MUKESH PATEL SCHOOL OF TECHNOLOGY MANAGEMENT AND ENGINEERING

Data Extraction and Processing
Report
ON

Analysis of Dataset - Spotify

Under the guidance of Dr. Prachi Natu

N054 - Atharva Kolhe

Introduction

What is Spotify?

Spotify is a popular digital music streaming service that provides access to a vast library of songs, podcasts, and other audio content. It was founded in April 2006 in Sweden and launched to the public in October 2008. Spotify allows users to listen to music from a wide range of artists and genres on demand, and it offers both free and premium subscription options.

Some key features and aspects of Spotify:

1. **Music Catalog:** Spotify has a massive catalog of songs, spanning various genres and languages. Users can search for and listen to individual tracks, albums, or playlists.

2. **Free and Premium Tiers:** Spotify offers a free, ad-supported tier where users can listen to music with occasional advertisements. The premium subscription removes ads, allows for offline downloads, and offers other benefits like higher audio quality and unlimited skips.

3. **Playlists:** Users can create and share their own playlists or follow playlists curated by others. Spotify also offers personalized playlists like "Discover Weekly" and "Release Radar" based on users' listening habits.

4. **Social Sharing:** Spotify integrates with social media platforms, allowing users to share their favorite songs, playlists, and music discoveries with friends.

5. **Podcasts:** In addition to music, Spotify hosts a wide variety of podcasts on numerous topics. It has become a significant player in the podcasting industry, with exclusive content and original shows.

6. **Device Compatibility**: Spotify can be accessed through various devices, including smartphones, tablets, desktop computers, smart speakers, and more. It's available on multiple operating systems, such as iOS, Android, Windows, and macOS.

7. **Personalization:** Spotify uses algorithms to recommend music based on users' listening history, helping them discover new tracks and artists.

8. **Collaborative Playli**sts: Users can create collaborative playlists with friends, allowing multiple people to contribute and edit the playlist.

Spotify has gained widespread popularity for its user-friendly interface, extensive music library, and the convenience it offers in accessing music content on the go. It's available in many countries and has played a significant role in the shift from physical media and downloads to music streaming as the primary way people consume music.

About the Dataset

This dataset contains a comprehensive list of the most famous songs of 2023 as listed on Spotify. The dataset offers a wealth of features beyond what is typically available in similar datasets. It provides insights into each song's attributes, popularity, and presence on various music platforms. The dataset includes information such as track name, artist(s) name, release date, Spotify playlists and charts, streaming statistics, Apple Music presence, Deezer presence, Shazam charts, and various audio features.

Dataset link:-

https://www.kaggle.com/datasets/nelgiriyewithana/top-spotify-songs-2023

Key Features:

- **track_name**: Name of the song
- **artist(s)_name**: Name of the artist(s) of the song
- **artist_count**: Number of artists contributing to the song
- **released_year**: Year when the song was released
- **released_month**: Month when the song was released
- **released_day**: Day of the month when the song was released
- **in_spotify_playlists**: Number of Spotify playlists the song is included in
- **in_spotify_charts**: Presence and rank of the song on Spotify charts
- **streams**: Total number of streams on Spotify
- **in_apple_playlists**: Number of Apple Music playlists the song is included in
- **in_apple_charts**: Presence and rank of the song on Apple Music charts
- **in_deezer_playlists**: Number of Deezer playlists the song is included in
- **in_deezer_charts**: Presence and rank of the song on Deezer charts
- **in_shazam_charts**: Presence and rank of the song on Shazam charts
- **bpm**: Beats per minute, a measure of song tempo
- **key**: Key of the song
- **mode**: Mode of the song (major or minor)
- **danceability_%**: Percentage indicating how suitable the song is for dancing
- **valence_%**: Positivity of the song's musical content

- **energy_%**: Perceived energy level of the song
- **acousticness_%**: Amount of acoustic sound in the song
- **instrumentalness_%**: Amount of instrumental content in the song
- **liveness_%**: Presence of live performance elements
- **speechiness_%**: Amount of spoken words in the song

Libraries Used

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

Getting to know about the Dataset

Reading the dataset

```
data =
pd.read_csv("C:/Users/athar/Downloads/archive/spotify-2023.csv",
encoding="latin-1")
print(data.head())
```

	track_name	artist(s)_name	artist_count	released_year	released_month	released_day
0	Seven (feat. Latto) (Explicit Ver.)	Latto, Jung Kook	2	2023	7	14
1	LALA	Myke Towers	1	2023	3	23
2	vampire	Olivia Rodrigo	1	2023	6	30
3	Cruel Summer	Taylor Swift	1	2019	8	23
4	WHERE SHE GOES	Bad Bunny	1	2023	5	18

in_spotify_playlists	in_spotify_charts	streams	in_apple_playlists	in_apple_charts	in_deezer_playlists	in_deezer_charts
553	147	141381703	43	263	45	10
1474	48	133716286	48	126	58	14
1397	113	140003974	94	207	91	14
7858	100	800840817	116	207	125	12
3133	50	303236322	84	133	87	15

in_shazam_charts	bpm	key	mode	danceability_%	valence_%	energy_%	acousticness_%	instrumentalness_%	liveness_%	speechiness_%
826	125	B	Major	80	89	83	31	0	8	4
382	92	C#	Major	71	61	74	7	0	10	4
949	138	F	Major	51	32	53	17	0	31	6
548	170	A	Major	55	58	72	11	0	11	15
425	144	A	Minor	65	23	80	14	63	11	6

