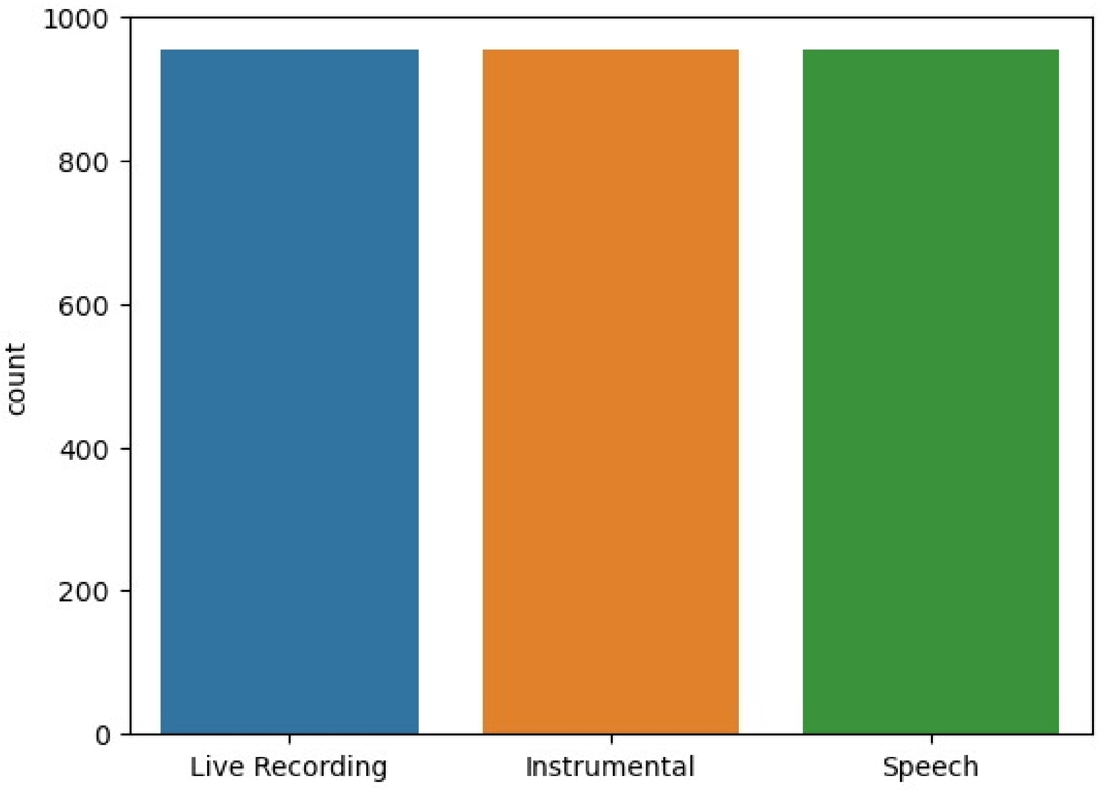
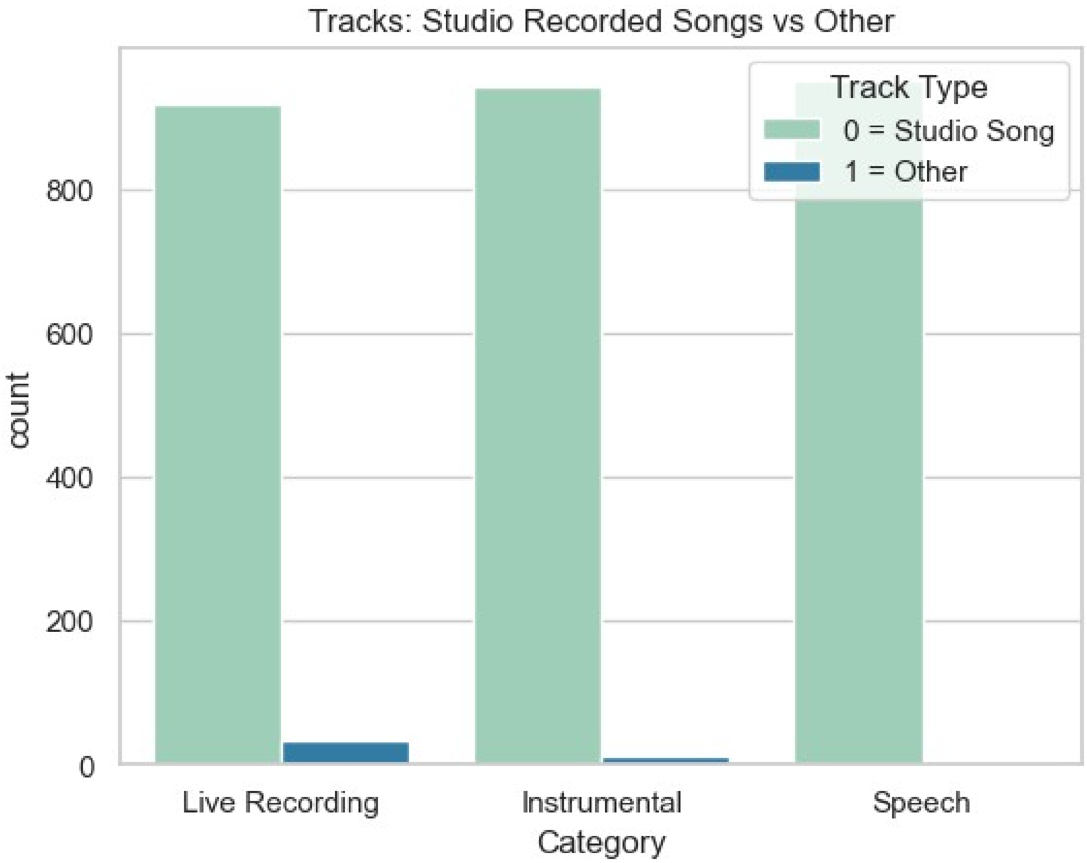
```
threshold = 50

# Create a new column 'live' based on the condition
Data['Live Recording'] = np.where(Data['liveness_%'] >= threshold, 1,
0)
Data['Instrumental'] = np.where(Data['instrumentalness_%'] >=
threshold, 1, 0)
Data['Speech'] = np.where(Data['speechiness_%'] >= threshold, 1, 0)

No_Music = Data[['Live Recording', 'Instrumental','Speech']]

NM_Cat = ['Live Recording', 'Instrumental','Speech']

sns.countplot(data=No_Music)
```

```
<Axes: ylabel='count'>
```

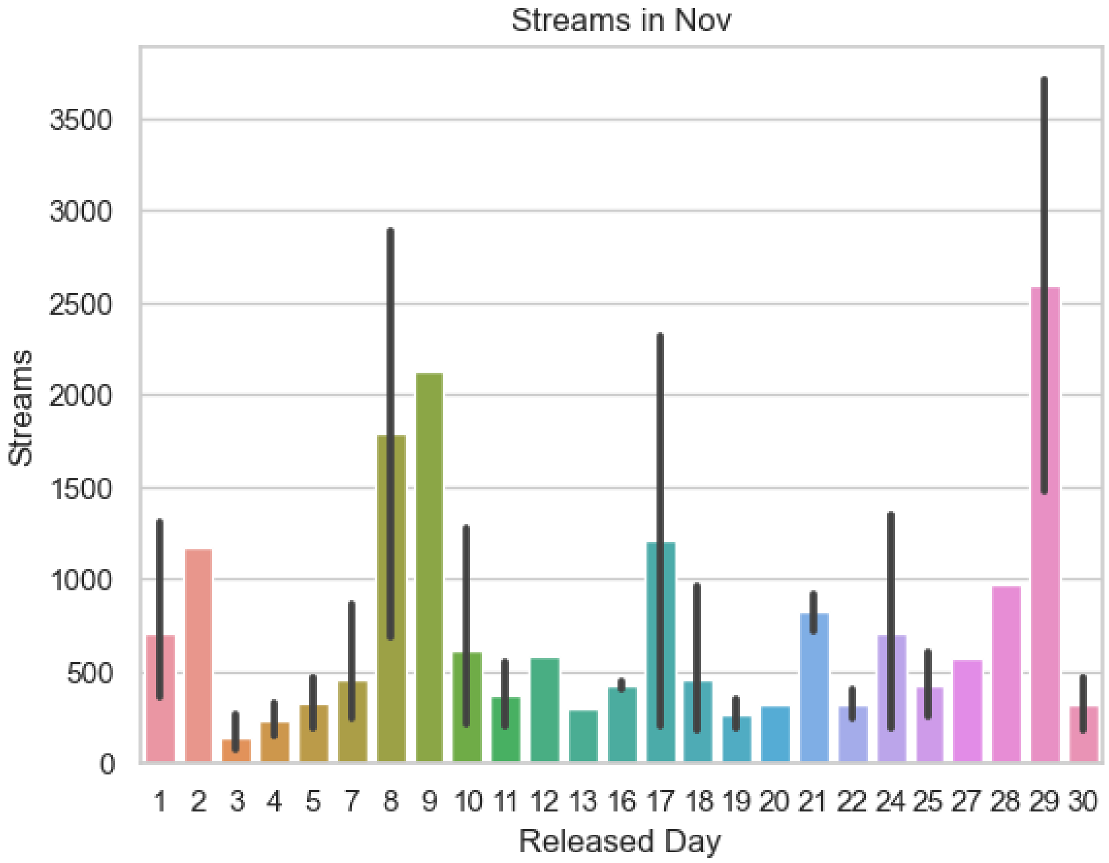


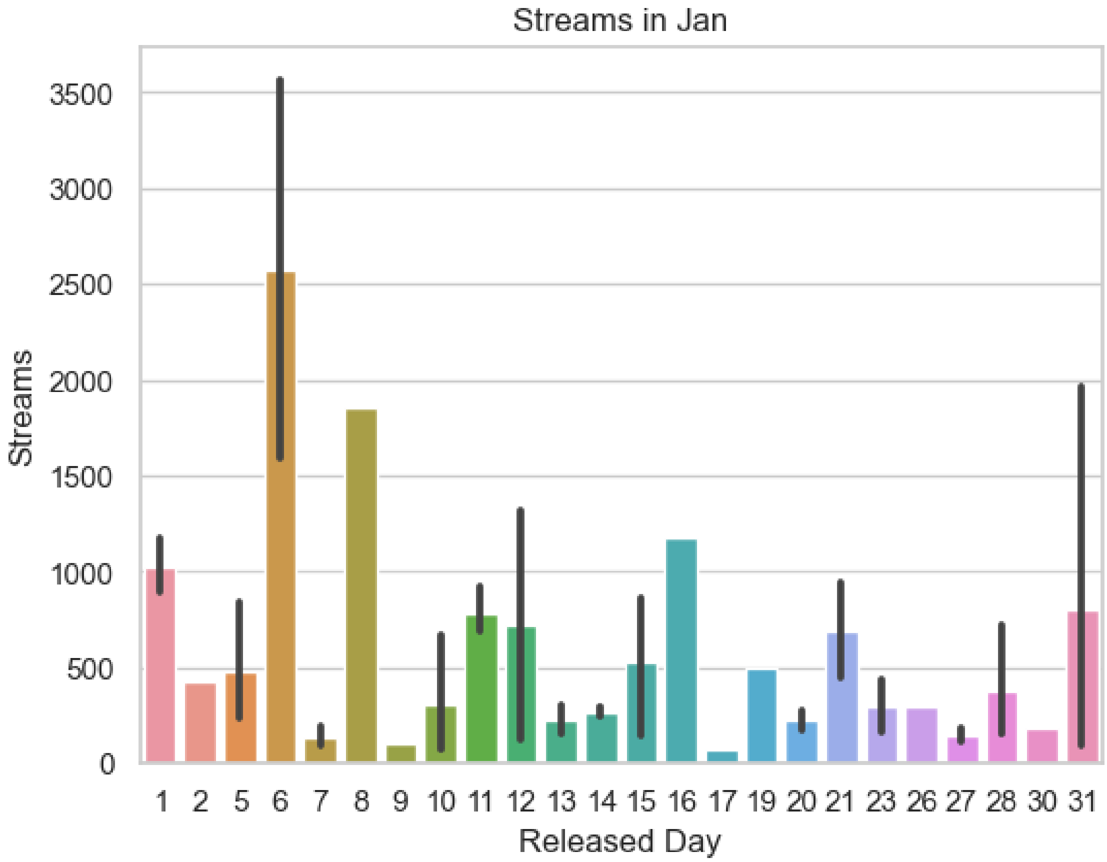
```
# Assuming No_Music is your DataFrame and NM_Cat is the list of column
names
sns.set(style="whitegrid")

# Melt the DataFrame to make it suitable for countplot
melted_data = No_Music.melt(value_vars=NM_Cat, var_name='Category',
value_name='Count')
```

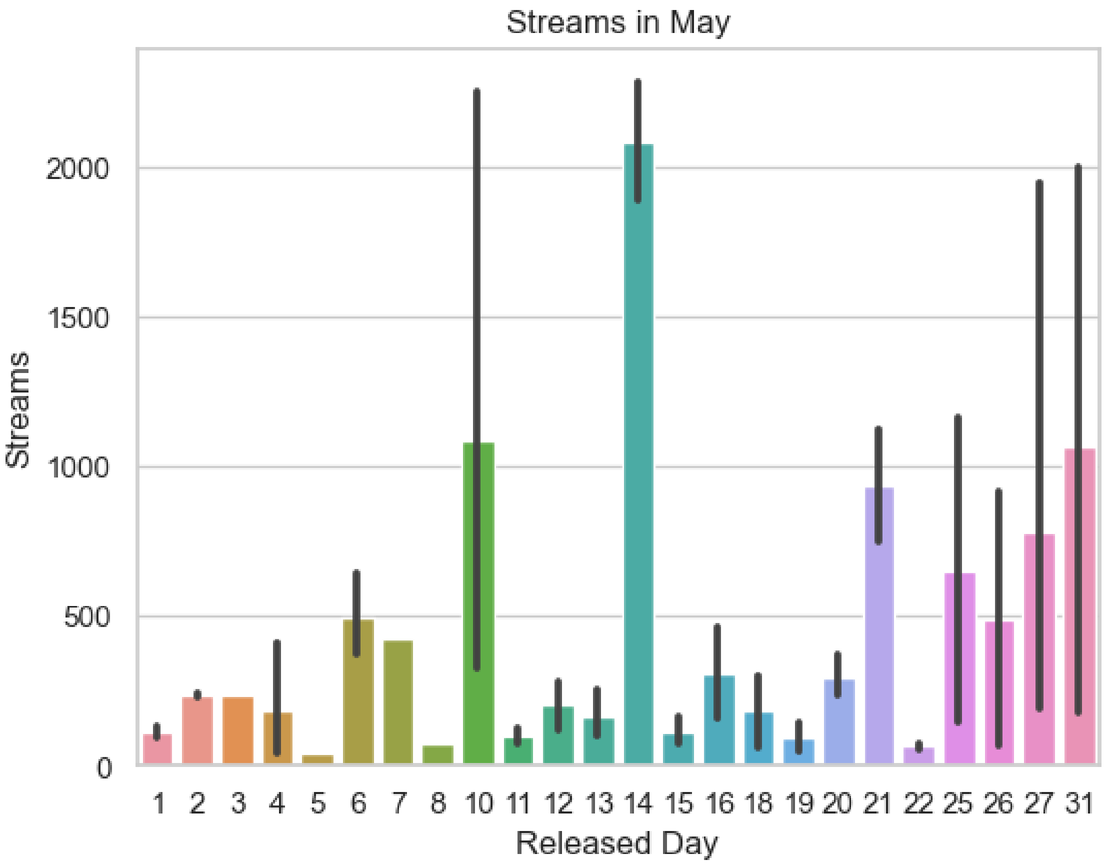
```
# Create the count plot
sns.countplot(x='Category', hue='Count', data=melted_data,
palette='YlGnBu')
plt.title('Tracks: Studio Recorded Songs vs Other')
plt.legend(title='Track Type', labels=['0 = Studio Song', '1 =
Other'])

# Show the plot
plt.show()
```

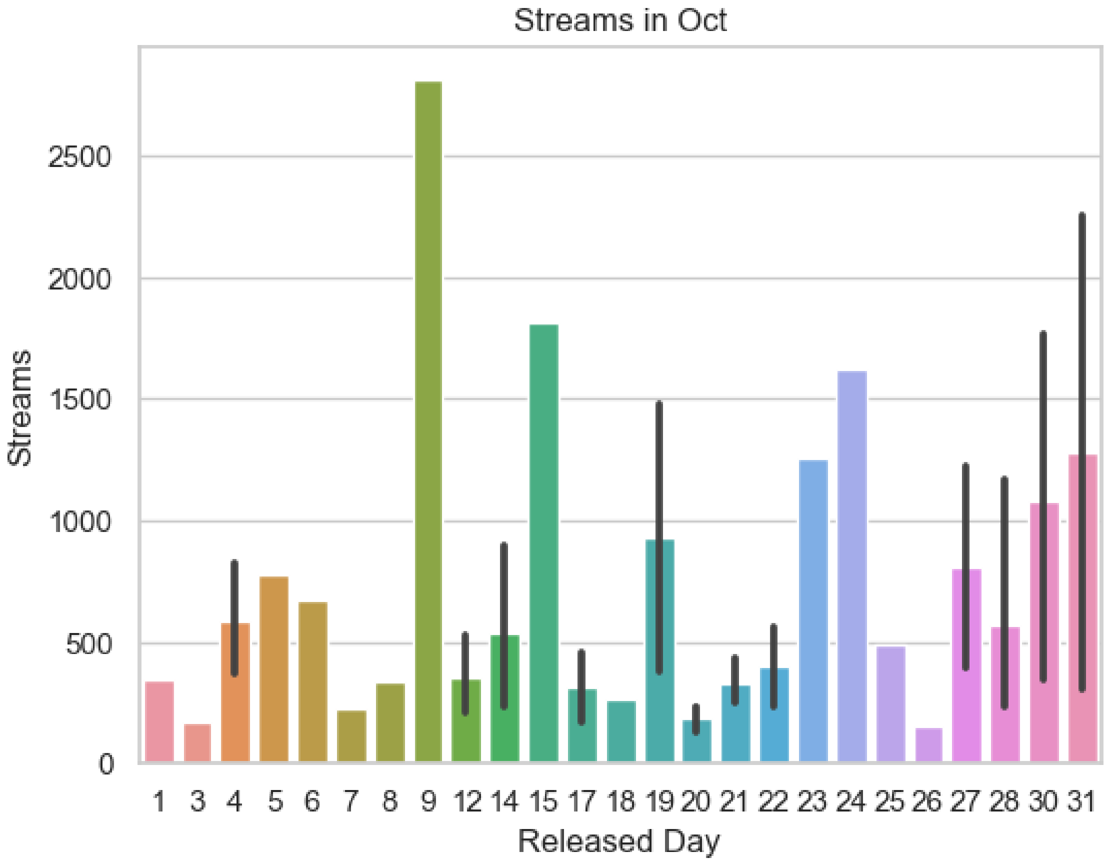


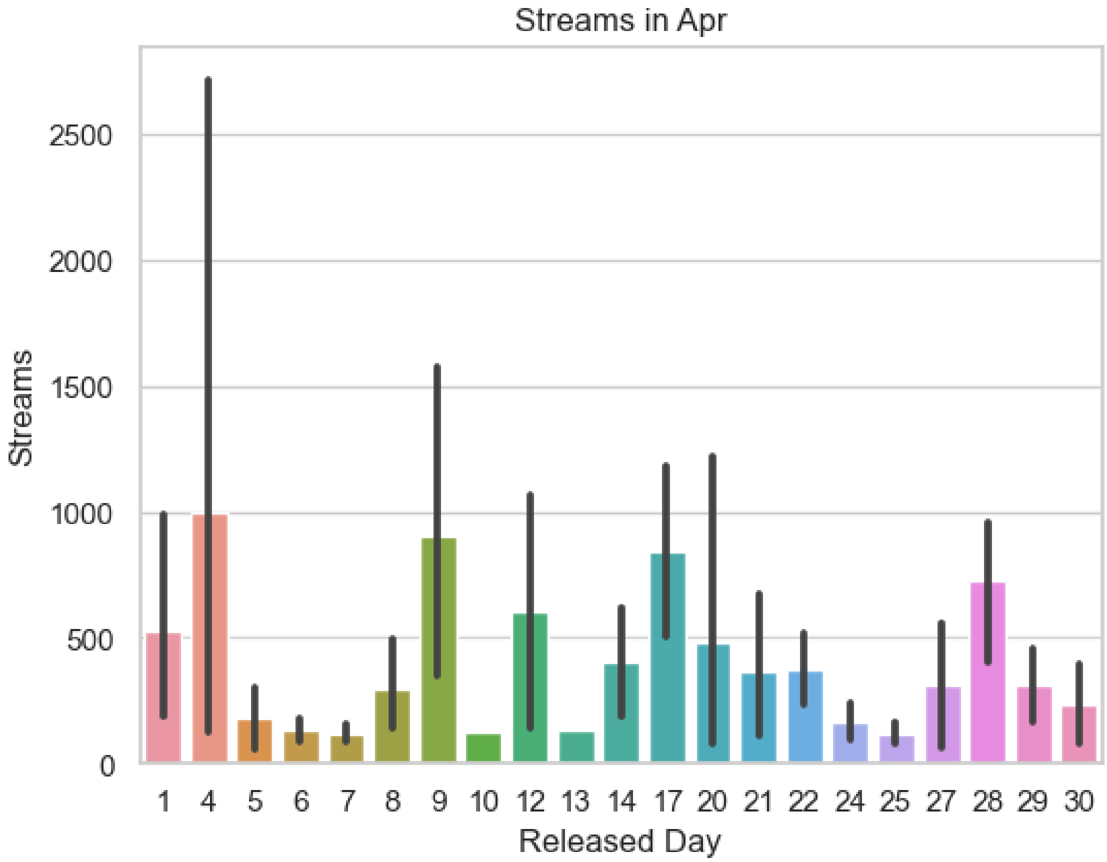
```
# Get unique months in the dataset
unique_months = Data['month'].unique()

# Create a bar plot for each month
for month in unique_months:
    # Filter data for the current month
    monthly_data = Data[Data['month'] == month]

    # Create a bar plot
    ax = sns.barplot(x="released_day", y="streams", data=monthly_data)

    # Set labels and title
    ax.set(xlabel="Released Day", ylabel="Streams", title=f"Streams in
{month}")

    # Show the plot
    plt.show()
```

