Importing Liberaries

```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
import seaborn as sns
import plotly.express as px
import matplotlib.pyplot as plt


#loading dataset
df = pd.read_csv('spotify_songs.csv')
```

Analysing the data set

```
df.shape
```

```
(32833, 23)
```

```
df.head()
```

	track_id	track_name	track_artist	track_popularity	track_album_id	track_album_name	track_album_rel
0	6f807x0ima9a1j3VPbc7VN	I Don't Care (with Justin Bieber) - Loud Luxur...	Ed Sheeran	66	2oCs0DGTsRO98Gh5ZSl2Cx	I Don't Care (with Justin Bieber) [Loud Luxury...	2
1	0r7CVbZTWZgbTCYdfa2P31	Memories - Dillon Francis Remix	Maroon 5	67	63rPSO264uRjW1X5E6cWv6	Memories (Dillon Francis Remix)	2
2	1z1Hg7Vb0AhHDiEmnDE79l	All the Time - Don Diablo Remix	Zara Larsson	70	1HoSmj2eLcsrR0vE9gThr4	All the Time (Don Diablo Remix)	2
3	75FpbthrwQmzHlBJLuGdC7	Call You Mine - Keanu Silva Remix	The Chainsmokers	60	1nqYsOef1yKKuGOVchbsk6	Call You Mine - The Remixes	2
4	1e8PAfcKUYoKkxPhrHqw4x	Someone You Loved - Future Humans Remix	Lewis Capaldi	69	7m7vv9wlQ4i0LFuJiE2zsQ	Someone You Loved (Future Humans Remix)	2

5 rows × 23 columns

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 32833 entries, 0 to 32832
Data columns (total 23 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   track_id                  32833 non-null  object
 1   track_name                32828 non-null  object
 2   track_artist              32828 non-null  object
 3   track_popularity          32833 non-null  int64
 4   track_album_id            32833 non-null  object
 5   track_album_name          32828 non-null  object
 6   track_album_release_date  32833 non-null  object
 7   playlist_name             32833 non-null  object
 8   playlist_id               32833 non-null  object
 9   playlist_genre            32833 non-null  object
 10  playlist_subgenre         32833 non-null  object
 11  danceability              32833 non-null  float64
 12  energy                    32833 non-null  float64
 13  key                       32833 non-null  int64
 14  loudness                  32833 non-null  float64
 15  mode                      32833 non-null  int64
 16  speechiness               32833 non-null  float64
 17  acousticness              32833 non-null  float64
 18  instrumentalness          32833 non-null  float64
 19  liveness                  32833 non-null  float64
 20  valence                   32833 non-null  float64
 21  tempo                     32833 non-null  float64
 22  duration_ms               32833 non-null  int64
```

```
dtypes: float64(9), int64(4), object(10)
memory usage: 5.8+ MB
```

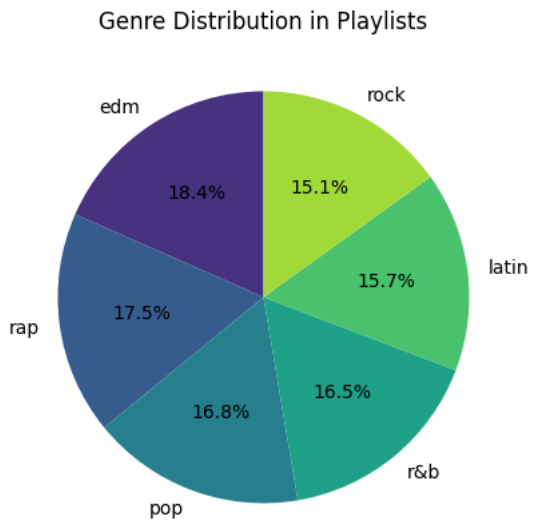
```
df.describe()
```

	track_popularity	danceability	energy	key	loudness	mode	speechiness	acousticness	instrumenta
count	32833.000000	32833.000000	32833.000000	32833.000000	32833.000000	32833.000000	32833.000000	32833.000000	32833.0
mean	42.477081	0.654850	0.698619	5.374471	-6.719499	0.565711	0.107068	0.175334	0.0
std	24.984074	0.145085	0.180910	3.611657	2.988436	0.495671	0.101314	0.219633	0.2
min	0.000000	0.000000	0.000175	0.000000	-46.448000	0.000000	0.000000	0.000000	0.0
25%	24.000000	0.563000	0.581000	2.000000	-8.171000	0.000000	0.041000	0.015100	0.0
50%	45.000000	0.672000	0.721000	6.000000	-6.166000	1.000000	0.062500	0.080400	0.0
75%	62.000000	0.761000	0.840000	9.000000	-4.645000	1.000000	0.132000	0.255000	0.0
max	100.000000	0.983000	1.000000	11.000000	1.275000	1.000000	0.918000	0.994000	0.9

Comparisons of parameters

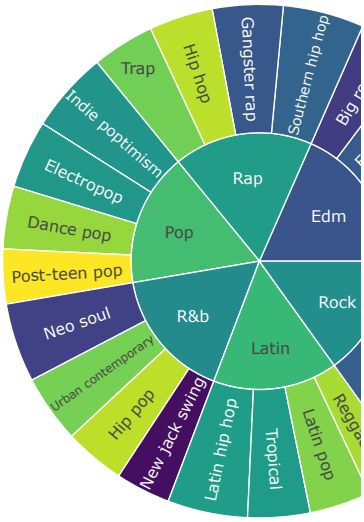
Genre Distribution