Listing the columns of the dataframe

```
columns = data.columns.tolist()

print("\nColumns in the DataFrame:")

print(columns)
```

Columns in the DataFrame:

['track_name', 'artist(s)_name', 'artist_count', 'released_year', 'released_month', 'released_day', 'in_spotify_playlists', 'in_spotify_charts', 'streams', 'in_apple_playlists', 'in_apple_charts', 'in_deezer_playlists', 'in_deezer_charts', 'in_shazam_charts', 'bpm', 'key', 'mode', 'danceability_%', 'valence_%', 'energy_%', 'acousticness_%', 'instrumentalness_%', 'liveness_%', 'speechiness_%']

Data Cleaning

Detecting missing values

```
# Checking for missing values in each column
missing_values = data.isnull().sum()
print(missing_values)
```

```
track_name              0
artist(s)_name          0
artist_count            0
released_year           0
released_month          0
released_day            0
in_spotify_playlists    0
in_spotify_charts       0
streams                 0
in_apple_playlists      0
in_apple_charts         0
in_deezer_playlists     0
in_deezer_charts        0
in_shazam_charts       50
bpm                     0
key                    95
mode                    0
danceability_%          0
valence_%               0
energy_%                0
acousticness_%          0
instrumentalness_%      0
liveness_%              0
speechiness_%           0
dtype: int64
```

Dealing with missing values

in_shazam_charts:

Analysis: Since this represents a ranking on the Shazam charts, missing values likely indicate that the song didn't achieve a rank.

Our Strategy: We opted to handle the missing data in two ways:

For interpretability, we created a copy of the column and filled missing values with a "Not Ranked" placeholder. This maintains clarity for anyone inspecting the dataset manually.

For machine learning readiness, another copy of the column was made. Here, missing values are replaced with a numerical value (max_rank + 1). This ensures models receive purely numerical input without losing the information that these songs weren't ranked.

```
# Create a copy for human-readable data
data['in_shazam_charts_readable'] = data['in_shazam_charts']

# Remove commas from the original column and convert to float
data['in_shazam_charts'] =
data['in_shazam_charts'].replace({',': ''},
regex=True).astype(float)

# Create another copy for ML processing
data['in_shazam_charts_forML'] = data['in_shazam_charts'].copy()
```

```
# Find the max rank
max_rank = data['in_shazam_charts_forML'].max()

# In the 'readable' version, replace NaN with "Not Ranked"
data['in_shazam_charts_readable'].fillna("Not Ranked",
inplace=True)

# In the 'forML' version, replace NaN with max_rank + 1
data['in_shazam_charts_forML'].fillna(max_rank + 1,
inplace=True)

# Drop the original 'in_shazam_charts' column
data.drop('in_shazam_charts', axis=1, inplace=True)
```

```
data['in_shazam_charts_forML'] =
data['in_shazam_charts_forML'].astype(int)
```

key:

Analysis: This attribute signifies the musical key in which the song is composed. Assigning mean, median, or mode might be inappropriate and potentially misleading unless the missing values occur randomly without a detectable pattern.

Our Strategy: Given the nature of the 'key' attribute and considering its importance in our analysis, we have decided to remove the rows with missing key values to maintain

the integrity and quality of our dataset. This approach ensures that we're working with complete data for each song in our subsequent analyses.

```
# Remove the rows with missing 'key' values
data = data.dropna(subset=['key'])


# Verify if the rows with missing values have been removed
print("Missing values after removal:",
data['key'].isnull().sum())
```

```
Missing values after removal: 0
```

Dealing with datatypes

```
# Inspect current data types
print("Original Data Types:")
print(data.dtypes)
```

```
Original Data Types:
track_name                    object
artist(s)_name                object
artist_count                   int64
released_year                  int64
released_month                 int64
released_day                   int64
in_spotify_playlists           int64
in_spotify_charts              int64
streams                       object
in_apple_playlists             int64
in_apple_charts                int64
in_deezer_playlists           object
in_deezer_charts               int64
bpm                            int64
key                           object
mode                          object
danceability_%                 int64
valence_%                      int64
energy_%                       int64
acousticness_%                 int64
instrumentalness_%             int64
liveness_%                     int64
speechiness_%                  int64
in_shazam_charts_readable     object
in_shazam_charts_forML         int32
dtype: object
```

Combine 'released_year', 'released_month', and 'released_day' into a single datetime column

```
data['release_date'] =
pd.to_datetime(data['released_year'].astype(str) + '-' +


data['released_month'].astype(str) + '-' +


data['released_day'].astype(str), errors='coerce')


# Check the new 'release_date' column
print(data['release_date'].head())
```

```
0   2023-07-14
1   2023-03-23
2   2023-06-30
3   2019-08-23
4   2023-05-18
Name: release_date, dtype: datetime64[ns]
```

Convert 'streams', 'in_deezer_playlists' to appropriate numerical or categorical data types

```
data['streams'] = pd.to_numeric(data['streams'],
errors='coerce')
data['in_deezer_playlists'] =
pd.to_numeric(data['in_deezer_playlists'], errors='coerce')
```

