Московский государственный технический университет им. Н. Э. Баумана

Курс «Технологии машинного обучения»
Отчёт по лабораторной работе №1
«Разведочный анализ данных. Исследование и визуализация данных»

Выполнил:
Зелинский Д.М.,
группа ИУ5-61Б

Проверил:
Нардид А.Н.,
каф. ИУ5

Дата:

Подпись:

Дата:

Подпись:

2023
Москва

Цель работы

Изучение различных методов визуализация данных.

Описание датасета

В качестве набора данных мы используем "Популярные песни ТикТока 2022г." https://www.kaggle.com/datasets/sveta151/tiktok-popular-songs-2022

Этот набор данных предоставляет всю важную информацию, которая может потребоваться для дальнейшего анализа, начиная с базовых знаний, таких как название трека и имя исполнителя, заканчивая самыми продвинутыми сведениями, такими как темп, time_signature и т.д. и т.п.

Проанализировав статистику по тикток-песням, можно получить данные об особенностях медиакультуры молодёжи.

Выполнение работы

Загрузка библиотек и датасета

In [1]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
import math as mth
import matplotlib.patches as patches
from scipy import stats as st
plt.rcParams.update({'figure.max_open_warning': 0})
import plotly.graph_objects as go
import plotly.express as ex
```

In [2]:
```
df = pd.read_csv('TikTok_songs_2022.csv')
```

Получение общей информации о датасете

In [3]:
```
df.head()
```

Out[3]:

	track_name	artist_name	artist_pop	album	track_pop	danceability	energy	loudness	mode	key	speechiness	acousticness	instrumentalness	livenes
0	Running Up That Hill (A Deal With God)	Kate Bush	81	Hounds Of Love	95	0.629	0.547	-13.123	0	10	0.0550	0.7200	0.003140	0.060
1	As It Was	Harry Styles	91	As It Was	96	0.520	0.731	-5.338	0	6	0.0557	0.3420	0.001010	0.311
2	Sunroof	Nicky Youre	73	Sunroof	44	0.768	0.716	-5.110	1	10	0.0404	0.3500	0.000000	0.150
3	Heat Waves	Glass Animals	80	Dreamland (+ Bonus Levels)	89	0.761	0.525	-6.900	1	11	0.0944	0.4400	0.000007	0.092
4	About Damn Time	Lizzo	81	About Damn Time	92	0.836	0.743	-6.305	0	10	0.0656	0.0995	0.000000	0.335

In [9]:
```
# Размер датасета - 263 строки, 18колонок
df.shape
```

Out[9]:
```
(263, 18)
```

In [10]:
```
total_count = df.shape[0]
print('Всего строк: {}'.format(total_count))
```

```
Всего строк: 263
```

In [11]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 263 entries, 0 to 262
Data columns (total 18 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   track_name        263 non-null    object 
 1   artist_name       263 non-null    object 
 2   artist_pop        263 non-null    int64  
 3   album             263 non-null    object 
 4   track_pop         263 non-null    int64  
 5   danceability      263 non-null    float64
 6   energy            263 non-null    float64
 7   loudness          263 non-null    float64
 8   mode              263 non-null    int64  
 9   key               263 non-null    int64  
 10  speechiness       263 non-null    float64
 11  acousticness      263 non-null    float64
 12  instrumentalness  263 non-null    float64
 13  liveness          263 non-null    float64
 14  valence           263 non-null    float64
 15  tempo             263 non-null    float64
 16  time_signature    263 non-null    int64  
 17  duration_ms       263 non-null    int64  
dtypes: float64(9), int64(6), object(3)
memory usage: 37.1+ KB
```

