MUSIC RECOMMENDATION SYSTEM

TEAM MEMBERS :-

--KOTA BHARADWAJ
--MAHIMAA CS
--VEDA P

2023

PROBLEM STATEMENT

TO CREATE A MACHINE LEARNING-BASED MUSIC RECOMMENDATION SYSTEM THAT DELIVERS PERSONALIZED MUSIC SUGGESTIONS BY ANALYZING USER LISTENING PREFERENCES USING COLLABORATIVE FILTERING.

WHY RECOMMENDATION?

MUSIC RECOMMENDATION SYSTEMS ARE ESSENTIAL IN TODAY'S DIGITAL AGE TO ADDRESS THE OVERWHELMING ABUNDANCE OF MUSIC AVAILABLE ACROSS VARIOUS PLATFORMS.

WITH MILLIONS OF SONGS AT OUR FINGERTIPS, IT CAN BE DAUNTING TO DISCOVER NEW MUSIC THAT ALIGNS WITH OUR INDIVIDUAL TASTES. RECOMMENDATION SYSTEMS STEP IN AS VALUABLE TOOLS, LEVERAGING ALGORITHMS AND USER DATA TO CURATE PERSONALIZED MUSIC SUGGESTIONS.

THEY ENHANCE MUSIC EXPLORATION, SAVE TIME, AND INTRODUCE USERS TO ARTISTS AND GENRES THEY MIGHT NOT HAVE DISCOVERED OTHERWISE.

SOLUTIONS USING ML

- COLLABORATIVE FILTERING: RECOMMEND MUSIC BASED ON USER SIMILARITIES.
- CONTENT-BASED FILTERING: MATCH MUSIC BASED ON AUDIO FEATURES AND METADATA.
- HYBRID METHODS: COMBINE USER BEHAVIOR AND SONG CHARACTERISTICS FOR PERSONALIZED RECOMMENDATIONS.
- MATRIX FACTORIZATION: DISCOVER LATENT FEATURES FOR ACCURATE MUSIC SUGGESTIONS.
- DEEP LEARNING: LEARN COMPLEX PATTERNS IN MUSIC DATA FOR IMPROVED RECOMMENDATIONS.
- REINFORCEMENT LEARNING: ADAPT RECOMMENDATIONS BASED ON USER FEEDBACK.
- CONTEXT-AWARE RECOMMENDATIONS: PROVIDE MUSIC SUGGESTIONS BASED ON USER CONTEXT.
- TRANSFER LEARNING: PERSONALIZE RECOMMENDATIONS USING PRE-TRAINED MODELS.