Revised data types

Inspect revised data types

```
print("Revised Data Types:")
print(data.dtypes)
```

```
Revised Data Types:
track_name                          object
artist(s)_name                      object
artist_count                         int64
released_year                        int64
released_month                       int64
released_day                         int64
in_spotify_playlists                 int64
in_spotify_charts                    int64
streams                            float64
in_apple_playlists                   int64
in_apple_charts                      int64
in_deezer_playlists                float64
in_deezer_charts                     int64
bpm                                  int64
key                                 object
mode                                object
danceability_%                       int64
valence_%                            int64
energy_%                             int64
acousticness_%                       int64
instrumentalness_%                   int64
liveness_%                           int64
speechiness_%                        int64
in_shazam_charts_readable           object
in_shazam_charts_forML               int32
release_date                datetime64[ns]
dtype: object
```

Defining Percentage columns

```
percentage_columns = [
   'danceability_%', 'valence_%', 'energy_%',
   'acousticness_%', 'instrumentalness_%', 'liveness_%',
'speechiness_%'
]


for col in percentage_columns:
   print(f"{col}: Min = {data[col].min()}, Max =
{data[col].max()}")
```

```
danceability_%: Min = 23, Max = 96
valence_%: Min = 4, Max = 97
energy_%: Min = 14, Max = 97
acousticness_%: Min = 0, Max = 97
instrumentalness_%: Min = 0, Max = 91
liveness_%: Min = 3, Max = 97
speechiness_%: Min = 2, Max = 64
```

Data Analysis

Summary

```
summary_statistics = data.describe()
print(summary_statistics)
```

	artist_count	released_year	released_month	released_day	in_spotify_playlists	in_spotify_charts
count	857	857	857	857	857	857
mean	2	2018	6	14	5205	12
min	1	1930	1	1	31	0
25%	1	2020	3	5	859	0
50%	1	2022	5	13	2226	3
75%	2	2022	9	22	5542	16
max	8	2023	12	31	52898	147
std	1	11	4	9	7944	19

streams	in_apple_playlists	in_apple_charts	in_deezer_playlists	in_deezer_charts	bpm
857	857	857	782	857	857
513355357	67	51	106	3	123
2762	0	0	0	0	65
139193812	13	7	12	0	100
284908316	34	38	34	0	121
674072710	85	85	110	2	142
3703895074	672	275	965	46	206
571485477	87	50	174	6	28

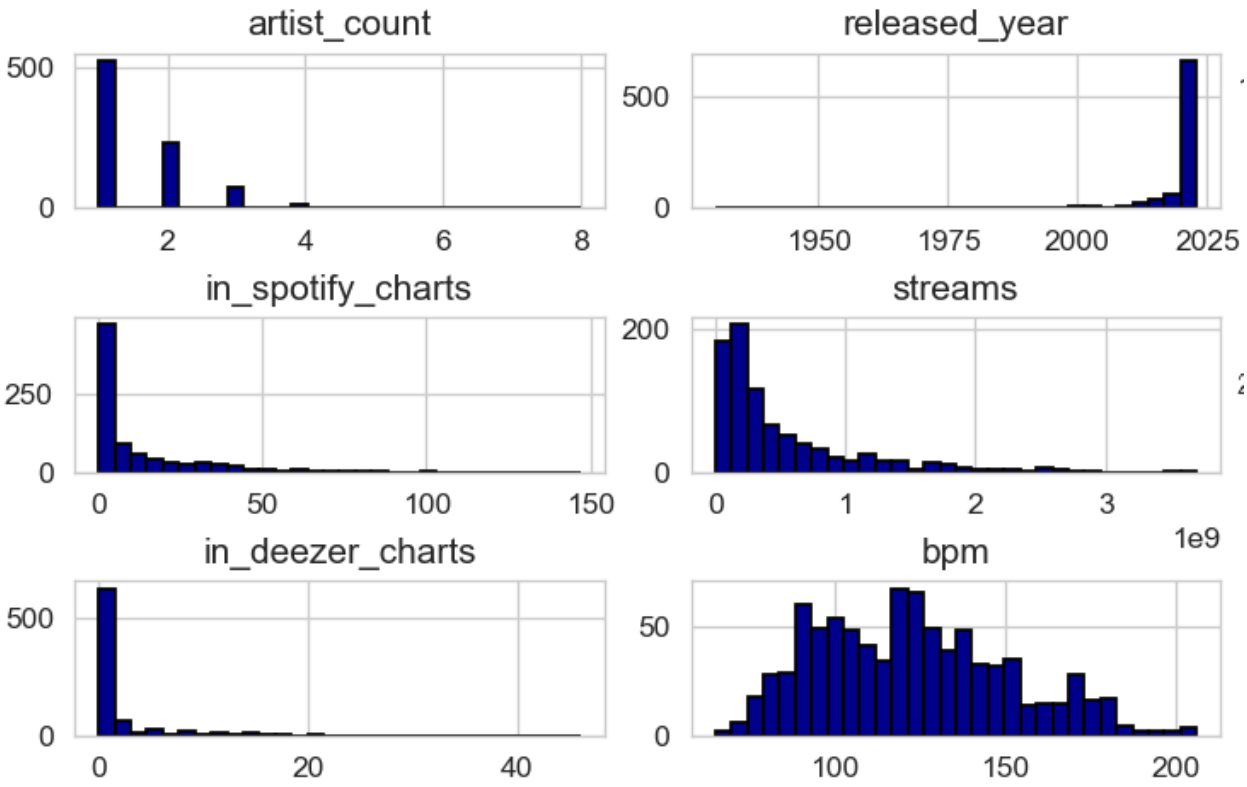
danceability_%	valence_%	energy_%	acousticness_%	instrumentalness_%	liveness_%	speechiness_%
857	857	857	857	857	857	857
67	51	64	27	2	18	10
23	4	14	0	0	3	2
57	32	53	5	0	10	4
70	51	66	17	0	12	6
78	70	76	42	0	24	12
96	97	97	97	91	97	64
15	24	16	26	9	14	10