Проверка на пропуски и дубликаты

In [7]:
```
df.isna().sum().sum()
```

Out[7]:
```
0
```

In [8]:
```
df.duplicated().sum()
```

Out[8]:
```
0
```

Пропусков и дубликатов в датафрейме не обнаружено

Получение общей информации о числовых данных датасета

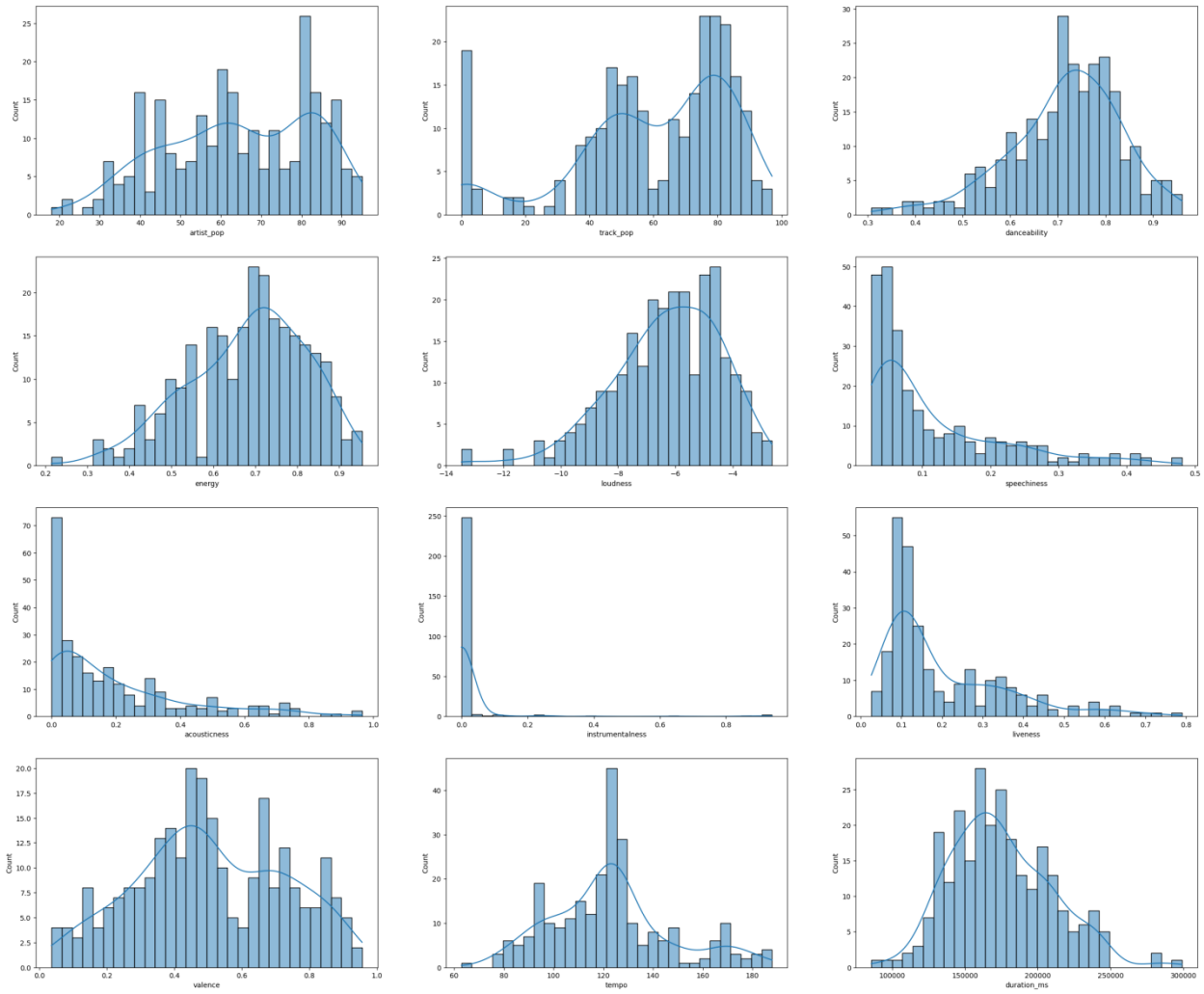
In [9]:
```
df.describe().T
```

Out[9]:

	count	mean	std	min	25%	50%	75%	max
artist_pop	263.0	64.186312	18.146338	18.000000	50.00000	64.000000	81.000000	95.000
track_pop	263.0	59.844106	24.976524	0.000000	47.00000	66.000000	79.500000	97.000
danceability	263.0	0.715338	0.117722	0.308000	0.64600	0.726000	0.796500	0.961
energy	263.0	0.682631	0.141756	0.214000	0.59300	0.701000	0.790500	0.955
loudness	263.0	-6.300513	1.905114	-13.468000	-7.42350	-6.067000	-4.887000	-2.634
mode	263.0	0.536122	0.499644	0.000000	0.00000	1.000000	1.000000	1.000
key	263.0	5.809886	3.657312	0.000000	2.00000	6.000000	9.000000	11.000
speechiness	263.0	0.117268	0.101417	0.025200	0.04475	0.070700	0.156000	0.481
acousticness	263.0	0.193188	0.214588	0.000038	0.02810	0.120000	0.298500	0.965
instrumentalness	263.0	0.020244	0.109370	0.000000	0.00000	0.000003	0.000334	0.939
liveness	263.0	0.202562	0.151808	0.026500	0.09435	0.132000	0.277500	0.790
valence	263.0	0.506149	0.220757	0.036300	0.34900	0.485000	0.680000	0.956
tempo	263.0	122.883696	24.396686	62.948000	106.85050	123.056000	130.004500	187.906
time_signature	263.0	3.969582	0.288168	1.000000	4.00000	4.000000	4.000000	5.000
duration_ms	263.0	174656.212928	34785.125564	85742.000000	148646.00000	171028.000000	199047.000000	298933.000

