스포티파이 인기음악 분석

PLAY

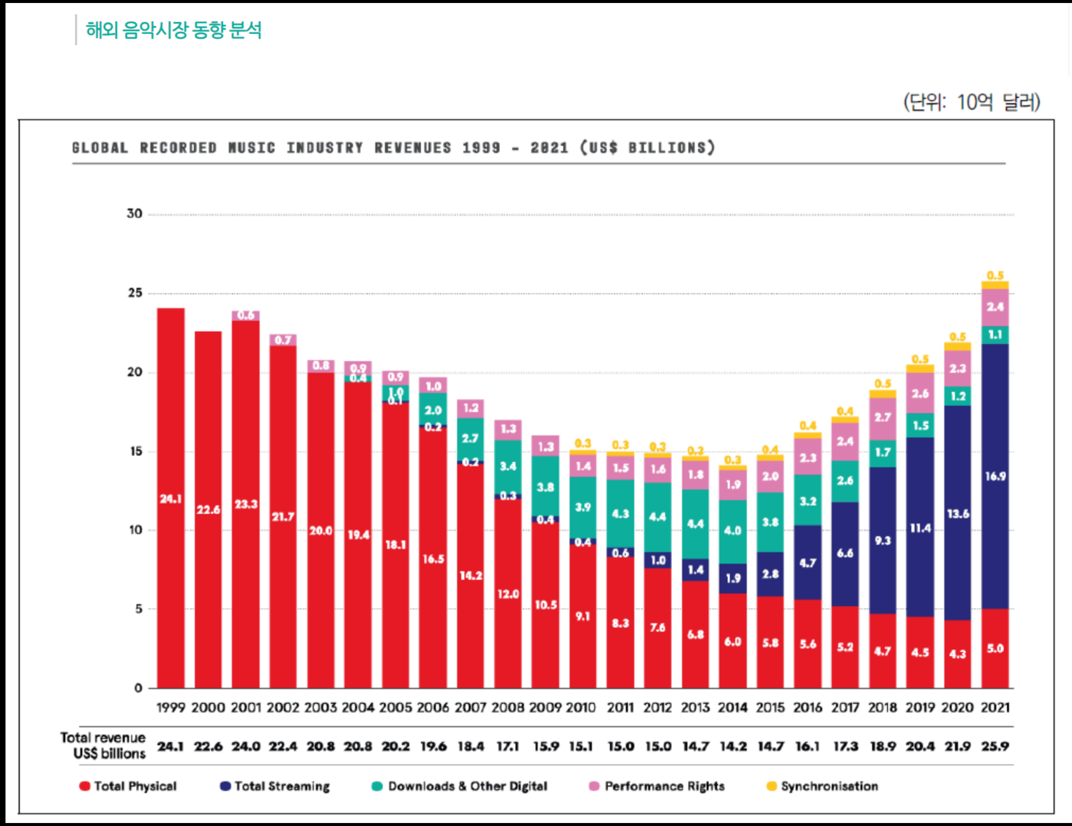
201910206 김태호

목차

1. 주제 선정 배경

대중적으로 인기 있는 음악의 공통점은 무엇일까?

→ 앞으로 음악시장에서의 히트곡 예상



출처 : "'21년 전세계음악 산업 32조원...유료스트리밍 3억명", 디엑스타임즈, 2023.02.28

2. 데이터 전처리

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt

df = pd.read_csv("/content/drive/MyDrive/Colab Notebooks/MusicData Analysis/genre_music - 복사본.csv")
```

```
df
```

	track	artist	danceability	energy	key	loudness	mode	speechiness	acousticness	instrumentalness	liveness	valence	tempo	duration_s	time_signature	chorus_hit	sections	popularity	decade	genre
0	Jealous Kind Of Fella	Garland Green	0.417	0.620	3	-7.727	1	0.0403	0.4900	0.000000	0.0779	0.8450	185.655	173.533	3	32.94975	9	1	60s	edm
1	Initials B.B.	Serge Gainsbourg	0.498	0.505	3	-12.475	1	0.0337	0.0180	0.107000	0.1760	0.7970	101.801	213.613	4	48.82510	10	0	60s	pop
2	Melody Twist	Lord Melody	0.657	0.649	5	-13.392	1	0.0380	0.8460	0.000004	0.1190	0.9080	115.940	223.960	4	37.22663	12	0	60s	pop
3	Mi Bomba Sonó	Celia Cruz	0.590	0.545	7	-12.058	0	0.1040	0.7060	0.024600	0.0610	0.9670	105.592	157.907	4	24.75484	8	0	60s	pop
4	Uravu Solla	P. Susheela	0.515	0.765	11	-3.515	0	0.1240	0.8570	0.000872	0.2130	0.9060	114.617	245.600	4	21.79874	14	0	60s	r&b
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
41094	Lotus Flowers	Yolta	0.172	0.358	9	-14.430	1	0.0342	0.8860	0.966000	0.3140	0.0361	72.272	150.857	4	24.30824	7	0	10s	rock
41095	Calling My Spirit	Kodak Black	0.910	0.366	1	-9.954	1	0.0941	0.0996	0.000000	0.2610	0.7400	119.985	152.000	4	32.53856	8	1	10s	pop
41096	Teenage Dream	Katy Perry	0.719	0.804	10	-4.581	1	0.0355	0.0132	0.000003	0.1390	0.6050	119.999	227.760	4	20.73371	7	1	10s	pop
41097	Stormy Weather	Oscar Peterson	0.600	0.177	7	-16.070	1	0.0561	0.9890	0.868000	0.1490	0.5600	120.030	213.387	4	21.65301	14	0	10s	pop
41098	Dust	Hans Zimmer	0.121	0.123	4	-23.025	0	0.0443	0.9640	0.696000	0.1030	0.0297	95.182	341.396	4	71.05343	15	0	10s	rock

41099 rows × 20 columns

원본데이터 : https://www.kaggle.com/code/akiboy96/spotify-song-popularity-genre-exploration/input?scriptVersionId=51654626

2. 데이터 전처리 – column 설명

danceability number [float]

Danceability describes how suitable a track is for dancing based on a combination of musical elements including tempo, rhythm stability, beat strength, and overall regularity. A value of 0.0 is least danceable and 1.0 is most danceable.

Example: 0.585

acousticness number [float]

A confidence measure from 0.0 to 1.0 of whether the track is acoustic. 1.0 represents high confidence the track is acoustic.

Range: 0 - 1
Example: 0.00242

duration_ms integer

The duration of the track in milliseconds.

Example: 237040

energy number [float]

Energy is a measure from 0.0 to 1.0 and represents a perceptual measure of intensity and activity. Typically, energetic tracks feel fast, loud, and noisy. For example, death metal has high energy, while a Bach prelude scores low on the scale. Perceptual features contributing to this attribute include dynamic range, perceived loudness, timbre, onset rate, and general entropy.

Example: 0.842

instrumentalness number [float]

Predicts whether a track contains no vocals. "Ooh" and "aah" sounds are treated as instrumental in this context. Rap or spoken word tracks are clearly "vocal". The closer the instrumentalness value is to 1.0, the greater likelihood the track contains no vocal content. Values above 0.5 are intended to represent instrumental tracks, but confidence is higher as the value approaches 1.0.

Example: 0.00686

speechiness number [float]

Speechiness detects the presence of spoken words in a track. The more exclusively speech-like the recording (e.g. talk show, audio book, poetry), the closer to 1.0 the attribute value. Values above 0.66 describe tracks that are probably made entirely of spoken words. Values between 0.33 and 0.66 describe tracks that may contain both music and speech, either in sections or layered, including such cases as rap music. Values below 0.33 most likely represent music and other non-speech-like tracks.

Example: 0.0556

liveness number [float]

Detects the presence of an audience in the recording. Higher liveness values represent an increased probability that the track was performed live. A value above 0.8 provides strong likelihood that the track is live.

Example: 0.0866

loudness number [float]

The overall loudness of a track in decibels (dB). Loudness values are averaged across the entire track and are useful for comparing relative loudness of tracks. Loudness is the quality of a sound that is the primary psychological correlate of physical strength (amplitude). Values typically range between -60 and 0 db.

Example: -5.883

mode integer

Mode indicates the modality (major or minor) of a track, the type of scale from which its melodic content is derived. Major is represented by 1 and minor is 0.

Example: 0

tempo number [float]

The overall estimated tempo of a track in beats per minute (BPM). In musical terminology, tempo is the speed or pace of a given piece and derives directly from the average beat duration.

Example: 118.211

time_signature integer

An estimated time signature. The time signature (meter) is a notational convention to specify how many beats are in each bar (or measure). The time signature ranges from 3 to 7 indicating time signatures of "3/4", to "7/4".

Range: 3 - 7
Example: 4

valence number [float]

A measure from 0.0 to 1.0 describing the musical positiveness conveyed by a track. Tracks with high valence sound more positive (e.g. happy, cheerful, euphoric), while tracks with low valence sound more negative (e.g. sad, depressed, angry).

Range: 0 - 1
Example: 0.428

key integer

The key the track is in. Integers map to pitches using standard Pitch Class notation. E.g. 0 = C, 1 = C♯/D♭, 2 = D, and so on. If no key was detected, the value is -1.

Range: -1 - 11
Example: 9

chorus_hit : This the the author's best estimate of when the chorus would start for the track. Its the timestamp of the start of the third section of the track (in milliseconds).
Sections : The number of sections the particular track has.
Popularity : It can be either '0' or '1'. '1' implies that this song has featured in the weekly list (Issued by Billboards) of Hot-100 tracks in that decade at least once and is therefore a 'hit'. '0' Implies that the track is a 'flop'.

2. 데이터 전처리

X값 / y값

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt

df = pd.read_csv("/content/drive/MyDrive/Colab Notebooks/MusicData Analysis/genre_music - 복사본.csv")
```

```
df
```

	track	artist	danceability	energy	key	loudness	mode	speechiness	acousticness	instrumentalness	liveness	valence	tempo	duration_s	time_signature	chorus_hit	sections	popularity	decade	genre
0	Jealous Kind Of Fella	Garland Green	0.417	0.620	3	-7.727	1	0.0403	0.4900	0.000000	0.0779	0.8450	185.655	173.533	3	32.94975	9	1	60s	edm
1	Initials B.B.	Serge Gainsbourg	0.498	0.505	3	-12.475	1	0.0337	0.0180	0.107000	0.1760	0.7970	101.801	213.613	4	48.82510	10	0	60s	pop
2	Melody Twist	Lord Melody	0.657	0.649	5	-13.392	1	0.0380	0.8460	0.000004	0.1190	0.9080	115.940	223.960	4	37.22663	12	0	60s	pop
3	Mi Bomba Sonó	Celia Cruz	0.590	0.545	7	-12.058	0	0.1040	0.7060	0.024600	0.0610	0.9670	105.592	157.907	4	24.75484	8	0	60s	pop
4	Uravu Solla	P. Susheela	0.515	0.765	11	-3.515	0	0.1240	0.8570	0.000872	0.2130	0.9060	114.617	245.600	4	21.79874	14	0	60s	r&b
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
41094	Lotus Flowers	Yolta	0.172	0.358	9	-14.430	1	0.0342	0.8860	0.966000	0.3140	0.0361	72.272	150.857	4	24.30824	7	0	10s	rock
41095	Calling My Spirit	Kodak Black	0.910	0.366	1	-9.954	1	0.0941	0.0996	0.000000	0.2610	0.7400	119.985	152.000	4	32.53856	8	1	10s	pop
41096	Teenage Dream	Katy Perry	0.719	0.804	10	-4.581	1	0.0355	0.0132	0.000003	0.1390	0.6050	119.999	227.760	4	20.73371	7	1	10s	pop
41097	Stormy Weather	Oscar Peterson	0.600	0.177	7	-16.070	1	0.0561	0.9890	0.868000	0.1490	0.5600	120.030	213.387	4	21.65301	14	0	10s	pop
41098	Dust	Hans Zimmer	0.121	0.123	4	-23.025	0	0.0443	0.9640	0.696000	0.1030	0.0297	95.182	341.396	4	71.05343	15	0	10s	rock