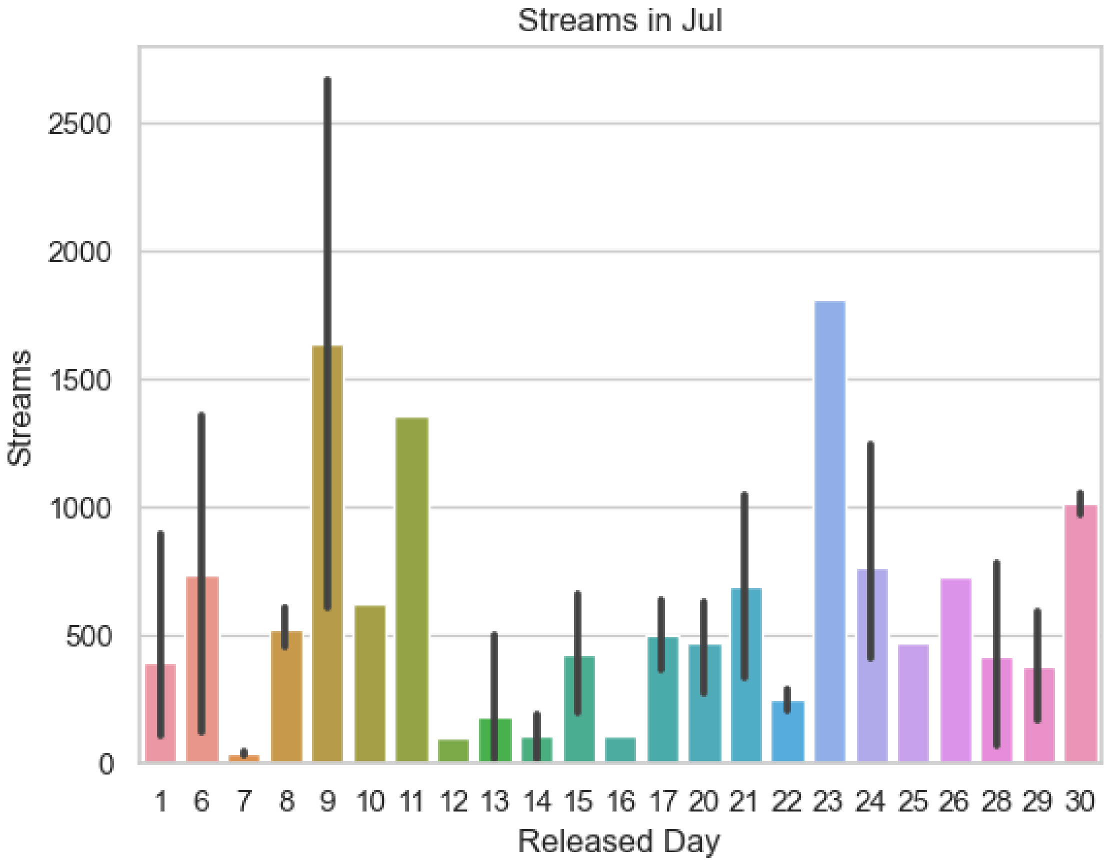


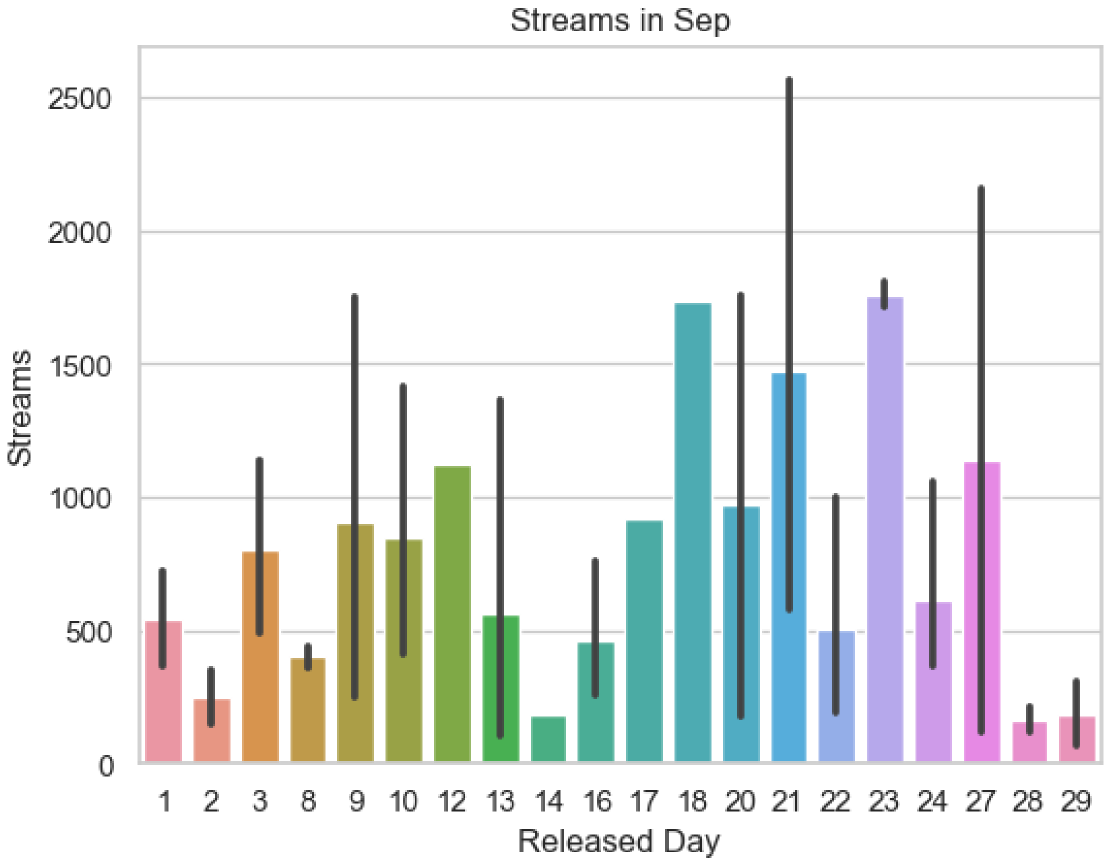

Streams in Jan
Streams
0
500
1000
1500
2000
2500
3000
3500
1
2
5
6
7
8
9
10
11
12
13
14
15
16
17
19
20
21
23
26
27
28
30
31
Released Day

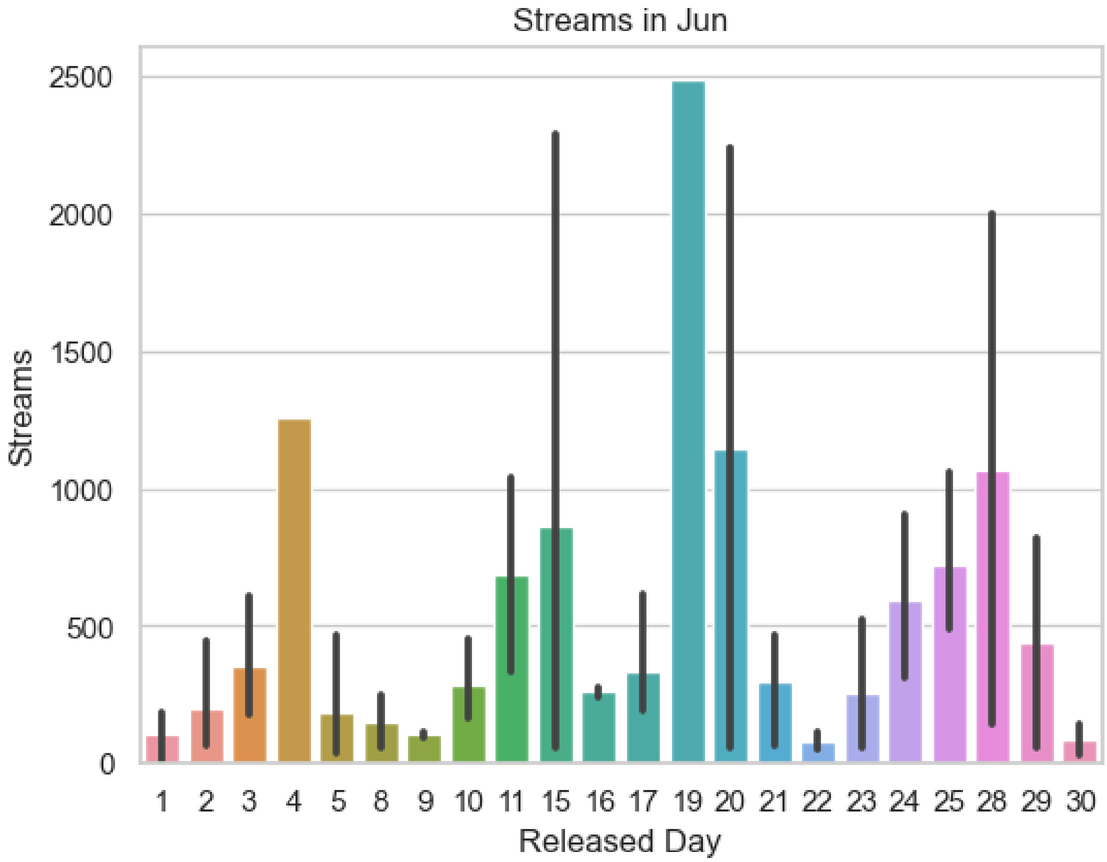

Streams in May
Streams
2000
1500
1000
500
0
1
2
3
4
5
6
7
8
10
11
12
13
14
15
16
18
19
20
21
22
25
26
27
31
Released Day

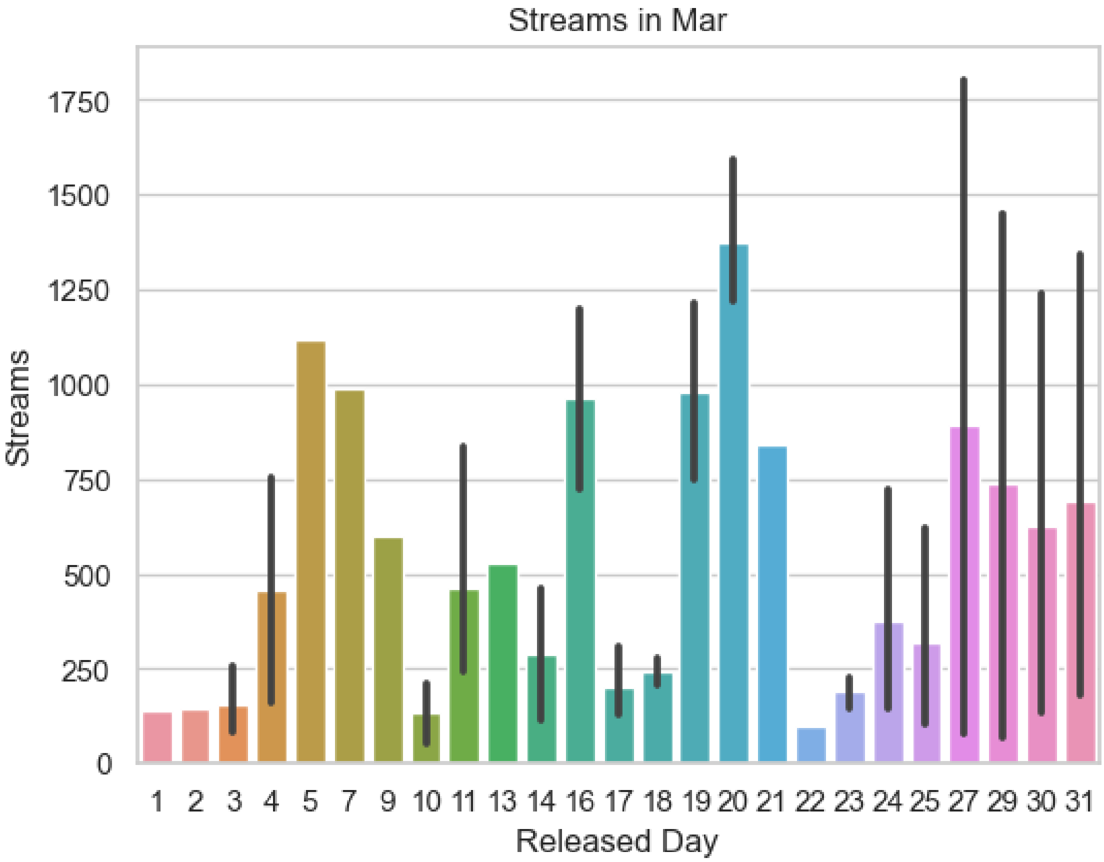

Streams in Oct
Streams
2500
2000
1500
1000
500
0
1
3
4
5
6
7
8
9
12
14
15
17
18
19
20
21
22
23
24
25
26
27
28
30
31
Released Day

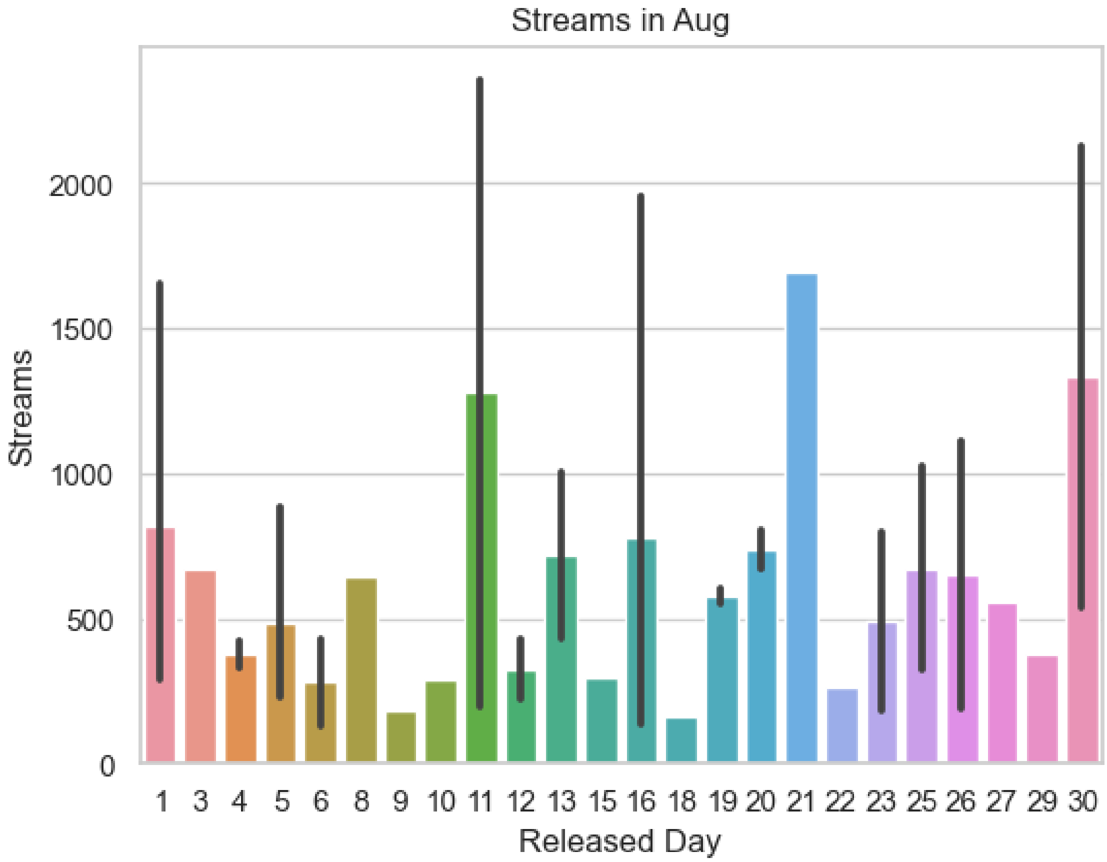

Streams in Apr
Streams
2500
2000
1500
1000
500
0
1
4
5
6
7
8
9
10
12
13
14
17
20
21
22
24
25
27
28
29
30
Released Day

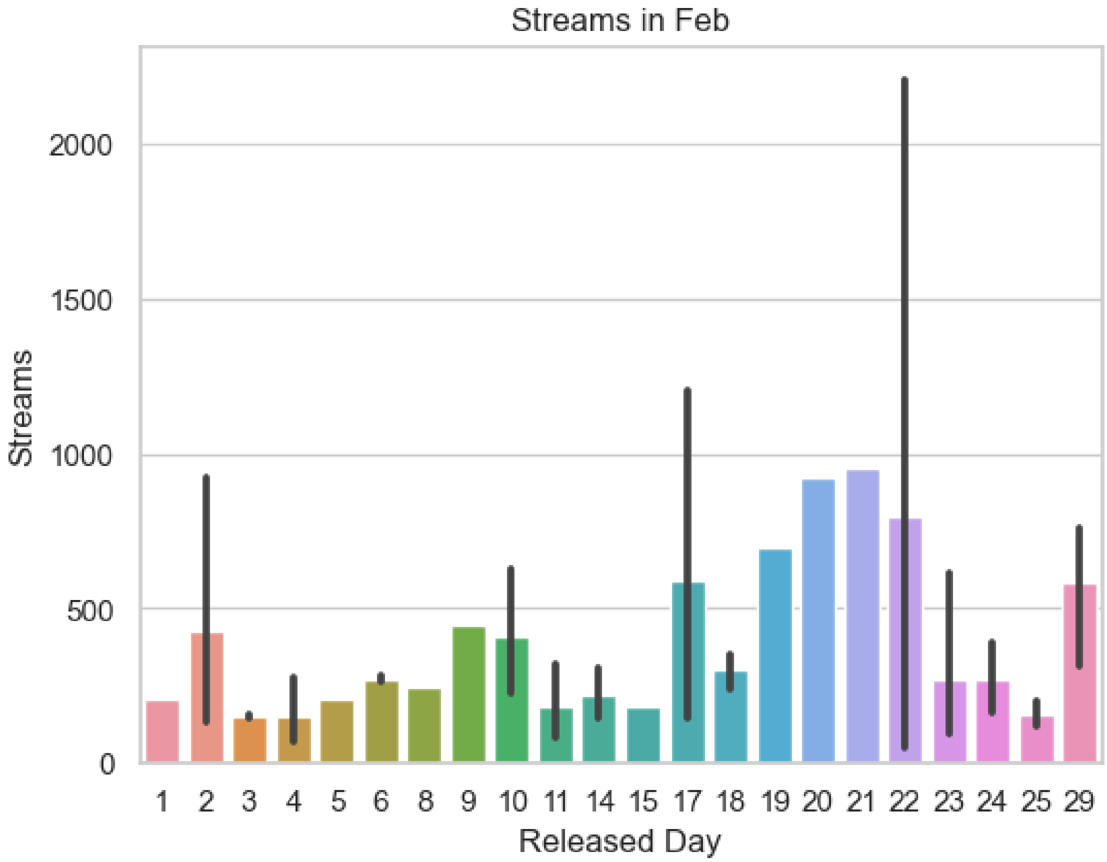

Streams in Jul
Streams
0
500
1000
1500
2000
2500
1
6
7
8
9
10
11
12
13
14
15
16
17
20
21
22
23
24
25
26
28
29
30
Released Day