Так как количество числовых данных велико, визуализируем их, чтобы понять зависимости, аномалии и тд

In [10]:
```
def hist(df, x, ax, bins=30):
    sns.histplot(data=df, x=x, bins=bins, ax=ax,kde=True)

columns = ['artist_pop', 'track_pop', 'danceability', 'energy', 'loudness', 'speechiness',
       'acousticness', 'instrumentalness', 'liveness', 'valence', 'tempo', 'duration_ms']

fig, axs = plt.subplots(4, 3, figsize=(30, 25))

for i, col in enumerate(columns):

    row_index = i // 3
    col_index = i % 3

    hist(df, col, axs[row_index][col_index])

plt.show()
```



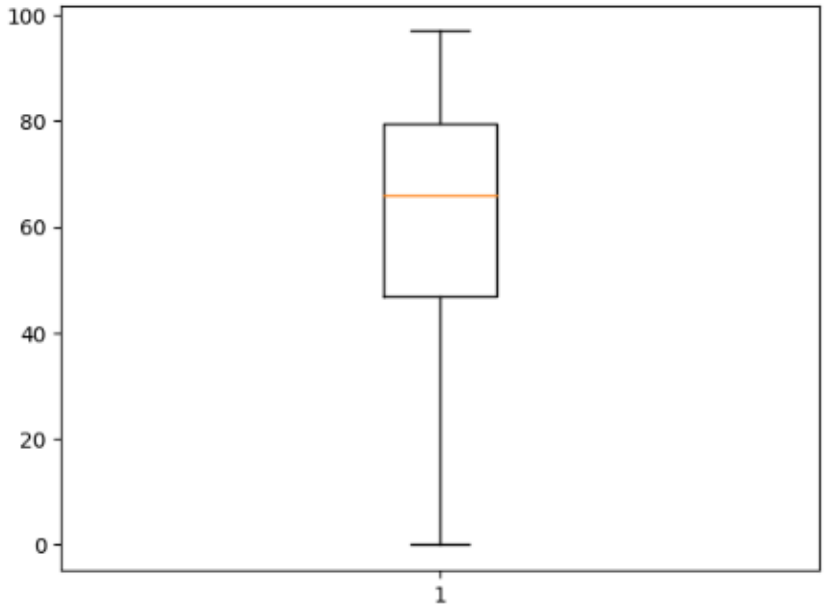
В переменной 'track_pop', есть скачок на 0. Вероятно, это означает, что для таких треков у нас отсутствуют данные, которые нам придется либо исключить, либо заменить средним значением по столбцу.

"Громкость" - единственная функция с отрицательными значениями

"Инструментальность" имеет почти все значения в 0: из-за этого данный столбец не несет особой смысловой нагрузки, исключим его в дальшейем.

Первая особенность: оценка трека имеет 0 значение

```
In [11]: plt.boxplot(df['track_pop']);
```



Видим, что Q1 и Q3 находятся в разрезе между 45 и 75 баллами, поэтому будем считать значения равные 0 - выбросами, посмотрим, какой их процент в датафрейме.

```
In [12]: df[['track_pop','track_name']].query('track_pop == 0')
```

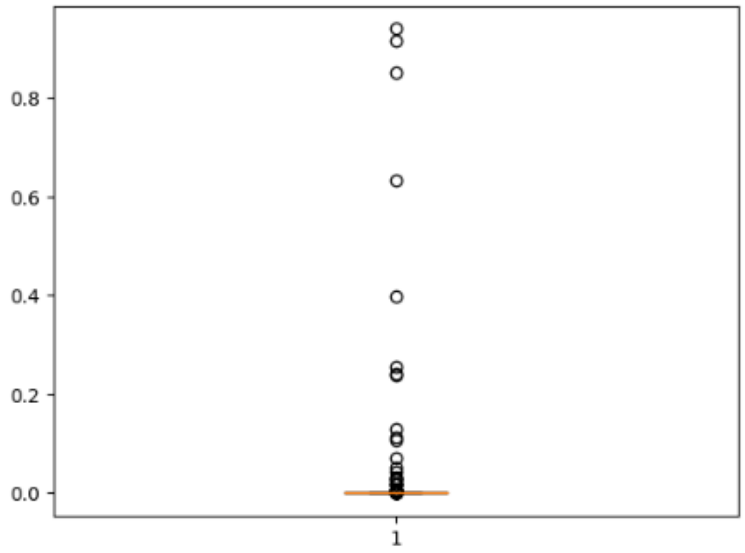
Out[12]:

	track_pop	track_name
16	0	AMG
32	0	Fever - Remix
42	0	Without You
68	0	Lost
75	0	Looking for Love
78	0	She Wolf (Falling to Pieces)
102	0	Buss It
130	0	Slow Down
131	0	What If
184	0	Backyard Boy
200	0	Right Here Waiting
243	0	Skechers
245	0	Shooting Stars