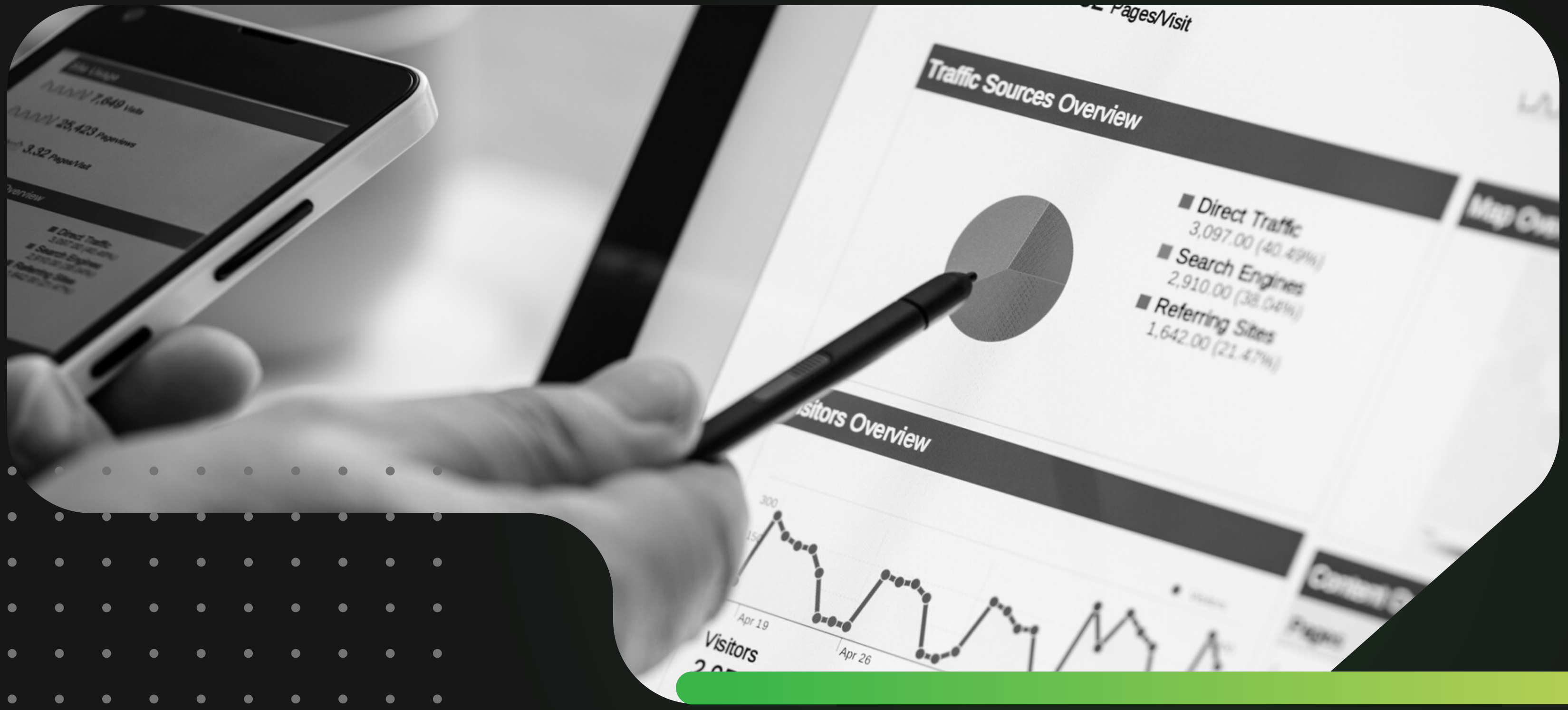
DATASETS

TRIPLETS FILE (USERS FILE) »»»

1	user_id	song_id	listen_count
2	b80344d063b5ccb3212f7	SOAKIMP12A8C130995	1
3	b80344d063b5ccb3212f7	SOBBMDR12A8C13253B	2
4	b80344d063b5ccb3212f7	SOBXHDL12A81C204C0	1
5	b80344d063b5ccb3212f7	SOBYHAJ12A6701BF1D	1
6	b80344d063b5ccb3212f7	SODACBL12A8C13C273	1
7	b80344d063b5ccb3212f7	SODDNQT12A6D4F5F7E	5
8	b80344d063b5ccb3212f7	SODXRTY12AB0180F3B	1
9	b80344d063b5ccb3212f7	SOFGUAY12AB017B0A8	1
10	b80344d063b5ccb3212f7	SOFRQTD12A81C233C0	1
11	[illegible]	[illegible]	1

SONG DATASET »»»

	A	B	C	D	E
1	song_id	title	release	artist_name	year
2	SOQMMHC12AB0180CB8	Silent Night	Monster Ballads X-Mas	Faster Pussy cat	2003
3	SOVFVAK12A8C1350D9	Tanssi vaan	Karkuteillä	Karkkiautomaatti	1995
4	SOGTUKN12AB017F4F1	No One Could Ever	Butter	Hudson Mohawke	2006
5	SOBNYVR12A8C13558C	Si Vos Querés	De Culo	Yerba Brava	2003
6	SOHSBXH12A8C13B0DF	Tangle Of Aspens	Rene Ablaze Presents Winter Sessions	Der Mystic	0
7	SOZVAPQ12A8C13B63C	Symphony No. 1 G minor "S	Berwald: Symphonies Nos. 1/2/3/4	David Montgomery	0
8	SOQVRHI12A6D4FB2D7	We Have Got Love	Strictly The Best Vol. 34	Sasha / Turbulence	0
9	SOEYRFT12AB018936C	2 Da Beat Ch'yall	Da Bomb	Kris Kross	1993
10	SOPMIYT12A6D4F851E	Goodbye	Danny Boy	Joseph Locke	0
11	SOJCFMH12A8C13B0C2	Mama_ mama can't you see	March to cadence with the US marines	The Sun Harbor's Cho	0
12	SOYGNWH12AB018191E	L'antarctique	Des cobras des tarentules	3 Gars Su'l Sofa	2007
13	SOLJTLX12AB01890ED	El hijo del pueblo	32 Grandes Éxitos CD 2	Jorge Negrete	1997
14	SOQQESG12A58A7AA28	Cold Beer feat. Prince Metro	International Hardcore Superstar	Danny Diablo	0
15	SOMPVQB12A8C1379BB	Pilots	The Loyal	Tiger Lou	2005
16	SOGPCJI12A8C13CCA0	N Gana	Afropea 3 - Telling Stories To The Sea	Waldemar Bastos	0