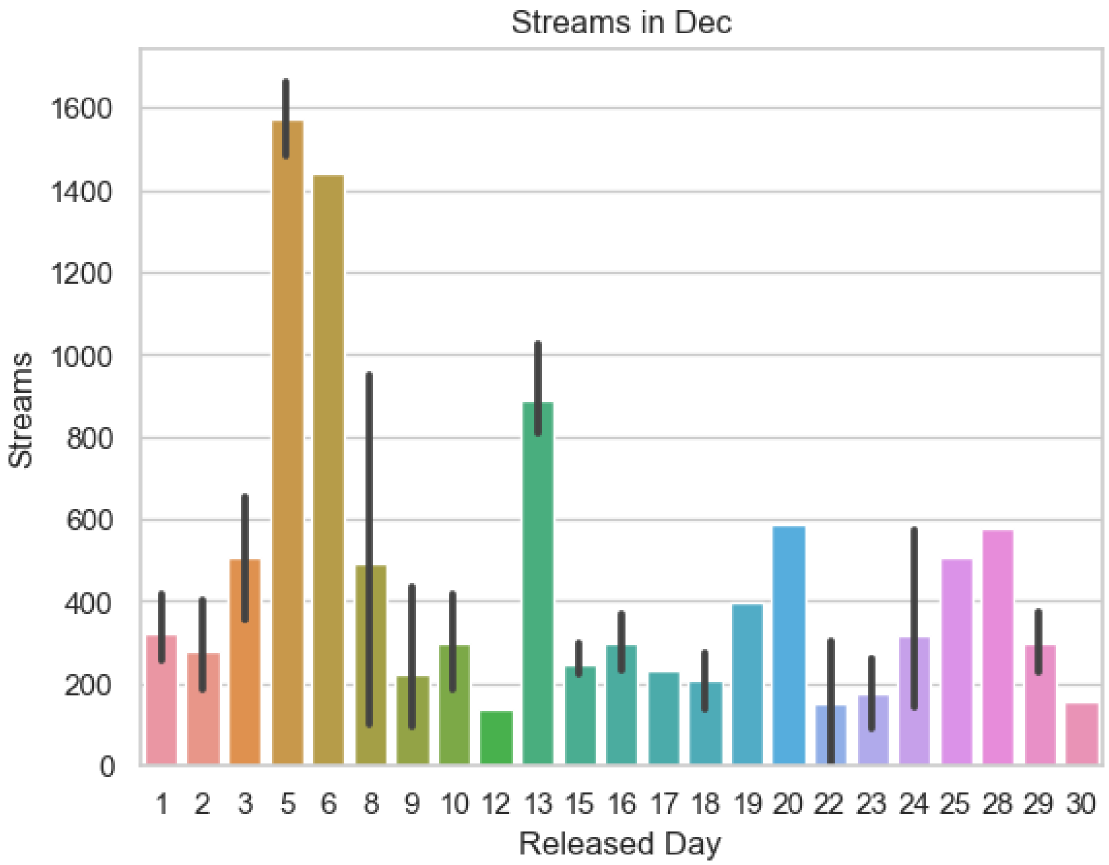

Streams in Sep
Streams
0
500
1000
1500
2000
2500
1
2
3
8
9
10
12
13
14
16
17
18
20
21
22
23
24
27
28
29
Released Day


Streams in Jun
Streams
2500
2000
1500
1000
500
0
1
2
3
4
5
8
9
10
11
15
16
17
19
20
21
22
23
24
25
28
29
30
Released Day

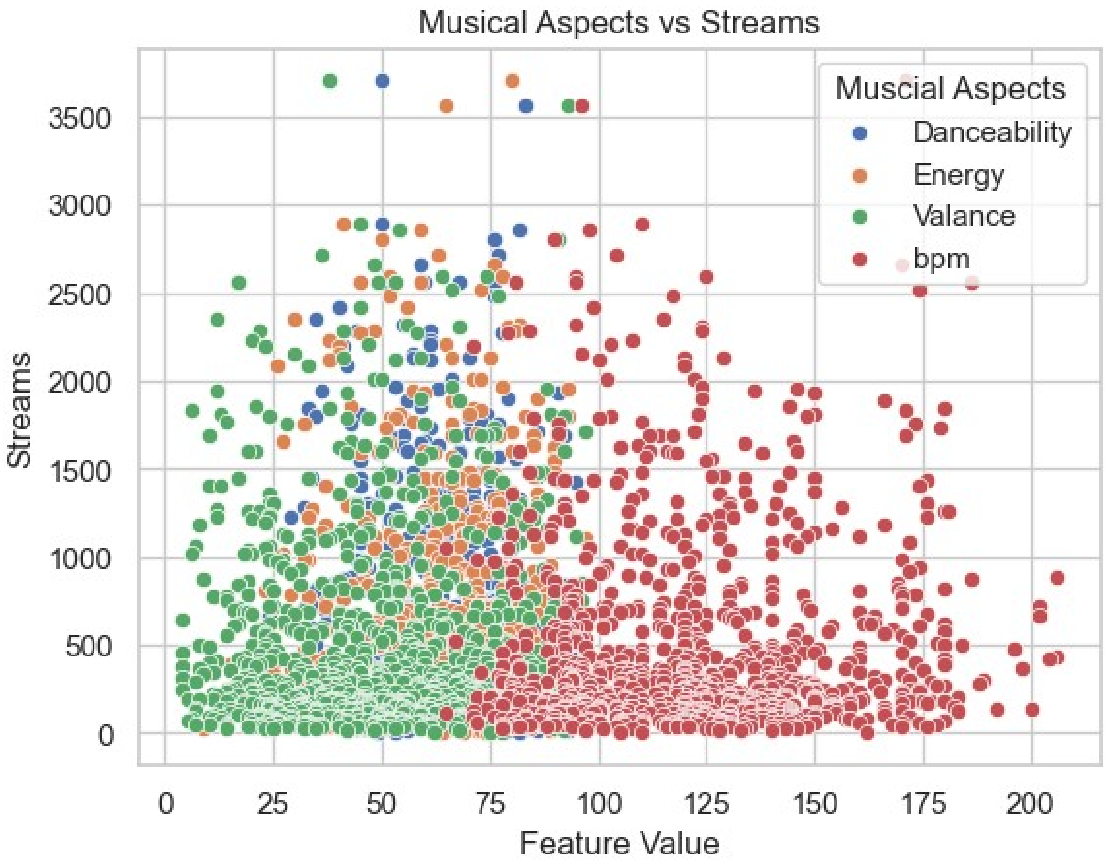

Streams in Mar
Streams
1750
1500
1250
1000
750
500
250
0
1
2
3
4
5
7
9
10
11
13
14
16
17
18
19
20
21
22
23
24
25
27
29
30
31
Released Day


Streams in Aug
Streams
0
500
1000
1500
2000
1
3
4
5
6
8
9
10
11
12
13
15
16
18
19
20
21
22
23
25
26
27
29
30
Released Day

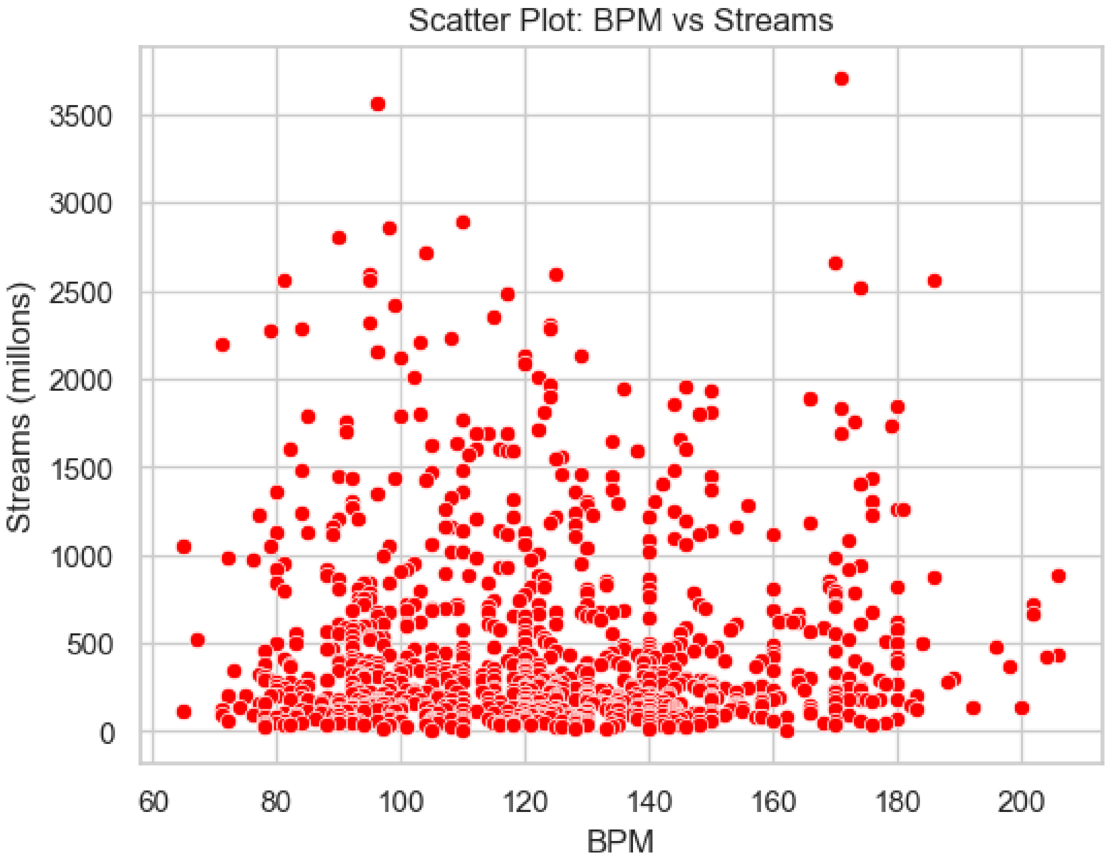

Streams in Feb
Streams
0
500
1000
1500
2000
1
2
3
4
5
6
8
9
10
11
14
15
17
18
19
20
21
22
23
24
25
29
Released Day



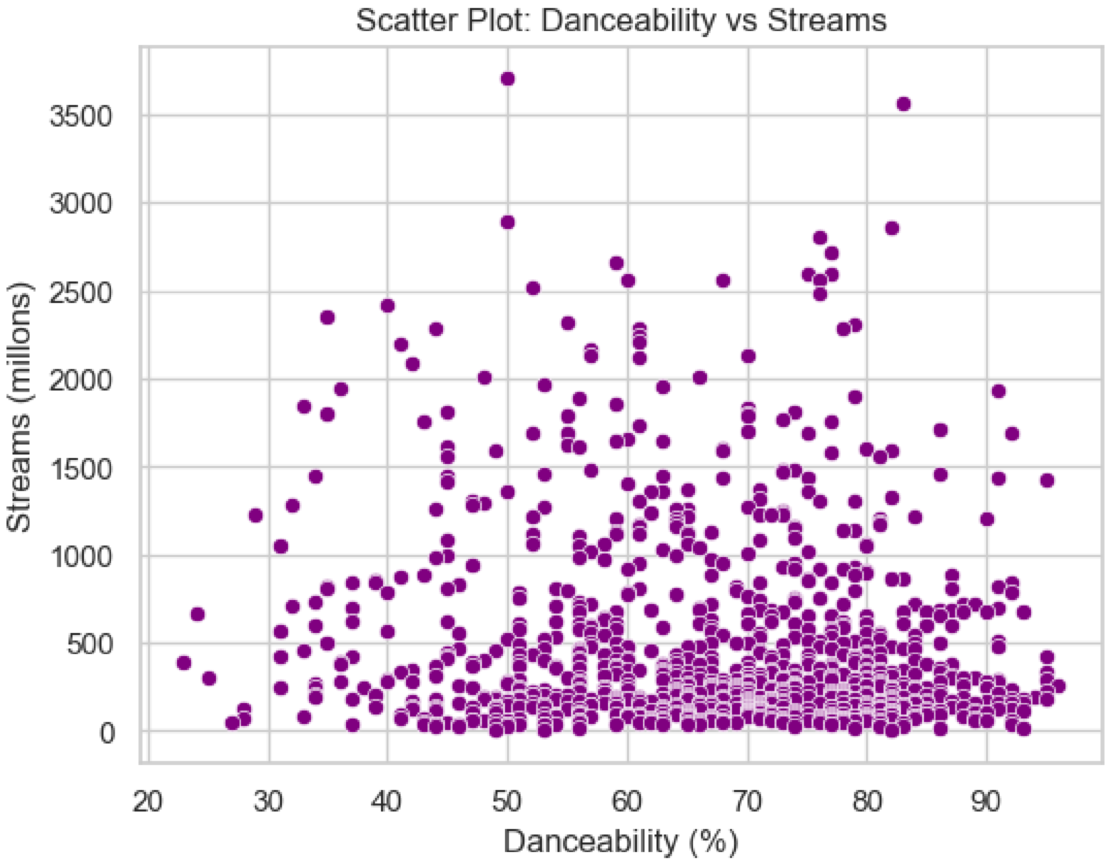
```python
# Create scatter plots with different markers
sns.scatterplot(x='danceability_%', y='streams', data=Data,
markers=['o'])
sns.scatterplot(x='energy_%', y='streams', data=Data, markers=['^'])
sns.scatterplot(x='valence_%', y='streams', data=Data, markers=['*'])
sns.scatterplot(x='bpm', y='streams', data=Data, markers=['+'])

# Set plot labels
plt.xlabel('Feature Value')
plt.ylabel('Streams')

plt.title('Musical Aspects vs Streams')

plt.legend(title='Muscial Aspects', labels=['Danceability', 'Energy',
'Valance','bpm'])


# Show the plot
plt.show()
```



```
sns.scatterplot(data=Data, x="bpm", y="streams", color='red')

plt.title('Scatter Plot: BPM vs Streams')
plt.xlabel('BPM')
plt.ylabel('Streams (millons)')

Text(0, 0.5, 'Streams (millons)')
```

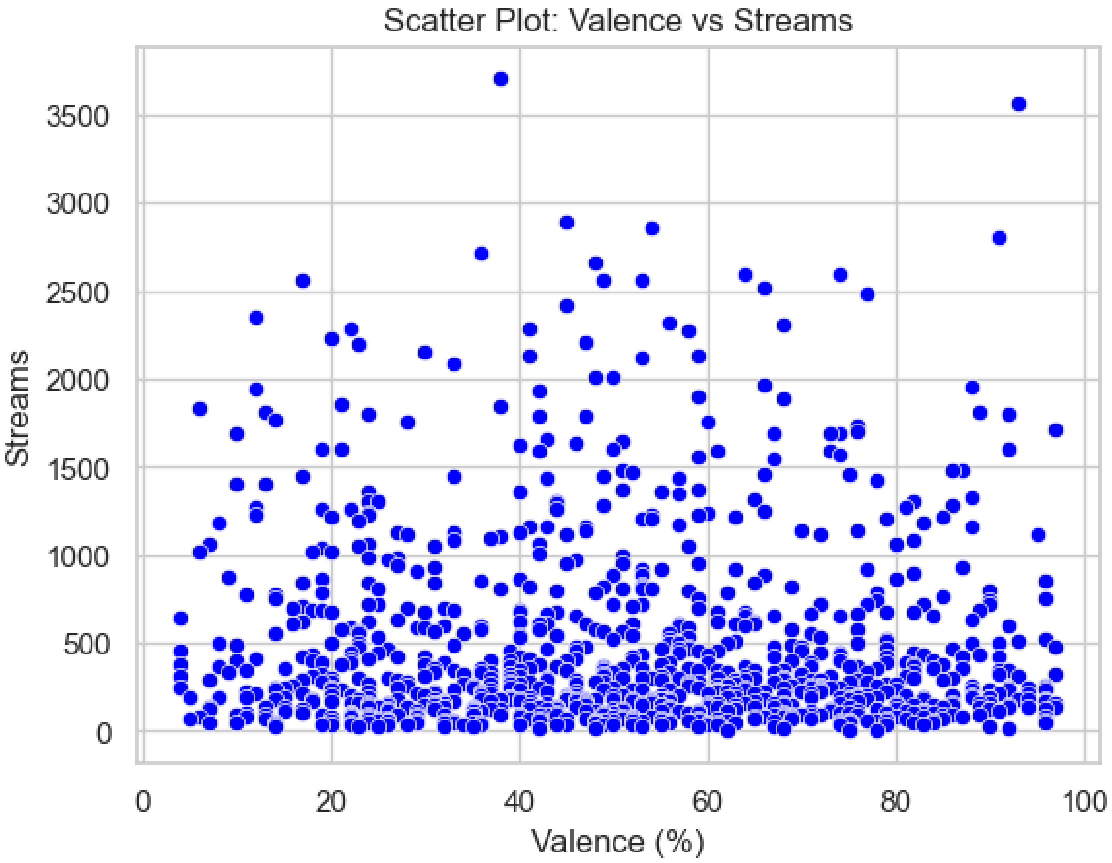


```
sns.scatterplot(data=Data, x="danceability_%", y="streams",
color='purple')

plt.title('Scatter Plot: Danceability vs Streams')
plt.xlabel('Danceability (%)')
plt.ylabel('Streams (millons)')

Text(0, 0.5, 'Streams (millons)')
```

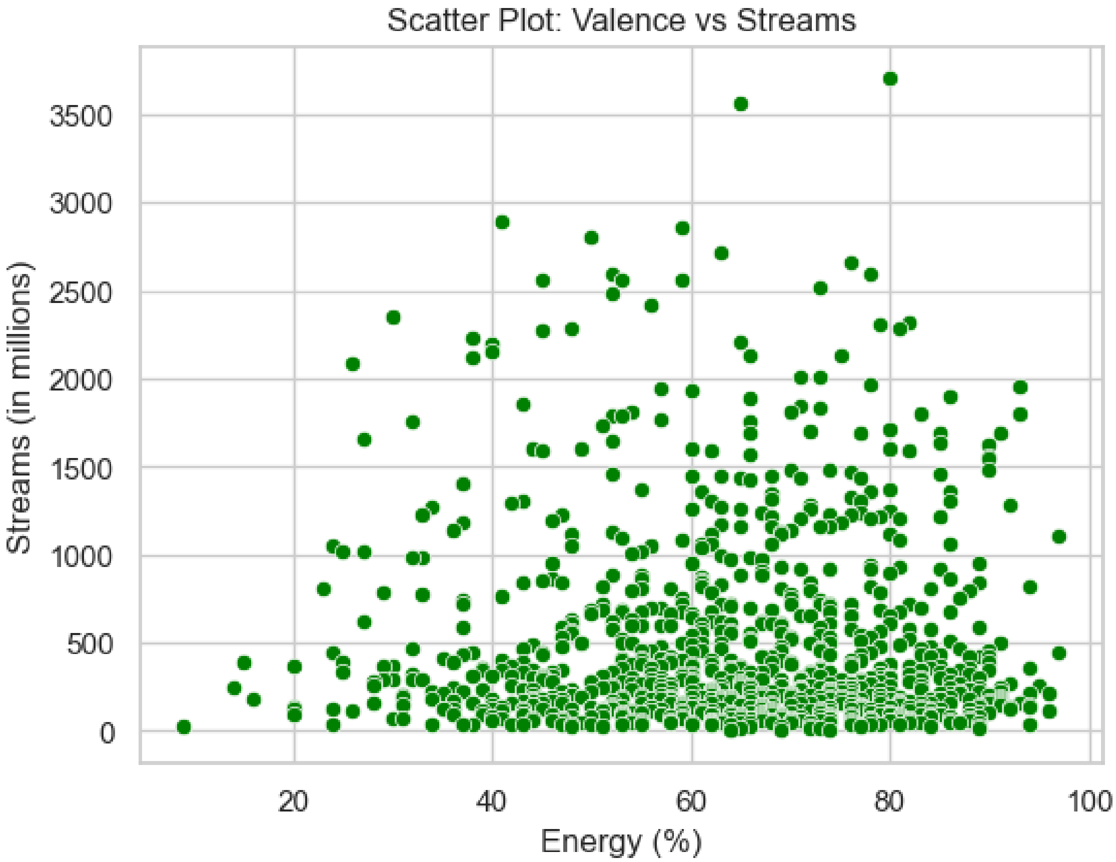


```
sns.scatterplot(data=Data, x="valence_%", y="streams", color='blue')

plt.title('Scatter Plot: Valence vs Streams')
plt.xlabel('Valence (%)')
plt.ylabel('Streams')
```

```
Text(0, 0.5, 'Streams')
```

