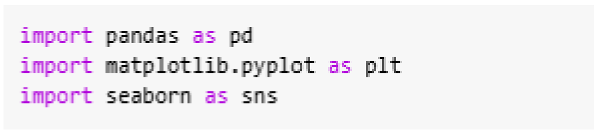
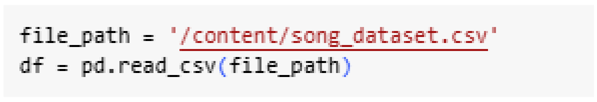
Loading and Inspecting the Dataset

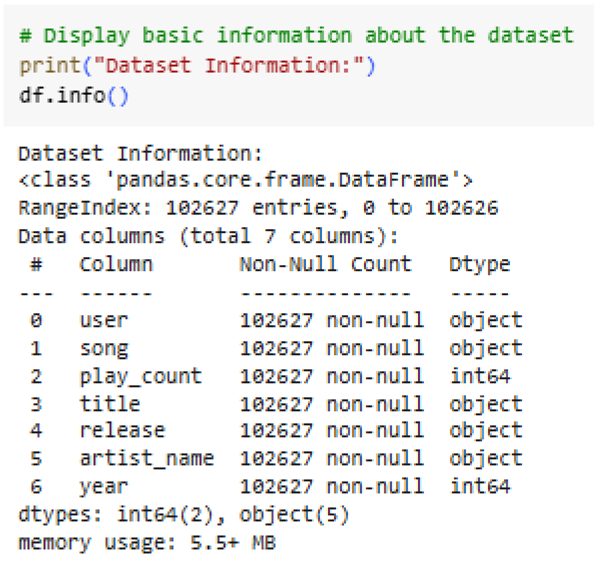
We first read the dataset with pandas. The info() function gave us an idea of the structure, and head() showed us the first few rows.



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

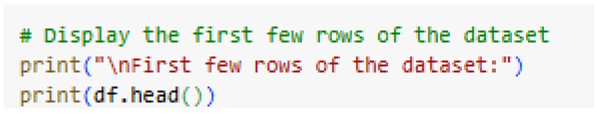


```
file_path = '/content/song_dataset.csv'
df = pd.read_csv(file_path)
```



```
# Display basic information about the dataset
print("Dataset Information:")
df.info()
```

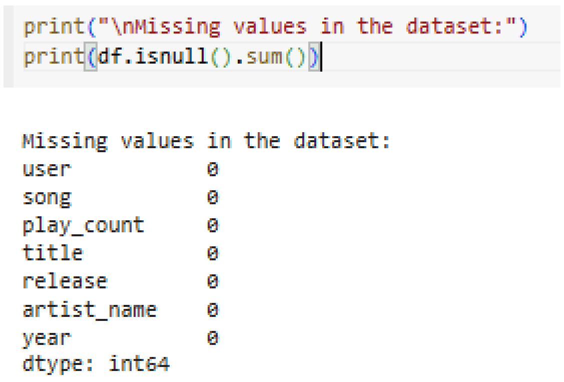
```
Dataset Information:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 102627 entries, 0 to 102626
Data columns (total 7 columns):
 #   Column       Non-Null Count   Dtype 
---  ------       --------------   ----- 
 0   user         102627 non-null  object
 1   song         102627 non-null  object
 2   play_count   102627 non-null  int64 
 3   title        102627 non-null  object
 4   release      102627 non-null  object
 5   artist_name  102627 non-null  object
 6   year         102627 non-null  int64 
dtypes: int64(2), object(5)
memory usage: 5.5+ MB
```



```
# Display the first few rows of the dataset
print("\nFirst few rows of the dataset:")
print(df.head())
```

Checking for Missing Values

To ensure data quality, the code checks if there are any missing values in the dataset.



```
print("\nMissing values in the dataset:")
print(df.isnull().sum())
```

```
Missing values in the dataset:
user           0
song           0
play_count     0
title          0
release        0
artist_name    0
year           0
dtype: int64
```

Finding the Most Listened Songs

The code summed up the play_count for each song to find the 10 most listened songs.