41099 rows × 20 columns

2. 데이터 전처리 – 목표 데이터

```python
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt

df = pd.read_csv("/content/drive/MyDrive/Colab Notebooks/MusicData Analysis/genre_music - 복사본.csv")
```

```python
df
```

	track	artist	danceability	energy	key	loudness	mode	speechiness	acousticness	instrumentalness	liveness	valence	tempo	duration_s	time_signature	chorus_hit	sections	popularity	decade	genre
0	Jealous Kind Of Fella	Garland Green	0.417	0.620	3	-7.727	1	0.0403	0.4900	0.000000	0.0779	0.8450	185.655	173.533	3	32.94975	9	1	60s	edm
1	Initials B.B.	Serge Gainsbourg	0.498	0.505	3	-12.475	1	0.0337	0.0180	0.107000	0.1760	0.7970	101.801	213.613	4	48.82510	10	0	60s	pop
2	Melody Twist	Lord Melody	0.657	0.649	5	-13.392	1	0.0380	0.8460	0.000004	0.1190	0.9080	115.940	223.960	4	37.22663	12	0	60s	pop
3	Mi Bomba Sonó	Celia Cruz	0.590	0.545	7	-12.058	0	0.1040	0.7060	0.024600	0.0610	0.9670	105.592	157.907	4	24.75484	8	0	60s	pop
4	Uravu Solla	P. Susheela	0.515	0.765	11	-3.515	0	0.1240	0.8570	0.000872	0.2130	0.9060	114.617	245.600	4	21.79874	14	0	60s	r&b
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
41094	Lotus Flowers	Yolta	0.172	0.358	9	-14.430	1	0.0342	0.8860	0.966000	0.3140	0.0361	72.272	150.857	4	24.30824	7	0	10s	rock
41095	Calling My Spirit	Kodak Black	0.910	0.366	1	-9.954	1	0.0941	0.0996	0.000000	0.2610	0.7400	119.985	152.000	4	32.53856	8	1	10s	pop
41096	Teenage Dream	Katy Perry	0.719	0.804	10	-4.581	1	0.0355	0.0132	0.000003	0.1390	0.6050	119.999	227.760	4	20.73371	7	1	10s	pop
41097	Stormy Weather	Oscar Peterson	0.600	0.177	7	-16.070	1	0.0561	0.9890	0.868000	0.1490	0.5600	120.030	213.387	4	21.65301	14	0	10s	pop
41098	Dust	Hans Zimmer	0.121	0.123	4	-23.025	0	0.0443	0.9640	0.696000	0.1030	0.0297	95.182	341.396	4	71.05343	15	0	10s	rock

41099 rows × 20 columns

2. 데이터 전처리 – 목표 데이터

```
df = df[df['decade'].isin(['00s', '10s'])] # 'decade' 컬럼에서 '00s'와 '10s'가 아닌 행 삭제
```

```
df
```

	track	artist	danceability	energy	key	loudness	mode	speechiness	acousticness	instrumentalness	liveness	valence	tempo	duration_s	time_signature	chorus_hit	sections	popularity	decade	genre
28832	Lucky Man	Montgomery Gentry	0.578	0.471	4	-7.270	1	0.0289	0.368000	0.000000	0.159	0.5320	133.061	196.707	4	30.88059	13	1	00s	pop
28833	On The Hotline	Pretty Ricky	0.704	0.854	10	-5.477	0	0.1830	0.018500	0.000000	0.148	0.6880	92.988	242.587	4	41.51106	10	1	00s	r&b
28834	Clouds Of Dementia	Candlemass	0.162	0.836	9	-3.009	1	0.0473	0.000111	0.004570	0.174	0.3000	86.964	338.893	4	65.32887	13	0	00s	rock
28835	Heavy Metal, Raise Hell!	Zwartketterij	0.188	0.994	4	-3.745	1	0.1660	0.000007	0.078400	0.192	0.3330	148.440	255.667	4	58.59528	9	0	00s	rock
28836	I Got A Feelin'	Billy Currington	0.630	0.764	2	-4.353	1	0.0275	0.363000	0.000000	0.125	0.6310	112.098	193.760	4	22.62384	10	1	00s	pop
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
41094	Lotus Flowers	Yolta	0.172	0.358	9	-14.430	1	0.0342	0.886000	0.966000	0.314	0.0361	72.272	150.857	4	24.30824	7	0	10s	rock
41095	Calling My Spirit	Kodak Black	0.910	0.366	1	-9.954	1	0.0941	0.099600	0.000000	0.261	0.7400	119.985	152.000	4	32.53856	8	1	10s	pop
41096	Teenage Dream	Katy Perry	0.719	0.804	10	-4.581	1	0.0355	0.013200	0.000003	0.139	0.6050	119.999	227.760	4	20.73371	7	1	10s	pop
41097	Stormy Weather	Oscar Peterson	0.600	0.177	7	-16.070	1	0.0561	0.989000	0.868000	0.149	0.5600	120.030	213.387	4	21.65301	14	0	10s	pop
41098	Dust	Hans Zimmer	0.121	0.123	4	-23.025	0	0.0443	0.964000	0.696000	0.103	0.0297	95.182	341.396	4	71.05343	15	0	10s	rock

12267 rows × 20 columns

2. 데이터 전처리

```
df.dtypes
```

```
track               object
artist              object
danceability       float64
energy             float64
key                  int64
loudness           float64
mode                 int64
speechiness        float64
acousticness       float64
instrumentalness   float64
liveness           float64
valence            float64
tempo              float64
duration_s         float64
time_signature       int64
chorus_hit         float64
sections             int64
popularity           int64
decade              object
genre               object
dtype: object
```

← 데이터 dtype 확인

```
df.isnull().sum()
```

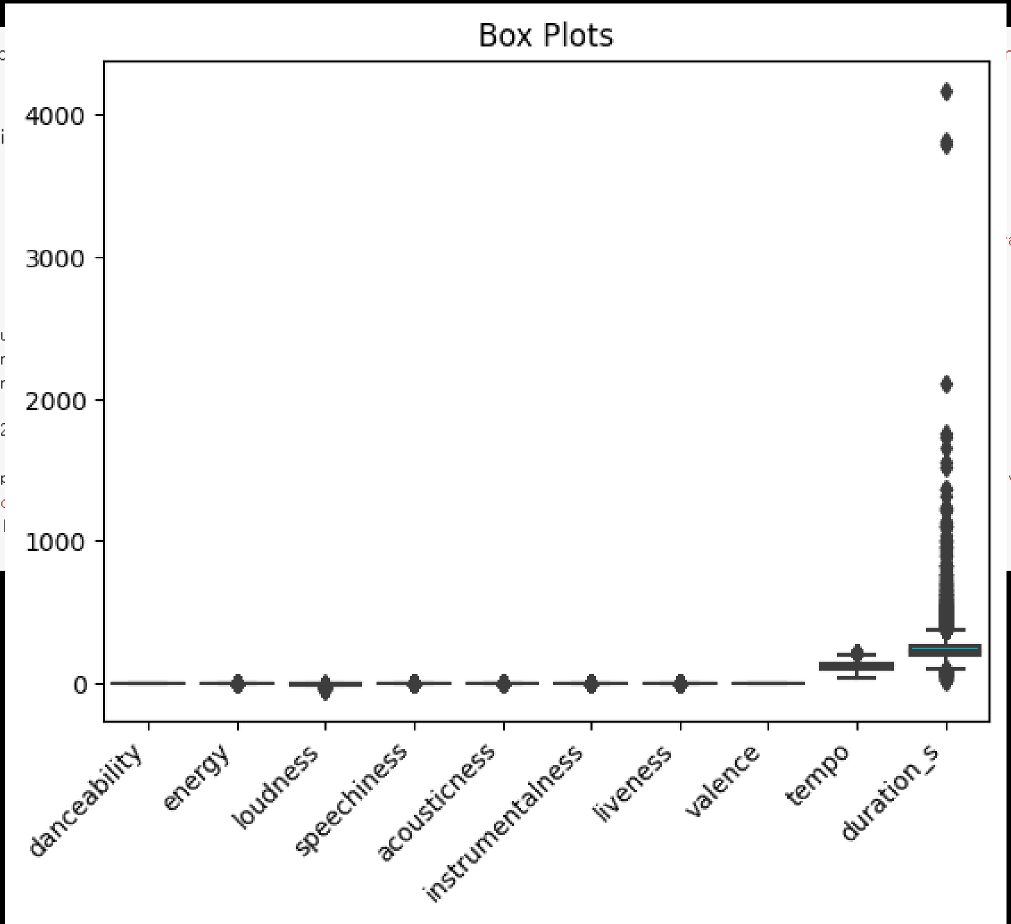
```
track               0
artist              0
danceability        0
energy              0
key                 0
loudness            0
mode                0
speechiness         0
acousticness        0
instrumentalness    0
liveness            0
valence             0
tempo               0
duration_s          0
time_signature      0
chorus_hit          0
sections            0
popularity          0
decade              0
genre               0
dtype: int64
```