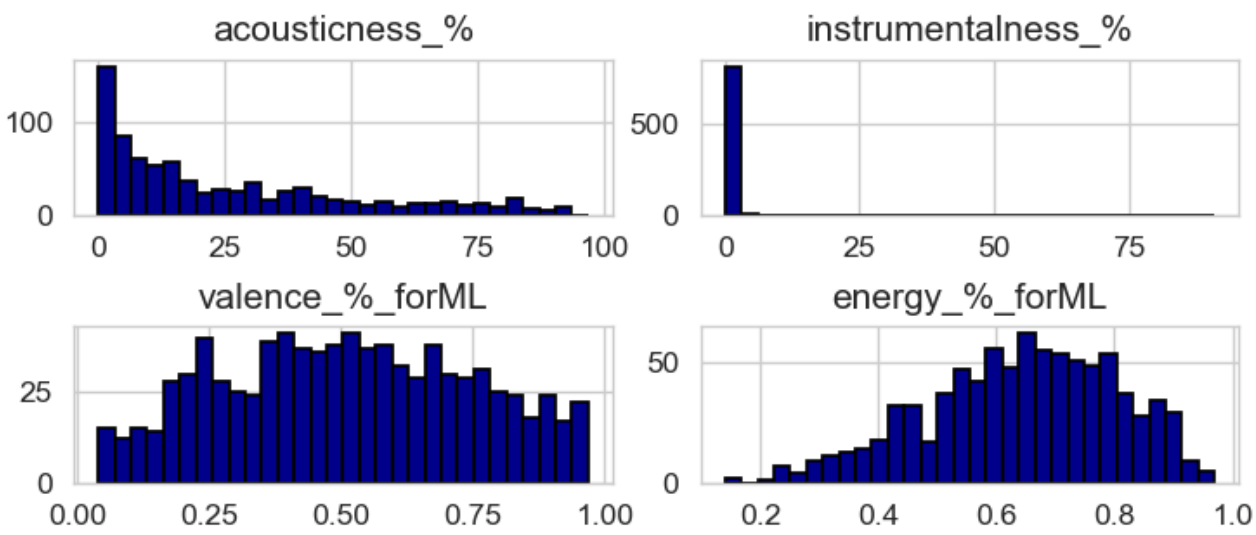
in_shazam_charts_forML	release_date	danceability_%_forML	valence_%_forML	energy_%_forML
857	857	857	857	857
124	2018-10-01 15:14:04.340723200	1	1	1
0	1930-01-01 00:00:00	0	0	0
0	2020-06-28 00:00:00	1	0	1
4	2022-04-08 00:00:00	1	1	1
52	2022-11-04 00:00:00	1	1	1
1452	2023-07-14 00:00:00	1	1	1
333	NaN	0	0	0

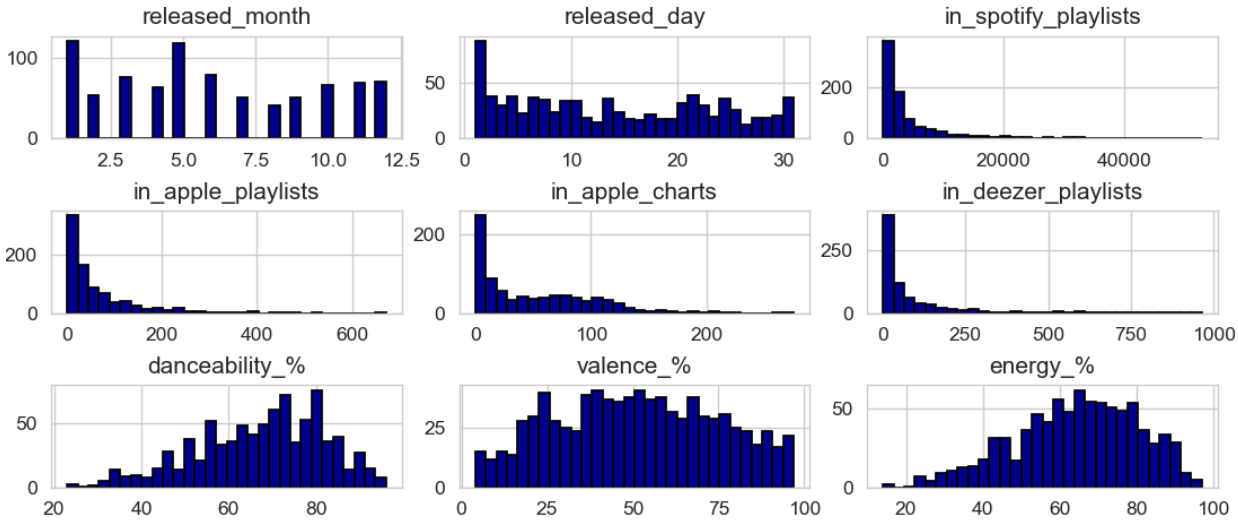
instrumentalness_%_forML	liveness_%_forML	speechiness_%_forML
857	857	857
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
1	1	1
0	0	0