13 треков имеет рейтинг равный 0 - это не большой процент. Эти данные можно удалить.

```
In [13]: df = df.query('track_pop != 0')
```

```
In [14]: plt.boxplot(df['instrumentalness']);
```




```
In [15]: df[['instrumentalness','track_name']].query('instrumentalness >= 0.5')
```

Out[15]:

	instrumentalness	track_name
112	0.633	Freaks
127	0.939	Write This Down (Instrumental)
132	0.852	Running Away
216	0.915	Astronomia

Да, действительно, видим, что 90% нашего датафрейма - вокальные треки, лишь 4 их них имеют порог выше 0.5, что говорит об инструментальности данных треков. Поэтому удалим данный столбец, тк выдвинуть гипотезы и анализировать данные с помощью него не получится.

```
In [16]: df.drop(columns = ['instrumentalness'],axis = 1, inplace=True)
```

Визуализация

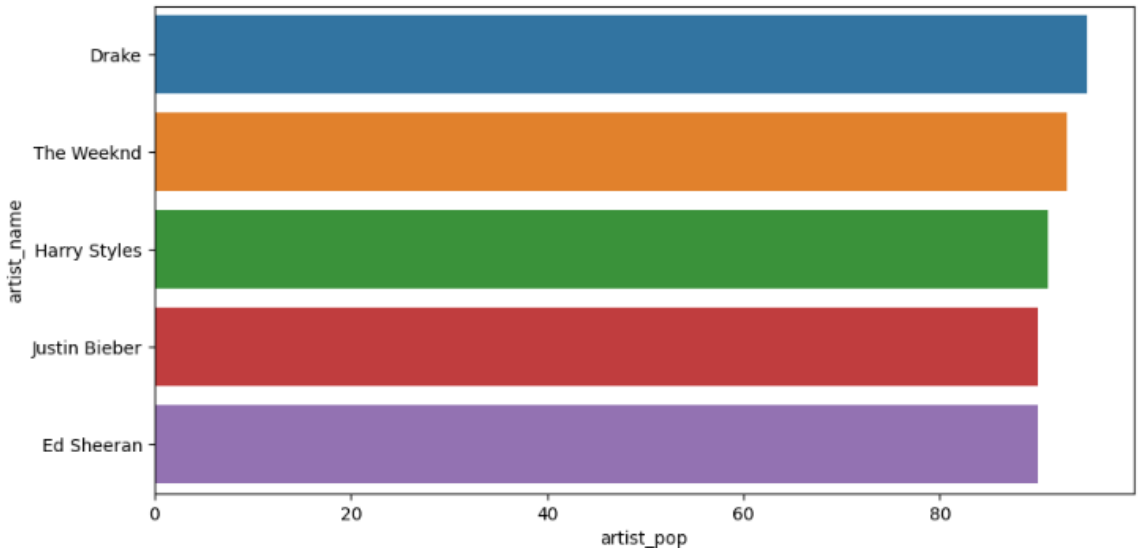
Выявим топ-5 артистов

```
In [21]: pop = df.drop_duplicates('artist_name').sort_values('artist_pop', ascending=False).head()
```

```
In [22]: pop.artist_name.unique()
```

```
Out[22]: array(['Drake', 'The Weeknd', 'Harry Styles', 'Justin Bieber',
                'Ed Sheeran'], dtype=object)
```

```
In [23]: plt.figure(figsize = (10,5))
         sns.barplot(y = 'artist_name',x = 'artist_pop',data = pop);
```