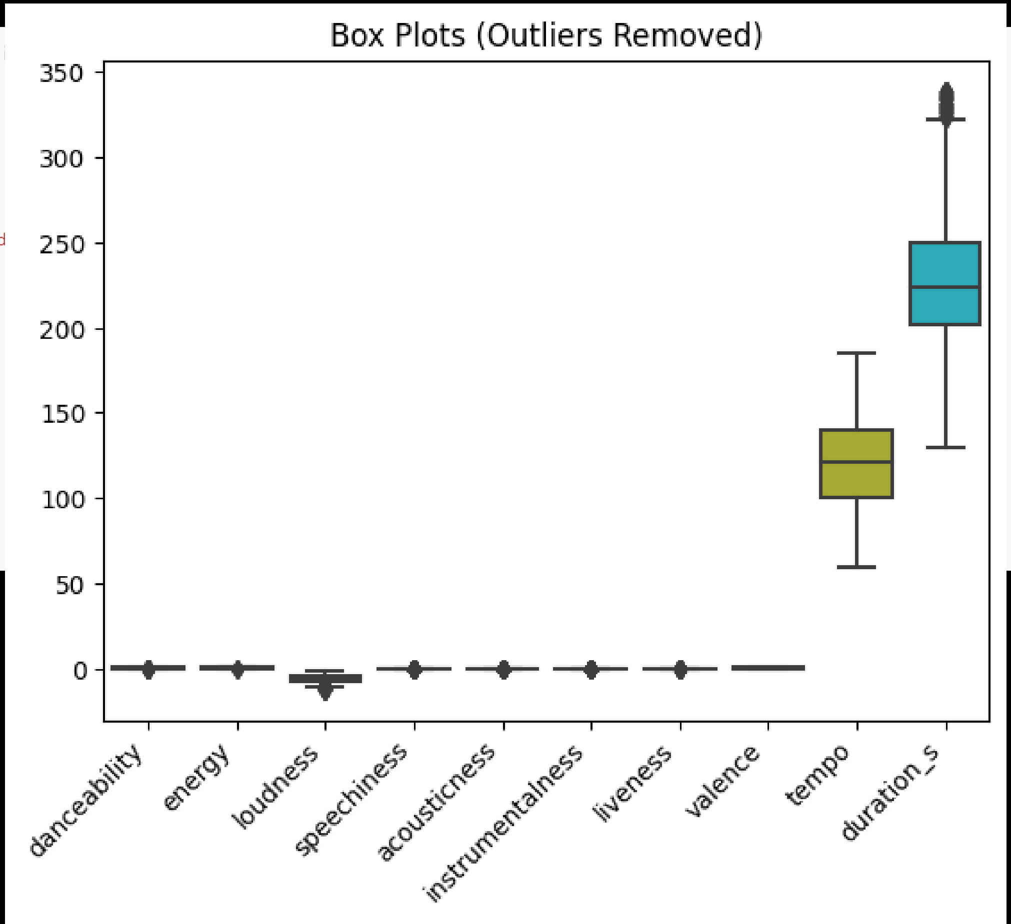
결측값 확인 →

2. 데이터 전처리 – outlier 제거

```
boxplot = sns.b
plt.title('Box

boxplot.set_xti

plt.show()

columns_to_filter

df2 = df.copy()

for column in colu
    Q1 = df[column
    Q3 = df[column
    IQR = Q3 - Q1
    df2 = df2[(df2

boxplot = sns.boxp
plt.title('Box Plo
boxplot.set_xtick
plt.show()
```

```
rumentalness', 'li

alence', 'tempo', 'd

veness', 'valence',
```

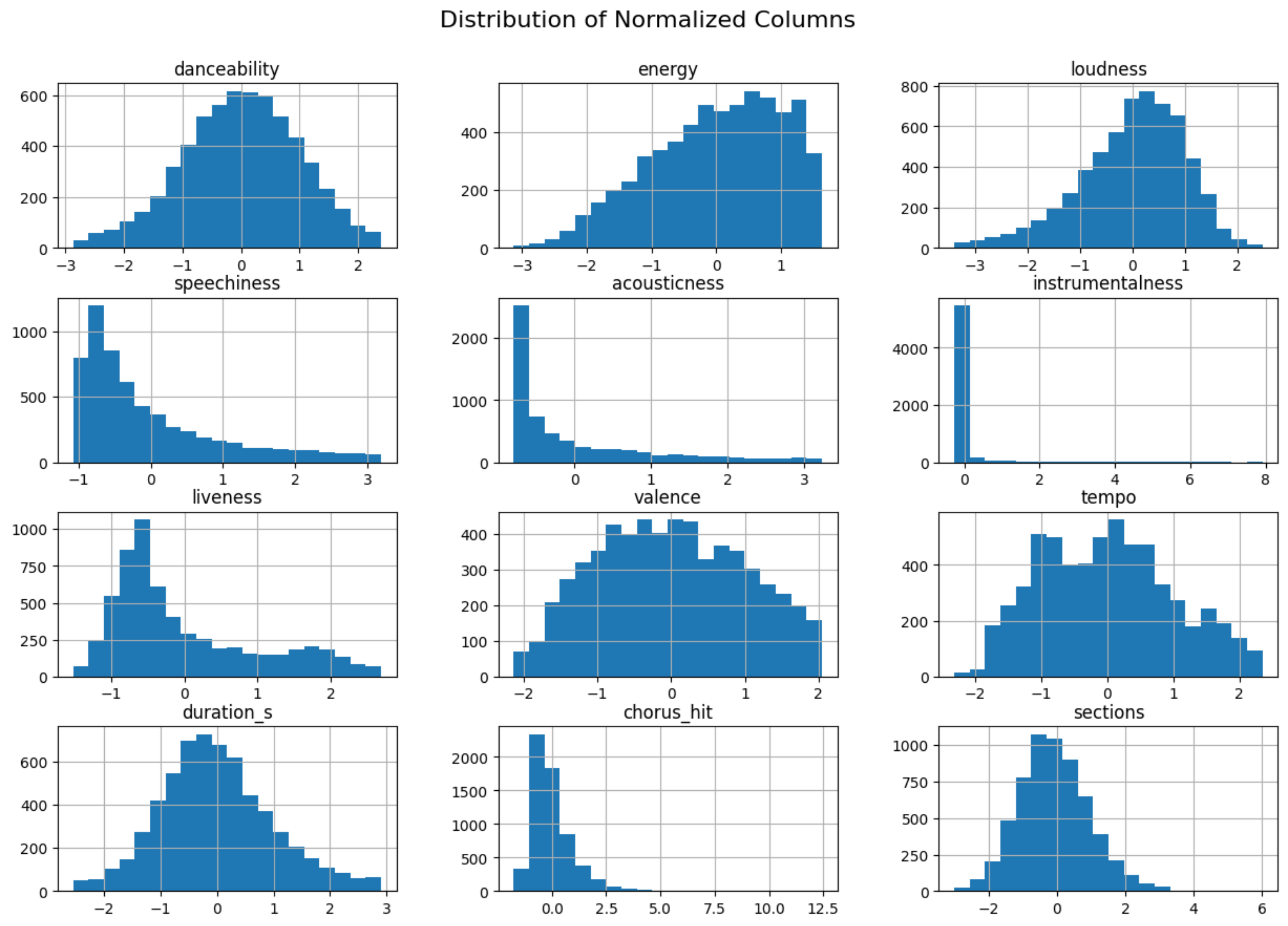




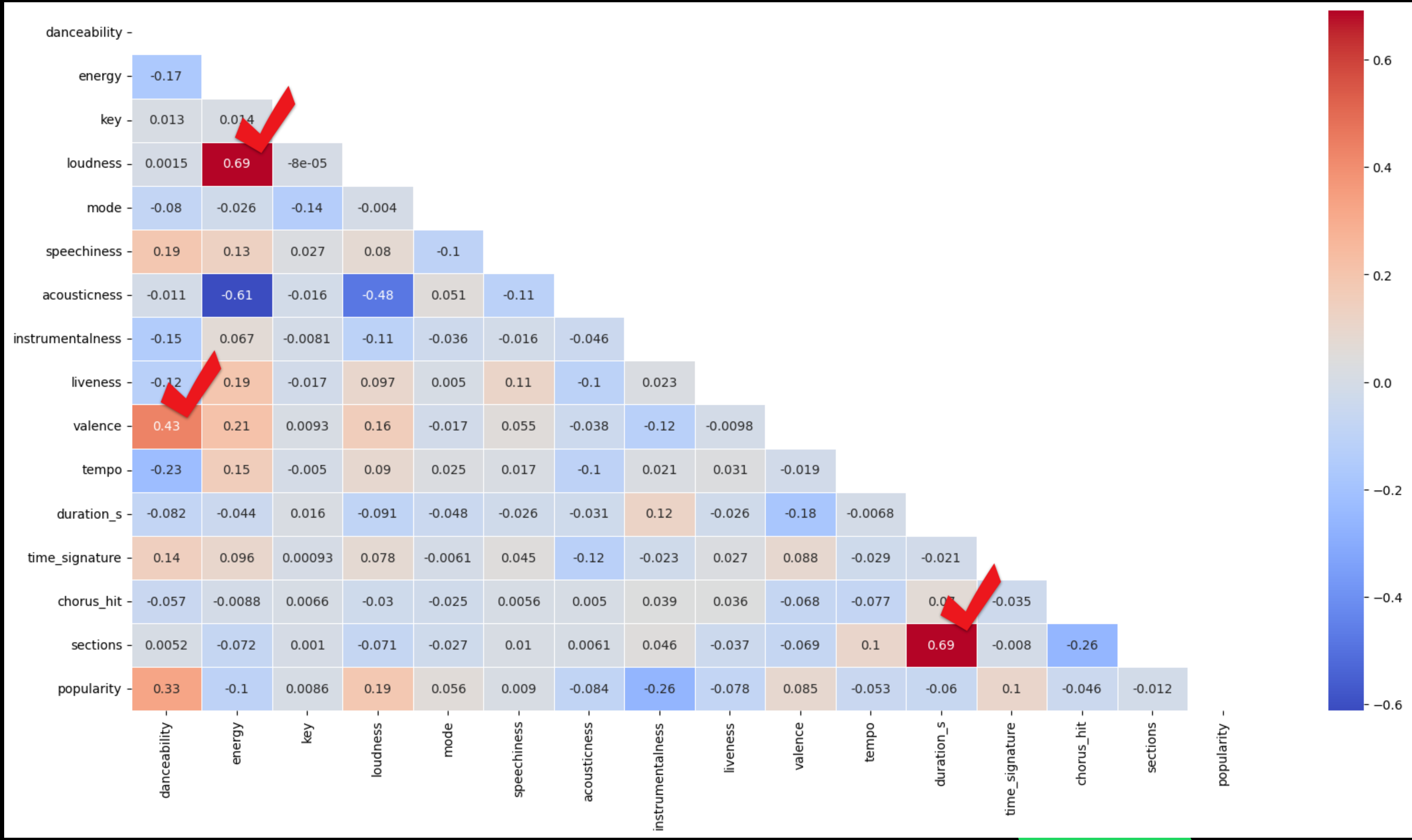
2. 데이터 전처리 – 정규화

```
from sklearn.preprocessing import StandardScaler

columns_to_normalize = ['danceability', 'energy', 'lou

scaler = StandardScaler()

df2[columns_to_normalize] = scaler.fit_transform(df2[c
```

```
import matplotlib.pyplot as plt

df2[columns_to_normalize].hist(bins=20, figsize=(15, 1
plt.suptitle('Distribution of Normalized Columns', y=0
plt.show()
```



2. 데이터 전처리 – 변수들간의 상관관계



2. 데이터 전처리 – 변수들간의 상관관계

다중 공선성

[multicollinearity]

회귀 분석에서 설명 변수 중에 서로 상관이 높은 것이 포함되어 있을 때는 분산·공분산 행렬의 행렬식이 0에 가까운 값이 되어 회귀 계수의 추정 정밀도가 매우 나빠지는 일이 발생하는데, 이러한 현상을 다중 공선성이라 한다.

진단법 [편집]

1. 결정계수 R^2값은 높아 회귀식의 설명력은 높지만 식안의 독립변수의 P값(P-value)이 커서 개별 인자들이 유의하지 않는 경우가 있다. 이런 경우 독립변수들 간에 높은 상관관계가 있다고 의심된다.
2. 독립변수들간의 상관계수를 구한다.
3. 분산팽창요인(VIF, Variance Inflation Factor)을 구하여 이 값이 10을 넘는다면 보통 다중공선성의 문제가 있다.