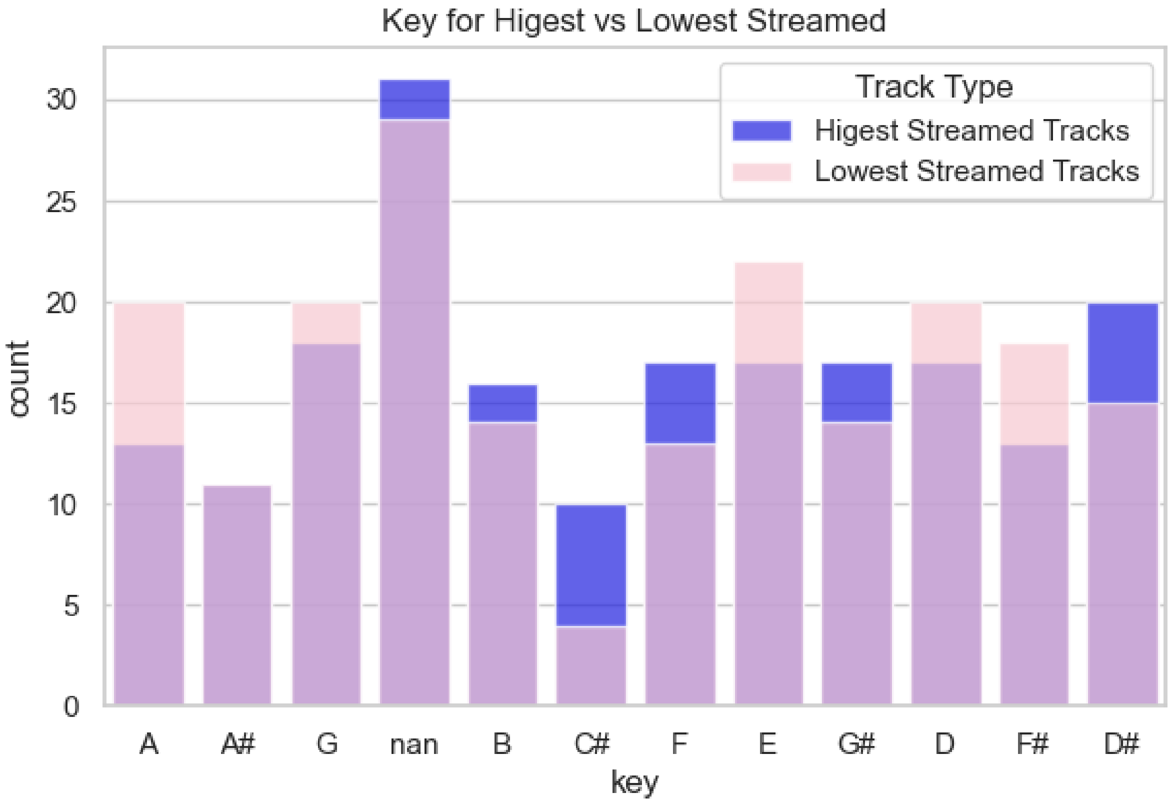


```
sns.scatterplot(data=Data, x="energy_%", y="streams", color='green')

plt.title('Scatter Plot: Valence vs Streams')
plt.xlabel('Energy (%)')
plt.ylabel('Streams (in millions)')

Text(0, 0.5, 'Streams (in millions)')
```



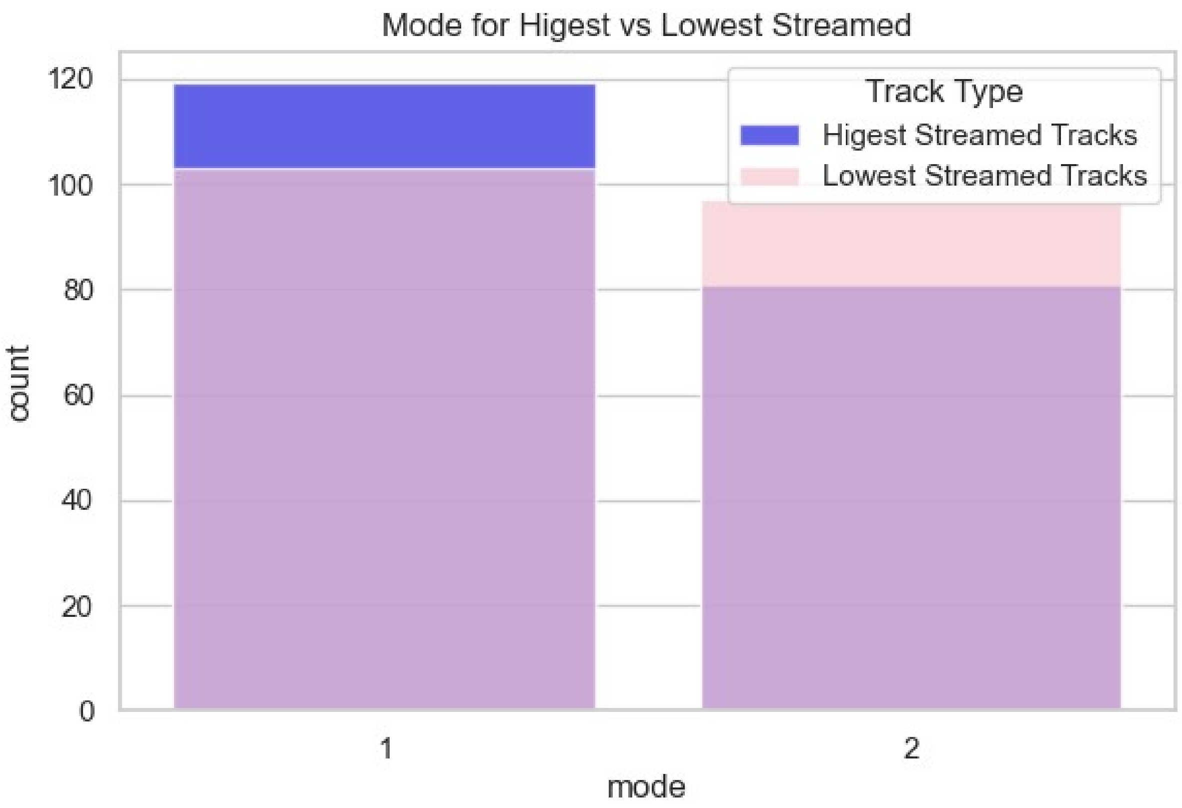
```
import seaborn as sns
import matplotlib.pyplot as plt

# Assuming 'Data' is your DataFrame

tops = [200]
tops_str = ['top200']

low = [200]
low_str = ['low200']

# Create a 2x2 subplot for top and low counts
fig, axes = plt.subplots(1, figsize=(12, 8))

for i, (top_col, low_col) in enumerate(zip(tops_str, low_str), 1):
    # Plot top counts
    plt.subplot(2, 2, i)
    sns.countplot(x="key", data=Data.head(tops[i-1]),
color='blue',alpha=0.7)
    plt.title(top_col, fontsize=10)

    # Plot low counts on the same subplot
    plt.subplot(2, 2, i)
    sns.countplot(x="key", data=Data.tail(low[i-1]), color='pink',
alpha=0.7)
    plt.title(low_col, fontsize=10)


keys_encoded = [0,  1,  9, 11,  2,  3,  7,  6, 10,  4,  8,  5]
keys_original = ['A', 'A#', 'G', 'nan', 'B', 'C#', 'F', 'E', 'G#',
'D', 'F#', 'D#']
plt.legend(title='Track Type', labels=['Higest Streamed Tracks',
'Lowest Streamed Tracks', 'Overlap'])
plt.title('Key for Higest vs Lowest Streamed')
plt.xticks(range(len(keys_encoded)), keys_original)
# Adjust layout for better spacing
plt.tight_layout()
plt.show()
```

```
/var/folders/j2/gt7wq0h5751027qf23w96tqw0000gn/T/
ipykernel_87432/1126530090.py:17: MatplotlibDeprecationWarning: Auto-
removal of overlapping axes is deprecated since 3.6 and will be
removed two minor releases later; explicitly call ax.remove() as
needed.
  plt.subplot(2, 2, i)
```

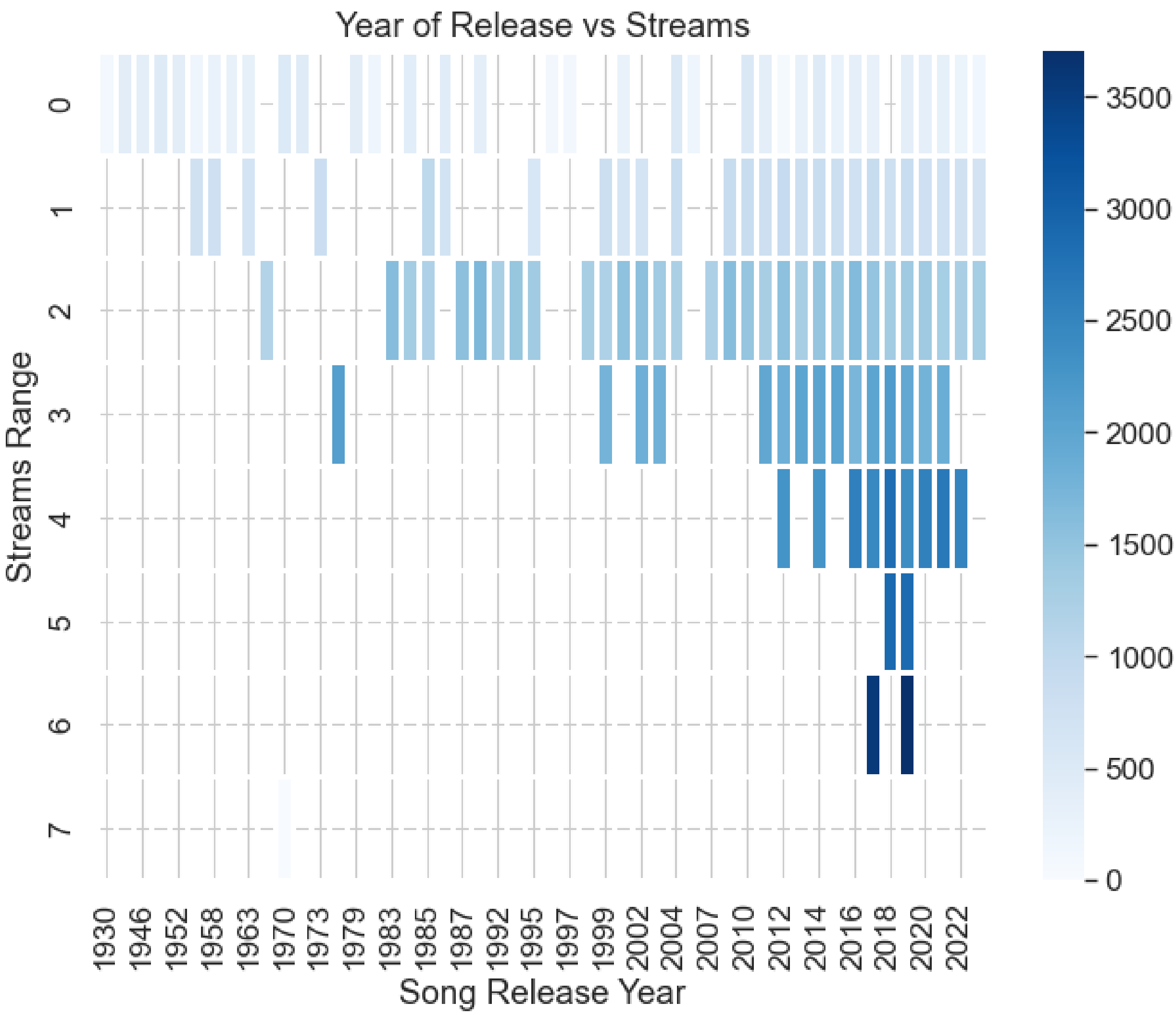


```
import seaborn as sns
import matplotlib.pyplot as plt

# Assuming 'Data' is your DataFrame

tops = [200]
tops_str = ['top200']

low = [200]
low_str = ['low200']

# Create a 2x2 subplot for top and low counts
fig, axes = plt.subplots(1, figsize=(12, 8))

for i, (top_col, low_col) in enumerate(zip(tops_str, low_str), 1):
    # Plot top counts
    plt.subplot(2, 2, i)
    sns.countplot(x="mode", data=Data.head(tops[i-1]), 
color='blue',alpha=0.7)
    plt.title(top_col, fontsize=10)

    # Plot low counts on the same subplot
    plt.subplot(2, 2, i)
    sns.countplot(x="mode", data=Data.tail(low[i-1]), color='pink', 
alpha=0.7)
    plt.title(low_col, fontsize=10)


#keys_encoded = [0,  1,  9, 11,  2,  3,  7,  6, 10,  4,  8,  5]
#keys_original = ['A', 'A#', 'G', 'nan', 'B', 'C#', 'F', 'E', 'G#', 
'D', 'F#', 'D#']
plt.legend(title='Track Type', labels=['Higest Streamed Tracks', 
'Lowest Streamed Tracks', 'Overlap'])
plt.title('Mode for Higest vs Lowest Streamed')
#plt.xticks(range(len(keys_encoded)), keys_original)
# Adjust layout for better spacing
plt.tight_layout()
plt.show()
```

```
/var/folders/j2/gt7wq0h5751027qf23w96tqw0000gn/T/
ipykernel_87432/700858123.py:17: MatplotlibDeprecationWarning: Auto-
removal of overlapping axes is deprecated since 3.6 and will be 
removed two minor releases later; explicitly call ax.remove() as 
needed.
  plt.subplot(2, 2, i)
```

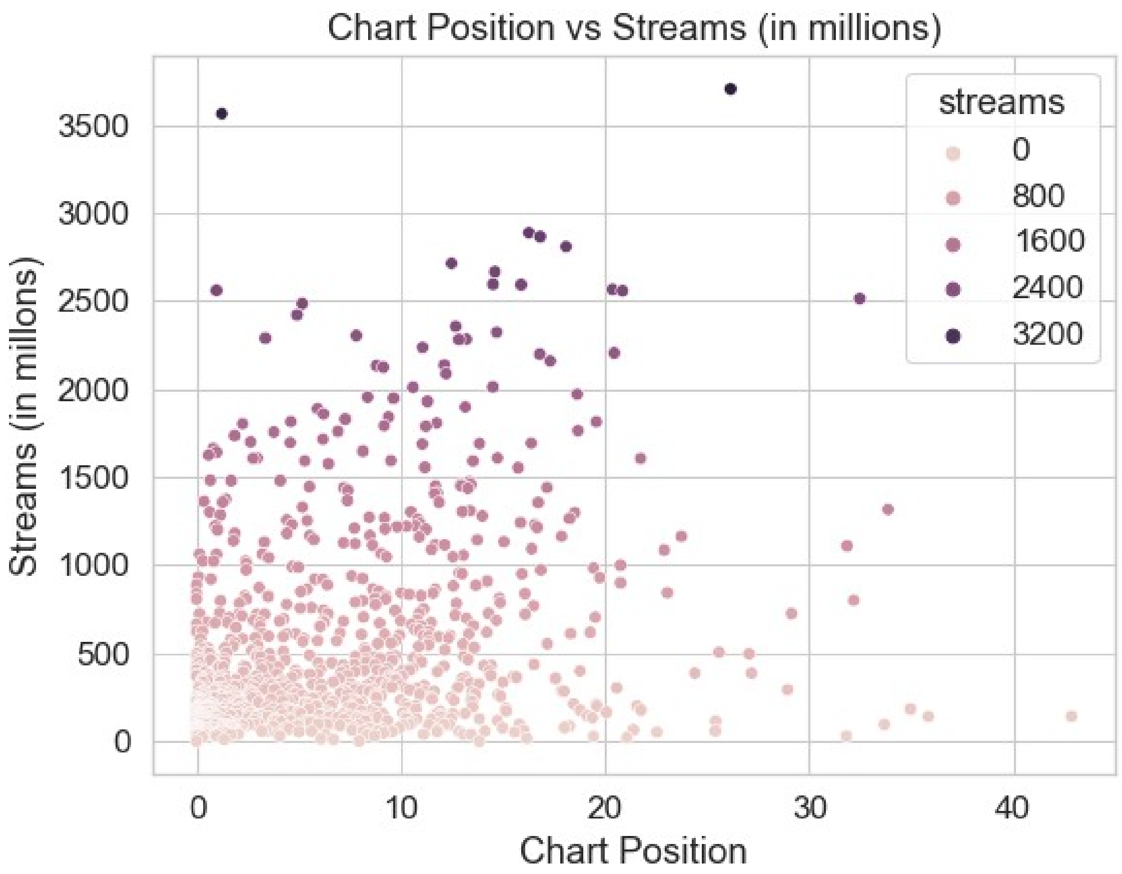


```
plt.figure(figsize=(8,6))
sns.set_context('paper', font_scale=1.4)



flights = Data.pivot_table(columns='released_year',
index='streams_range', values='streams')
#Data.sort_values('month_integer')
# You can separate data with lines
sns.heatmap(flights, cmap='Blues', linecolor='white', linewidth=1)
plt.title("Year of Release vs Streams")
plt.ylabel('Streams Range')
plt.xlabel('Song Release Year')

Text(0.5, 33.91666666666665, 'Song Release Year')
```



```
rel = sns.scatterplot(data=Data, x='chart', y='streams',
hue='streams')
plt.title('Chart Position vs Streams (in millions)')
plt.ylabel('Streams (in millons)')
plt.xlabel('Chart Position')
#rel.invert_xaxis()
```