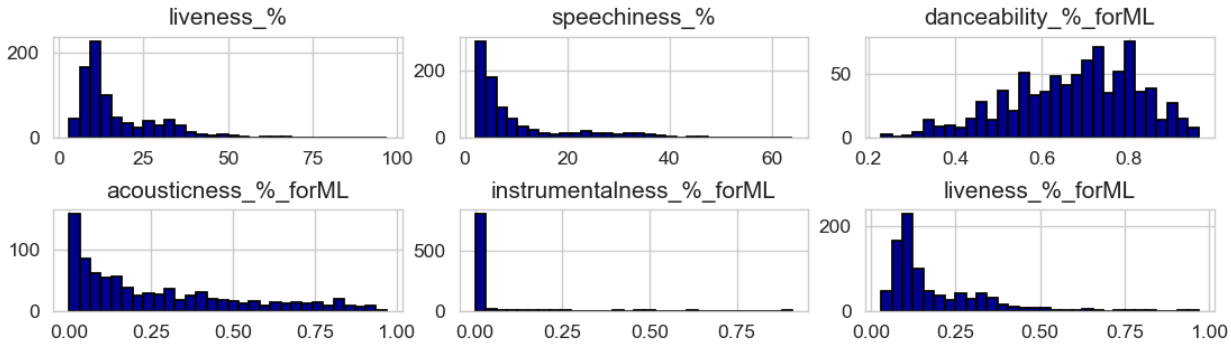
Histogram

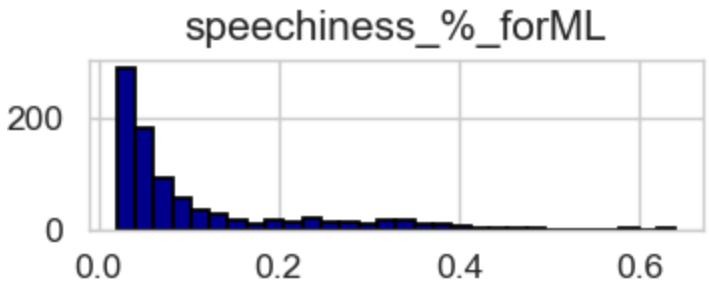
```
# Set style
sns.set_style("whitegrid")

# Plotting histograms for all numerical columns
data.select_dtypes(include=['float64', 'int64']).hist(bins=30,
figsize=(20,20), color='darkblue', edgecolor='black')
plt.tight_layout()
plt.show()
```




acousticness_%
instrumentalness_%
valence_%_forML
energy_%_forML


released_month
released_day
in_spotify_playlists
in_apple_playlists
in_apple_charts
in_deezer_playlists
danceability_%
valence_%
energy_%


liveness_%
speechiness_%
danceability_%_forML
acousticness_%_forML
instrumentalness_%_forML
liveness_%_forML



Based on the histograms, here are some key observations:

artist_count: Most songs are performed by a single artist, but there's a noticeable number of songs with multiple artists as well.

released_year: The bulk of songs in the dataset have been released after 1980, with a very sharp increase in the number of songs released after 2000. The peak seems to occur in the 2020s, suggesting that most of the data pertains to very recent songs.

released_month: January (1) and May (5) appear to be particularly popular months for song releases. This could be due to various reasons such as industry practices or seasonal trends.

released_day: There's a slight peak around the start and middle of the month, but overall, song releases are relatively distributed throughout the month.

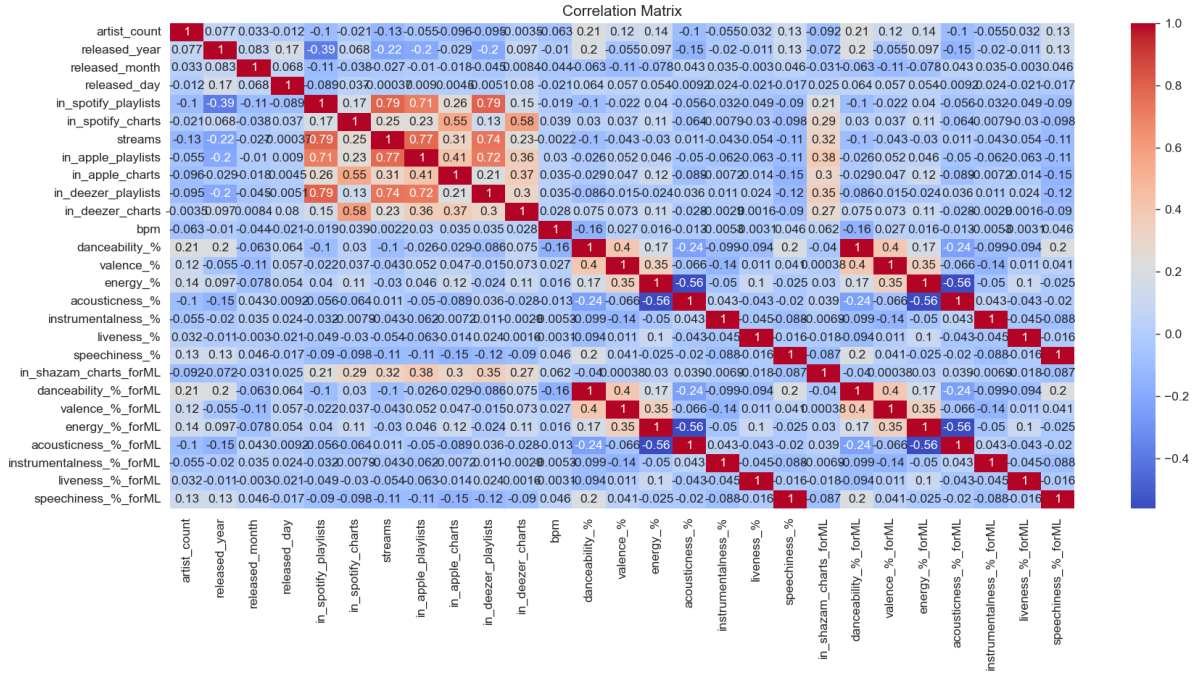
in_playlists and in_charts columns: Most songs are either not present or only appear in a few playlists or charts. However, there's a small subset of songs that seem to dominate in terms of being added to many playlists or making it to the charts.