2. 데이터 전처리 – 변수들간의 상관관계

```
from statsmodels.stats.outliers_influence import variance_inflation_factor

selected_features = df[['loudness', 'energy']]

vif_data = pd.DataFrame()
vif_data["Variable"] = selected_features.columns
vif_data["VIF"] = [variance_inflation_factor(selected_features.values, i) for i in range(selected_features.shape[1])]

print(vif_data)
```

```
   Variable       VIF
0  loudness  1.663157
1    energy  1.663157
```

'loudness' , 'energy'

Vif : 1.663157 < 10

```
from statsmodels.stats.outliers_influence import variance_inflation_factor

selected_features = df[['duration_s', 'sections']]

vif_data = pd.DataFrame()
vif_data["Variable"] = selected_features.columns
vif_data["VIF"] = [variance_inflation_factor(selected_features.values, i) for i in range(selected_features.shape[1])]

print(vif_data)
```

```
     Variable        VIF
0  duration_s  24.470168
1    sections  24.470168
```

'duration_s' , 'sections'

Vif : 24.470168 > 10

'sections' 분석에서 제외

```
from statsmodels.stats.outliers_influence import variance_inflation_factor

selected_features = df[['valence', 'danceability']]

vif_data = pd.DataFrame()
vif_data["Variable"] = selected_features.columns
vif_data["VIF"] = [variance_inflation_factor(selected_features.values, i) for i in range(selected_features.shape[1])]

print(vif_data)
```

```
       Variable       VIF
0       valence  6.070815
1  danceability  6.070815
```

'valence' , 'danceability'

Vif : 6.070815 < 10

2. 데이터 전처리 – key column

```
unique_keys = df2['key'].unique()
unique_keys
```

```
array([ 4, 10,  9,  2,  1, 11,  7,  8,  0,  3,  5,  6])
```

df2

	track	artist	danceability	energy	key	loudness	mode	speechiness	acousticness	instrumentalness	...	key_2	key_3	key_4	key_5	key_6	key_7	key_8	key_9	key_10	key_11
28832	Lucky Man	Montgomery Gentry	-0.092609	-1.289587	4	-0.463083	1	-0.786918	0.862941	-0.277155	...	0	0	1	0	0	0	0	0	0	0
28833	On The Hotline	Pretty Ricky	0.670357	0.740870	10	0.233707	0	1.413774	-0.652801	-0.277155	...	0	0	0	0	0	0	0	0	1	0
28834	Clouds Of Dementia	Candlemass	-2.611607	0.645444	9	1.192814	1	-0.524148	-0.732552	-0.103450	...	0	0	0	0	0	0	0	1	0	0
28835	Heavy Metal, Raise Hell!	Zwartketterij	-2.454169	1.483073	4	0.906792	1	1.170998	-0.733001	2.702823	...	0	0	1	0	0	0	0	0	0	0
28836	I Got A Feelin'	Billy Currington	0.222266	0.263739	2	0.670513	1	-0.806911	0.841257	-0.277155	...	1	0	0	0	0	0	0	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
41091	Tear In My Heart	twenty one pilots	0.373648	-0.436053	2	0.496024	1	-0.501299	-0.651066	-0.277155	...	1	0	0	0	0	0	0	0	0	0
41092	Sweater Weather	The Neighbourhood	0.113271	0.491701	10	1.270149	1	-0.719797	-0.518357	0.395620	...	0	0	0	0	0	0	0	0	1	0
41093	Untouchable	YoungBoy Never Broke Again	1.130558	0.369768	1	0.403921	1	1.456617	-0.539174	-0.277155	...	0	0	0	0	0	0	0	0	0	0
41095	Calling My Spirit	Kodak Black	1.917745	-1.846240	1	-1.506131	1	0.144199	-0.301079	-0.277155	...	0	0	0	0	0	0	0	0	0	0
41096	Teenage Dream	Katy Perry	0.761186	0.475797	10	0.581908	1	-0.692663	-0.675786	-0.277040	...	0	0	0	0	0	0	0	0	1	0

8832 rows × 32 columns

```
6      F#,Gb
7          G
8      G#,Ab
9          A
10     A#,Bb
11         B
Name: key_meaning, dtype: object
```

```
# 'key' 원-핫 인코딩
key_encoded = pd.get_dummies(df2['key'], prefix='key', prefix_sep='_')
df2 = pd.concat([df2, key_encoded], axis=1)
```

2. 데이터 전처리 – genre column

```
unique_genre = df2['genre'].unique()
unique_genre
```

```
array(['pop', 'r&b', 'rock', 'latin', 'edm', 'rap'], dtype=object)
```

```
# 'genre' 컬럼을 원-핫 인코딩
genre_encoded = pd.get_dummies(df2['genre'], prefix='genre', prefix_sep='_')
df2 = pd.concat([df2, genre_encoded], axis=1)
```

```
df2
```

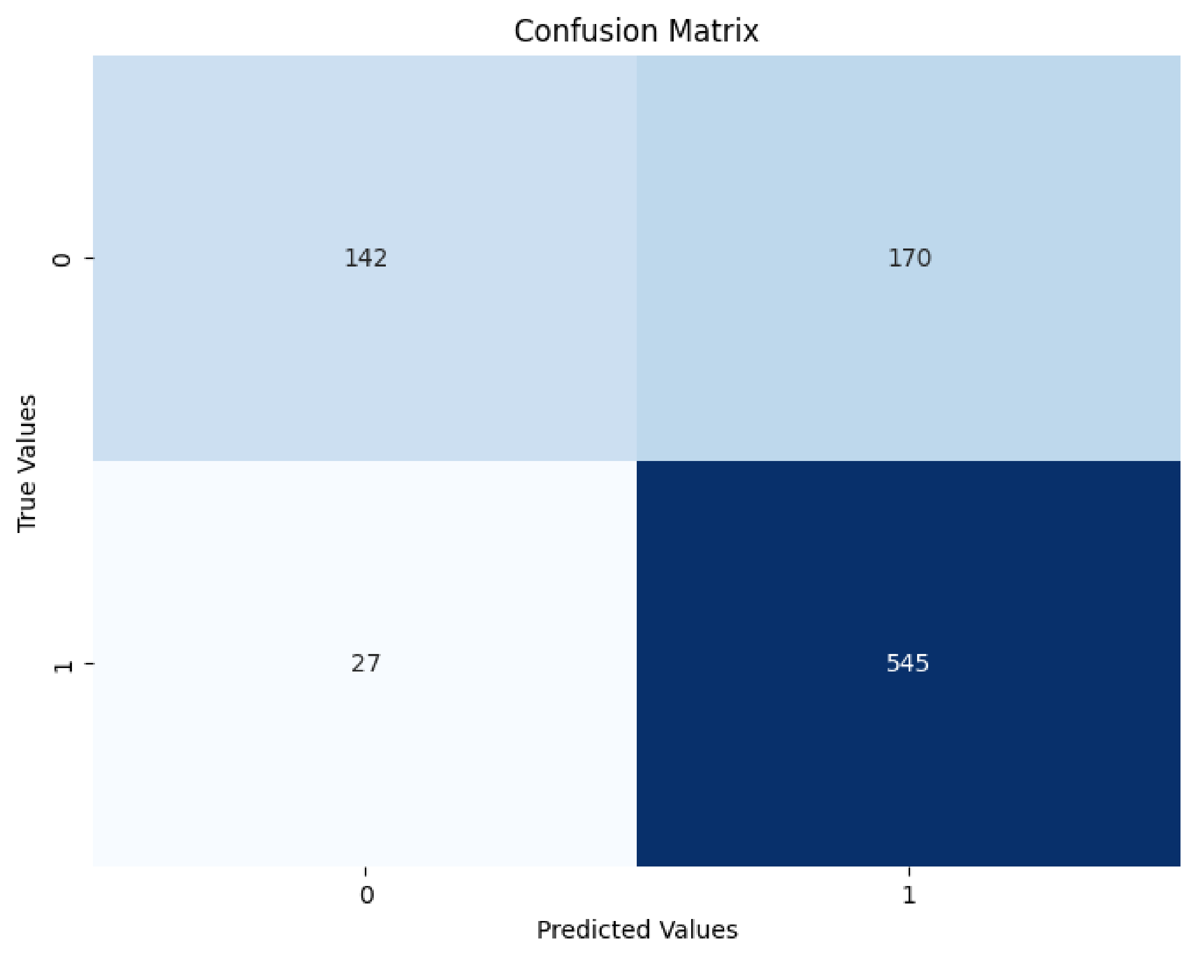
	track	artist	danceability	energy	loudness	mode	speechiness	acousticness	instrumentalness	liveness	...	key_8	key_9	key_10	key_11	genre_edm	genre_latin	genre_pop	genre_r&b	genre_rap	genre_rock
28832	Lucky Man	Montgomery Gentry	-0.092609	-1.289587	-0.463083	1	-0.786918	0.862941	-0.277155	-0.187851	...	0	0	0	0	0	0	1	0	0	0
28833	On The Hotline	Pretty Ricky	0.670357	0.740870	0.233707	0	1.413774	-0.652801	-0.277155	-0.271321	...	0	0	1	0	0	0	0	1	0	0
28834	Clouds Of Dementia	Candlemass	-2.611607	0.645444	1.192814	1	-0.524148	-0.732552	-0.103450	-0.074029	...	0	1	0	0	0	0	0	0	0	1
28835	Heavy Metal, Raise Hell!	Zwartketterij	-2.454169	1.483073	0.906792	1	1.170998	-0.733001	2.702823	0.062558	...	0	0	0	0	0	0	0	0	0	1
28836	I Got A Feelin'	Billy Currington	0.222266	0.263739	0.670513	1	-0.806911	0.841257	-0.277155	-0.445849	...	0	0	0	0	0	0	1	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
41091	Tear In My Heart	twenty one pilots	0.373648	-0.436053	0.496024	1	-0.501299	-0.651066	-0.277155	-0.846504	...	0	0	0	0	0	0	1	0	0	0
41092	Sweater Weather	The Neighbourhood	0.113271	0.491701	1.270149	1	-0.719797	-0.518357	0.395620	-0.627965	...	0	0	1	0	0	0	0	1	0	0
41093	Untouchable	YoungBoy Never Broke Again	1.130558	0.369768	0.403921	1	1.456617	-0.539174	-0.277155	-0.468614	...	0	0	0	0	0	0	1	0	0	0
41095	Calling My Spirit	Kodak Black	1.917745	-1.846240	-1.506131	1	0.144199	-0.301079	-0.277155	0.586142	...	0	0	0	0	0	0	1	0	0	0
41096	Teenage Dream	Katy Perry	0.761186	0.475797	0.581908	1	-0.692663	-0.675786	-0.277040	-0.339615	...	0	0	1	0	0	0	1	0	0	0