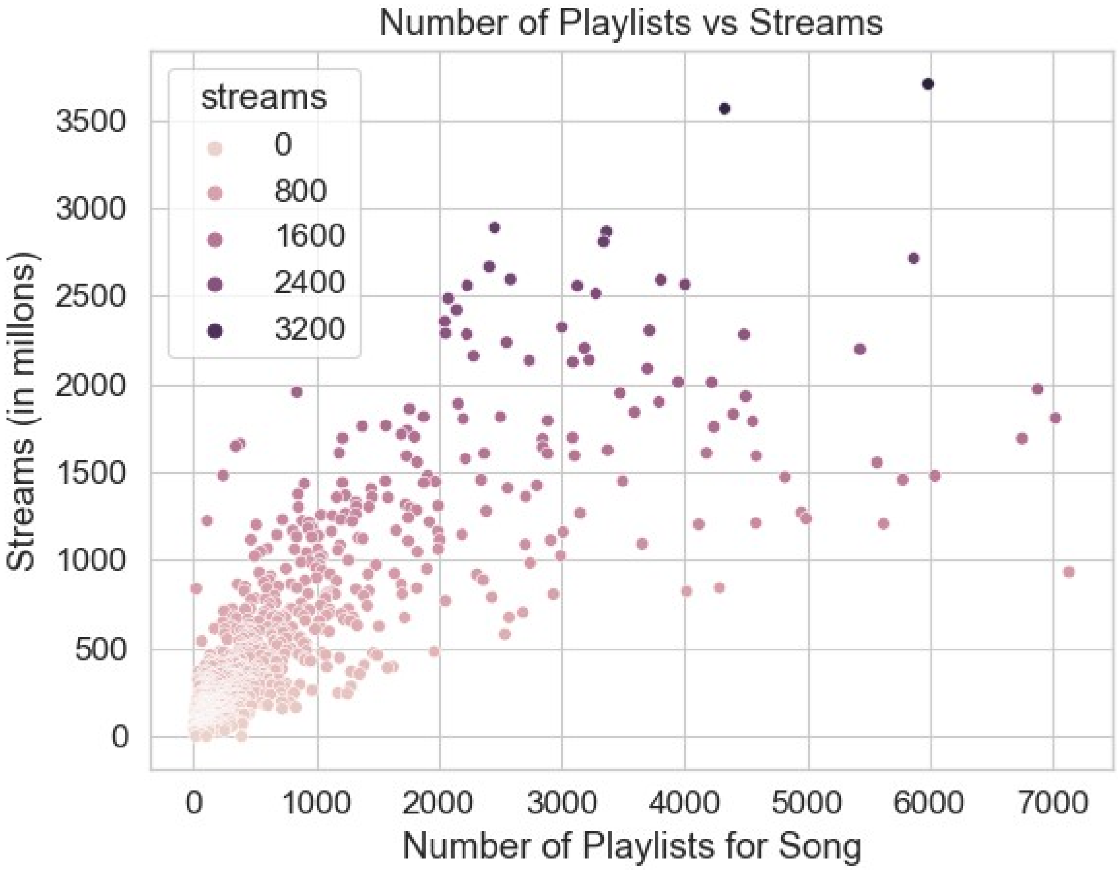
```
Text(0.5, 0, 'Chart Position')
```



```
playlist =
Data[['in_apple_playlists','in_spotify_playlists','in_deezer_playlists
','playlist','streams']]
charts = Data [['in_spotify_charts', 'in_apple_charts',
'in_deezer_charts', 'in_shazam_charts','chart','streams']]

# Assuming 'Data' is your DataFrame
weights_playlists = {
    'in_apple_playlists':
Data['in_apple_playlists'].corr(Data['streams']),
    'in_spotify_playlists':
Data['in_spotify_playlists'].corr(Data['streams']),
    'in_deezer_playlists':
Data['in_deezer_playlists'].corr(Data['streams']),
}

# Normalize weights to sum to 1
total_weight_playlists = sum(weights_playlists.values())
weights_playlists = {key: value / total_weight_playlists for key,
value in weights_playlists.items()}

# Calculate the weighted sum for 'playlist'
Data['playlist'] = ((
    weights_playlists['in_apple_playlists'] *
Data['in_apple_playlists'] +
    weights_playlists['in_spotify_playlists'] *
Data['in_spotify_playlists'] +
    weights_playlists['in_deezer_playlists'] *
Data['in_deezer_playlists']
)/3)

# Weights for 'chart' columns
weights_charts = {
    'in_spotify_charts':
Data['in_spotify_charts'].corr(Data['streams']),
    'in_apple_charts': Data['in_apple_charts'].corr(Data['streams']),
    'in_deezer_charts':
Data['in_deezer_charts'].corr(Data['streams']),
    'in_shazam_charts':
Data['in_shazam_charts'].corr(Data['streams']),
}

# Normalize weights to sum to 1
total_weight_charts = sum(weights_charts.values())
weights_charts = {key: value / total_weight_charts for key, value in
weights_charts.items()}

# Calculate the weighted sum for 'chart'
Data['chart'] = ((
    weights_charts['in_spotify_charts'] * Data['in_spotify_charts'] +
    weights_charts['in_apple_charts'] * Data['in_apple_charts'] +
    weights_charts['in_deezer_charts'] * Data['in_deezer_charts'] +
    weights_charts['in_shazam_charts'] * Data['in_shazam_charts']
)/4)

rel2 = sns.scatterplot(data=Data, x='playlist', y='streams',
hue='streams', cmap="magma")
plt.title('Number of Playlists vs Streams')
plt.ylabel('Streams (in millons)')
plt.xlabel('Number of Playlists for Song')
```

```
/Users/leah/anaconda3/lib/python3.11/site-packages/seaborn/
relational.py:573: UserWarning: No data for colormapping provided via
'c'. Parameters 'cmap' will be ignored
  points = ax.scatter(x=x, y=y, **kws)
```

```
Text(0.5, 0, 'Number of Playlists for Song')
```



```
---------------------------------------------------------------------------
-----
NameError                                 Traceback (most recent call
last)
Cell In[3], line 1
----> 1 df_artists =
Data.assign(artists_all=Data['artists_all'].str.split(',')).explode('a
rtists_all')
      4 df_artists[['track_name', 'artists_all', 'artist_count',
'released_year',
      5         'released_month', 'released_day',
'in_spotify_playlists',
      6         'in_spotify_charts', 'streams', 'in_apple_playlists',
'in_apple_charts',
   (...)
     15         'instru_per', 'live_per', 'speech_per', 'track_id',
'art_combined_id',
     16         'date']].ffill()
     18 df_artists['artists_id'] =
df_artists['artists_all'].apply(lambda x: le.fit_transform([x])[0])
```

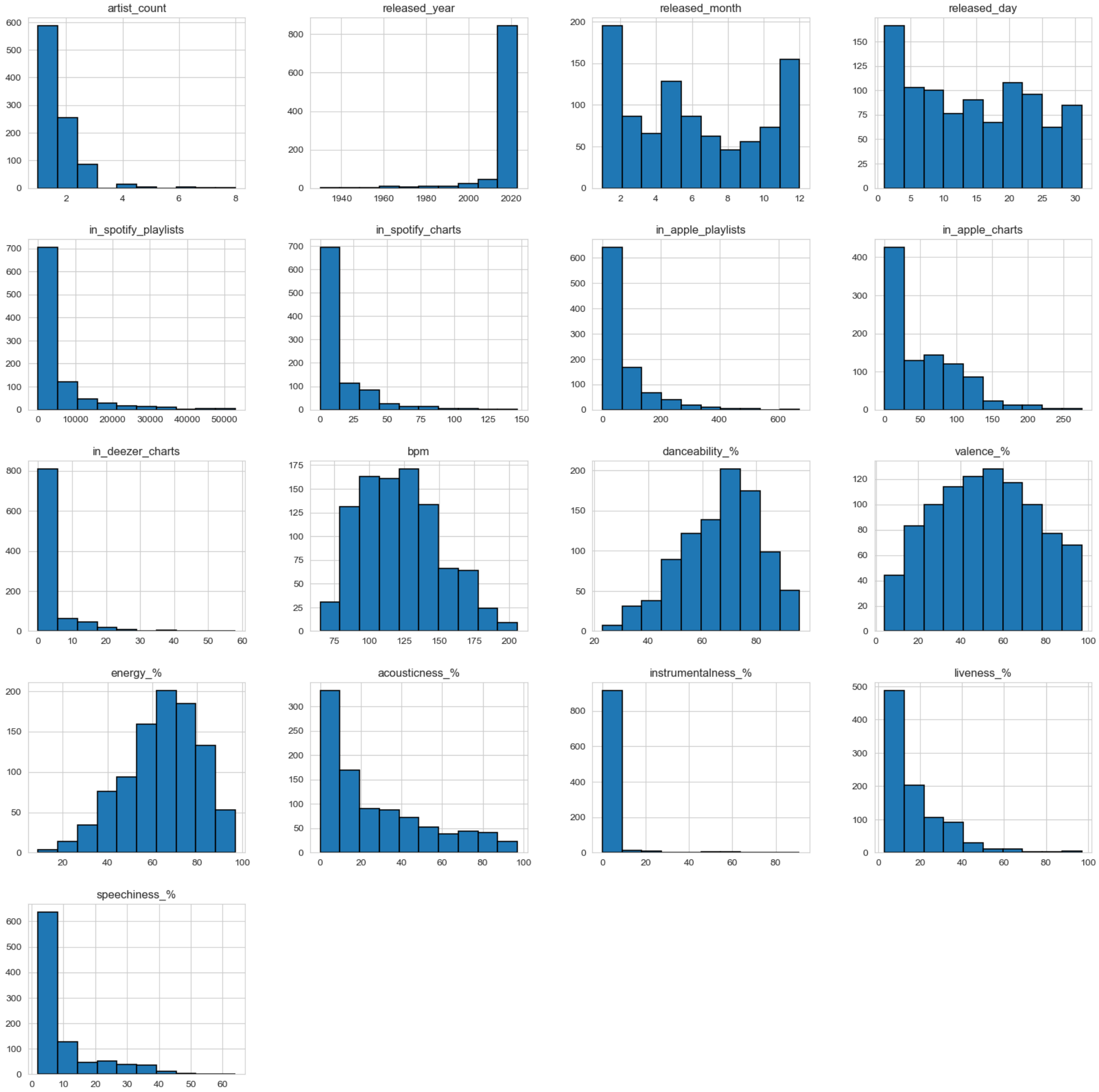
```
NameError: name 'Data' is not defined
```

```python
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix, accuracy_score
from sklearn.preprocessing import StandardScaler, OneHotEncoder
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
from sklearn.metrics import confusion_matrix, accuracy_score
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import confusion_matrix, accuracy_score
from sklearn.model_selection import cross_val_score
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
import matplotlib.pyplot as plt
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import statsmodels.api as sm
from sklearn.preprocessing import LabelEncoder
#Libraries for data processing
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split

#SVM
from sklearn import svm

#Evaluation metrics
from sklearn.metrics import classification_report, confusion_matrix
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn import svm
from sklearn.metrics import confusion_matrix, classification_report
from sklearn.model_selection import train_test_split
from sklearn.datasets import make_classification

# import Dataset and Data cleaning

Data = pd.read_csv("spotify1.csv", encoding='ISO-8859-1')

Data.head(5)
```

```
                            track_name    artist(s)_name  artist_count
\
0  Seven (feat. Latto) (Explicit Ver.)  Latto, Jung Kook             2

1                                 LALA       Myke Towers             1

2                              vampire    Olivia Rodrigo             1

3                         Cruel Summer      Taylor Swift             1

4                       WHERE SHE GOES         Bad Bunny             1


   released_year  released_month  released_day
in_spotify_playlists  \
0           2023               7            14                   553

1           2023               3            23                  1474

2           2023               6            30                  1397

3           2019               8            23                  7858

4           2023               5            18                  3133


   in_spotify_charts    streams  in_apple_playlists  ...  bpm key
mode  \
0                147  141381703                  43  ...  125   B
Major
1                 48  133716286                  48  ...   92  C#
Major
2                113  140003974                  94  ...  138   F
Major
3                100  800840817                 116  ...  170   A
Major
4                 50  303236322                  84  ...  144   A
Minor

  danceability_%  valence_% energy_% acousticness_%  instrumentalness_
%  \
0             80         89       83             31
0
1             71         61       74              7
0
2             51         32       53             17
0
3             55         58       72             11
0
4             65         23       80             14
63
```

```
   liveness_%  speechiness_%
0           8              4
1          10              4
2          31              6
3          11             15
4          11              6

[5 rows x 24 columns]
```

```
Data.hist(edgecolor='black', linewidth=1.2, figsize=(20, 20))
plt.show()
```



```
Data.describe()

       artist_count  released_year  released_month  released_day  \
count    953.000000     953.000000      953.000000    953.000000
mean       1.556139    2018.238195        6.033578     13.930745
std        0.893044      11.116218        3.566435      9.201949
min        1.000000    1930.000000        1.000000      1.000000
25%        1.000000    2020.000000        3.000000      6.000000
50%        1.000000    2022.000000        6.000000     13.000000
75%        2.000000    2022.000000        9.000000     22.000000
max        8.000000    2023.000000       12.000000     31.000000

       in_spotify_playlists  in_spotify_charts  in_apple_playlists  \
count            953.000000         953.000000          953.000000
mean            5200.124869          12.009444           67.812172
std             7897.608990          19.575992           86.441493
min               31.000000           0.000000            0.000000
25%              875.000000           0.000000           13.000000
50%             2224.000000           3.000000           34.000000
75%             5542.000000          16.000000           88.000000
max            52898.000000         147.000000          672.000000

       in_apple_charts  in_deezer_charts         bpm  danceability_
%  \
count       953.000000        953.000000  953.000000       953.00000

mean         51.908709          2.666317  122.540399        66.96957

std          50.630241          6.035599   28.057802        14.63061

min           0.000000          0.000000   65.000000        23.00000

25%           7.000000          0.000000  100.000000        57.00000

50%          38.000000          0.000000  121.000000        69.00000

75%          87.000000          2.000000  140.000000        78.00000

max         275.000000         58.000000  206.000000        96.00000


        valence_%    energy_%  acousticness_%  instrumentalness_%
liveness_%  \
count  953.000000  953.000000      953.000000          953.000000
953.000000
mean    51.431270   64.279119       27.057712            1.581322
18.213012
std     23.480632   16.550526       25.996077            8.409800
13.711223
min      4.000000    9.000000        0.000000            0.000000
3.000000
```

```
25%     32.000000   53.000000        6.000000            0.000000
10.000000
50%     51.000000   66.000000       18.000000            0.000000
12.000000
75%     70.000000   77.000000       43.000000            0.000000
24.000000
max     97.000000   97.000000       97.000000           91.000000
97.000000

       speechiness_%
count     953.000000
mean       10.131165
std         9.912888
min         2.000000
25%         4.000000
50%         6.000000
75%        11.000000
max        64.000000
```

```
Data.shape
```

```
(953, 24)
```

```
Data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 953 entries, 0 to 952
Data columns (total 24 columns):
 #   Column                Non-Null Count  Dtype 
---  ------                --------------  ----- 
 0   track_name            953 non-null    object
 1   artist(s)_name        953 non-null    object
 2   artist_count          953 non-null    int64 
 3   released_year         953 non-null    int64 
 4   released_month        953 non-null    int64 
 5   released_day          953 non-null    int64 
 6   in_spotify_playlists  953 non-null    int64 
 7   in_spotify_charts     953 non-null    int64 
 8   streams               953 non-null    object
 9   in_apple_playlists    953 non-null    int64 
 10  in_apple_charts       953 non-null    int64 
 11  in_deezer_playlists   953 non-null    object
 12  in_deezer_charts      953 non-null    int64 
 13  in_shazam_charts      903 non-null    object
 14  bpm                   953 non-null    int64 
 15  key                   858 non-null    object
 16  mode                  953 non-null    object
 17  danceability_%        953 non-null    int64 
 18  valence_%             953 non-null    int64 
 19  energy_%              953 non-null    int64 
 20  acousticness_%        953 non-null    int64
 21  instrumentalness_%    953 non-null    int64
 22  liveness_%            953 non-null    int64
 23  speechiness_%         953 non-null    int64
dtypes: int64(17), object(7)
memory usage: 178.8+ KB
```

```
def convert_mode(mode):
    if mode == 'Major':
        return 1
    elif mode == 'Minor':
        return 2
    else:
        return None

Data['mode'] = Data['mode'].apply(convert_mode)

# Singers in total
artist_names = Data['artist(s)_name'].unique().tolist()
for artist in artist_names:
    print(artist)

total_artists = Data['artist(s)_name'].nunique()
print(f'Total number of artists: {total_artists}')
```

```
Latto, Jung Kook
Myke Towers
Olivia Rodrigo
Taylor Swift
Bad Bunny
Dave, Central Cee
Eslabon Armado, Peso Pluma
Quevedo
Gunna
Peso Pluma, Yng Lvcas
Bad Bunny, Grupo Frontera
NewJeans
Miley Cyrus
David Kushner
Harry Styles
SZA
Fifty Fifty
Billie Eilish
Feid, Young Miko
Jimin
Gabito Ballesteros, Junior H, Peso Pluma
Arctic Monkeys
Bizarrap, Peso Pluma
The Weeknd, Madonna, Playboi Carti
Fuerza Regida
```

```
Rï¿½ï¿½ma, Selena G
Tainy, Bad Bunny
Morgan Wallen
Dua Lipa
Troye Sivan
Peso Pluma, Grupo Frontera
The Weeknd, 21 Savage, Metro Boomin
Karol G, Shakira
Big One, Duki, Lit Killah, Maria Becerra, FMK, Rusherking, Emilia, Tiago pzk
Yahritza Y Su Esencia, Grupo Frontera
Junior H, Peso Pluma
Post Malone, Swae Lee
Bebe Rexha, David Guetta
Tyler, The Creator, Kali Uchis
Nicki Minaj, Aqua, Ice Spice
OneRepublic
The Weeknd
The Weeknd, Daft Punk
Ariana Grande, The Weeknd
Feid, Myke Towers, Sky Rompiendo
David Guetta, Anne-Marie, Coi Leray
Karol G
Peggy Gou
Tom Odell
Kali Uchis
Manuel Turizo
dennis, MC Kevin o Chris
PinkPantheress, Ice Spice
Charlie Puth, BTS, Jung Kook
Rauw Alejandro, ROSALï¿½
Ozuna, Feid
Chris Molitor
Libianca
Post Malone
Rauw Alejandro, Bizarrap
Glass Animals
JVKE
The Neighbourhood
Bizarrap, Quevedo
Coldplay
d4vd
Sam Smith, Kim Petras
Yandel, Feid
Maria Becerra
Vance Joy
Em Beihold, Stephen Sanchez
Mc Livinho, DJ Matt D
Justin Bieber, The Kid Laroi
```

```
Marshmello, Manuel Turizo
Lewis Capaldi
Chencho Corleone, Bad Bunny
Jain
Ayparia, unxbected
Luke Combs
Doechii
Bad Bunny, Eladio Carrion
Jung Kook
J. Cole, Lil Durk
Lana Del Rey
BTS
Future, Metro Boomin, Don Toliver
Eminem
Carin Leon
Dua Lipa, Elton John, Pnau
Ruth B.
Imagine Dragons
Adele
Kendrick Lamar, Jay Rock
Anggi Marito
Jasiel Nuï¿½ï¿½ez, Peso P
Tears For Fears
J. Cole
Loreen
Taiu, Milo j
Sebastian Yatra, Manuel Turizo, Beï¿½ï
Calvin Harris, Ellie Goulding
Sabrina Carpenter
Carin Leon, Grupo Frontera
Peso Pluma
Feid
Ray Dalton, Ryan Lewis, Macklemore
Billie Eilish, Khalid
(G)I-DLE
Shakira, Bizarrap
Swae Lee, A Boogie Wit da Hoodie, Metro Boomin, NAV
Travis Scott, Young Thug, Metro Boomin
Sachin-Jigar, Shadab Faridi, Altamash Faridi, Amitabh Bhattacharya,
Varun Jain
Ed Sheeran
Veigh, Bvga Beatz, Supernova Ent, Prod Malax
Mc Pedrinho, DJ 900
Sog, Ryan Castro, Peso Pluma
The Weeknd, Lana Del Rey
Bomba Estï¿½ï¿½reo, Bad B
Keane
New West
Chino Pacas
```

```
Natanael Cano
Don Omar, Lucenzo
Ana Castela, AgroPlay
Bruno Mars
Big One, FMK, Ke personajes
Drake, 21 Savage
Justin Bieber
Chris Brown
Natanael Cano, Peso Pluma
Daddy Yankee
Drake, WizKid, Kyla
Miguel
The Police
Lord Huron
Rihanna, Calvin Harris
Alec Benjamin
A$AP Rocky, Metro Boomin, Roisee
YOASOBI
Rosa Linn
One Direction
Oscar Maydon, Junior H
Radiohead
James Arthur
Cigarettes After Sex
Shubh
Frank Ocean
Hozier
Fuerza Regida, Grupo Frontera
Hotel Ugly
Creedence Clearwater Revival
Plan B
Feid, Mora, Saiko, Quevedo
a-ha
Swae Lee, Lil Wayne, Offset, Metro Boomin
Kodak Black, NLE Choppa, Muni Long, JVKE, Jimin
Shakira
Natanael Cano, Gabito Ballesteros, Peso Pluma
Arijit Singh, Sachin-Jigar, Amitabha Bhattacharya
Stray Kids
Becky G, Peso Pluma
Metro Boomin, Coi Leray
Kaliii, Kaliii
Junior H, Eden Muï¿½ï
Abhijay Sharma, Riar Saab
Semicenk, Doï¿½ï¿½u
JISOO
Future, Lil Uzi Vert, Metro Boomin
RAYE, 070 Shake
Arcangel, Bad Bunny
```

```
Taylor Swift, Ice Spice
Future, Chris Brown, Metro Boomin
BLESSD, Peso Pluma
Fuerza Regida, Natanael Cano
Lana Del Rey, Taylor Swift
Coi Leray
ENHYPEN
Lil Durk, Morgan Wallen
James Blake, Metro Boomin
Israel & Rodolffo, Mari Fernandez
Steve Lacy
Beyoncï¿
Halsey, Suga
Nicky Jam, Feid
Sean Paul, Feid
Drake
Kendrick Lamar, Beyoncï¿
WizKid, Toian, Metro Boomin, Don Toliver, Beam
Simone Mendes
Tini
Lil Uzi Vert
Sia
Stephen Sanchez
MC Caverinha, KayBlack
Offset, JID
Zï¿½ï¿½ Neto & Crist
Migrantes, LiL CaKe, Nico Valdi
Tory Lanez
Joji
Mambo Kingz, DJ Luian, Anuel Aa
Nile Rodgers, LE SSERAFIM
Agust D
Rich The Kid, Matuï¿
Labrinth
Karol G, Romeo Santos
Lizzy McAlpine
IVE
sped up 8282
Cartel De Santa, La Kelly
SEVENTEEN
NLE Choppa
The Weeknd, Future
Nicki Minaj, Ice Spice
Grupo Marca Registrada, Grupo Frontera
Fuerza Regida, Peso Pluma
TAEYANG, Lisa
Raim Laode
ROSALï¿½
Meghan Trainor
```

```
sped up nightcore, ARIZONATEARS, Lil Uzi Vert
Grupo Frontera
TV Girl
INTERWORLD
Quevedo, Jhayco
El Chachito, Junior H
The Weeknd, Gesaffelstein
RM, Colde
BLACKPINK
Nengo Flow, Bad Bunny
Nicky Youre, Dazy
Eminem, Dina Rae
IU, Agust D
Arcangel, Bizarrap
Tyler, The Creator
Tiï¿½ï¿½sto, Tate M
NF
Yng Lvcas
Kanii, PR1ISVX
Marï¿½ï¿½lia Mendo
Zach Bryan
Melanie Martinez
Fuerza Regida, Chino Pacas
Mae Stephens
Feid, Mora
Kenia OS
Kordhell
Coolio, L.V.
Karol G, Ovy On The Drums
The Walters
Mac DeMarco
Jack Black
Duki, NICKI NICOLE, Cris Mj, Standly, Stars Music Chile
Brray, Rauw Alejandro, Lyanno
Linkin Park
Gorillaz, Bad Bunny
j-hope, J. Cole
Robin Schulz, Oliver Tree
Don Toliver, Future, Justin Bieber
Niall Horan
Central Cee
Mahalini
Arijit Singh, Sachin-Jigar
ThxSoMch
Myke Towers, Quevedo
Morgan Wallen, Eric Church
Nicki Minaj
Kaifi Khalil
Karol G, Quevedo
```

```
Lady Gaga
Ozuna, Shakira
Leo Santana
Treyce
Post Malone, Doja Cat
MC Xenon, Os Gemeos da Putaria
Imagine Dragons, League of Legends, JID, Arcane
XXXTENTACION
Israel & Rodolffo, Ana Castela
J Balvin, Bad Bunny
Yuridia, Angela Aguilar
The Kid Laroi
Lizzo
Fujii Kaze
Wisin & Yandel, ROSALï¿½
King
Coldplay, BTS
Sam Smith, Calvin Harris, Jessie Reyez
P!nk
LE SSERAFIM
PinkPantheress
TAEYANG, Jimin
Rihanna
Lil Nas X
Beach Weather
Halsey, BTS
Baby Tate
Rï¿½ï
Arijit Singh, Vishal Dadlani, Sukriti Kakar, Vishal-Shekhar, Shekhar Ravjiani, Kumaar
Beach House
BTS, Jung Kook, FIFA Sound
Shakira, Rauw Alejandro
Steve Aoki, Tini, La Joaqui
Sam Smith
Conan Gray
Vishal-Shekhar, Shilpa Rao, Caralisa Monteiro, Kumaar, Vishal Dadlani, Shekhar Ravjiani
Calvin Harris, Dua Lipa
TOMORROW X TOGETHER
Drake, Future, Tems
Mr.Kitty
Gustavo Mioto, Mari Fernandez
Jack Harlow, Lil Nas X
Elley Duhï¿
Seafret
Baby Rasta, Rauw Alejandro
DJ Escobar, MC MENOR SG, MC MENOR HR
Dean Lewis
```

```
Bellakath
Skrillex, Flowdan, Fred again..
Travis Scott, 21 Savage, Metro Boomin
Kate Bush
Aerosmith
Freddie Dredd
Lost Frequencies, Calum Scott
Twisted, Oliver Tree
Kenshi Yonezu
Omar Apollo
Dua Lipa, DaBaby
Doja Cat
James Hype, Miggy Dela Rosa
Pritam, Arijit Singh, Amitabh Bhattacharya
Styrx, utku INC, Thezth
Maroon 5, Wiz Khalifa
Mariah Carey
Wham!
Brenda Lee
Bobby Helms
Michael Bublï¿
Ariana Grande
Andy Williams
Dean Martin
Kelly Clarkson
Josï¿½ï¿½ Felic
The Ronettes
John Lennon, The Harlem Community Choir, The Plastic Ono Band, Yoko
Ono
SZA, Don Toliver
Burl Ives
Nat King Cole
Paul McCartney
Band Aid
SZA, Phoebe Bridgers
Shakin' Stevens
Frank Sinatra, B. Swanson Quartet
SZA, Travis Scott
Bing Crosby, John Scott Trotter & His Orchestra, Ken Darby Singers
Chris Rea
Darlene Love
Ed Sheeran, Elton John
Perry Como, The Fontane Sisters, Mitchell Ayres & His Orchestra
Britney Spears
Mabel Matiz, Mert Demir
RM
Travis Scott, Metro Boomin
Taylor Swift, Lana Del Rey
Dj LK da Escï¿½ï¿½cia, Tchakabum, mc jhenny, M
```

```
Bad Bunny, Jhay Cortez
Gwen Stefani, Blake Shelton
Chencho Corleone, Rauw Alejandro
Bad Bunny, Rauw Alejandro
Chuck Berry
Frank Sinatra
John Legend, Metro Boomin
Maldy, Karol G
Gayle
Lauren Spencer Smith, Lauren Spencer Smith, Lauren Spencer Smith
Cherish, ACRAZE
Adassa, Mauro Castillo, Stephanie Beatriz, Encanto - Cast, Rhenzy
Feliz, Diane Guerrero, Carolina Gaitan
Farruko
Jaymes Young
The Weeknd, Tyler, The Creator
SZA, Doja Cat
The Weeknd, Lil Wayne
Ckay
Jessica Darrow
The Weeknd, Post Malone
Mï¿½ï¿½ne
Aitana, zzoilo
Julieta Venegas, Bad Bunny, Tainy
Lisa
Tiï¿½ï¿½sto, Ava
The Weeknd, Swedish House Mafia
Juan Cruz Toledo, Huilen Toledo
BoyWithUke
Xamï¿½ï¿½, Gustah, Neo B
Lit Killah, Maria Becerra, Tiago pzk, NICKI NICOLE
Emmy Meli
Justin Quiles, Lenny Tavï¿½ï¿½rez, BL
Shawn Mendes
Kodak Black
Aventura, Bad Bunny
Rauw Alejandro
Ckay, AX'EL, Dj Yo!
Bruno Mars, Anderson .Paak, Silk Sonic
THE ANXIETY, Willow, Tyler Cole
Sebastian Yatra
Justin Bieber, Daniel Caesar, Giveon
Bizarrap, Tiago pzk
Stromae
Chris Brown, Rvssian, Rauw Alejandro
NEIKED, Mae Muller, Polo G
Oliver Tree
Kali Uchis, Amaarae, Moliy
Young Thug, Future, Gunna
```

```
Doja Cat, The Weeknd
Jnr Choi
Trueno, Tiago pzk
Sech, Bad Bunny, Mora
Drake, Project Pat, 21 Savage
Tiï¿½ï¿½sto, Kar
Edison Lighthouse
The Weeknd, ROSALï¿½
Olga Merediz, Stephanie Beatriz, Encanto - Cast
Surf Curse
Jason Derulo
21 Savage, Gunna
24kgoldn, Iann Dior
Stephanie Beatriz, Diane Guerrero
Trueno
Kanye West
Playboi Carti
Maluma
Ed Sheeran, Fireboy DML
Prezioso, Gabry Ponte, LUM!X
Tiï¿½ï¿
Marï¿½ï¿½lia Mendonï¿½ï¿½a, George Henrique &
TWICE
Giveon
Sean Paul, Dua Lipa
Charli XCX, Jax Jones, Joel Corry, Saweetie
SALES
Sleepy hallow
Masked Wolf
Anuel Aa, Jhay Cortez
NIKI
Tate McRae
Avicii
Drake, Future, Young Thug
Imagine Dragons, League of Legends, Arcane
Tones and I
Juice WRLD
J Balvin, Maria Becerra
John Legend
Nirvana
Duncan Laurence
Drake, Travis Scott
Tini, L-Gante
SiM
Kendrick Lamar, Baby Keem
The Killers
J Balvin, Nio Garcia, Bad Bunny
Lady Gaga, Bradley Cooper
Marshmello, Jonas Brothers
```

```
Riton, Nightcrawlers, Mufasa & Hypeman, Dopamine
Polo G
Eminem, Nate Dogg
Maroon 5
Arcangel, De La Ghetto, Justin Quiles, Lenny Tavï¿½ï¿½rez, Sech,
Dalex, Dimelo Flow, Rich Music
Gunna, Lil Baby
Katy Perry, Alesso
Marï¿½ï¿½lia Mendonï¿½ï¿½a, Maiara &
Tini, Maria Becerra
Anuel Aa, Myke Towers, Jhay Cortez
Karol G, Becky G
Dr. Dre, Snoop Dogg
Dove Cameron
Ed Sheeran, Taylor Swift
Charlie Puth
Kanye West, Alicia Keys, Fivio Foreign
Mahmood, Blanco
50 Cent
Yung Lean
Labrinth, Zendaya
The Rare Occasions
Intense, AP Dhillon, Gurinder Gill
Quevedo, La Pantera, Juseph, Cruz Cafunï¿½ï¿½, Bï¿½ï¿½jo, Abhir Hathi
Dr. Dre, 2Pac, Roger
Eminem, Dr. Dre
Luude, Colin Hay
Ryan Castro
Kendrick Lamar
Eminem, Dido
GODZZ__-, Zakaria
Chase Atlantic
Em Beihold
Mainstreet, Chefin
Alvaro Diaz, Rauw Alejandro
Nicki Minaj, Lil Baby
Future
Zï¿½ï¿½ Fe
Muni Long
Vundabar
V
Marï¿½ï¿½lia Mendonï¿½ï¿½a, Hugo & G
Shawn Mendes, Camila Cabello
Daddy Yankee, Bad Bunny
Dua Lipa, Megan Thee Stallion
Camila Cabello, Ed Sheeran
Cris Mj
Anitta
Dave
Tulus
```

```
Mc Pedrinho, Pedro Sampaio
Zion & Lennox
Residente, Bizarrap
Jack Harlow
Eden Muï¿½ï
Treasure
Christian Nodal
Kevin Gates, Juicy J
Tyga, Doja Cat
Queen
YEAT
C. Tangana
The Chainsmokers, Coldplay
The Chainsmokers, Halsey
NMIXX
Gotye, Kimbra
AnnenMayKantereit, Giant Rooks
Paulo Londra
Danny Ocean
Lil Tjay
Willow
Feid, Alejo, Robi
Demi Lovato
Camila Cabello, Willow
Matuï¿½ï¿½, Wiu,
De La Ghetto, Duki, Quevedo
Lil Baby
Ak4:20, Cris Mj, Pailita
Yahritza Y Su Esencia
Luï¿½ï¿½sa Sonza, MC Frog, Dj Gabriel do Borel, Davi K
Ed Sheeran, J Balvin
teto
Blackbear, BoyWithUke
Pharrell Williams, Nile Rodgers, Daft Punk
Jordan Fisher, Josh Levi, Finneas O'Connell, 4*TOWN (From Disney and
Pixarï¿½ï¿½ï¿½s Turning Red), Topher Ngo, Grayson Vill
BIGBANG
Kendrick Lamar, Blxst, Amanda Reifer
Kendrick Lamar, Sampha
Post Malone, Roddy Ricch
Kendrick Lamar, Kodak Black
Tony Dize, Bad Bunny
Buscabulla, Bad Bunny
Kendrick Lamar, Taylour Paige
Kendrick Lamar, Sam Dew, Baby Keem
Bad Bunny, The Marï¿½ï
Kendrick Lamar, Ghostface Killah, Summer Walker
Polima WestCoast, Pailita
Bad Bunny, Tainy
```

```
Panic! At The Disco
Kendrick Lamar, Tanna Leone
PSY, Suga
Benson Boone
KALUSH
Luciano
Bizarrap, Paulo Londra
Shae Gill, Ali Sethi
Kendrick Lamar, Beth Gibbons
Sech, Mora
MNEK, Jax Jones
Ugly Dray, Tesla Jnr
Justin Bieber, Don Toliver
Jessi
Chanel
Leah Kate
Sam Ryder
HA SUNG WOON, Jimin
The Goo Goo Dolls
Kevin Kaarl
Suki Waterhouse
Lauren Spencer Smith
Pharrell Williams, Tyler, The Creator, 21 Savage
Calvin Harris, Dua Lipa, Young Thug
Nengo Flow, Anuel Aa, Chris Jedi, Chencho Corleone
Kanye West, XXXTENTACION
Musical Youth
Bizarrap, Villano Antillano
Halsey
BYOR, Imanbek
Sleepy hallow, 347aidan
Paloma Faith
Marshmello, Khalid
Ozuna, Tiago pzk
Childish Gambino
David Guetta, Ella Henderson, Becky Hill
Duki
Sidhu Moose Wala
Keisya Levronka
Victor Cibrian
Metallica
De La Ghetto, Feid, Polima WestCoast, Paloma Mami, Pailita
Yung Gravy
Lasso
Nayeon
Schï¿½ï¿½rze, DJ R
j-hope
Guns N' Roses
Burna Boy
```