bpm (beats per minute): Songs range widely in bpm, but there's a prominent peak around 80-100 bpm, indicating that a moderate tempo is quite popular.

Danceability, Valence, Energy, and other % metrics: These attributes display varying levels of skewness, but most of them, like danceability and energy, show a wide distribution indicating diverse musical properties.

Correlation Matrix

```
# Select only numeric columns
numeric_data = data.select_dtypes(include=[np.number])

# Compute the correlation matrix
correlation_matrix = numeric_data.corr()

# Visualize the correlation matrix using a heatmap
plt.figure(figsize=(15, 10))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm')
plt.title("Correlation Matrix")
plt.show()
```



Highly Correlated Values:

danceability% and valence% are positively correlated. This suggests that tracks with high danceability often have high valence, meaning they are typically more upbeat or cheerful.

energy% and valence% also show a positive correlation. Tracks with more energy are likely to be more positive.

Streaming Platforms Consistency:

The presence of a song in playlists for different platforms (in_spotify_playlists, in_apple_playlists, in_deezer_playlists) are moderately correlated, indicating that tracks popular in one platform's playlists are likely to appear in another's.

Charts vs. Playlists:

Being in charts and being in playlists for each platform (e.g., in_spotify_playlists vs. in_spotify_charts) have positive correlations, though not extremely high. This suggests that while songs that make it to the charts are likely to be in playlists, it's not a strict guarantee.
Musical Properties:

bpm (beats per minute) doesn't have strong correlations with most of the other features, implying that tempo isn't a significant driver for a song's presence in playlists or charts.

Data Visualization

Most Streamed Songs

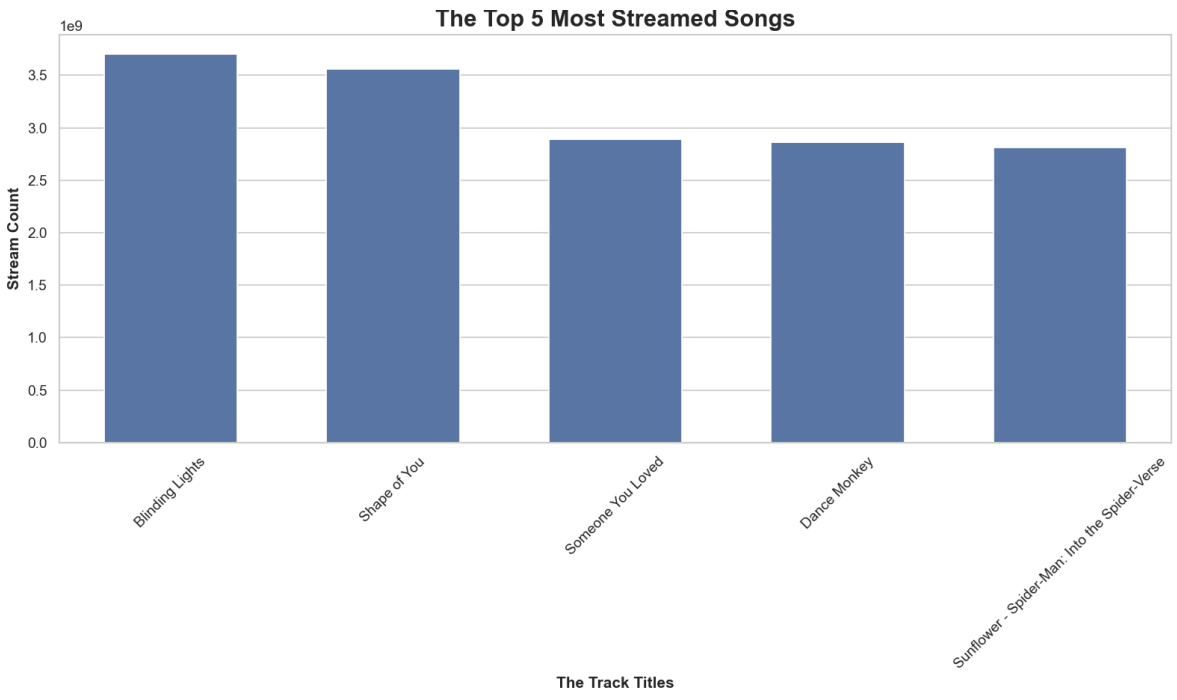
```
most_streamed_songs =
data[['artist(s)_name','track_name','streams']].nlargest(10,'str
eams')
print(most_streamed_songs)
```

	artist(s)_name	track_name	streams
55	The Weeknd	Blinding Lights	3703895074
179	Ed Sheeran	Shape of You	3562543890
86	Lewis Capaldi	Someone You Loved	2887241814
620	Tones and I	Dance Monkey	2864791672
41	Post Malone, Swae Lee	Sunflower - Spider-Man: Into the Spider-Verse	2808096550
162	Drake, WizKid, Kyla	One Dance	2713922350
84	Justin Bieber, The Kid Laroi	STAY (with Justin Bieber)	2665343922
140	Imagine Dragons	Believer	2594040133
725	The Chainsmokers, Halsey	Closer	2591224264
48	The Weeknd, Daft Punk	Starboy	2565529693

Blinding Lights by **the weeknd** is the most streamed song with over 3.7B+ views.