IMPLEMENTATION

RECOMMENDATION LIBRARY

WE HAVE INTEGRATED A RECOMMENDATION LIBRARY INTO OUR PROJECT THAT LEVERAGES THE K-NEAREST NEIGHBORS (KNN) ALGORITHM.

THE KNN ALGORITHM IS A VERSATILE AND INTUITIVE MACHINE LEARNING ALGORITHM USED FOR BOTH CLASSIFICATION AND REGRESSION TASKS. IT OPERATES ON THE PRINCIPLE OF SIMILARITY, WHERE IT CLASSIFIES OR PREDICTS THE TARGET VARIABLE BASED ON THE SIMILARITY OF ITS NEIGHBORS.

THIS LIBRARY EFFICIENTLY GENERATES PERSONALIZED RECOMMENDATIONS BY ANALYZING USER PREFERENCES AND ITEM SIMILARITIES.
IT SUPPORTS USER-BASED AND ITEM-BASED COLLABORATIVE FILTERING, OFFERING FLEXIBLE RECOMMENDATION STRATEGIES.

MODULES USED

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings("ignore", category=FutureWarning)
```

```
from scipy.sparse import csr_matrix
```

```
import recommenders
import import_ipynb
import Recommender
from Recommender import Recommender
```

MERGED DATASET

In [9]:

```
song_df['song'] = song_df['title']+' - '+song_df['artist_name']
song_df.head()
```