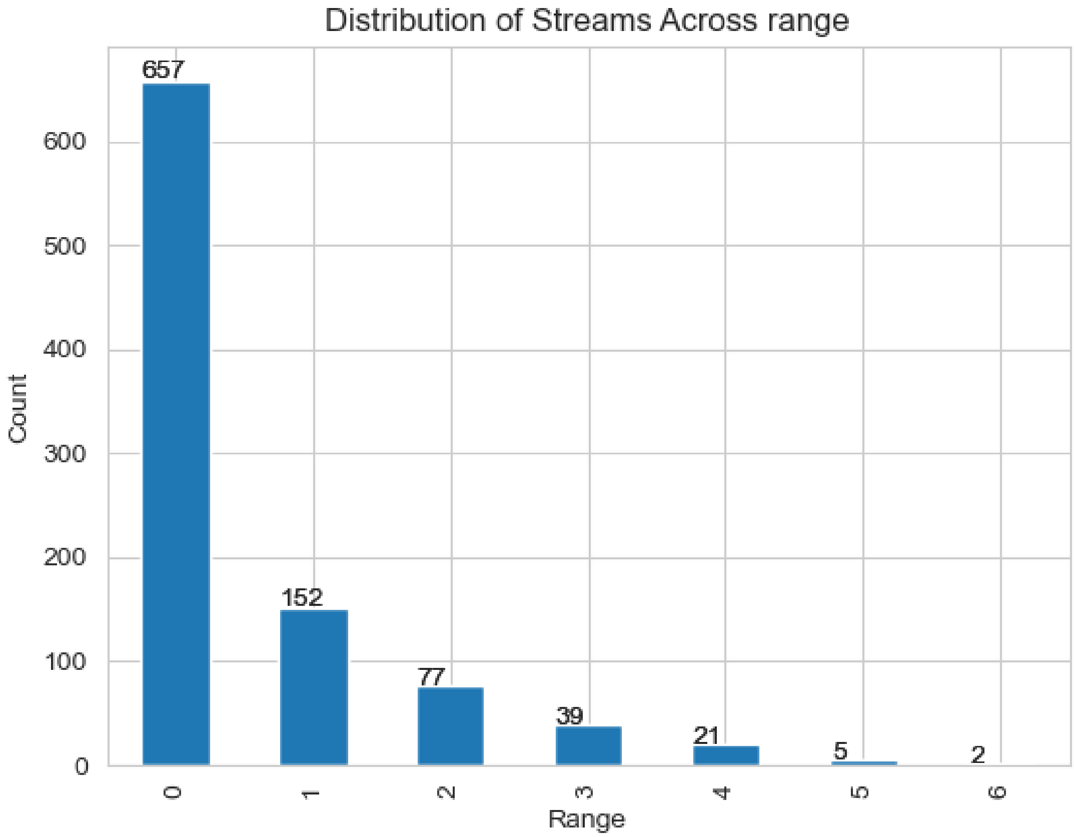
```
Kanye West, Lil Durk, Cardi B
L7nnon, DJ Biel do Furduncinho, Bianca
Anitta, Tini, Becky G
Nessa Barrett
LF System
Mc Vitin Da Igrejinha, MC Tairon, DJ Win
Rels B
Snoop Dogg, BTS, Benny Blanco
Drake, DJ Khaled, Lil Baby
Luar La L
Ghost
Rex Orange County
Sofia Carson
Grupo Marca Registrada, Junior H
Calvin Harris, Halsey, Pharrell Williams, Justin Timberlake
Peso Pluma, Luis R Conriquez
Luis R Conriquez, La Adictiva
David Guetta, Shakira, Black Eyed Peas
Melody, Ana Castela, Dj Chris No Beat
Elton John, Britney Spears
Interplanetary Criminal, Eliza Rose
Armani White
Oxlade
PnB Rock
Ovy On The Drums, Quevedo
Lil Yachty
Marshmello, Juice WRLD
Fran C, Polima WestCoast, Nickoog Clk, Pablito Pesadilla
Southstar
girl in red
Rosa Walton, Hallie Coggins
Luciano, Aitch, Bï¿½
MC Ryan SP, Love Funk, Mc Paiva ZS
Drake, Travis Scott, 21 Savage
Ludwig Goransson, Foudeqush
Jin
Selena Gomez
Feid, Paulo Londra
Feid, Sech, Jhayco
Total number of artists: 645
```

```python
Data['streams'] = pd.to_numeric(Data['streams'],
errors='coerce').fillna(0).astype(int)

Data['in_deezer_playlists'] =
pd.to_numeric(Data['in_deezer_playlists'],
errors='coerce').fillna(0).astype(int)

Data['in_shazam_charts'] = pd.to_numeric(Data['in_shazam_charts'],
errors='coerce').fillna(0).astype(int)
```

```
Data['key'] = pd.to_numeric(Data['key'],
errors='coerce').fillna(0).astype(int)

Data['artist(s)_name'] = pd.to_numeric(Data['artist(s)_name'],
errors='coerce').fillna(0).astype(int)

bins = 7
Data['streams_range'] = pd.cut(Data['streams'], bins, labels=False)

max_value = Data['streams'].max()
bins = 7
bin_edges = np.linspace(0, max_value, bins + 1)
Data['streams_range'] = pd.cut(Data['streams'], bins=bin_edges,
labels=False)
new_bin_edges = pd.cut(Data['streams'], bins=bin_edges).cat.categories
for edge in new_bin_edges:
    print(f"Bin range: {edge.left} to {edge.right}")
```

```
Bin range: 0.0 to 529127867.714
Bin range: 529127867.714 to 1058255735.429
Bin range: 1058255735.429 to 1587383603.143
Bin range: 1587383603.143 to 2116511470.857
Bin range: 2116511470.857 to 2645639338.571
Bin range: 2645639338.571 to 3174767206.286
Bin range: 3174767206.286 to 3703895074.0
```

```
#std_dev = Data['streams'].std()
#bin_edges = np.arange(Data['streams'].min(), Data['streams'].max() +
std_dev, std_dev)

# Create a new column with bin labels
#labels = ['streams_sd__0', 'streams_sd_1', 'streams_sd_2',
'streams_sd_3', 'streams_sd_4', 'streams_sd_5', 'streams_sd_6']
#labels = list(range(7, -1, -1))
#Data['streams_range'] = pd.cut(Data['streams'], bins=bin_edges,
labels=False)

# Count the occurrences in each bin
#bin_counts = Data['streams_range'].value_counts().sort_index()

Data['streams_range'].value_counts().plot(kind='bar')
ax =
Data['streams_range'].value_counts().sort_index().plot(kind='bar')
for p in ax.patches:
    ax.annotate(str(p.get_height()), (p.get_x() * 1.005,
p.get_height() * 1.005))
plt.xlabel('Range')
plt.ylabel('Count')
plt.title('Distribution of Streams Across range')
plt.figure(figsize=(20,15))
plt.show()
```



```
<Figure size 2000x1500 with 0 Axes>

Data.info()
Data = Data.dropna(axis=1, how='all')

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 953 entries, 0 to 952
Data columns (total 25 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   track_name            953 non-null    object
 1   artist(s)_name        953 non-null    int64
 2   artist_count          953 non-null    int64
 3   released_year         953 non-null    int64
 4   released_month        953 non-null    int64
 5   released_day          953 non-null    int64
 6   in_spotify_playlists  953 non-null    int64
 7   in_spotify_charts     953 non-null    int64
 8   streams               953 non-null    int64
 9   in_apple_playlists    953 non-null    int64
 10  in_apple_charts       953 non-null    int64
 11  in_deezer_playlists   953 non-null    int64
 12  in_deezer_charts      953 non-null    int64
 13  in_shazam_charts      953 non-null    int64
 14  bpm                   953 non-null    int64
 15  key                   953 non-null    int64
 16  mode                  953 non-null    int64
 17  danceability_%        953 non-null    int64
 18  valence_%             953 non-null    int64
 19  energy_%              953 non-null    int64
 20  acousticness_%        953 non-null    int64
 21  instrumentalness_%    953 non-null    int64
 22  liveness_%            953 non-null    int64
 23  speechiness_%         953 non-null    int64
 24  streams_range         953 non-null    int64
dtypes: int64(24), object(1)
memory usage: 186.3+ KB
```

```
Data = Data.drop(['track_name'], axis=1)

# Backward selection
X = Data.drop(['streams', 'streams_range'], axis=1)
y = Data['streams_range']

X = sm.add_constant(X)

p_values_dict = {}


while True:
    model = sm.OLS(y, X).fit()
    p_values = model.pvalues.drop('const')
    max_p_value = p_values.max()

    if max_p_value > 0.05:
        feature_to_remove = p_values.idxmax()
        p_values_dict[feature_to_remove] = p_values[feature_to_remove]
        X = X.drop(feature_to_remove, axis=1)
    else:
        break

selected_features = X.columns.tolist()
print("Selected Features:", selected_features)

print("Columns and their p-values:")
for col in p_values_dict:
    print(f"{col}: {p_values_dict[col]}")
```

```
Selected Features: ['const', 'artist(s)_name', 'released_year',
'in_spotify_playlists', 'in_spotify_charts', 'in_apple_playlists',
'in_deezer_playlists', 'in_deezer_charts', 'in_shazam_charts', 'key',
'valence_%', 'acousticness_%']
Columns and their p-values:
liveness_%: 0.992686023541067
mode: 0.9204383445605774
```

```
released_month: 0.8552747485618422
bpm: 0.6687624171216515
speechiness_%: 0.5860408462133606
energy_%: 0.5368390339044422
danceability_%: 0.4936508719392937
instrumentalness_%: 0.22882560800703433
released_day: 0.24235413468782666
in_apple_charts: 0.148316368842711
artist_count: 0.058450810034350834
```

```
#KNN without Backward selection

X = Data.drop(['streams', 'streams_range'], axis=1)
y = Data['streams_range'].values
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y,
test_size=0.3, random_state=0)
classifier = KNeighborsClassifier(n_neighbors=5)
classifier.fit(X_train, y_train)
y_pred = classifier.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
conf_matrix = confusion_matrix(y_test, y_pred)
print(f'Accuracy: {accuracy * 100}%')
print('Confusion Matrix:\n', conf_matrix)
```

```
Accuracy: 74.82517482517483%
Confusion Matrix:
 [[196   4   0   0   0   0   0]
 [ 35   9   4   0   0   0   0]
 [  9   4   5   1   0   0   0]
 [  1   3   2   3   1   0   0]
 [  1   1   1   2   1   0   0]
 [  0   0   0   0   2   0   0]
 [  0   0   0   0   1   0   0]]
```

```
# KNN with Backward selection
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import accuracy_score, confusion_matrix

feature_columns = ['artist(s)_name', 'artist_count', 'released_year',
'in_spotify_playlists', 'in_spotify_charts',
                   'in_apple_playlists', 'in_deezer_playlists',
                   'in_deezer_charts', 'in_shazam_charts', 'key',
'valence_%', 'acousticness_%']

X = Data[feature_columns].values
```

```
y = Data['streams_range'].values
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y,
test_size=0.3, random_state=0)
classifier = KNeighborsClassifier(n_neighbors=5)
classifier.fit(X_train, y_train)
y_pred = classifier.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
conf_matrix = confusion_matrix(y_test, y_pred)
print(f'Accuracy: {accuracy * 100}%')
print('Confusion Matrix:\n', conf_matrix)
```

```
Accuracy: 76.57342657342657%
Confusion Matrix:
 [[197   3   0   0   0   0   0]
 [ 34  10   4   0   0   0   0]
 [  4   8   5   2   0   0   0]
 [  1   2   2   5   0   0   0]
 [  0   0   3   1   2   0   0]
 [  0   0   0   1   1   0   0]
 [  0   0   0   0   1   0   0]]
```

```
k_list = list(range(1,30,2))
cv_scores = []

for k in k_list:
    knn = KNeighborsClassifier(n_neighbors=k)
    scores = cross_val_score(knn, X_train, y_train, cv=10,
scoring='accuracy')
    cv_scores.append(scores.mean())
```