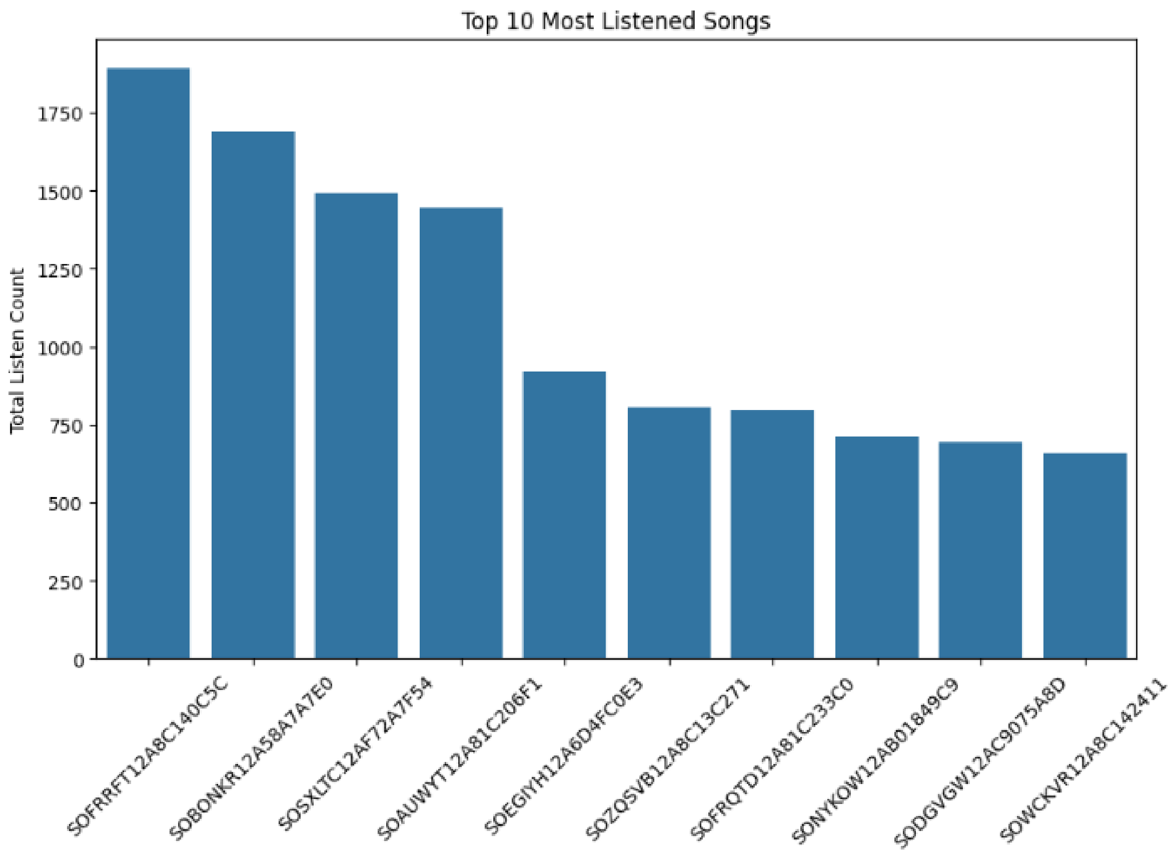
```
most_listened_songs = df.groupby('song')['play_count'].sum().sort_values(ascending=False).head(10)

print(most_listened_songs)
```

```
song
SOFRRFT12A8C140C5C    1890
SOBONKR12A58A7A7E0    1689
SOSXLTC12AF72A7F54    1490
SOAUWYT12A81C206F1    1443
SOEGIYH12A6D4FC0E3     921
SOZQSVB12A8C13C271     805
SOFRQTD12A81C233C0     795
SONYKOW12AB01849C9     712
SODGVGW12AC9075A8D     692
SOWCKVR12A8C142411     660
Name: play_count, dtype: int64
```

Visualization: A bar chart shows these top 10 songs, with the x-axis displaying the song IDs and the y-axis showing the total number of plays.

```
plt.figure(figsize=(10, 6))
sns.barplot(x=most_listened_songs.index, y=most_listened_songs.values)
plt.title('Top 10 Most Listened Songs')
plt.xlabel('Song ID')
plt.ylabel('Total Listen Count')
plt.xticks(rotation=45)
plt.show()
```



Identifying the Most Popular Artists

We found the top 10 artists by totaling the play counts for each artist.