Top 5 most Streamed songs

```
data['streams'] = pd.to_numeric(data['streams'],
errors='coerce')
most_streamed_songs = data[['track_name',
'streams']].sort_values(by='streams', ascending=False).head(5)

plt.figure(figsize=(15, 8))
sns.set(style='whitegrid')

sns.barplot(x='track_name', y='streams',
data=most_streamed_songs,width=0.6)
plt.xlabel('The Track Titles',fontweight='bold')
plt.ylabel('Stream Count',fontweight='bold')
plt.title('The Top 5 Most Streamed Songs',fontsize=18,
fontweight='bold')
plt.xticks(rotation=45)
plt.show()
```



Most Streamed Artist

```
most_streamed_artists =
data.groupby('artist(s)_name')[['streams']].sum().nlargest(10,'s
treams')
print(most_streamed_artists)
```

artist(s)_name	streams
Ed Sheeran	13908947204
Taylor Swift	11851151082
The Weeknd	10069328661
Bad Bunny	8582384095
Harry Styles	8546679005
Olivia Rodrigo	7442148916
Eminem	6183805596
Imagine Dragons	5272484650
Lewis Capaldi	4734698360
Doja Cat	4702294655

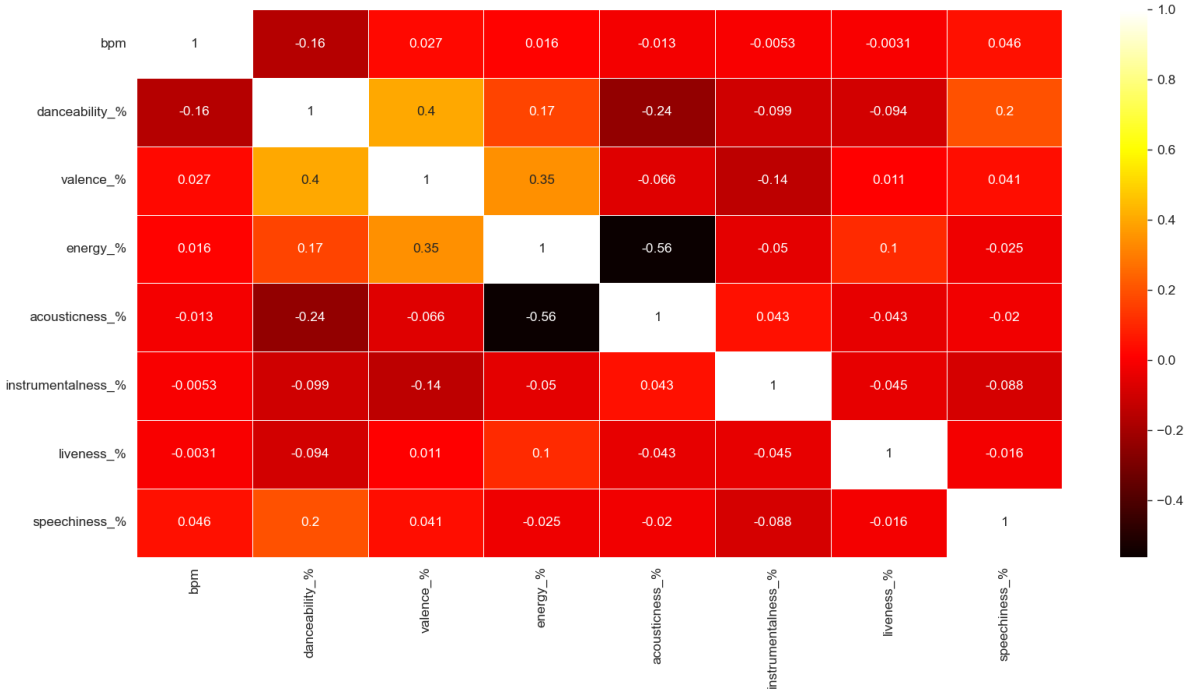
How often each Artist shows up in playlist (Top 15)

```
most_streamed_artists =
data.groupby('artist(s)_name')[['in_spotify_playlists','in_apple
_playlists','in_deezer_playlists']].sum().nlargest(10,
['in_spotify_playlists','in_apple_playlists','in_deezer_playlist
s'])
print(most_streamed_artists)
```

artist(s)_name	in_spotify_playlists	in_apple_playlists	in_deezer_playlists
Ed Sheeran	128758	1448	1702
Taylor Swift	114460	1606	1487
The Weeknd	98395	1291	1901
Eminem	87331	475	0
Harry Styles	80528	1214	2334
Coldplay	75716	381	805
Avicii	68241	407	0
Dr. Dre, Snoop Dogg	65728	283	0
Adele	65049	646	856
Nirvana	59505	310	0