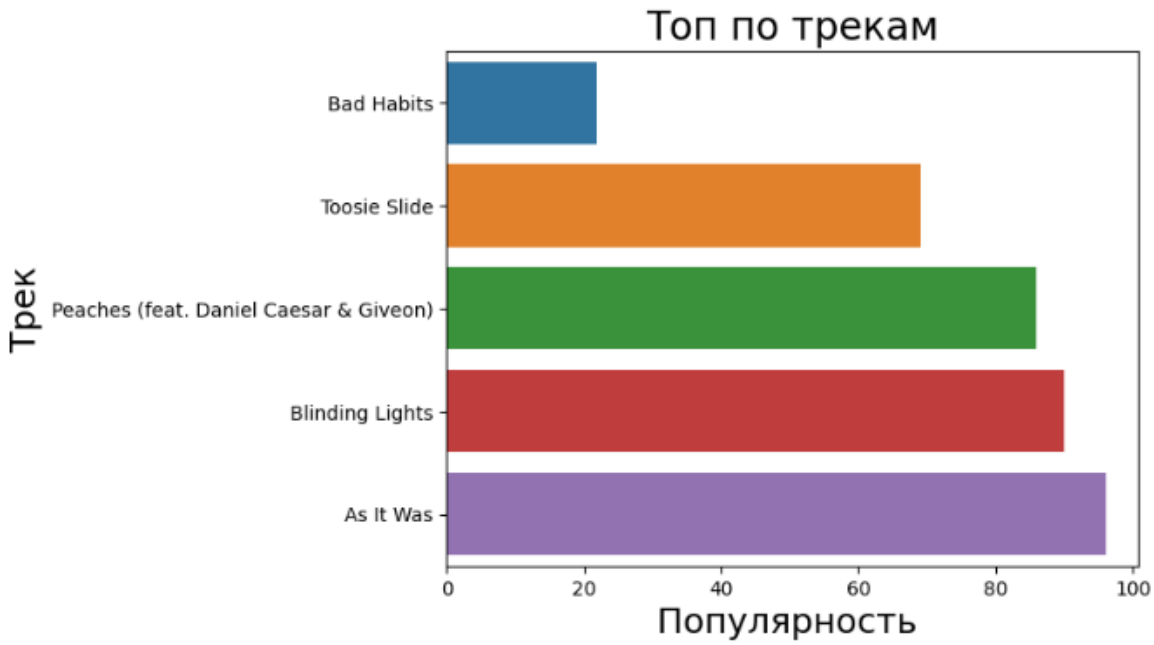
Примерно у всех артистов одинаковый уровень оценки > 80.

In [27]:
```
pop.album.value_counts()
```

Out[27]:
```
Dark Lane Demo Tapes    1
After Hours             1
As It Was               1
Justice                 1
Bad Habits              1
Name: album, dtype: int64
```

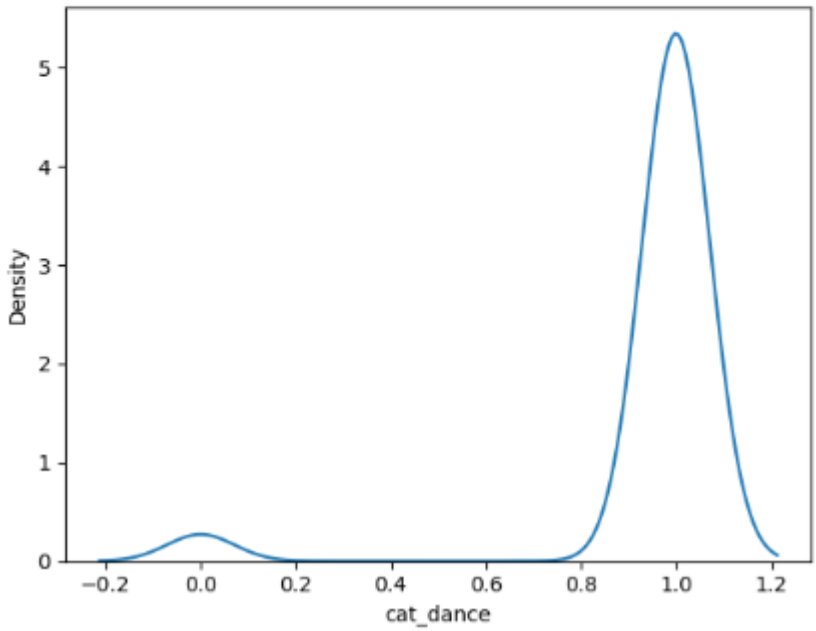
Посмотрим на самые популярные треки топ5 артистов

In [29]:
```
df_10 = pop.sort_values(by = 'track_pop' ).head(10)

%matplotlib inline
plt.title('Топ по трекам',fontsize=20)
sns.barplot(x=df_10.track_pop,
            y=df_10.track_name)
plt.xlabel('Популярность ',fontsize=18)
plt.ylabel('Трек',fontsize=18)
plt.show()
```



Проведём статистический анализ наших данных. Сначала рассмотрим статистику зажигательности треков.

```
In [35]: def cat_dance (df):
             if df['danceability'] < 0.5:
                 return 0
             else:
                 return 1
         df['cat_dance'] = df.apply(cat_dance, axis = 1)

In [36]: sns.kdeplot(df.cat_dance);
```