```
genre_counts = df['playlist_genre'].value_counts()
plt.figure(figsize=(5, 5))
plt.pie(genre_counts, labels=genre_counts.index, autopct='%1.1f%%', startangle=90, colors=sns.color_palette('viridis'))
plt.title('Genre Distribution in Playlists')
plt.show()
```

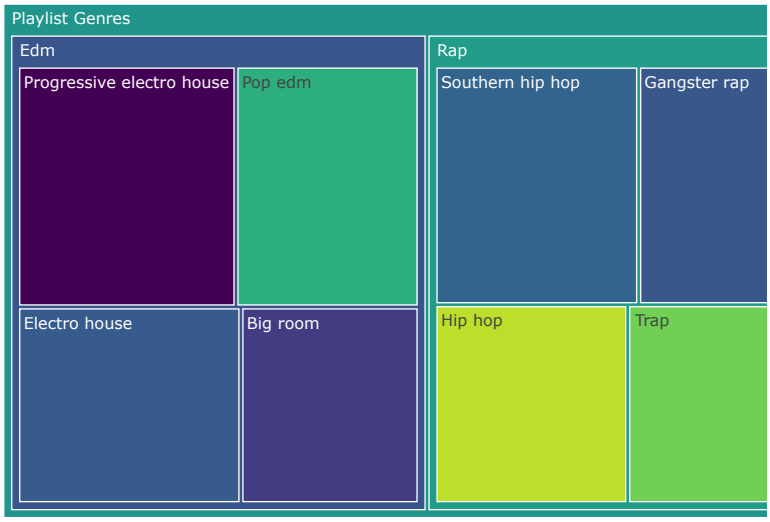


Sub-Genre Distribution in Playlist

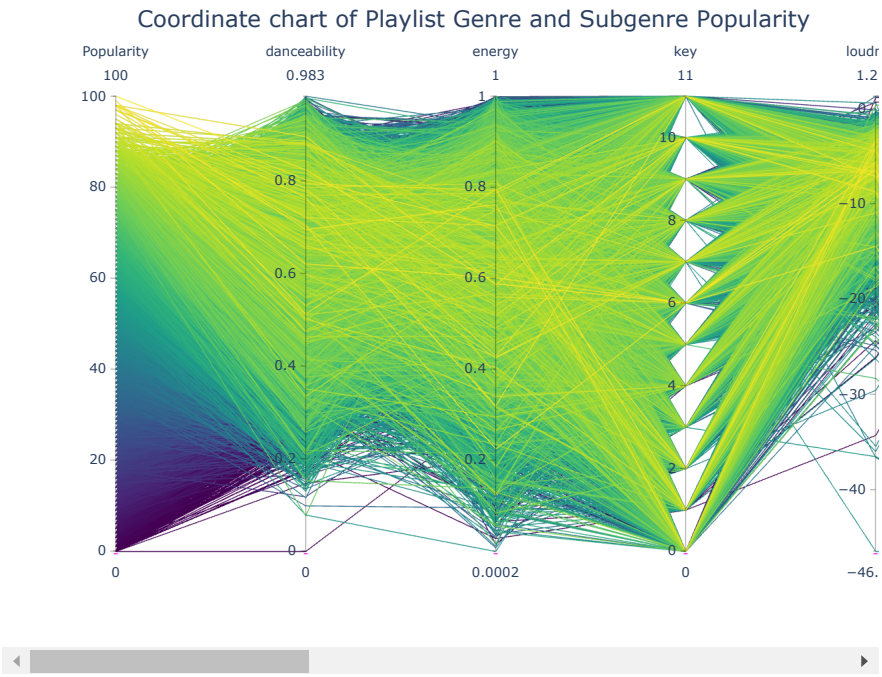
```
df[['playlist_genre', 'playlist_subgenre']] = df[['playlist_genre', 'playlist_subgenre']] \
                                              .apply(lambda x: x.str.capitalize(), axis=1)
fig = px.sunburst(df,
                  path=['playlist_genre', 'playlist_subgenre'],
                  color='track_popularity',
                  color_continuous_scale='viridis',
                  labels={'track_popularity': 'Popularity'})
fig.show()
```



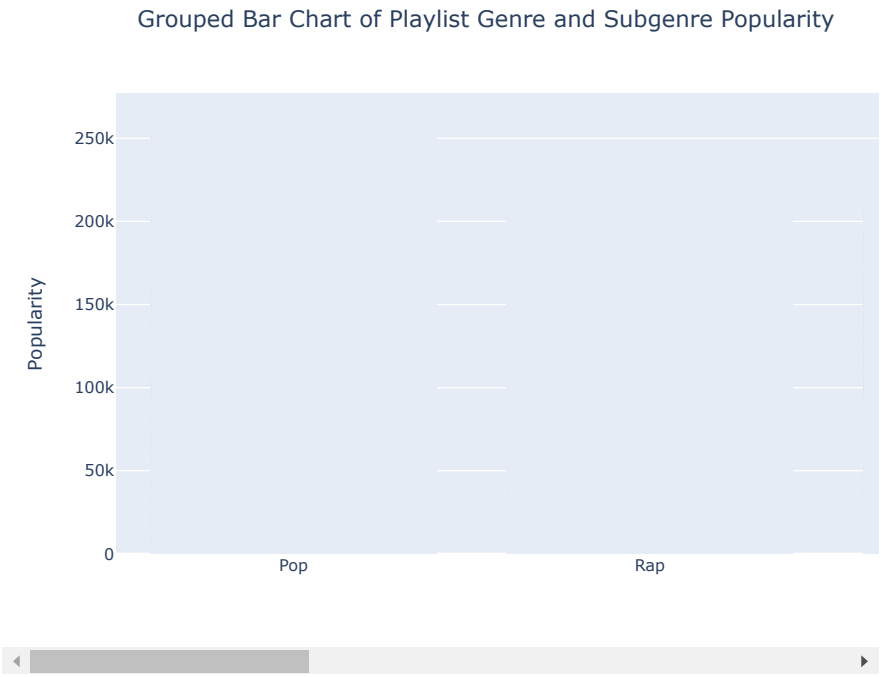