What makes the best song the best

```
cols=['bpm','danceability_%', 'valence_%', 'energy_%',
      'acousticness_%', 'instrumentalness_%', 'liveness_%',
'speechiness_%']
correlation_matrix = data[cols].corr()
plt.figure(figsize=(12, 8))
sns.heatmap(correlation_matrix, annot=True, cmap='hot',
linewidths=.5)
plt.show()
```



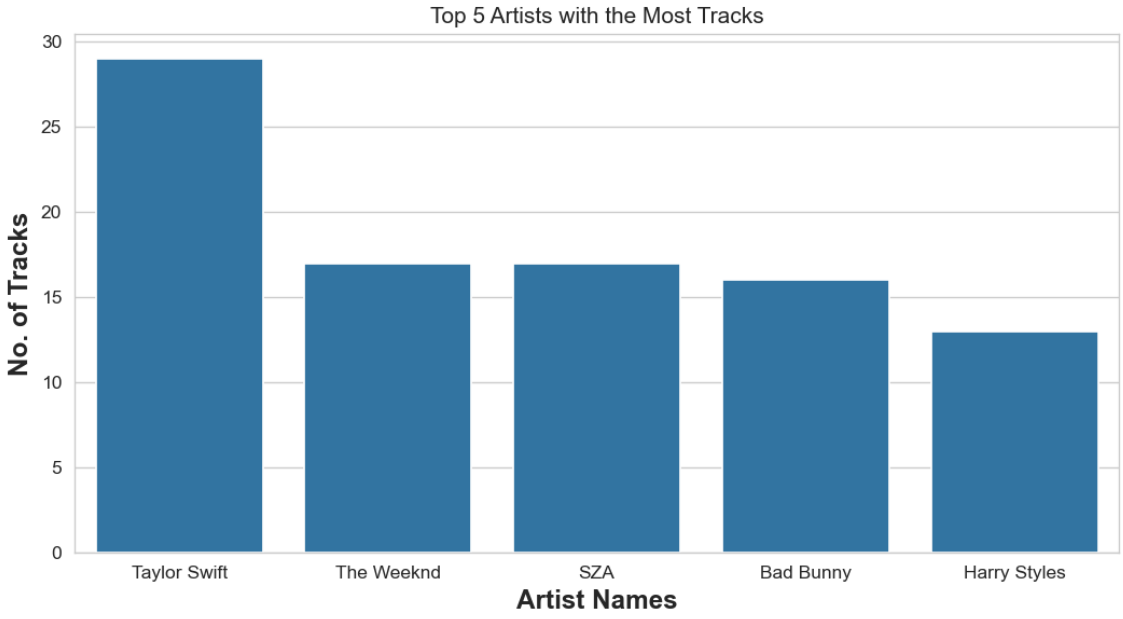
Majority of columns have weak negative correlation with the number of streams, indicating a mild inverse relationship. It should be noted that the practical significance of these weak correlations may be limited.

- Danceable Songs: There is a moderate positive correlation between danceability and valence. This suggests that when aiming to create a danceable song, it's likely to have a more positive or cheerful mood (valence).

- Energetic Songs: You found a moderate negative correlation between acousticness and energy. This implies that songs with lower acousticness (less acoustic, more electronic) tend to have higher energy levels. Therefore, if you're looking to create a more energetic song, reducing acoustic elements and incorporating more electronic or produced elements may be effective

Top 5 Artist with the most tracks

```
leading_artists = data['artist(s)_name'].value_counts().head(5)

plt.figure(figsize=(10, 5))


sns.barplot(x=leading_artists.index , y=leading_artists )

plt.ylabel('No. of Tracks',fontsize=14, fontweight='bold')
plt.xlabel('Artist Names',fontsize=14, fontweight='bold')
plt.title('Top 5 Artists with the Most Tracks')
plt.show()
```



Conclusion

In this report, we conducted a comprehensive analysis of a Spotify dataset using Python, focusing on a wide range of variables related to music tracks. The dataset included essential information such as track name, artist details, release date, Spotify and Apple Music playlist and chart placements, streaming statistics, and audio features.

Our analysis revealed several key insights:

- **Popular Tracks and Artists:** We identified the most popular tracks and artists based on their presence in Spotify playlists and charts. This information can be valuable for music industry professionals and music enthusiasts

- **Temporal Trends:** We examined temporal trends in music releases, such as the distribution of tracks by year, month, and day. This information can help stakeholders plan releases for optimal exposure.

- **Streaming Performance:** We assessed the streaming statistics for tracks, identifying those with the highest streams. Understanding streaming trends can be crucial for both artists and music platforms.

- **Cross-Platform Performance:** We examined the presence of tracks in Apple Music, Deezer playlists, and charts, which provides insights into their popularity across different streaming services.

- **Audio Features:** We analyzed audio features like BPM, key, mode, danceability, valence, energy, acousticness, instrumentalness, liveness, and speechiness. This information can be used to understand the musical characteristics of popular tracks and potentially inform music creation and curation.

In conclusion, this analysis of the Spotify dataset demonstrates the valuable insights that can be derived from music-related data. These insights can inform various stakeholders, from artists and record labels to streaming platforms and music enthusiasts, to make informed decisions regarding content creation, marketing strategies, and playlist curation. As the music industry continues to evolve in the digital age, data-driven insights become increasingly important for success and audience engagement.