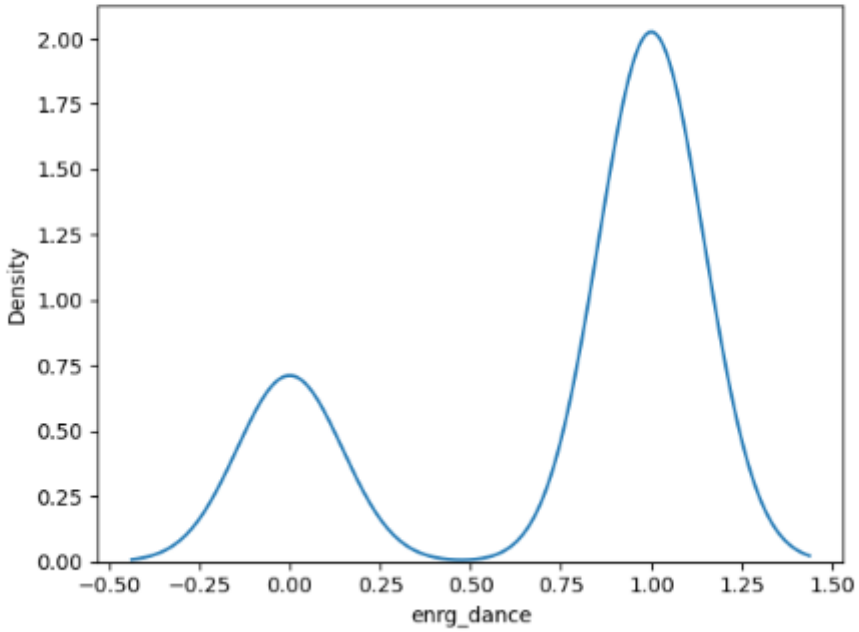
Статистика зажигательности треков

Большинство треков – танцевальные.

Теперь рассмотрим статистику энергичности треков

```
In [37]: def enrg_dance (df):
             if df['energy'] < 0.6:
                 return 0
             else:
                 return 1
         df['enrg_dance'] = df.apply(enrg_dance, axis = 1)
```

```
In [38]: sns.kdeplot(df.enrg_dance);
```



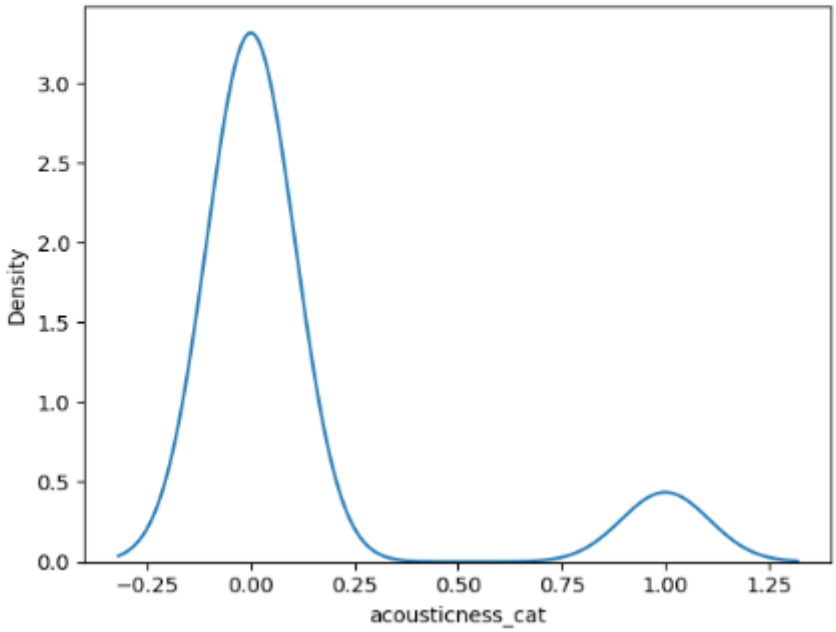
Статистика энергичности треков

Большинство треков – энергичные.

Теперь рассмотрим статистику акустики треков

In [39]:
```
def acousticness_cat (df):
    if df['acousticness'] < 0.5:
        return 0
    else:
        return 1
df['acousticness_cat'] = df.apply(acousticness_cat, axis = 1)
```

In [40]:
```
sns.kdeplot(df.acousticness_cat);
```



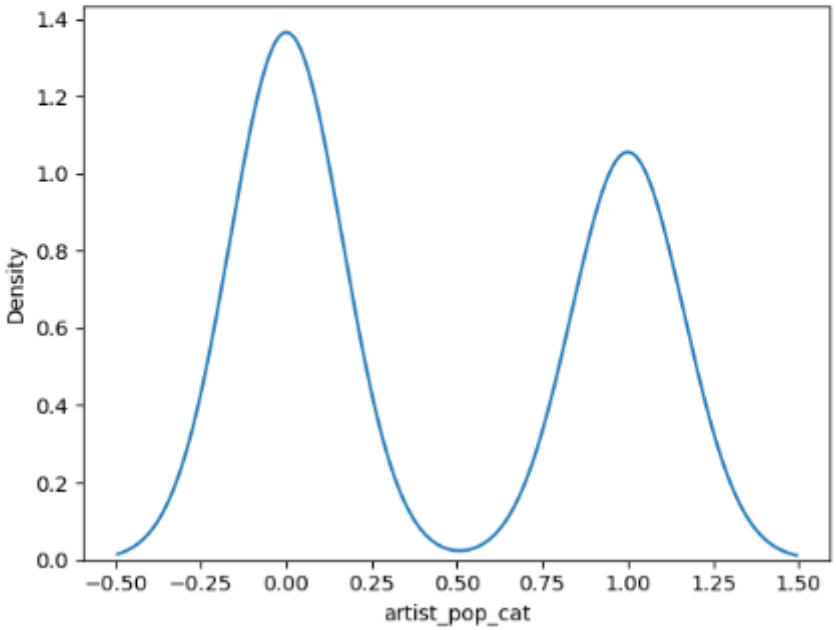
Статистика акустики треков

Большинство треков – электронные.