```
/var/folders/2w/n266pzyn46s2klyb2jlnw7ph0000gn/T/ipykernel_9839/3952665672.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  song_df['song'] = song_df['title']+' - '+song_df['artist_name']
```

Out[9]:

user_id	song_id	listen_count	title	release	artist_name	year	song
b5ccb3212f76538f3d9e43d87dca9e	SOAKIMP12A8C130995	1	The Cove	Thicker Than Water	Jack Johnson	0	The Cove - Jack Johnson
b5ccb3212f76538f3d9e43d87dca9e	SOBBMDR12A8C13253B	2	Entre Dos Aguas	Flamenco Para Niños	Paco De Lucia	1976	Entre Dos Aguas - Paco De Lucia
b5ccb3212f76538f3d9e43d87dca9e	SOBXHDL12A81C204C0	1	Stronger	Graduation	Kanye West	2007	Stronger - Kanye West
b5ccb3212f76538f3d9e43d87dca9e	SOBYHAJ12A6701BF1D	1	Constellations	In Between Dreams	Jack Johnson	2005	Constellations - Jack Johnson
				There Is			

2022

LISTENING COUNT AND PERCENTAGE

In [19]:

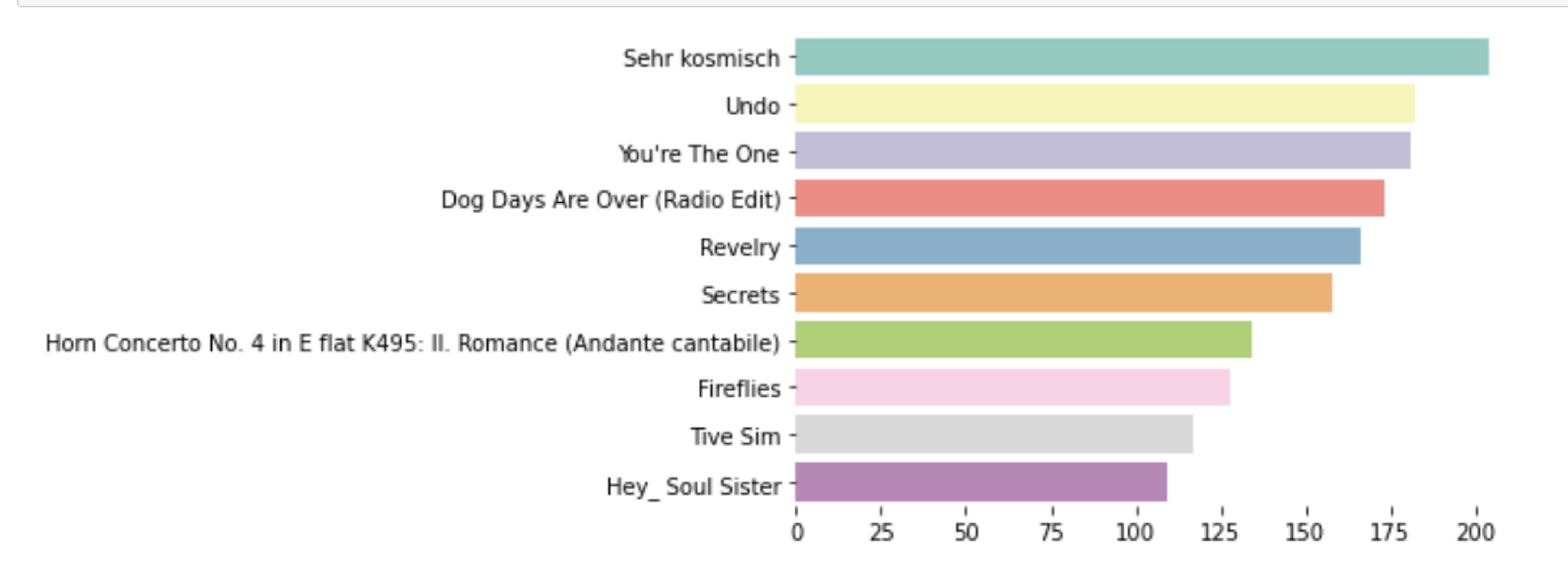
```
song_grouped['percentage'] = (song_grouped['listen_count'] / grouped_sum ) * 100
song_grouped.sort_values(['listen_count', 'song'], ascending=[0,1])
```

Out[19]:

	song	listen_count	percentage
6682	Sehr kosmisch - Harmonia	204	0.408
8509	Undo - Björk	182	0.364
1936	Dog Days Are Over (Radio Edit) - Florence + Th...	173	0.346
9256	You're The One - Dwight Yoakam	169	0.338
6348	Revelry - Kings Of Leon	166	0.332