```
fig = px.treemap(df.groupby(['playlist_genre', 'playlist_subgenre']).track_popularity.agg(['count', 'mean']).reset_index(),
                 path=[px.Constant('Playlist Genres'), 'playlist_genre', 'playlist_subgenre'],
                 values='count',
                 color='mean',
                 color_continuous_scale='viridis',
                 labels={'mean': 'Popularity'},
                )
fig.show()
```



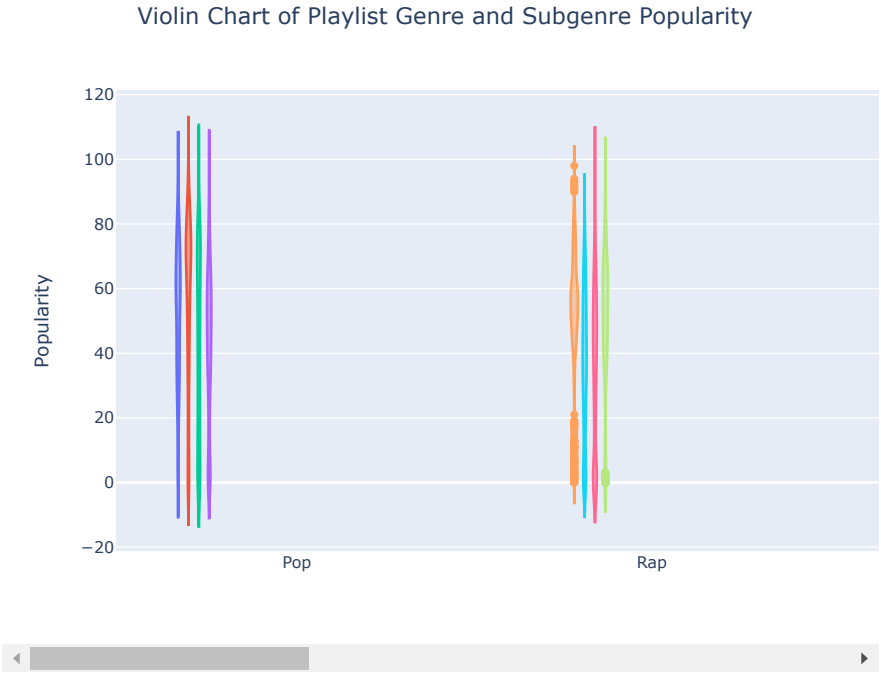
```
# Assuming you have a DataFrame df with columns 'playlist_genre', 'playlist_subgenre', 'track_popularity'
df[['playlist_genre', 'playlist_subgenre']] = df[['playlist_genre', 'playlist_subgenre']] \
    .apply(lambda x: x.str.capitalize(), axis=1)