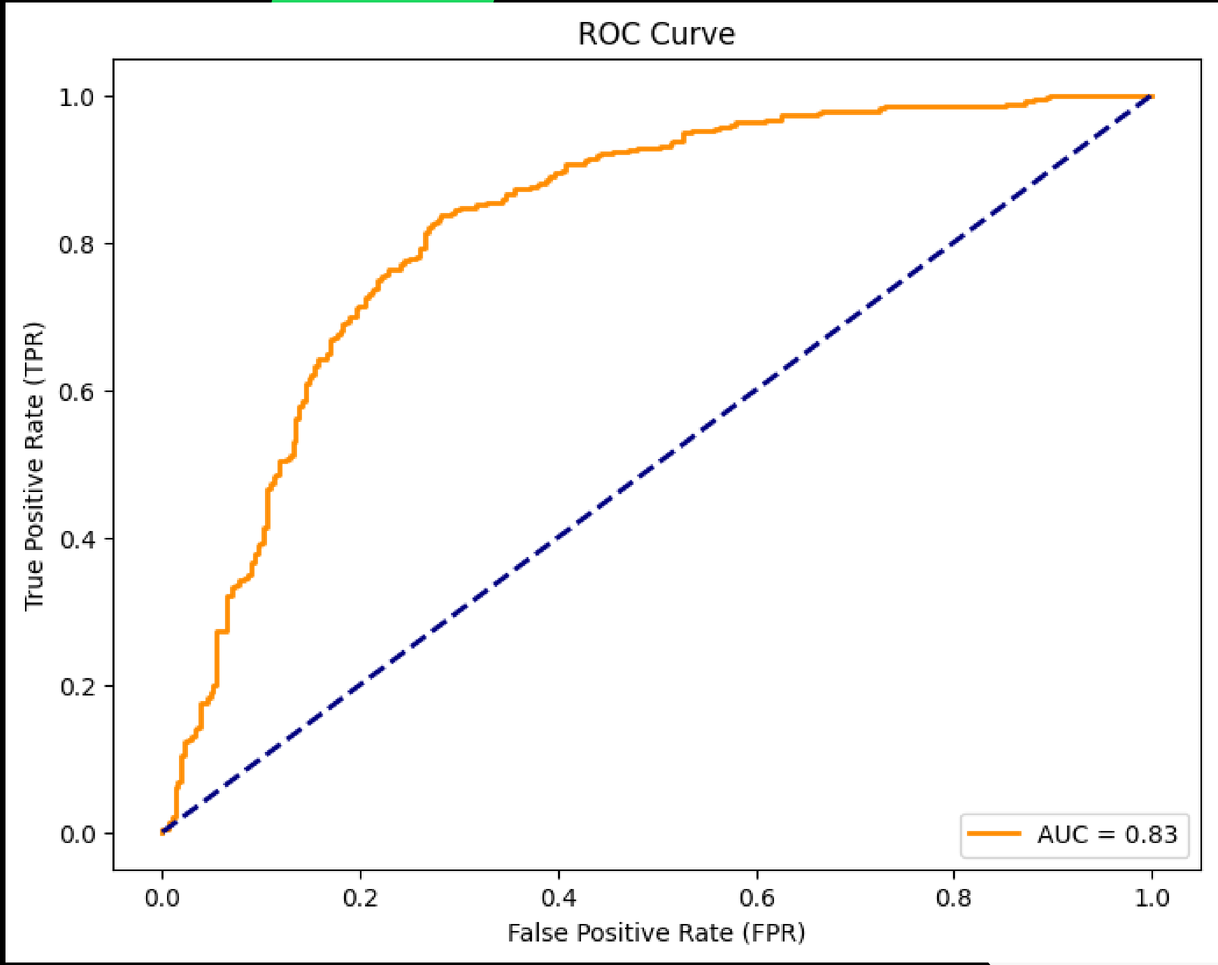
3. Logistic Regression

```python
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report, f1_score
from sklearn.metrics import roc_curve, roc_auc_score
import statsmodels.api as sm
import matplotlib.pyplot as plt
import seaborn as sns

features = ['danceability', 'energy', 'loudness', 'mode', 'speechiness', 'acousticness',
            'instrumentalness', 'liveness', 'valence', 'tempo', 'duration_s', 'time_signature',
            'chorus_hit', 'key_0', 'key_1', 'key_2', 'key_3', 'key_4', 'key_5', 'key_6',
            'key_7', 'key_8', 'key_9', 'key_10', 'key_11', 'genre_edm', 'genre_latin', 'genre_pop',
            'genre_r&b', 'genre_rap', 'genre_rock']

X = sm.add_constant(df2[features])
y = df2['popularity']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.1, random_state=0)

model = sm.Logit(y_train, X_train)
result = model.fit()

y_pred_prob = result.predict(X_test)
y_pred = (y_pred_prob > 0.4).astype(int)

accuracy = accuracy_score(y_test, y_pred)
conf_matrix = confusion_matrix(y_test, y_pred)
class_report = classification_report(y_test, y_pred)
f1 = f1_score(y_test, y_pred)

print(f"Accuracy: {accuracy}")
print(f"Confusion Matrix:₩n{conf_matrix}")
print(f"Classification Report:₩n{class_report}")
print(f"F1 Score: {f1}")
```

```
Accuracy: 0.7771493212669683
Confusion Matrix:
[[142 170]
 [ 27 545]]
Classification Report:
              precision    recall  f1-score   support

           0       0.84      0.46      0.59       312
           1       0.76      0.95      0.85       572

    accuracy                           0.78       884
   macro avg       0.80      0.70      0.72       884
weighted avg       0.79      0.78      0.76       884

F1 Score: 0.846930846930846
```

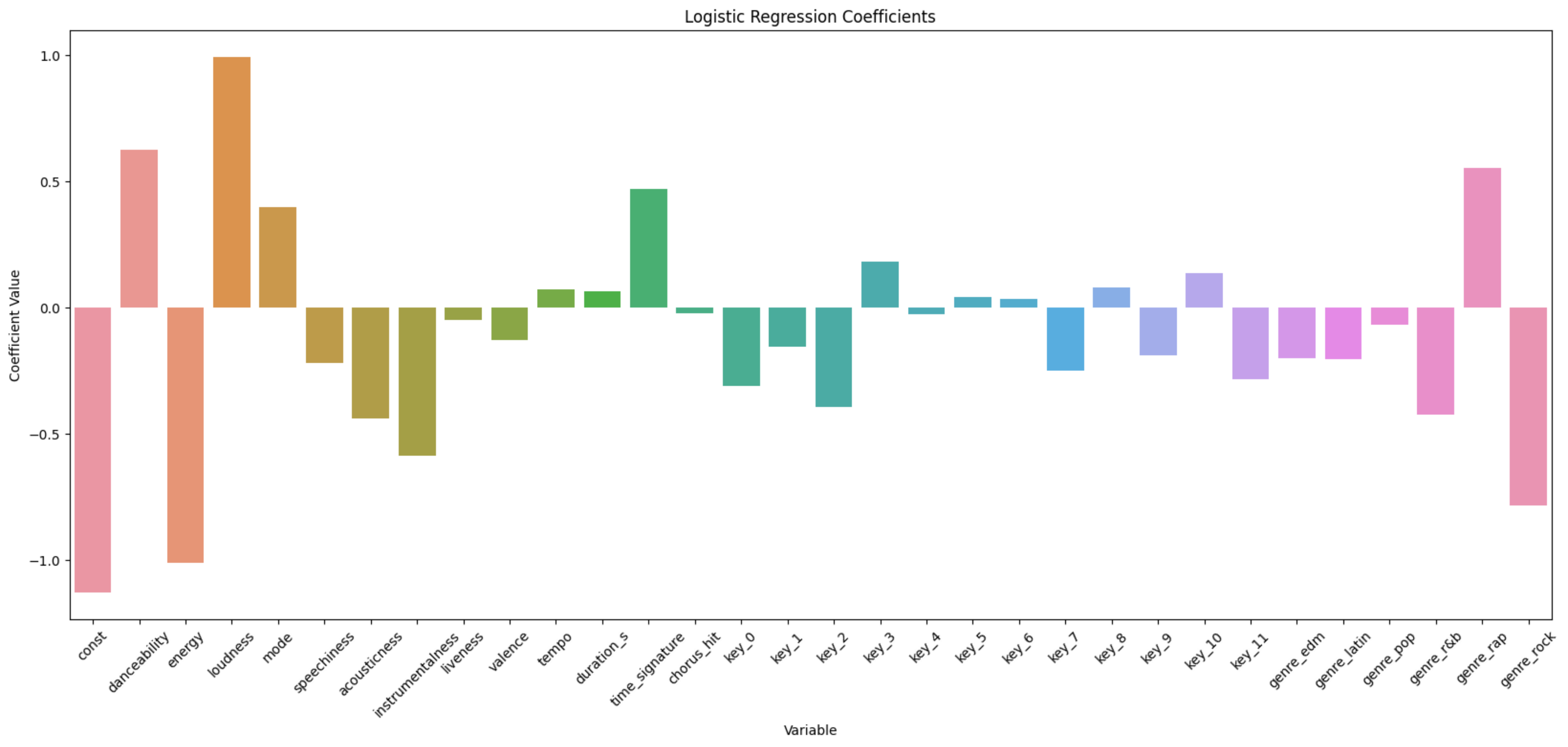
3. Logistic Regression





3. Logistic Regression

```
Coefficients:
const              -1.129454
danceability        0.625274
energy             -1.013829
loudness            0.994243
mode                0.400368
speechiness        -0.219417
acousticness       -0.440131
instrumentalness   -0.585970
liveness           -0.047910
valence            -0.128629
tempo               0.072907
duration_s          0.063884
time_signature      0.470405
chorus_hit         -0.021622
key_0              -0.310994
key_1              -0.153672
key_2              -0.392040
key_3               0.182336
key_4              -0.026138
key_5               0.042526
key_6               0.034774
key_7              -0.250810
key_8               0.078474
key_9              -0.188130
key_10              0.137446
key_11             -0.283228
genre_edm          -0.201195
genre_latin        -0.204108
genre_pop          -0.066315
genre_r&b          -0.426018
genre_rap           0.552685
genre_rock         -0.784503
dtype: float64
```