...	...	...	...
9290	Your Time Has Come - Audioslave	1	0.002
9300	Zebra (full-length/album version) - John Butle...	1	0.002
9311	clouding - Four Tet	1	0.002
9312	high fives - Four Tet	1	0.002

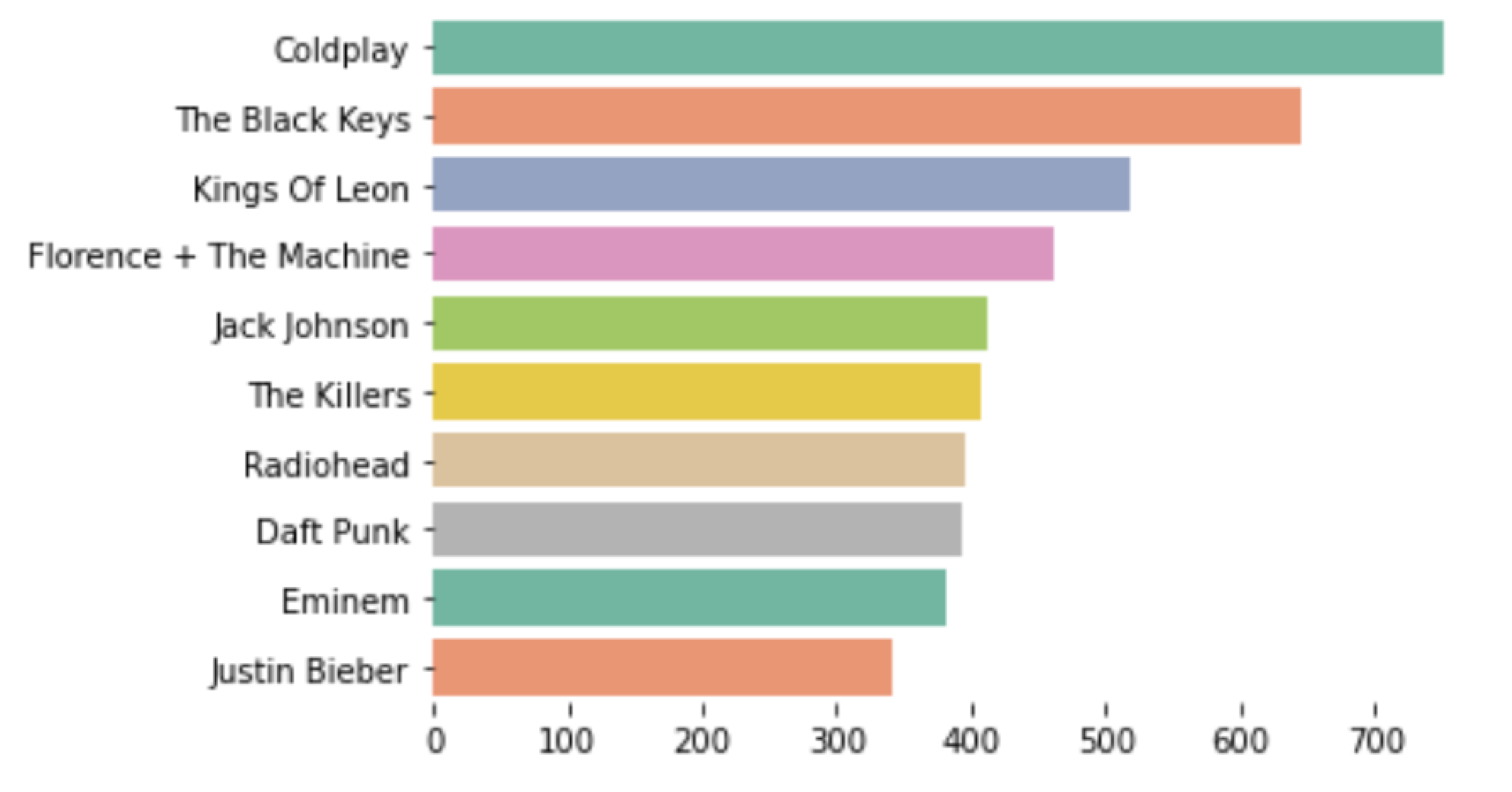
MOST LISTENED SONGS!!

	title	listen_count	percentage
6421	Sehr kosmisch	204	0.41
8186	Undo	182	0.36
8909	You're The One	181	0.36
1846	Dog Days Are Over (Radio Edit)	173	0.35
6100	Revelry	166	0.33
6410	Secrets	158	0.32
3214	Horn Concerto No. 4 in E flat K495: II. Romanc...	134	0.27
2427	Fireflies	128	0.26
7969	Tive Sim	117	0.23
3109	Hey_ Soul Sister	109	0.22



MOST LISTENED ARTISTS!!

	artist_name	listen_count
623	Coldplay	751
2717	The Black Keys	644
1586	Kings Of Leon	517
1065	Florence + The Machine	461
1318	Jack Johnson	412
2804	The Killers	407
2273	Radiohead	396
707	Daft Punk	394
942	Eminem	382
1490	Justin Bieber	342



USER LISTEN_COUNT BOX PLOT

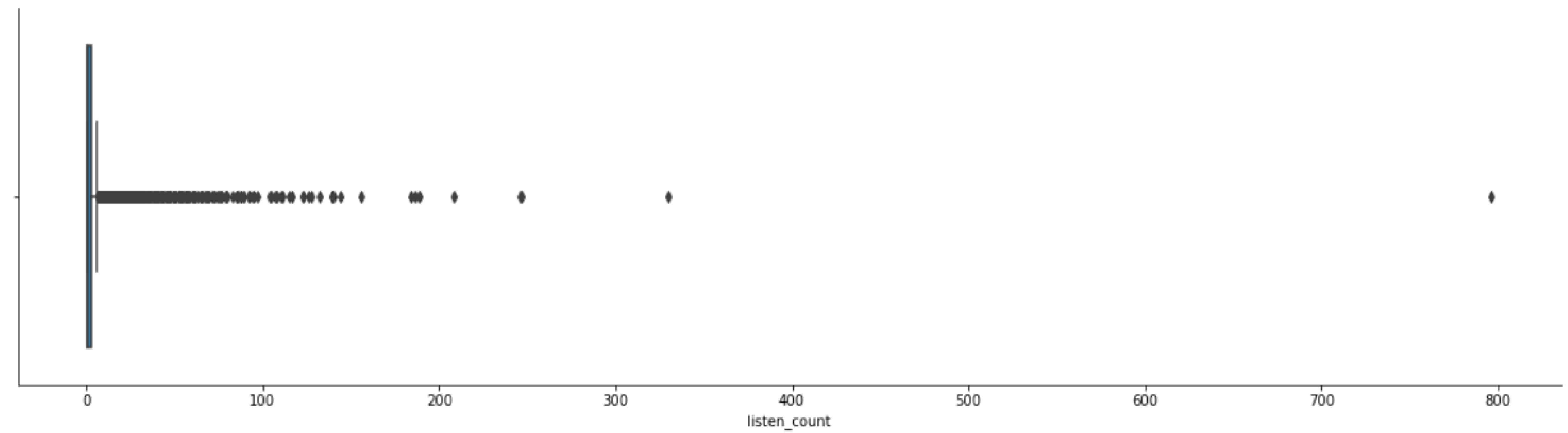
```
# What was the maximum time the same user listen to a same song?
listen_counts = pd.DataFrame(song_df.groupby('listen_count').size(), columns=['count'])
print(f"The maximum time the same user listened to the same songs was: {listen_counts.reset_index(drop=False)['listen_count'].iloc[-1]}")
```

The maximum time the same user listened to the same songs was: 796