# Parallel Coordinates Plot
parallel_coordinates_fig = px.parallel_coordinates(df, color='track_popularity',
                                                   color_continuous_scale='viridis',
                                                   title='Coordinate chart of Playlist Genre and Subgenre Popularity',
                                                   labels={'track_popularity': 'Popularity'})


parallel_coordinates_fig.show()
```



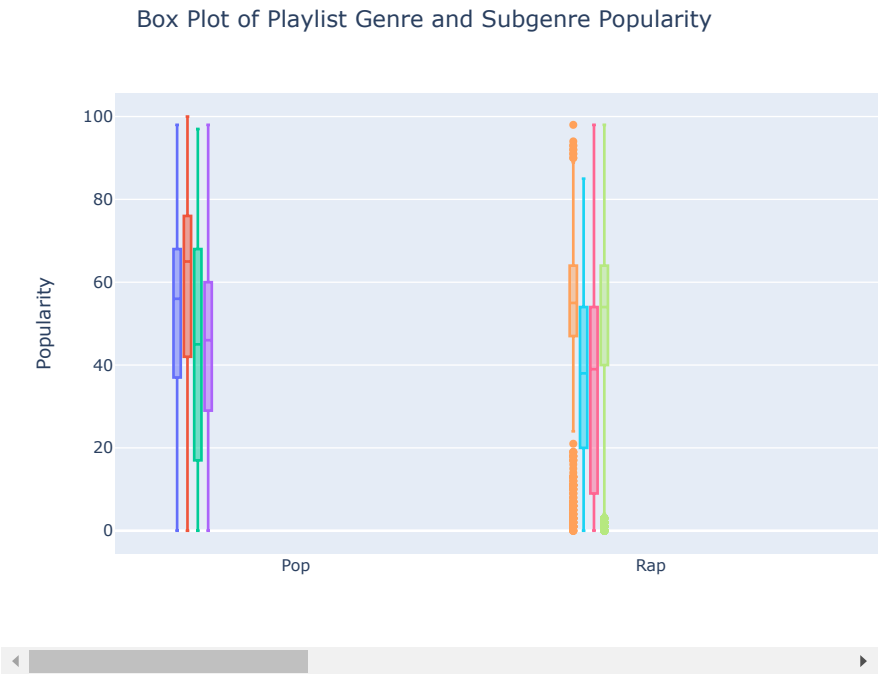

```
# Assuming you have a DataFrame df with the given structure
df[['playlist_genre', 'playlist_subgenre']] = df[['playlist_genre', 'playlist_subgenre']] \
    .apply(lambda x: x.str.capitalize(), axis=1)

# Create a grouped bar chart
fig = px.bar(df, x='playlist_genre', y='track_popularity', color='playlist_subgenre',
             title='Grouped Bar Chart of Playlist Genre and Subgenre Popularity',
             labels={'track_popularity': 'Popularity'})

# Show the grouped bar chart
fig.show()
```




```
# Assuming you have a DataFrame df with the given structure
df[['playlist_genre', 'playlist_subgenre']] = df[['playlist_genre', 'playlist_subgenre']] \
    .apply(lambda x: x.str.capitalize(), axis=1)

# Create a violin chart
fig = px.violin(df, x='playlist_genre', y='track_popularity', color='playlist_subgenre',
                title='Violin Chart of Playlist Genre and Subgenre Popularity',
                labels={'track_popularity': 'Popularity'})