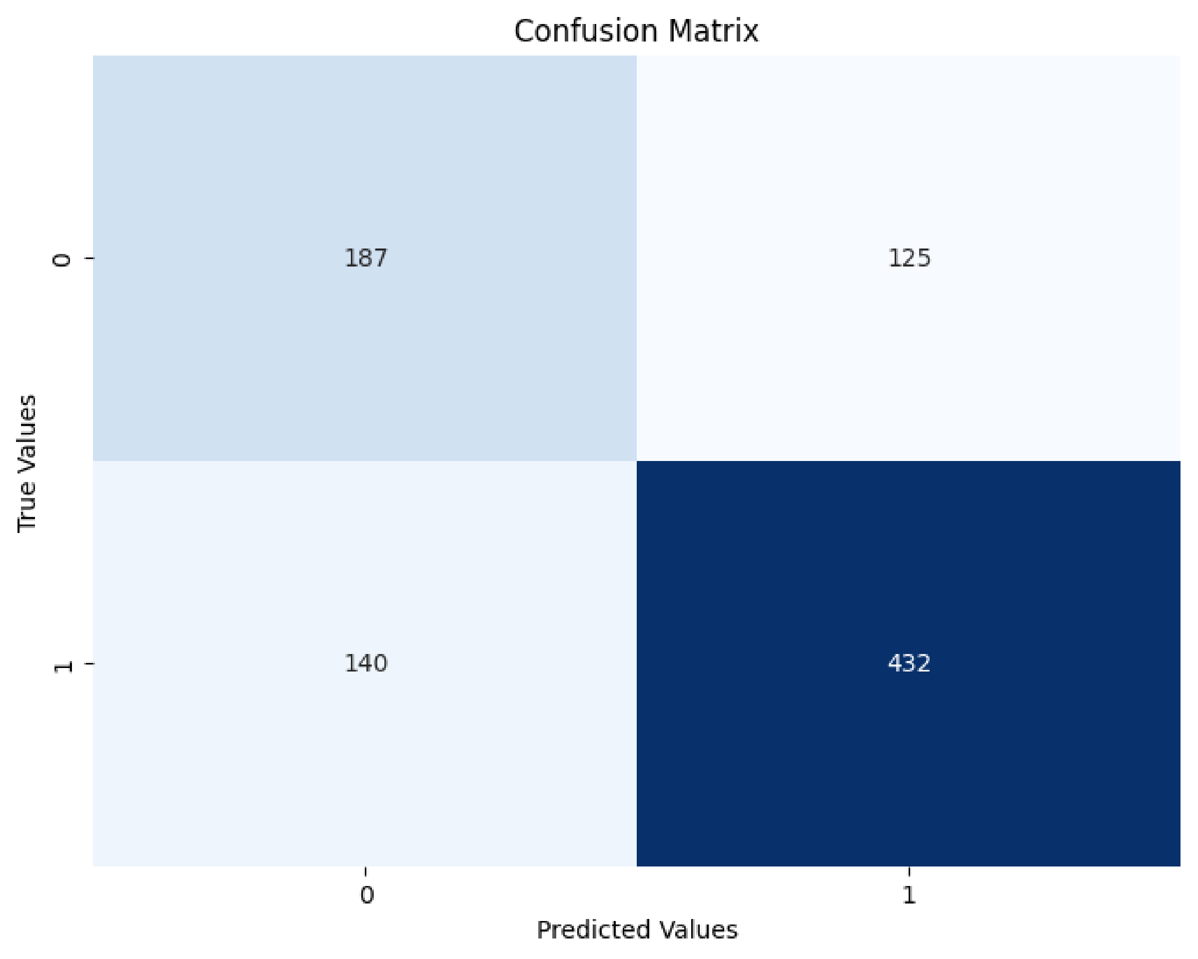


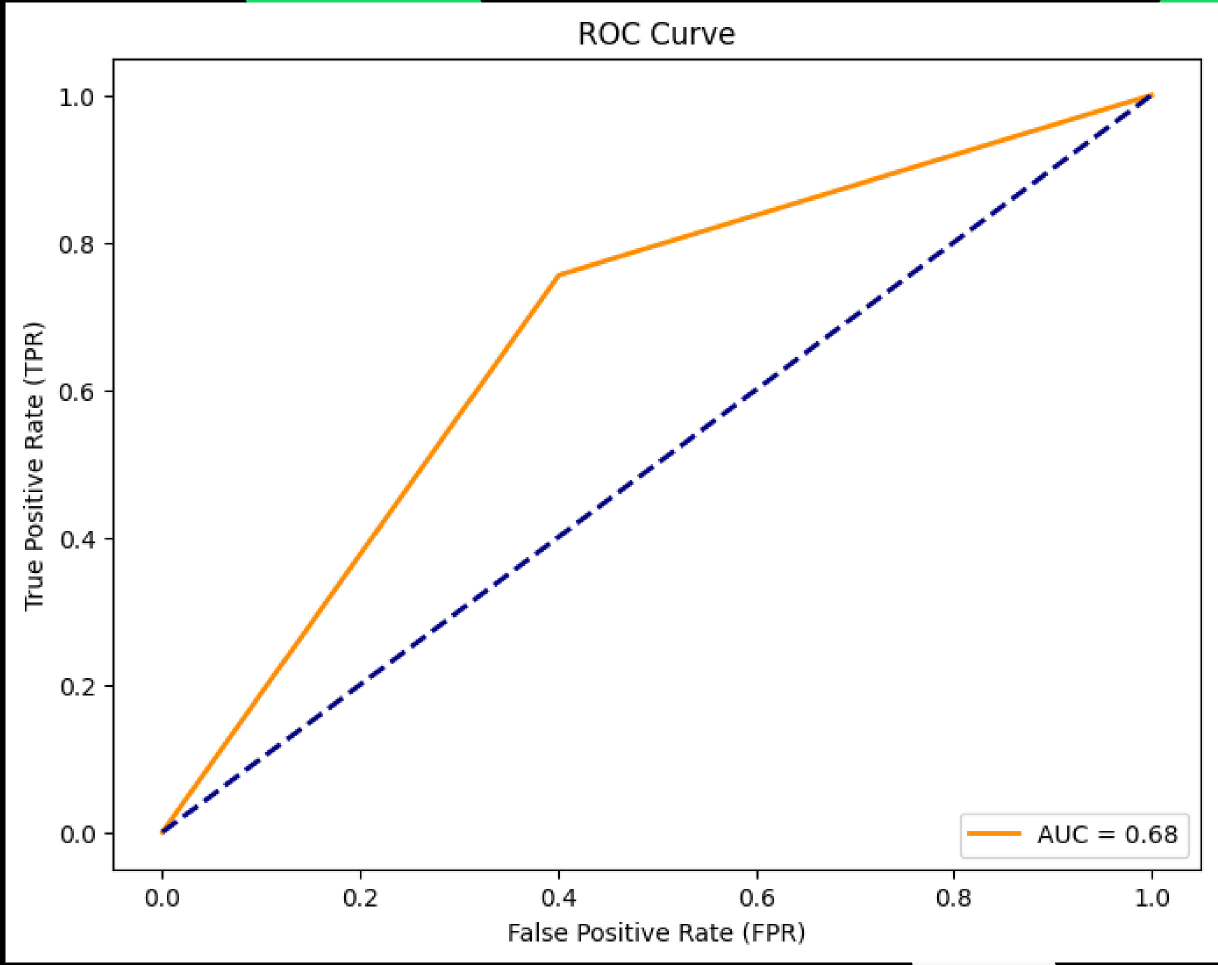
4. Decision Tree

```python
from sklearn.tree import DecisionTreeClassifier

features = ['danceability', 'energy', 'loudness', 'mode', 'speechiness', 'acousticness',
            'instrumentalness', 'liveness', 'valence', 'tempo', 'duration_s', 'time_signature',
            'chorus_hit', 'key_0', 'key_1', 'key_2', 'key_3', 'key_4', 'key_5', 'key_6',
            'key_7', 'key_8', 'key_9', 'key_10', 'key_11', 'genre_edm', 'genre_latin', 'genre_pop',
            'genre_r&b', 'genre_rap', 'genre_rock']

X = df2[features]
y = df2['popularity']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.1, random_state=0)

dt_model = DecisionTreeClassifier(criterion='gini', max_depth=None,
                                   min_samples_split=2, min_samples_leaf=1,
                                   max_features=None, random_state=0)

dt_model.fit(X_train, y_train)

y_pred = dt_model.predict(X_test)
y_pred_prob = dt_model.predict_proba(X_test)[:, 1]

accuracy = accuracy_score(y_test, y_pred)
conf_matrix = confusion_matrix(y_test, y_pred)
class_report = classification_report(y_test, y_pred)
f1 = f1_score(y_test, y_pred)

print(f"Accuracy: {accuracy}")
print(f"Confusion Matrix:₩n{conf_matrix}")
print(f"Classification Report:₩n{class_report}")
print(f"F1 Score: {f1}")
```

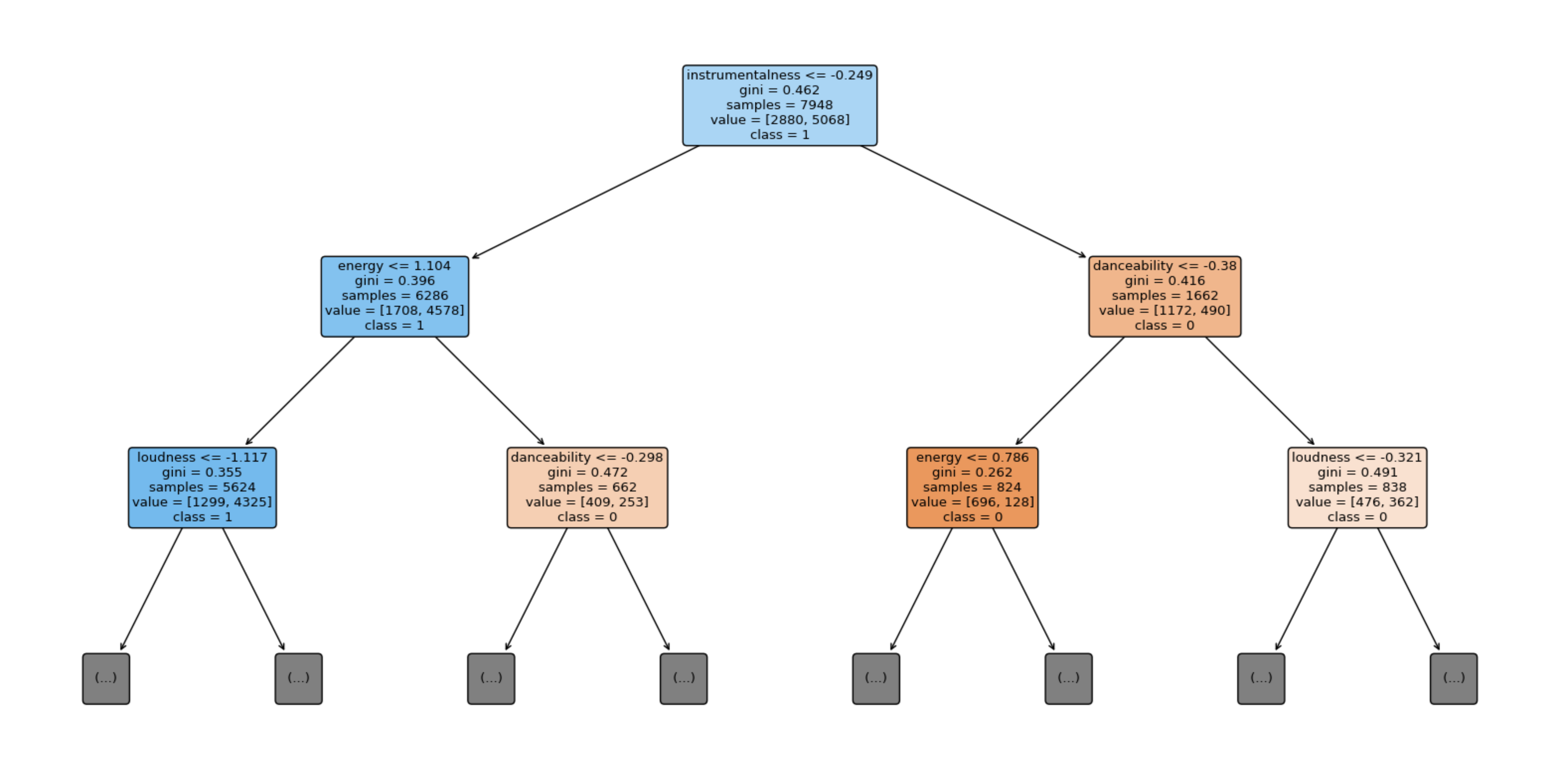
```
Accuracy: 0.7002262443438914
Confusion Matrix:
[[187 125]
 [140 432]]
Classification Report:
              precision    recall  f1-score   support

           0       0.57      0.60      0.59       312
           1       0.78      0.76      0.77       572

    accuracy                           0.70       884
   macro avg       0.67      0.68      0.68       884
weighted avg       0.70      0.70      0.70       884

F1 Score: 0.7652790079716563
```