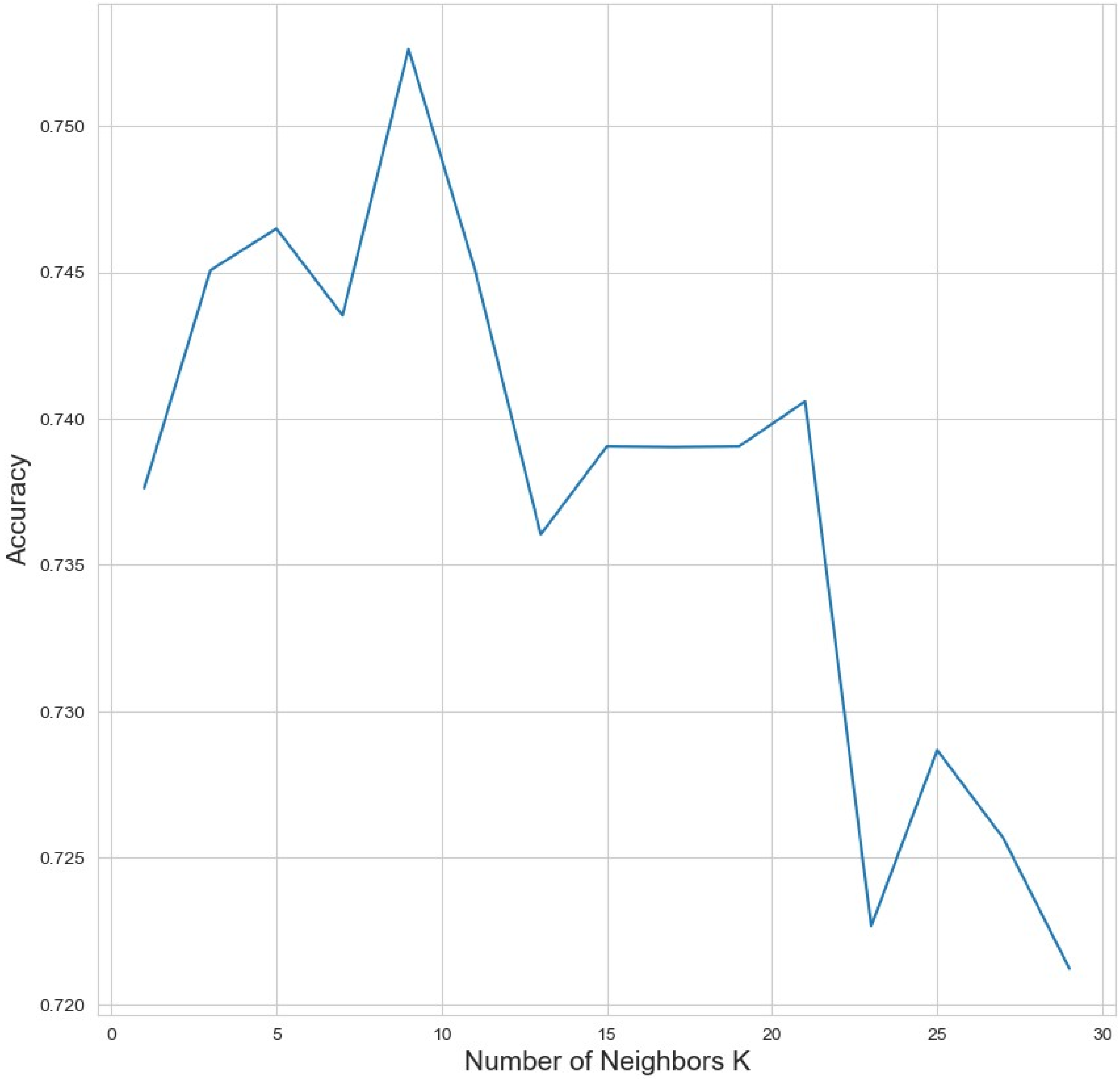
```
/Users/marco9951qq.com/anaconda3/lib/python3.11/site-packages/
sklearn/model_selection/_split.py:725: UserWarning: The least
populated class in y has only 1 members, which is less than
n_splits=10.
  warnings.warn(
/Users/marco9951qq.com/anaconda3/lib/python3.11/site-packages/sklearn/
model_selection/_split.py:725: UserWarning: The least populated class
in y has only 1 members, which is less than n_splits=10.
  warnings.warn(
/Users/marco9951qq.com/anaconda3/lib/python3.11/site-packages/sklearn/
model_selection/_split.py:725: UserWarning: The least populated class
in y has only 1 members, which is less than n_splits=10.
  warnings.warn(
/Users/marco9951qq.com/anaconda3/lib/python3.11/site-packages/sklearn/
model_selection/_split.py:725: UserWarning: The least populated class
in y has only 1 members, which is less than n_splits=10.
  warnings.warn(
/Users/marco9951qq.com/anaconda3/lib/python3.11/site-packages/sklearn/
```

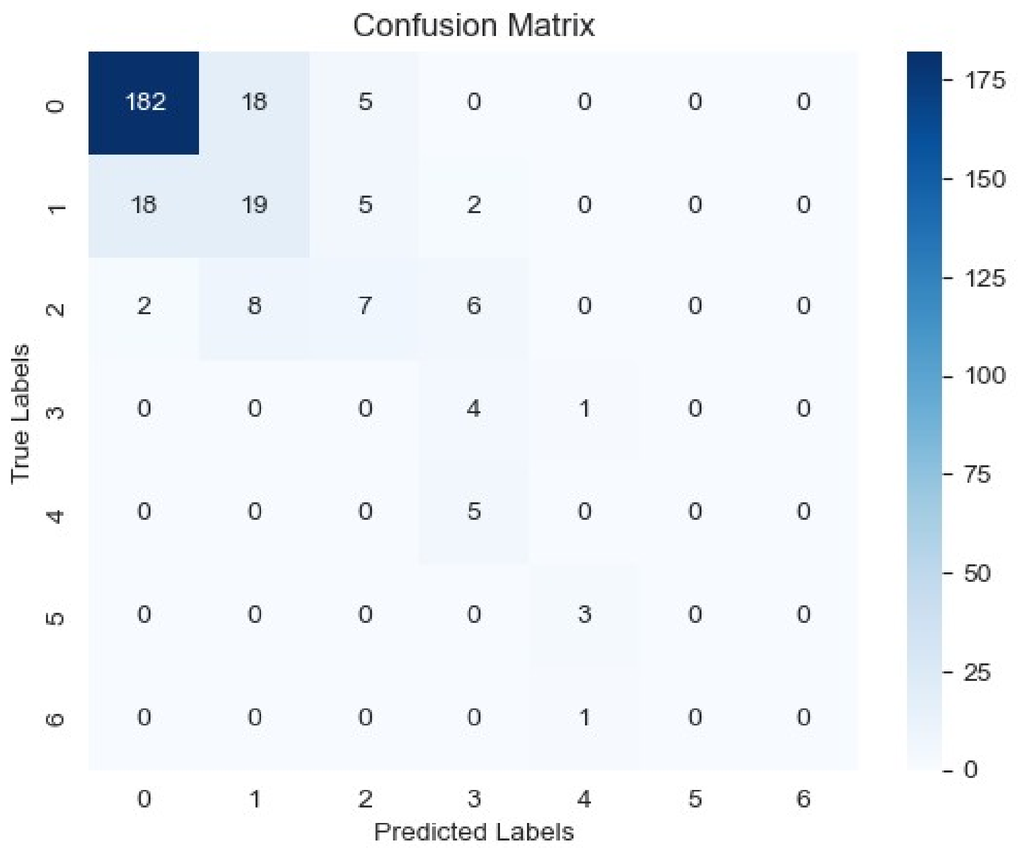
```
model_selection/_split.py:725: UserWarning: The least populated class
in y has only 1 members, which is less than n_splits=10.
  warnings.warn(
/Users/marco9951qq.com/anaconda3/lib/python3.11/site-packages/sklearn/
model_selection/_split.py:725: UserWarning: The least populated class
in y has only 1 members, which is less than n_splits=10.
  warnings.warn(
/Users/marco9951qq.com/anaconda3/lib/python3.11/site-packages/sklearn/
model_selection/_split.py:725: UserWarning: The least populated class
in y has only 1 members, which is less than n_splits=10.
  warnings.warn(
/Users/marco9951qq.com/anaconda3/lib/python3.11/site-packages/sklearn/
model_selection/_split.py:725: UserWarning: The least populated class
in y has only 1 members, which is less than n_splits=10.
  warnings.warn(
/Users/marco9951qq.com/anaconda3/lib/python3.11/site-packages/sklearn/
model_selection/_split.py:725: UserWarning: The least populated class
in y has only 1 members, which is less than n_splits=10.
  warnings.warn(
/Users/marco9951qq.com/anaconda3/lib/python3.11/site-packages/sklearn/
model_selection/_split.py:725: UserWarning: The least populated class
in y has only 1 members, which is less than n_splits=10.
  warnings.warn(
/Users/marco9951qq.com/anaconda3/lib/python3.11/site-packages/sklearn/
model_selection/_split.py:725: UserWarning: The least populated class
in y has only 1 members, which is less than n_splits=10.
  warnings.warn(
/Users/marco9951qq.com/anaconda3/lib/python3.11/site-packages/sklearn/
model_selection/_split.py:725: UserWarning: The least populated class
in y has only 1 members, which is less than n_splits=10.
  warnings.warn(
/Users/marco9951qq.com/anaconda3/lib/python3.11/site-packages/sklearn/
model_selection/_split.py:725: UserWarning: The least populated class
in y has only 1 members, which is less than n_splits=10.
  warnings.warn(
/Users/marco9951qq.com/anaconda3/lib/python3.11/site-packages/sklearn/
model_selection/_split.py:725: UserWarning: The least populated class
in y has only 1 members, which is less than n_splits=10.
  warnings.warn(
/Users/marco9951qq.com/anaconda3/lib/python3.11/site-packages/sklearn/
model_selection/_split.py:725: UserWarning: The least populated class
in y has only 1 members, which is less than n_splits=10.
  warnings.warn(
```

```
plt.figure()
plt.figure(figsize=(10,10))
plt.xlabel('Number of Neighbors K', fontsize=15)
plt.ylabel('Accuracy', fontsize=15)
sns.set_style("whitegrid")
```

```
plt.plot(k_list, cv_scores)
plt.show()

conf_matrix = confusion_matrix(y_test, y_pred)
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues')
plt.title('Confusion Matrix')
plt.xlabel('Predicted Labels')
plt.ylabel('True Labels')
plt.show()
```

```
<Figure size 640x480 with 0 Axes>
```



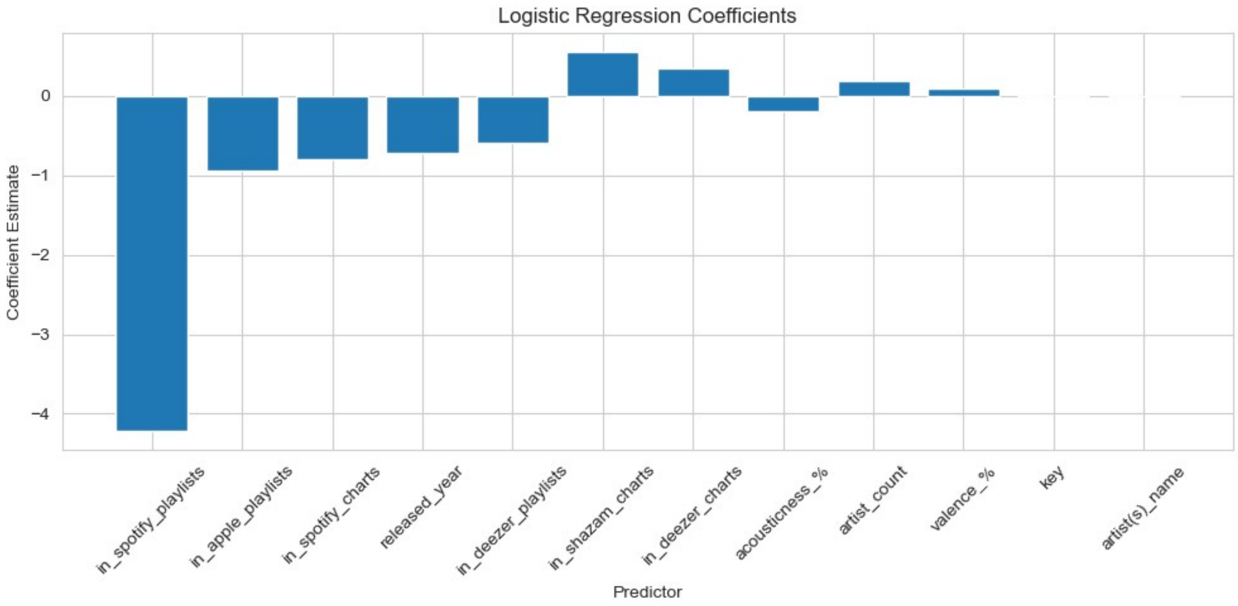


```
#Logistic Regression
from sklearn.metrics import confusion_matrix, accuracy_score
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression

#feature_columns = ['artist(s)_name', 'artist_count', 'released_year',
'in_spotify_playlists', 'in_spotify_charts',
                   #'in_apple_playlists', 'in_deezer_playlists',
                   #'in_deezer_charts', 'in_shazam_charts', 'key',
'valence_%', 'acousticness_%']
X = Data.drop(['streams', 'streams_range'], axis=1)
#X = Data[feature_columns].values
y = Data['streams_range']
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)

logreg = LogisticRegression(random_state=16, max_iter=1000)
logreg.fit(X_train_scaled, y_train)

y_pred = logreg.predict(X_test_scaled)

conf_matrix = confusion_matrix(y_test, y_pred)
```