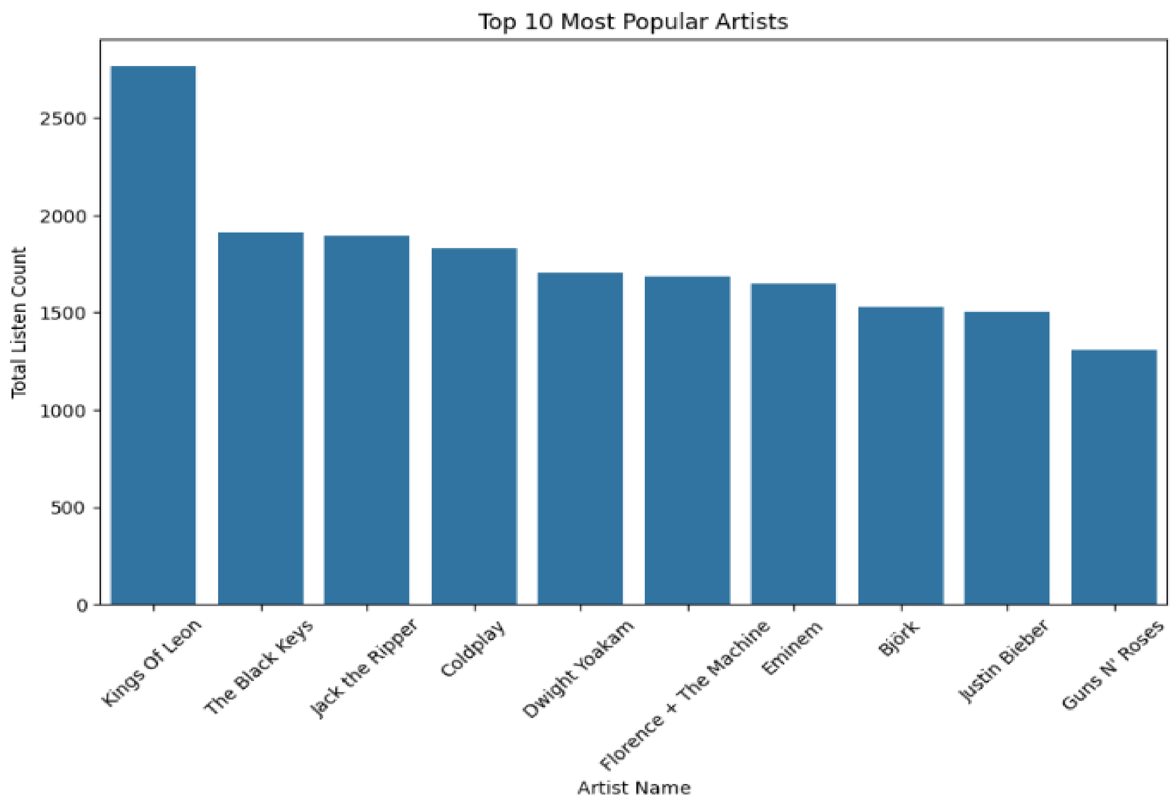
```
most_popular_artists = df.groupby('artist_name')['play_count'].sum().sort_values(ascending=False).head(10)

print(most_popular_artists)
```

```
artist_name
Kings Of Leon              2765
The Black Keys             1912
Jack the Ripper            1890
Coldplay                   1830
Dwight Yoakam              1705
Florence + The Machine     1682
Eminem                     1650
Björk                      1526
Justin Bieber              1504
Guns N' Roses              1309
Name: play_count, dtype: int64
```

Visualization: The bar chart shows artist names on the x-axis and the number of times their songs were played on the y-axis.

```
plt.figure(figsize=(10, 6))
sns.barplot(x=most_popular_artists.index, y=most_popular_artists.values)
plt.title('Top 10 Most Popular Artists')
plt.xlabel('Artist Name')
plt.ylabel('Total Listen Count')
plt.xticks(rotation=45)
plt.show()
```



Distribution of Unique Song Counts per User

We counted the number of different songs each user listened to, to know how much variety there is in their listening patterns.