# Show the violin chart
fig.show()
```



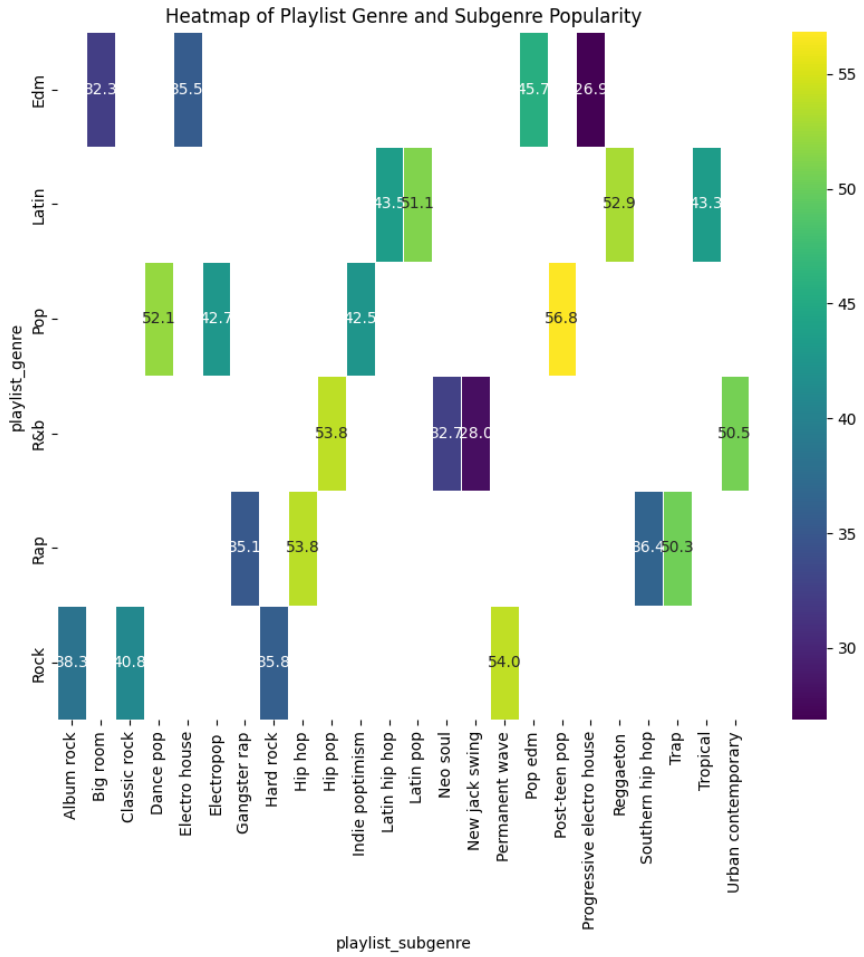
```
# Assuming you have a DataFrame df with the given structure
df[['playlist_genre', 'playlist_subgenre']] = df[['playlist_genre', 'playlist_subgenre']] \
    .apply(lambda x: x.str.capitalize(), axis=1)

# Create a box plot
fig = px.box(df, x='playlist_genre', y='track_popularity', color='playlist_subgenre',
             title='Box Plot of Playlist Genre and Subgenre Popularity',
             labels={'track_popularity': 'Popularity'})