```
#How many times on average the same user listen to a same song?
print(f"On average, a user listen to the same song {song_df['listen_count'].mean()} times")
```

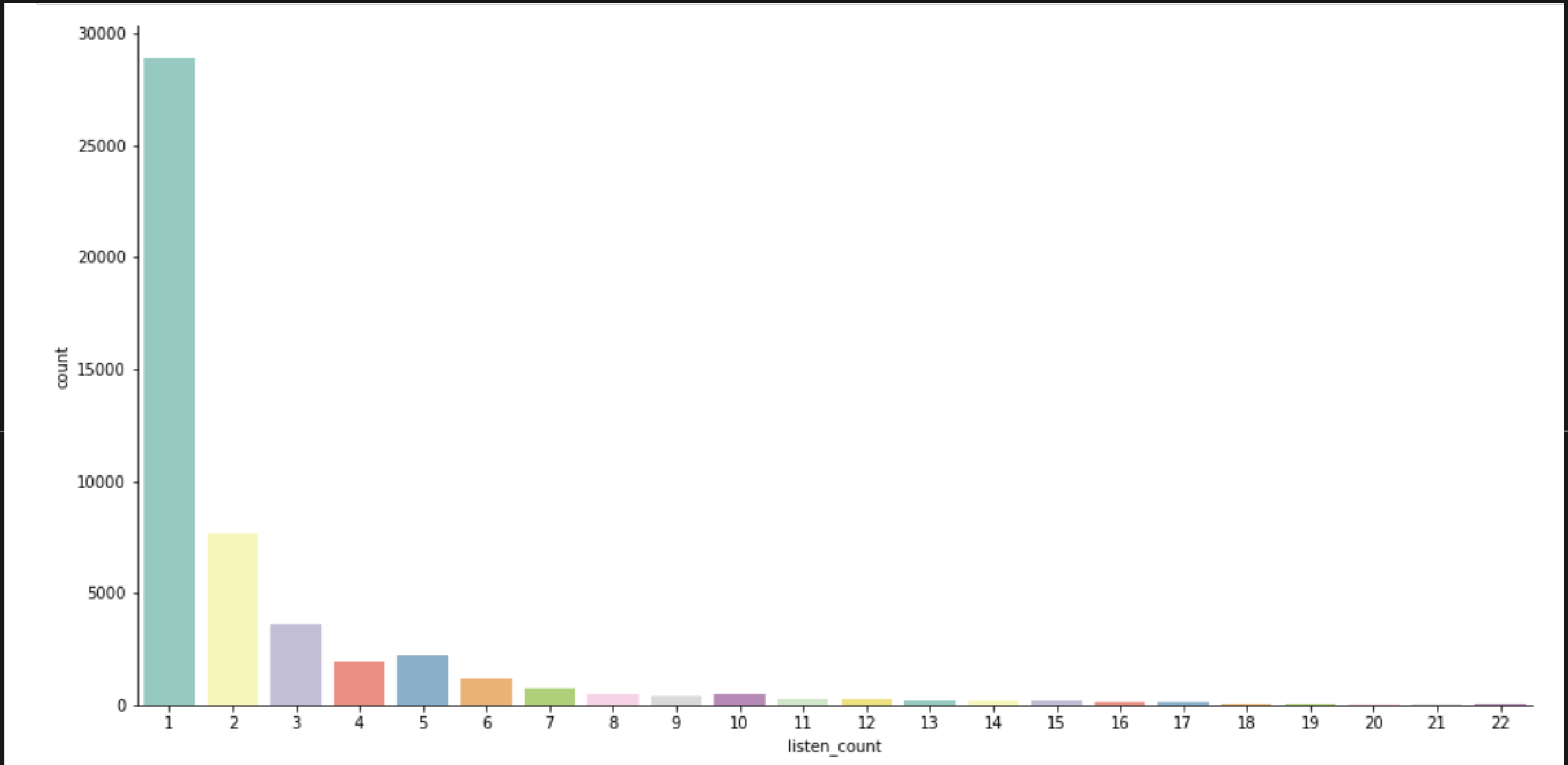
On average, a user listen to the same song 3.02526 times