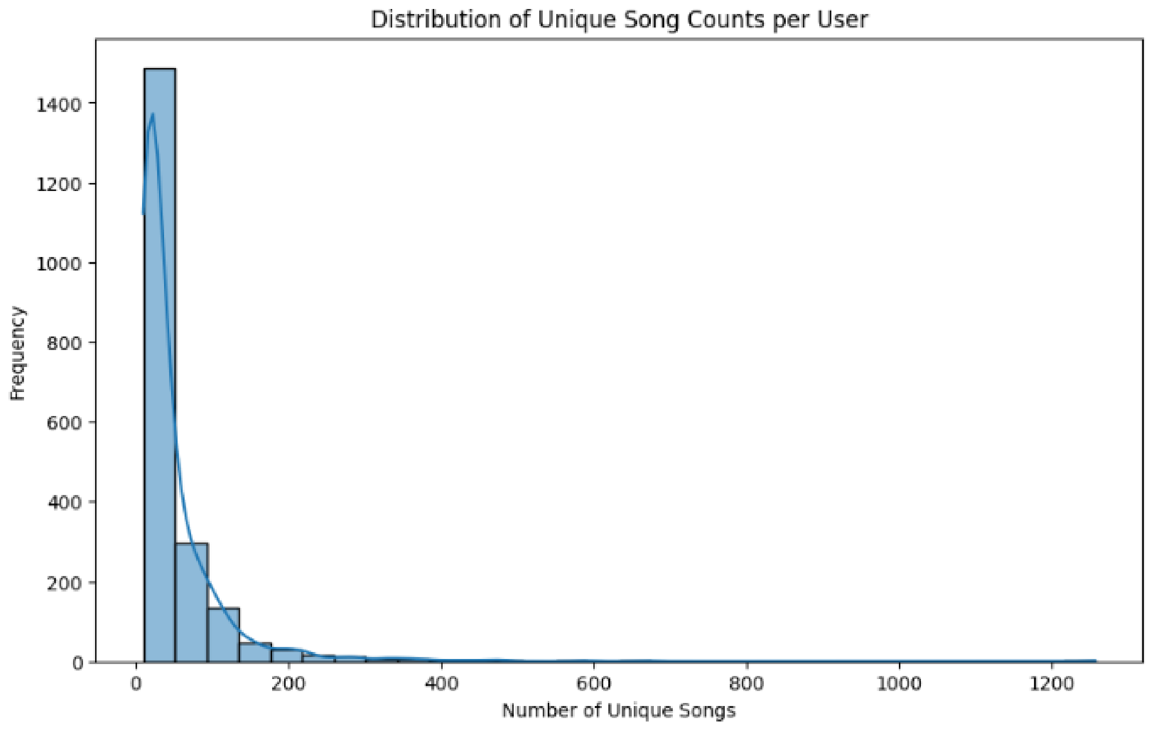
```
user_song_count = df.groupby('user')['song'].nunique()

print(user_song_count)
```

```
user
0007c0e74728ca9ef0fe4eb7f75732e8026a278b     13
000ebc858861aca26bac9b49f650ed424cf882fc    478
00342a0cdf56a45465f09a39040a5bc25b7d0046     93
0039bd8483d578997718cdc0bf6c7c88b679f488     64
00498f4bab2bfeb17680113c7d9525ad5b0ad401     16
                                           ... 
ffa24617ea80c268c74e86cd3ee3d9e7ac5504ec     39
ffadf9297a99945c0513cd87939d91d8b602936b     28
ffdaab327f2fc6b9fa01a4e3e7f41fdd0e468046    400
ffe50146303f1b12ea8254450b95580b1b99a5c4     13
fffce9c1537fbc350ea68823d956eaa8f5236dbe     54
Name: song, Length: 2042, dtype: int64
```

Visualization: The histogram shows the unique song counts for each user. Most users listen to a limited set of songs, while a few explore a wider range.

```
plt.figure(figsize=(10, 6))
sns.histplot(user_song_count, kde=True, bins=30)
plt.title('Distribution of Unique Song Counts per User')
plt.xlabel('Number of Unique Songs')
plt.ylabel('Frequency')
plt.show()
```