# Show the box plot
fig.show()
```



Artist in sub-Genre Distribution in Playlist

```
fig = px.treemap(df.groupby(['track_artist', 'track_name']).track_popularity.mean().sort_values(ascending=False).reset_index()[:100],
                 path=['track_artist', 'track_name'],
                 values='track_popularity',
                 color_continuous_scale='viridis',
                 title="Artists"
                )
fig.show()
```

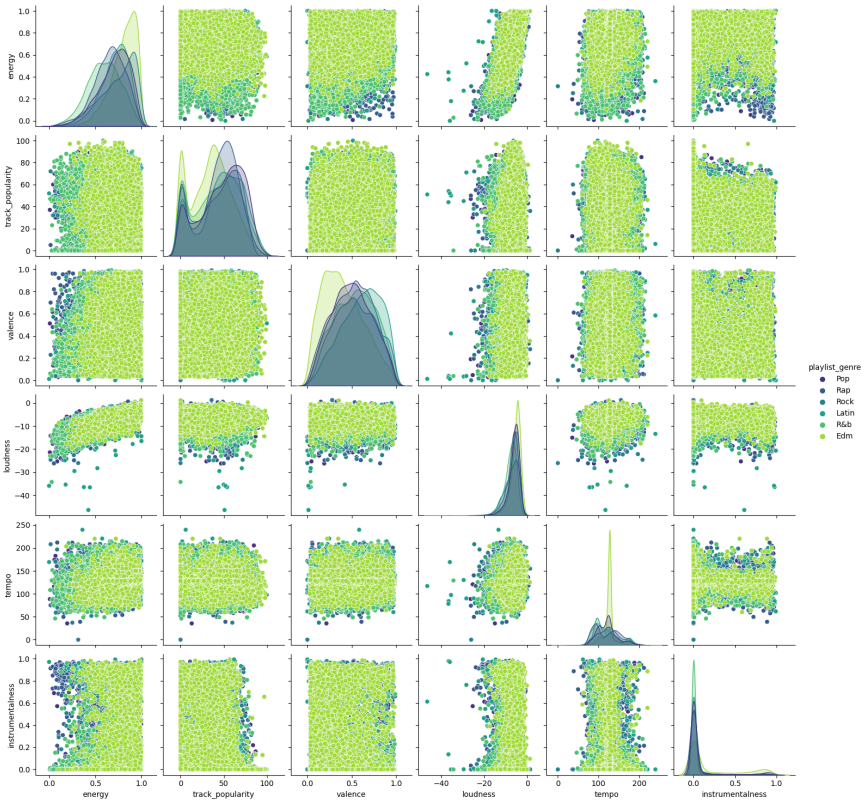
```
# Assuming you have a DataFrame df with the given structure
df[['playlist_genre', 'playlist_subgenre']] = df[['playlist_genre', 'playlist_subgenre']] \
    .apply(lambda x: x.str.capitalize(), axis=1)

# Create a pivot table for the heatmap
heatmap_data = df.pivot_table(index='playlist_genre', columns='playlist_subgenre', values='track_popularity', aggfunc='mean')

# Create a heatmap
plt.figure(figsize=(10, 8))
sns.heatmap(heatmap_data, cmap='viridis', annot=True, fmt=".1f", linewidths=.5)
plt.title('Heatmap of Playlist Genre and Subgenre Popularity')
plt.show()
```



Overall Conclusion

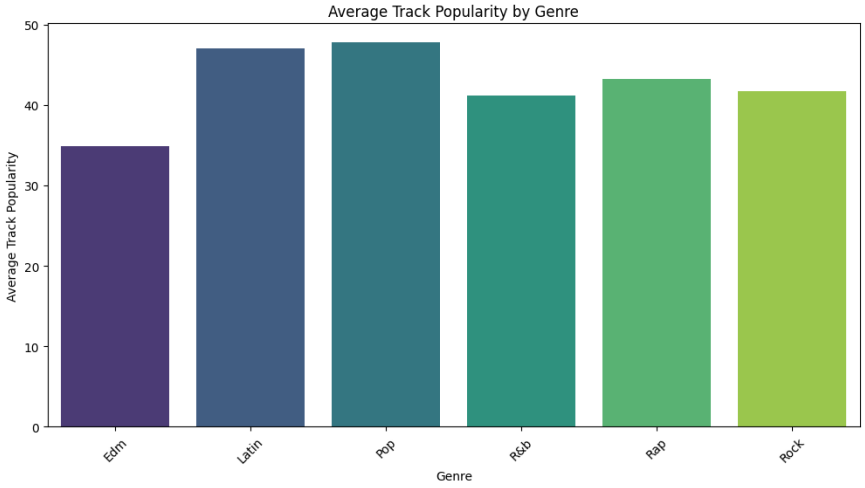
Different parameters compared with genre types

```
sns.pairplot(df[['energy', 'track_popularity', 'valence', 'loudness', 'tempo', 'instrumentalness', 'playlist_genre']], hue='playlist_ge
plt.show()
```



Overall genre popularity

```
average_popularity_by_genre = df.groupby('playlist_genre')['track_popularity'].mean().reset_index()

plt.figure(figsize=(12, 6))
sns.barplot(x='playlist_genre', y='track_popularity', data=average_popularity_by_genre, palette='viridis')
plt.title('Average Track Popularity by Genre')
plt.xlabel('Genre')
plt.ylabel('Average Track Popularity')
plt.xticks(rotation=45)
plt.show()
```



Interpretations