Теперь рассмотрим статистику популярности исполнителей

```
In [41]: def artist_pop_cat (df):
             if df['artist_pop'] < 70:
                 return 0
             else:
                 return 1
         df['artist_pop_cat'] = df.apply(artist_pop_cat, axis = 1)
```

```
In [42]: sns.kdeplot(df.artist_pop_cat);
```



Статистика популярности артистов

Количество популярных и менее популярных исполнителей примерно равно.

Проверим количество песен у исполнителей и выведем тех, у кого большее количество известных треков

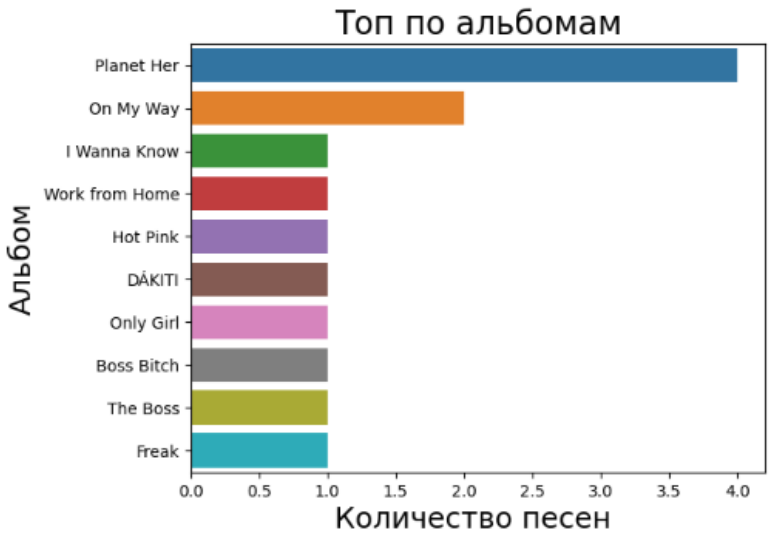
```
In [34]: tik_tok = df['artist_name'].value_counts(ascending=False)

In [35]: tik_tok = pd.DataFrame(tik_tok)

In [36]: tik_tok.artist_name.head()

Out[36]: Doja Cat          8
         Coopex            5
         Dame Dame         5
         Alex Alexander    4
         Farux             4
         Name: artist_name, dtype: int64
```

Здесь мы видим, что артисты с самым большим количеством популярных песен не входят в топ лучших. Самое большое число

популярных песен наблюдается у исполнителей Doja Cat, Coopex и Dame Dame.

Выявим самые популярные альбомы

```
In [38]: %matplotlib inline
plt.title('Топ по альбомам',fontsize=20)
sns.barplot(x=top.album.value_counts().head(10),
            y=top.album.value_counts().head(10).index)
plt.xlabel('Количество песен ',fontsize=18)
plt.ylabel('Альбом',fontsize=18)
plt.show()
```



Выявим самые популярные треки

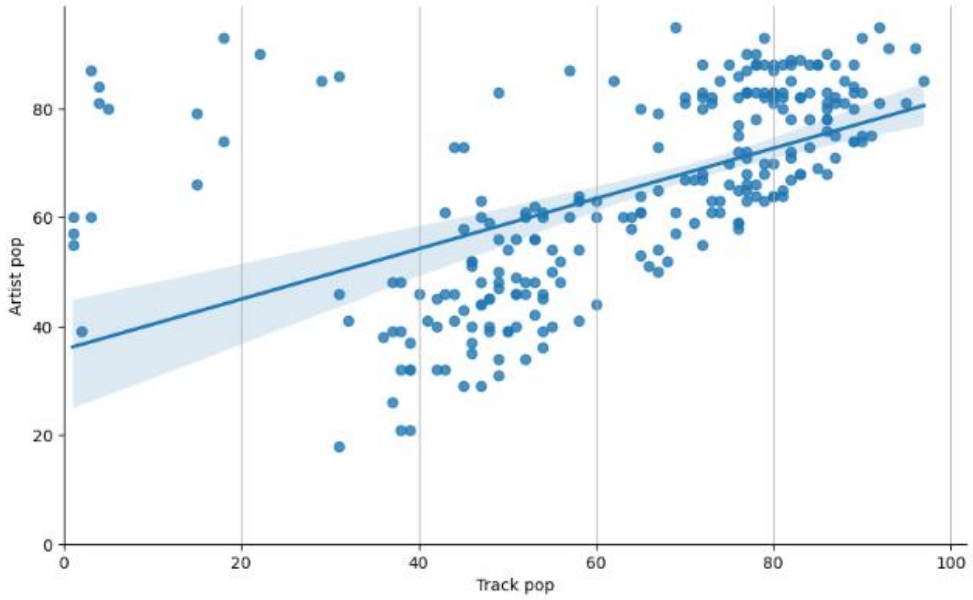
```
In [40]: top = top.sort_values(by = 'track_pop' )
```

```
In [41]: %matplotlib inline
plt.title('Топ по трекам',fontsize=20)
sns.barplot(x=top.track_pop.sort_values(),
            y=top.track_name)
plt.xlabel('Популярность ',fontsize=18)
plt.ylabel('Трек',fontsize=18)
plt.show()
```