Distribution of Play Counts

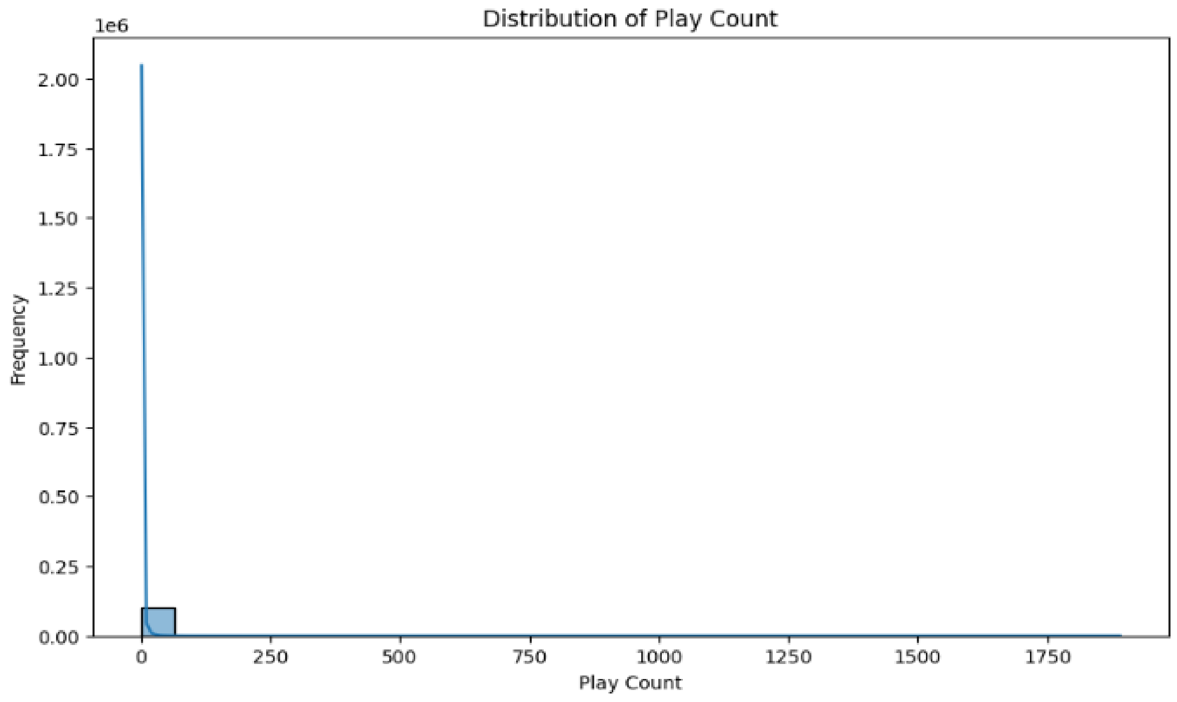
The distribution of the play counts (listening frequency) is analyzed to understand how often users play songs.

```
print("\nDistribution of song count for users':")
print(df['play_count'].describe())
```

```
Distribution of song count for users':
count    102627.000000
mean          2.885790
std           8.569376
min           1.000000
25%           1.000000
50%           1.000000
75%           3.000000
max        1890.000000
Name: play_count, dtype: float64
```

Visualization. We used a histogram with a KDE curve to show the play counts. Most songs get a few plays, but some are played far more often, as shown in the chart.

```
plt.figure(figsize=(10, 6))
sns.histplot(df['play_count'], bins=30, kde=True)
plt.title('Distribution of Play Count')
plt.xlabel('Play Count')
plt.ylabel('Frequency')
plt.show()
```



Summary

After performing the necessary exploratory data analysis on the dataset song_dataset.csv, we found the 10 songs with the most plays and the artists who were played the most. The data shows that only a few songs and artists are played far more often than others. Many users seem to prefer listening to a small number of familiar songs. We used bar charts and histograms to display these trends.

Reference

Adapted from Tutorial4-solution.ipynb. The code demonstrated Importing, Pre-processing and Exploratory Data Analysis.