4. Decision Tree





4. Decision Tree

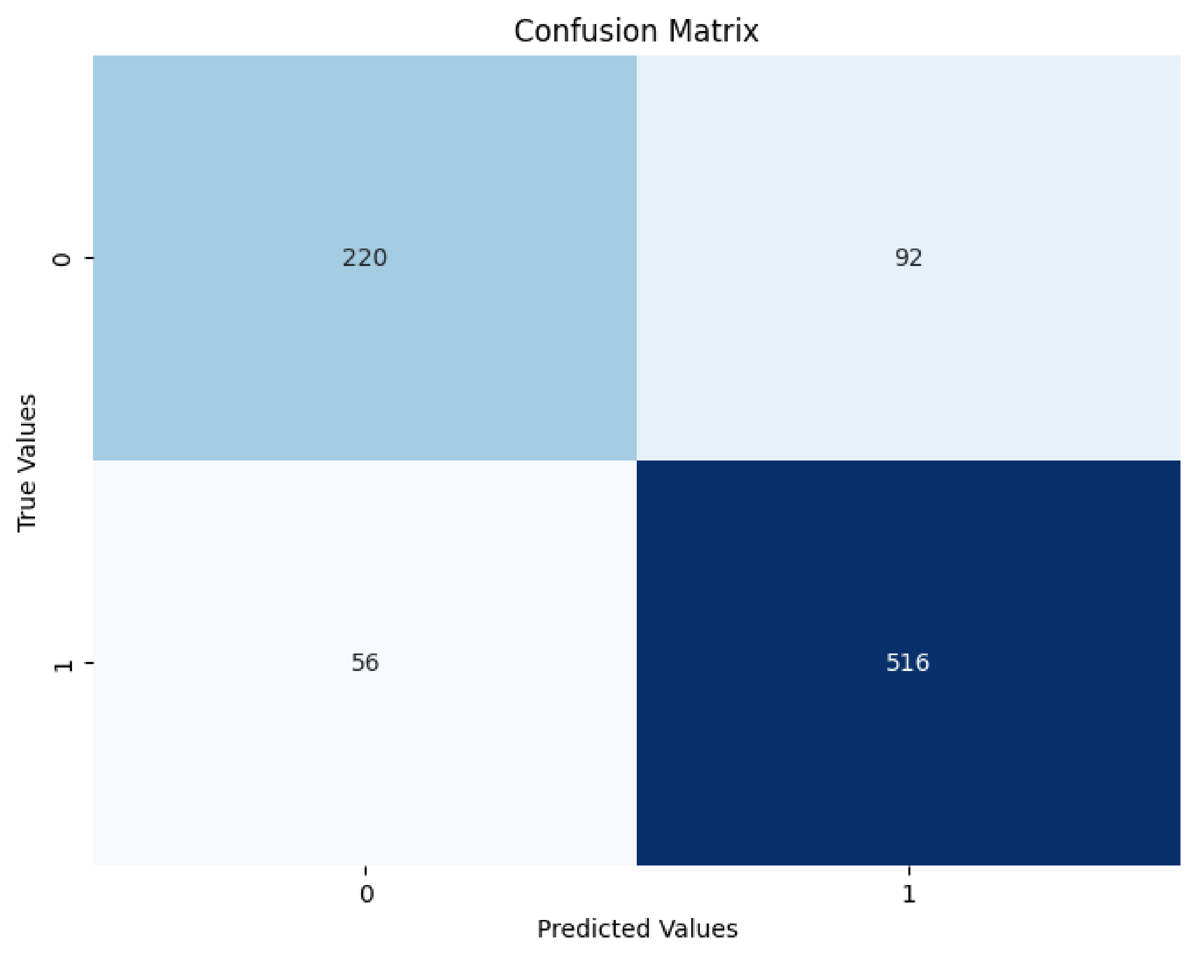


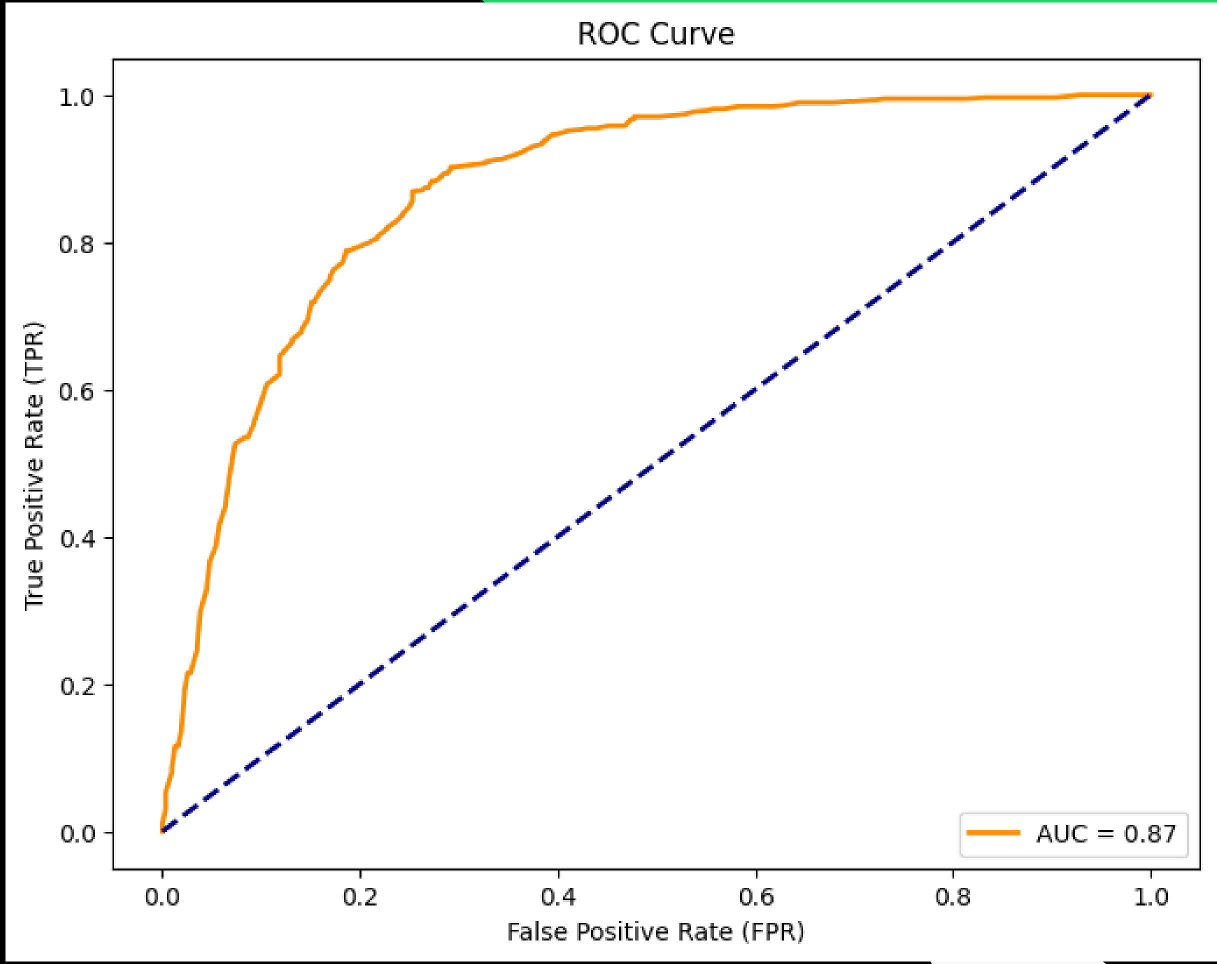
5. Random Forest

```
from sklearn.ensemble import RandomForestClassifier

features = ['danceability', 'energy', 'loudness', 'mode', 'speechiness', 'acousticness',
            'instrumentalness', 'liveness', 'valence', 'tempo', 'duration_s', 'time_signature',
            'chorus_hit', 'key_0', 'key_1', 'key_2', 'key_3', 'key_4', 'key_5', 'key_6',
            'key_7', 'key_8', 'key_9', 'key_10', 'key_11', 'genre_edm', 'genre_latin', 'genre_pop',
            'genre_r&b', 'genre_rap', 'genre_rock']

X = df2[features]
y = df2['popularity']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.1, random_state=0)

rf_model = RandomForestClassifier(n_estimators=100, random_state=0)
rf_model.fit(X_train, y_train)

y_pred = rf_model.predict(X_test)
y_pred_prob = rf_model.predict_proba(X_test)[:, 1]

accuracy = accuracy_score(y_test, y_pred)
conf_matrix = confusion_matrix(y_test, y_pred)
class_report = classification_report(y_test, y_pred)
f1 = f1_score(y_test, y_pred)

print(f"Accuracy: {accuracy}")
print(f"Confusion Matrix:₩n{conf_matrix}")
print(f"Classification Report:₩n{class_report}")
print(f"F1 Score: {f1}")
```