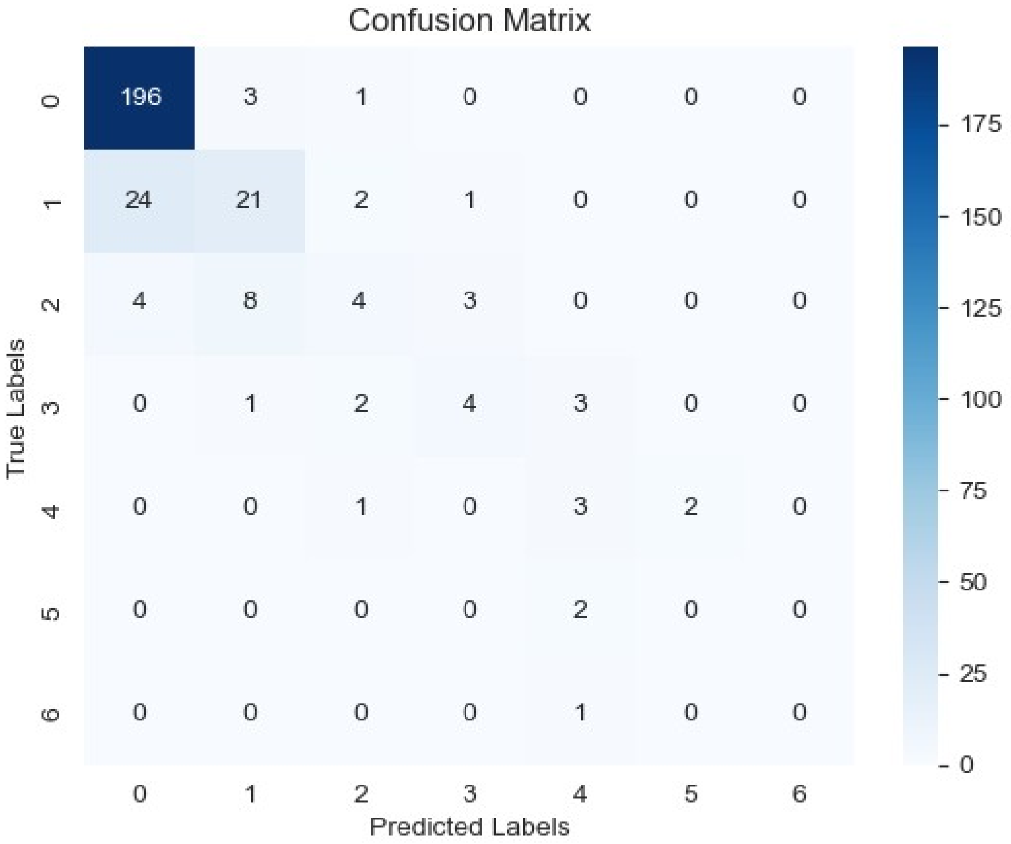
```
print(conf_matrix)

accuracy = accuracy_score(y_test, y_pred)
print(accuracy)
```

```
[[197   8   0   0   0   0   0]
 [ 24  14   6   0   0   0   0]
 [  2  12   7   2   0   0   0]
 [  0   0   0   4   1   0   0]
 [  0   0   1   2   2   0   0]
 [  0   0   0   0   3   0   0]
 [  0   0   0   0   1   0   0]]
0.7832167832167832
```

```
import matplotlib.pyplot as plt
import numpy as np

# Assuming logreg is your trained logistic regression model
coefficients = logreg.coef_[0]  # Get the coefficients from the model
features = feature_columns  # List of feature names
indices = np.argsort(np.abs(coefficients))[::-1]  # Sort by magnitude

# Create a bar chart
plt.figure(figsize=(10, 5))
plt.title('Logistic Regression Coefficients')
plt.bar(range(len(indices)), coefficients[indices], align='center')
plt.xticks(range(len(indices)), np.array(features)[indices],
rotation=45)
plt.xlabel('Predictor')
plt.ylabel('Coefficient Estimate')
plt.tight_layout()
plt.show()
```



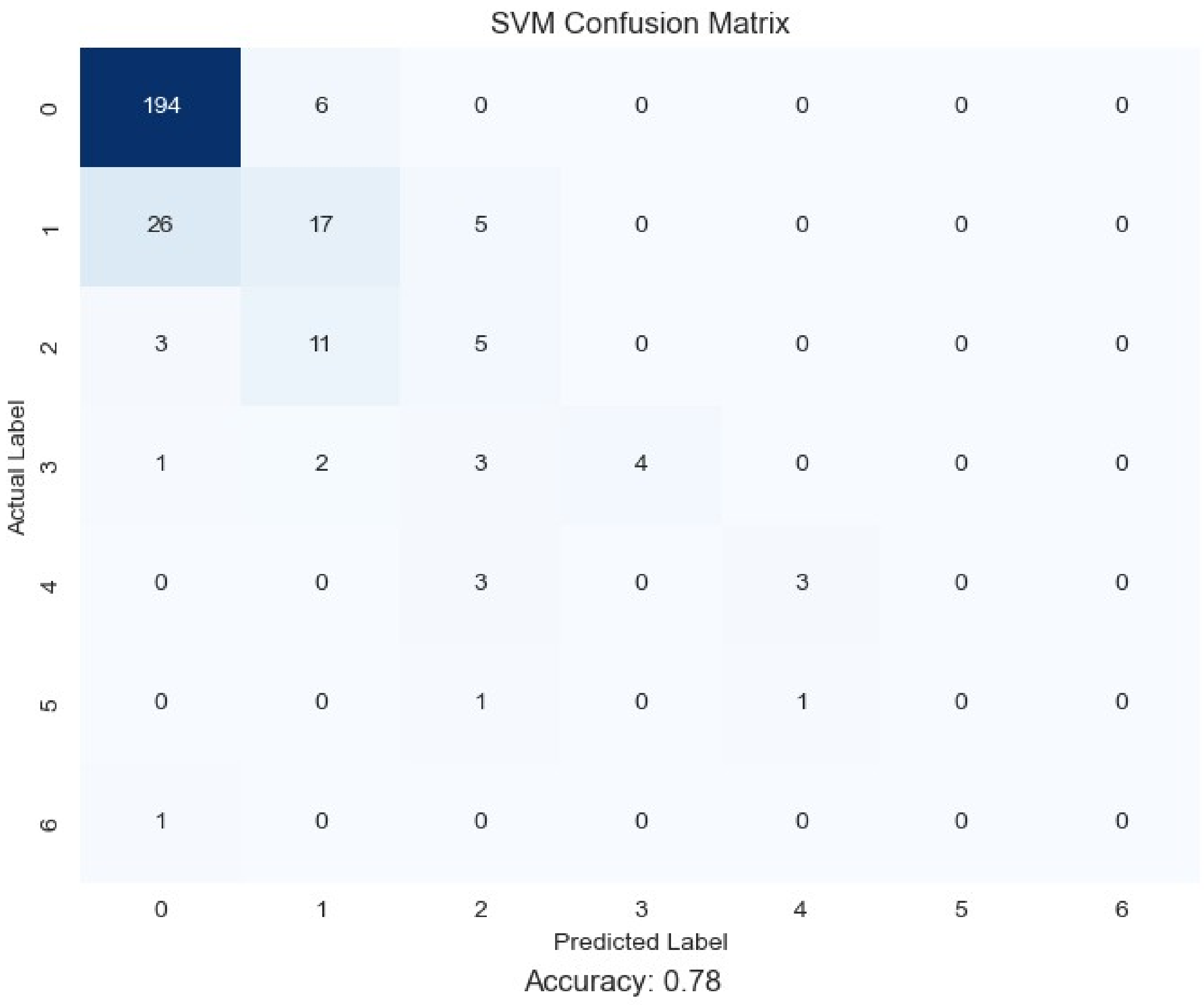
```
cm = confusion_matrix(y_test, y_pred)
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues')
plt.title('Confusion Matrix')
plt.xlabel('Predicted Labels')
plt.ylabel('True Labels')
plt.show()
```



```
#SVM

clf = svm.SVC()
clf.fit(X_train, y_train)
y_pred = clf.predict(X_test)

print(confusion_matrix(y_test,y_pred))
print(classification_report(y_test,y_pred))
```

```
[[194   6   0   0   0   0   0]
 [ 26  17   5   0   0   0   0]
 [  3  11   5   0   0   0   0]
 [  1   2   3   4   0   0   0]
 [  0   0   3   0   3   0   0]
 [  0   0   1   0   1   0   0]
 [  1   0   0   0   0   0   0]]
              precision    recall  f1-score   support

           0       0.86      0.97      0.91       200
           1       0.47      0.35      0.40        48
           2       0.29      0.26      0.28        19
           3       1.00      0.40      0.57        10
           4       0.75      0.50      0.60         6
           5       0.00      0.00      0.00         2
           6       0.00      0.00      0.00         1

    accuracy                           0.78       286
   macro avg       0.48      0.36      0.40       286
weighted avg       0.75      0.78      0.76       286


/Users/marco9951qq.com/anaconda3/lib/python3.11/site-packages/
sklearn/metrics/_classification.py:1469: UndefinedMetricWarning:
Precision and F-score are ill-defined and being set to 0.0 in labels
with no predicted samples. Use `zero_division` parameter to control
this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/Users/marco9951qq.com/anaconda3/lib/python3.11/site-packages/sklearn/
metrics/_classification.py:1469: UndefinedMetricWarning: Precision and
F-score are ill-defined and being set to 0.0 in labels with no
predicted samples. Use `zero_division` parameter to control this
behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/Users/marco9951qq.com/anaconda3/lib/python3.11/site-packages/sklearn/
metrics/_classification.py:1469: UndefinedMetricWarning: Precision and
F-score are ill-defined and being set to 0.0 in labels with no
predicted samples. Use `zero_division` parameter to control this
behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

```python
y_pred = clf.predict(X_test)
cm = confusion_matrix(y_test, y_pred)
plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', cbar=False)
plt.ylabel('Actual Label')
plt.xlabel('Predicted Label')
plt.title('SVM Confusion Matrix')
accuracy = np.trace(cm) / np.sum(cm).astype('float')
plt.figtext(0.5, 0.01, f'Accuracy: {accuracy:.2f}', ha='center',
fontsize=12)
plt.show()
print(classification_report(y_test, y_pred))
```



```
              precision    recall  f1-score   support

           0       0.86      0.97      0.91       200
           1       0.47      0.35      0.40        48
           2       0.29      0.26      0.28        19
           3       1.00      0.40      0.57        10
           4       0.75      0.50      0.60         6
           5       0.00      0.00      0.00         2
           6       0.00      0.00      0.00         1

    accuracy                           0.78       286
   macro avg       0.48      0.36      0.40       286
weighted avg       0.75      0.78      0.76       286


/Users/marco9951qq.com/anaconda3/lib/python3.11/site-packages/
sklearn/metrics/_classification.py:1469: UndefinedMetricWarning:
Precision and F-score are ill-defined and being set to 0.0 in labels
with no predicted samples. Use `zero_division` parameter to control
```

```
this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/Users/marco9951qq.com/anaconda3/lib/python3.11/site-packages/sklearn/
metrics/_classification.py:1469: UndefinedMetricWarning: Precision and
F-score are ill-defined and being set to 0.0 in labels with no
predicted samples. Use `zero_division` parameter to control this
behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/Users/marco9951qq.com/anaconda3/lib/python3.11/site-packages/sklearn/
metrics/_classification.py:1469: UndefinedMetricWarning: Precision and
F-score are ill-defined and being set to 0.0 in labels with no
predicted samples. Use `zero_division` parameter to control this
behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

```
#Random Forest

from sklearn.ensemble import RandomForestClassifier
feature_columns = ['artist(s)_name', 'artist_count', 'released_year',
'in_spotify_playlists', 'in_spotify_charts',
                   'in_apple_playlists', 'in_deezer_playlists',
                   'in_deezer_charts', 'in_shazam_charts', 'key',
'valence_%', 'acousticness_%']

X = Data[feature_columns].values
y = Data['streams_range'].values
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.3, random_state=42)

rf = RandomForestClassifier(n_estimators=100, random_state=42)

rf.fit(X_train, y_train)

rf_predictions = rf.predict(X_test)

rf_accuracy = accuracy_score(y_test, rf_predictions)
print(f"Random Forest Accuracy: {rf_accuracy}")
```

```
Random Forest Accuracy: 0.8041958041958042
```

```
#Random Forest with Cross validation

from sklearn.model_selection import train_test_split, cross_val_score


cv_scores = cross_val_score(rf, X, y, cv=5)
print(f"Cross-Validation Scores: {cv_scores}")
print(f"Average CV Score: {cv_scores.mean()}")
```

```
/Users/marco9951qq.com/anaconda3/lib/python3.11/site-packages/
sklearn/model_selection/_split.py:725: UserWarning: The least
```

```
populated class in y has only 2 members, which is less than
n_splits=5.
  warnings.warn(
```

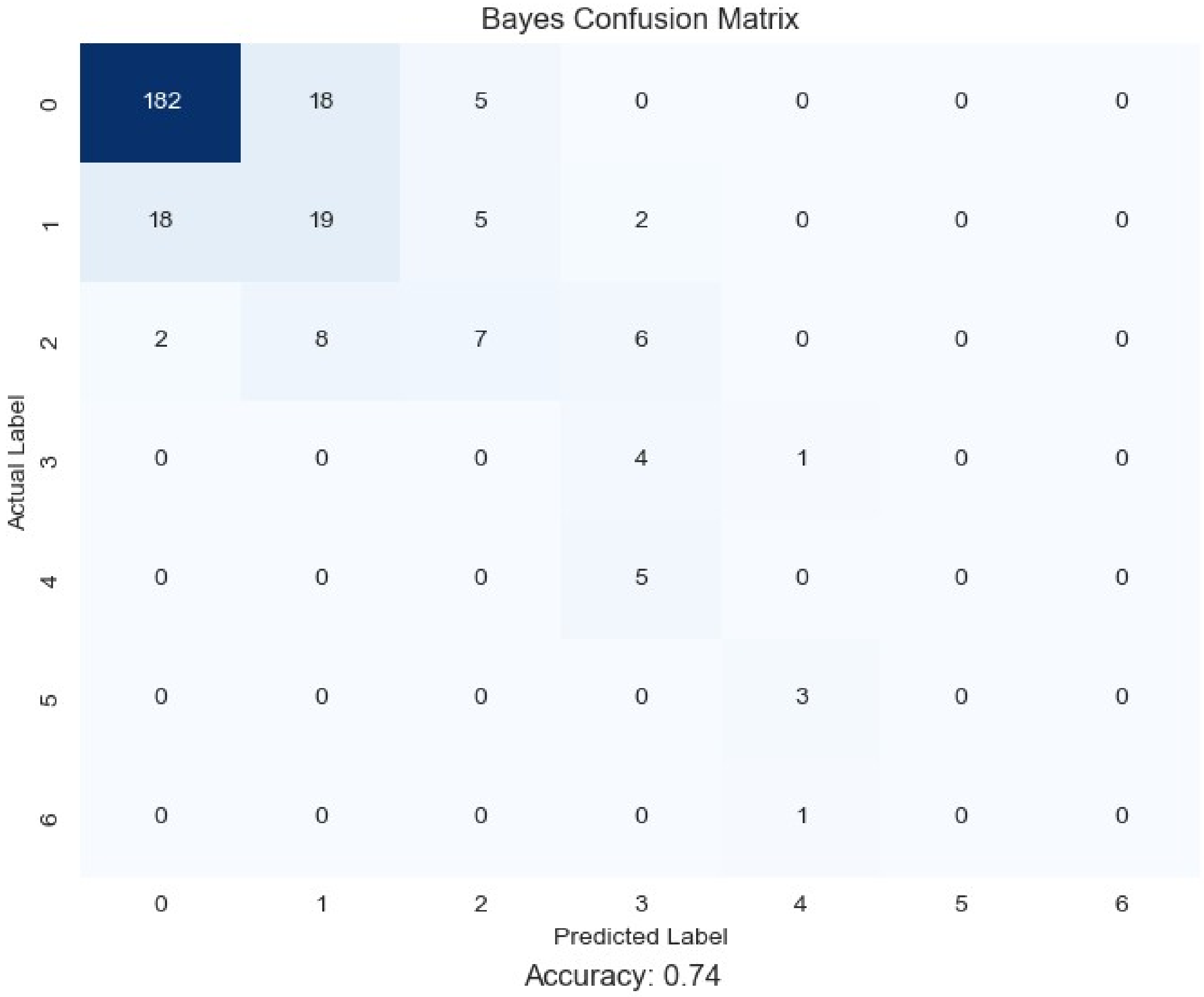
```
Cross-Validation Scores: [0.80628272 0.83246073 0.65968586 0.76315789
0.77368421]
Average CV Score: 0.7670542849269771
```

```
#Feature Importance
import numpy as np

importances = rf.feature_importances_

indices = np.argsort(importances)
plt.title('Feature Importances')
plt.barh(range(len(indices)), importances[indices], color='b',
align='center')
plt.yticks(range(len(indices)), [Data.columns[i] for i in indices])
plt.xlabel('Relative Importance')
plt.show()

monthly_streams = Data.groupby('released_month')['streams'].mean()
best_month = monthly_streams.idxmax()
print(f"The best month for release, based on average streams, is:
{best_month}")
```

```
The best month for release, based on average streams, is: 9
```

```
#Naive Bayes
from sklearn.naive_bayes import GaussianNB

model = GaussianNB()

X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.3, random_state=42)

model.fit(X_train, y_train);

from sklearn.metrics import (
    accuracy_score,
    confusion_matrix,
    ConfusionMatrixDisplay,
    f1_score,
    classification_report,
)

y_pred = model.predict(X_test)

accuray = accuracy_score(y_pred, y_test)
f1 = f1_score(y_pred, y_test, average="weighted")
```

```
print("Accuracy:", accuray)
print("F1 Score:", f1)


Accuracy: 0.7412587412587412
F1 Score: 0.7412719510012666


cm = confusion_matrix(y_test, y_pred)
plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', cbar=False)
plt.ylabel('Actual Label')
plt.xlabel('Predicted Label')
plt.title('Bayes Confusion Matrix')
accuracy = np.trace(cm) / np.sum(cm).astype('float')
plt.figtext(0.5, 0.01, f'Accuracy: {accuracy:.2f}', ha='center',
fontsize=12)
plt.show()
print(classification_report(y_test, y_pred))
```



	0	1	2	3	4	5	6
0	182	18	5	0	0	0	0
1	18	19	5	2	0	0	0
2	2	8	7	6	0	0	0
3	0	0	0	4	1	0	0
4	0	0	0	5	0	0	0
5	0	0	0	0	3	0	0
6	0	0	0	0	1	0	0

```
              precision    recall  f1-score   support

           0       0.90      0.89      0.89       205
           1       0.42      0.43      0.43        44
           2       0.41      0.30      0.35        23
           3       0.24      0.80      0.36         5
           4       0.00      0.00      0.00         5
           5       0.00      0.00      0.00         3
           6       0.00      0.00      0.00         1

    accuracy                           0.74       286
   macro avg       0.28      0.35      0.29       286
weighted avg       0.75      0.74      0.74       286


/Users/marco9951qq.com/anaconda3/lib/python3.11/site-packages/
sklearn/metrics/_classification.py:1469: UndefinedMetricWarning:
Precision and F-score are ill-defined and being set to 0.0 in labels
with no predicted samples. Use `zero_division` parameter to control
this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/Users/marco9951qq.com/anaconda3/lib/python3.11/site-packages/sklearn/
metrics/_classification.py:1469: UndefinedMetricWarning: Precision and
F-score are ill-defined and being set to 0.0 in labels with no
predicted samples. Use `zero_division` parameter to control this
behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/Users/marco9951qq.com/anaconda3/lib/python3.11/site-packages/sklearn/
metrics/_classification.py:1469: UndefinedMetricWarning: Precision and
F-score are ill-defined and being set to 0.0 in labels with no
predicted samples. Use `zero_division` parameter to control this
behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

```
# labels = [ "streams_sd 1","streams_sd 2","streams_sd 3","streams_sd
4","streams_sd 5","streams_sd 6", "streams_sd 7"]
# cm = confusion_matrix(y_test, y_pred)
# disp =
ConfusionMatrixDisplay(confusion_matrix=cm ,display_labels=labels)

# disp.plot()
# plt.xticks(rotation=45)
```