```
plt.figure(figsize=(20, 5))
sns.boxplot(x='listen_count', data=song_df)
sns.despine()
```



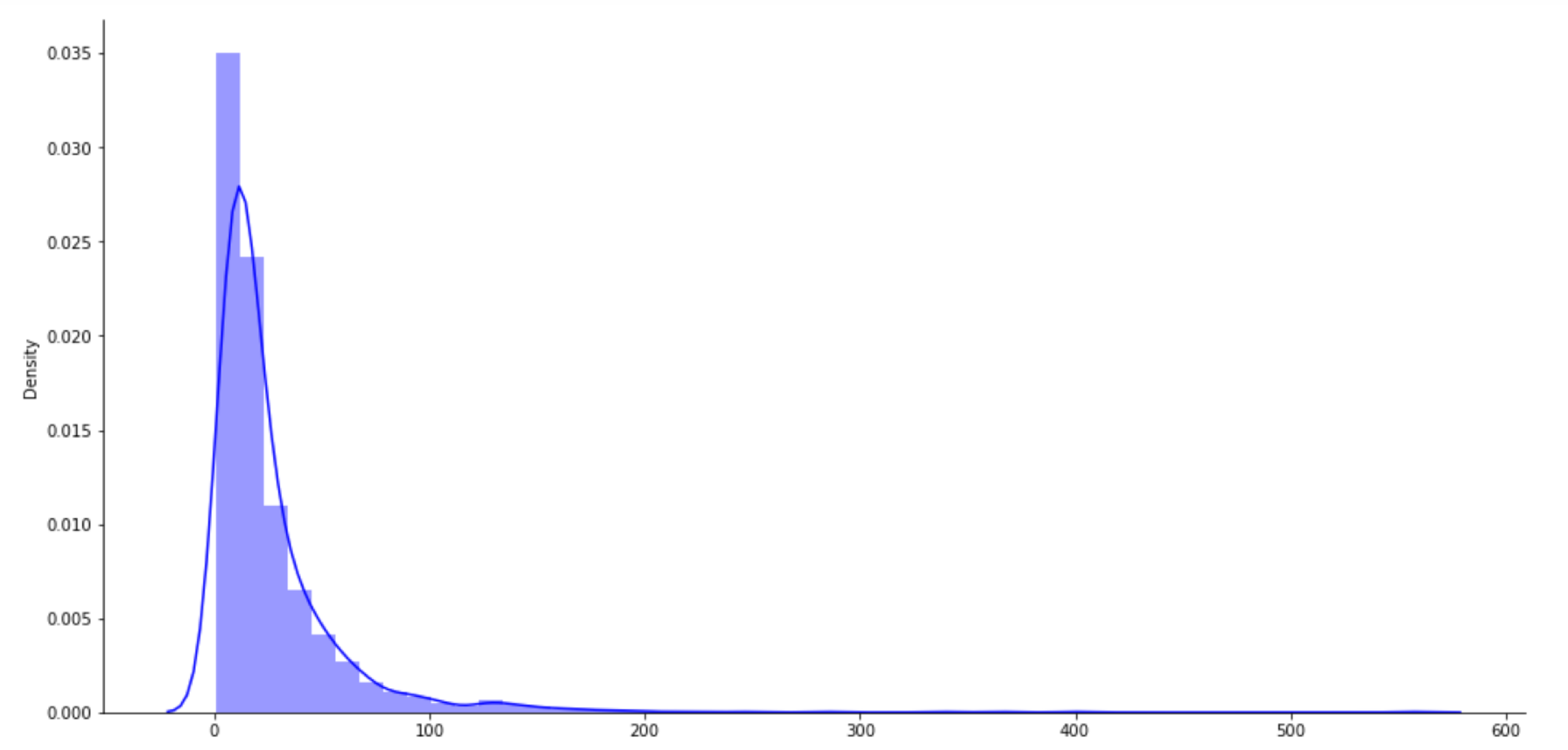
SONG FREQUENCY !!