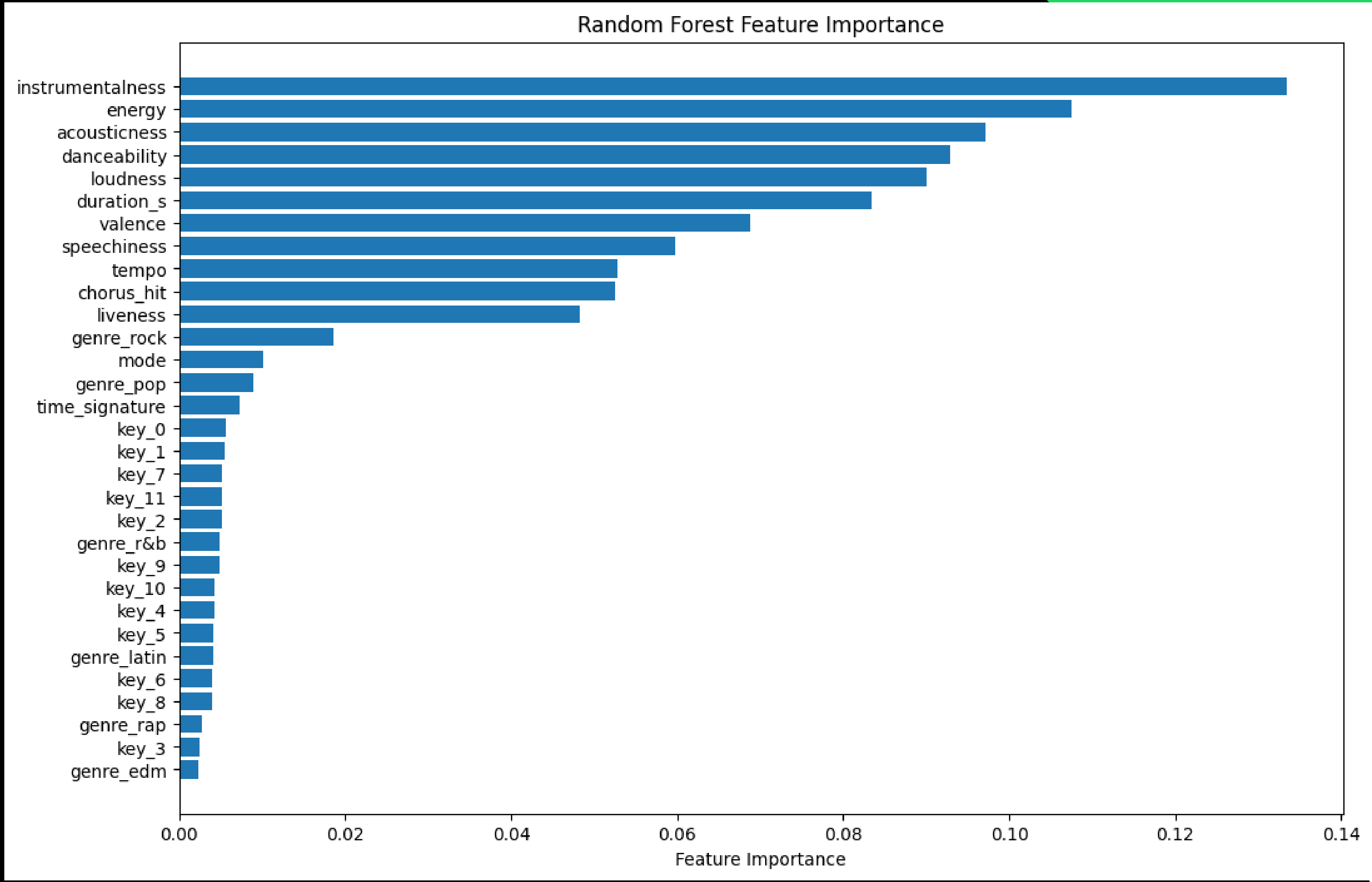
```
Accuracy: 0.832579185520362
Confusion Matrix:
[[220  92]
 [ 56 516]]
Classification Report:
              precision    recall  f1-score   support

           0       0.80      0.71      0.75       312
           1       0.85      0.90      0.87       572

    accuracy                           0.83       884
   macro avg       0.82      0.80      0.81       884
weighted avg       0.83      0.83      0.83       884

F1 Score: 0.8745762711864407
```

5. Random Forest





5. Random Forest



6. 결론 – 모델 정확도 비교

- 해당 데이터 분석결과 정확도

Random Forest (83%) > Logistic Regression (78%) > Decision Tree (70%)

Accuracy: 0.832579185520362
Confusion Matrix:
[[220 92]
 [56 516]]
Classification Report:

	precision	recall	f1-score	support
0	0.80	0.71	0.75	312
1	0.85	0.90	0.87	572
accuracy			0.83	884
macro avg	0.82	0.80	0.81	884
weighted avg	0.83	0.83	0.83	884

F1 Score: 0.8745762711864407

Accuracy: 0.7771493212669683
Confusion Matrix:
[[142 170]
 [27 545]]
Classification Report:

	precision	recall	f1-score	support
0	0.84	0.46	0.59	312
1	0.76	0.95	0.85	572
accuracy			0.78	884
macro avg	0.80	0.70	0.72	884
weighted avg	0.79	0.78	0.76	884

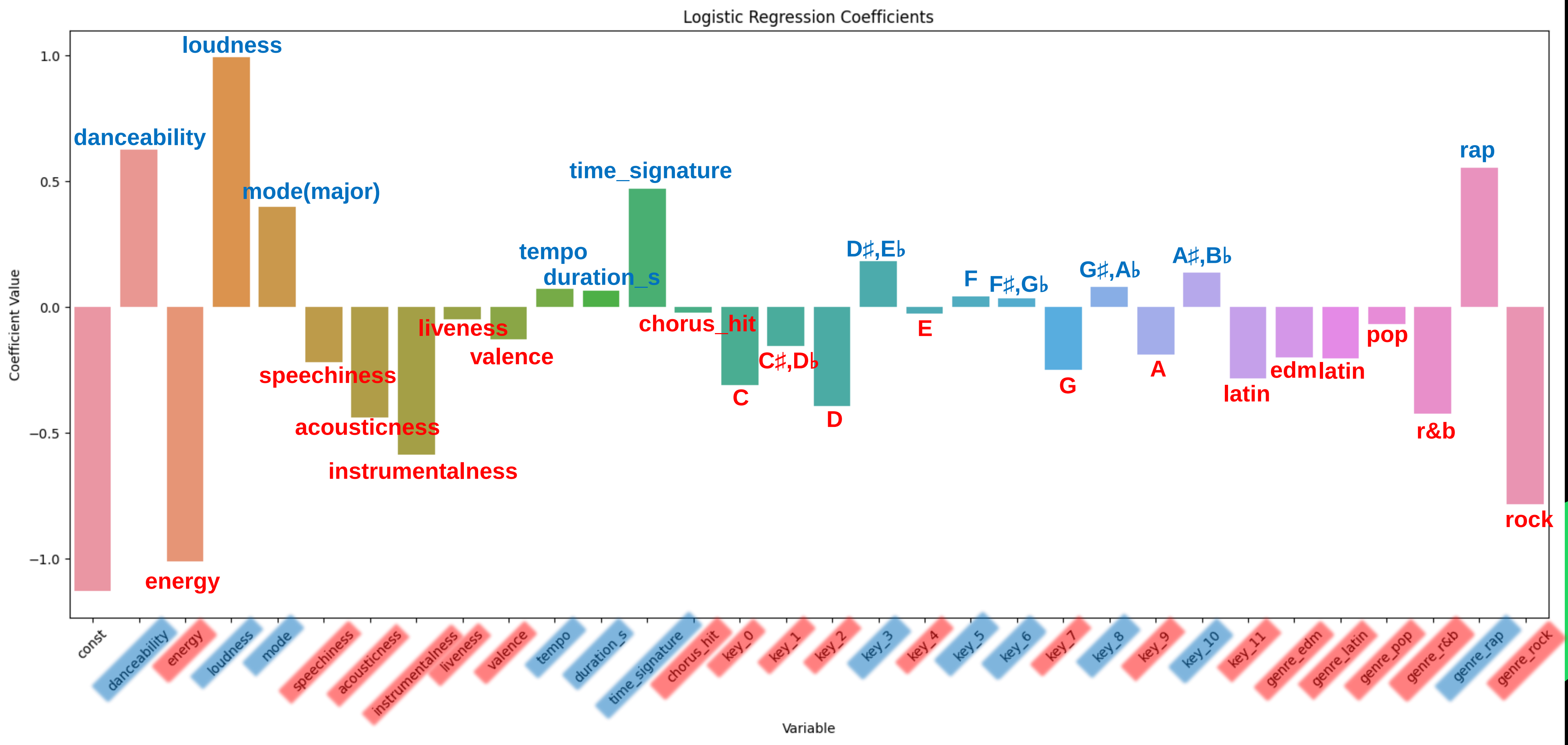
F1 Score: 0.8469308469308469

Accuracy: 0.7002262443438914
Confusion Matrix:
[[187 125]
 [140 432]]
Classification Report:

	precision	recall	f1-score	support
0	0.57	0.60	0.59	312
1	0.78	0.76	0.77	572
accuracy			0.70	884
macro avg	0.67	0.68	0.68	884
weighted avg	0.70	0.70	0.70	884

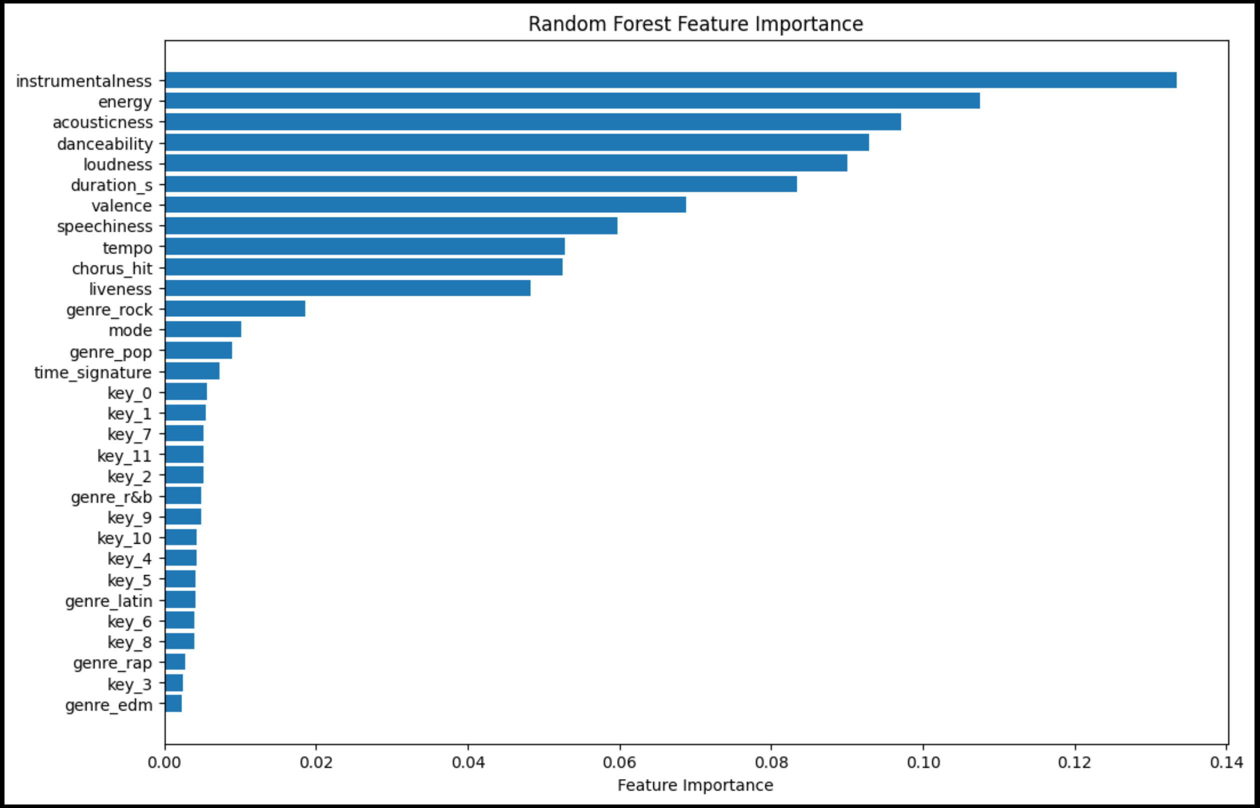
F1 Score: 0.7652790079716563

6. 결론 – Logistic Regression 상관계수



6. 결론 – Random Forest 중요변수

- Random Forest 로 알아본 히트곡 예측에 중요한 변수
  : instrumentalness → energy → accousticness → danceability → loudness → duration_s → valence → speechiness → tempo → chorus_hit → liveness



6. 결론 – 한계

- 원 – 핫 인코딩 을 진행한 key 변수와 genre 변수의 특성이 잘 반영되지 못함.
- 원본데이터에서 genre 컬럼의 pop 변수에 너무 많은 장르가 포함되어 있어 정확한

  분석이 어려움.
- Confusion matrix 관찰 결과 모든 모델에서 False Positive 값이 높음.

Thanks for tuning in.