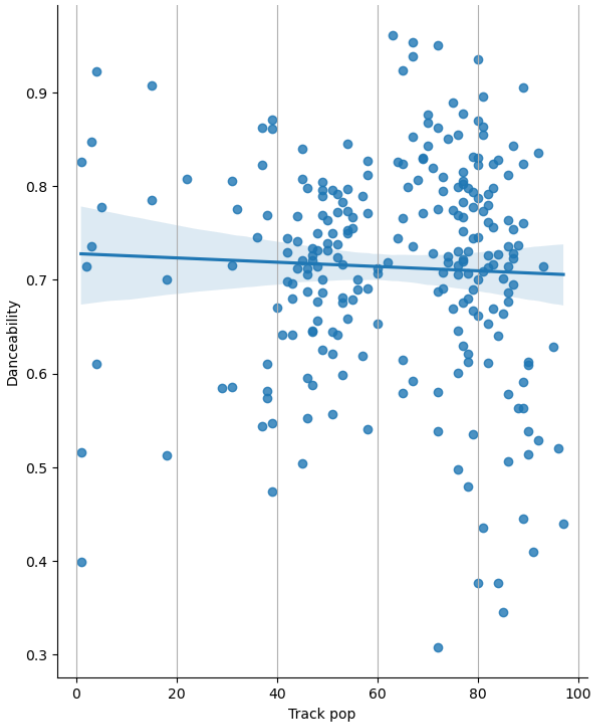
Анализируя полученные графики, можно увидеть, что самым популярным альбомом является альбом «Planet Her» за авторством Doja Cat, и самая известная песня под названием «Woman» входит в этот альбом.

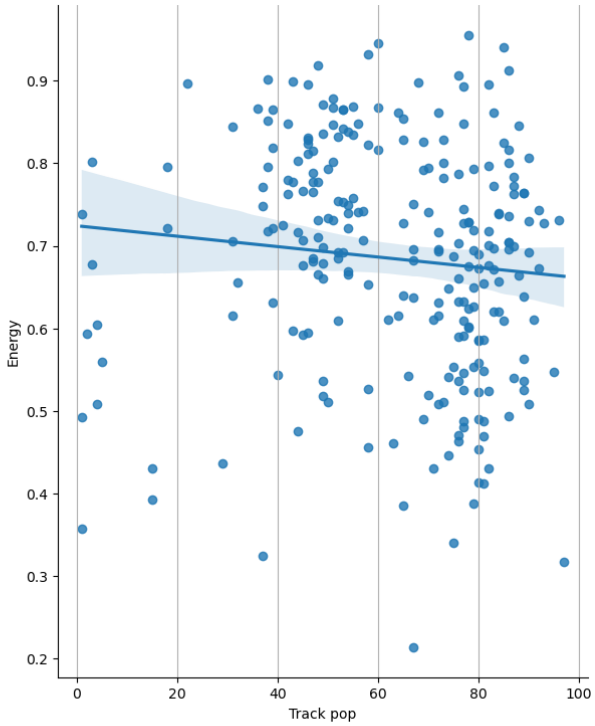
Определим зависимость популярности трека от рейтинга артиста.

```
In [49]: fig, ax = plt.subplots(figsize=(10, 6))

scatter(df, 'track_pop', 'artist_pop', ax)

ax.set_ylim(bottom=0)
ax.set_xlim(left=0)

plt.show()
```



Теперь рассмотрим зависимости популярности трека от зажигательности и энергичности.





Исходя из проведённого анализа, видно, что популярность трека не зависит от его зажигательности и энергичности, но зависит от известности артиста.

Построим тепловую карту, чтобы посмотреть корреляцию между переменными