```python
# What are the most frequent number of times a user listen to the same song?

listen_counts_temp = listen_counts[listen_counts['count'] > 50].reset_index(drop=False)
plt.figure(figsize=(16, 8))
sns.barplot(x='listen_count', y='count', palette='Set3', data=listen_counts_temp)
plt.gca().spines['top'].set_visible(False)
plt.gca().spines['right'].set_visible(False)
plt.show();
```



AVERAGE NO. OF SONGS !!

```
# How many songs does a user listen in average?
song_user = song_df.groupby('user_id')['song_id'].count()

plt.figure(figsize=(16, 8))
sns.distplot(song_user.values, color='blue')
plt.gca().spines['top'].set_visible(False)
plt.gca().spines['right'].set_visible(False)
plt.show();
```



USER SONG LISTENING DESCRIPTION

```
print(f"A user listens to an average of {np.mean(song_user)} songs")
```

A user listens to an average of 26.609898882384247 songs

```
print(f"A user listens to an average of {np.median(song_user)} songs, with minimum {np.min(song_user)} and maximum {np.max(song_user)} songs")
```

A user listens to an average of 16.0 songs, with minimum 1 and maximum 556 songs

```
#So, not all user listen to all songs, so a lot of values in the song x users matrix are going to be zero.
# Thus, we'll be dealing with extremely sparse data.

# Get how many values should it be if all songs have been listen by all users
values_matrix = unique_users * unique_songs
```

```
# Substract the total values with the actural shape of the DataFrame songs
zero_values_matrix = values_matrix - song_df.shape[0]
```

```
print(f"The matrix of users x songs has {zero_values_matrix} values that are zero")
```

The matrix of users x songs has 16815904 values that are zero

SPARSE MATRIX

```python
from scipy.sparse import csr_matrix
```

```python
# convert the dataframe into a pivot table
df_songs_features = df_song_id_more_ten.pivot(index='song_id', columns='user_id', values='listen_count').fillna(0)

# obtain a sparse matrix
mat_songs_features = csr_matrix(df_songs_features.values)
```

```python
df_songs_features.head()
```

user_id	000ebc858861aca26bac9b49f650ed424cf882fc	00342a0cdf56a45465f09a39040a5bc25b7d0046	0039bd8483d5
song_id			
SOAAAGQ12A8C1420C8	0.0	0.0	
SOAACPJ12A81C21360	0.0	0.0	
SOAAEJI12AB0188AB5	0.0	0.0	
SOAAFAC12A67ADF7EB	0.0	0.0	
SOAAFYH12A8C13717A	0.0	0.0	

5 rows × 920 columns

RESULTS

```
song = 'I believe in miracles'
```

```
new_recommendations = model.make_recommendation(new_song=song, n_recommendations=10)
```

Starting the recommendation process for I believe in miracles ...
... Done

```
print(f"The recommendations for {song} are: \n")
print(f"{new_recommendations}")
```

The recommendations for I believe in miracles are:

['On The Road Again', 'Radio Nowhere', 'Blue Shoes', 'Piece By Piece', 'Blues In The Night', 'Smash Into You', 'Oltremare', 'Shy Boy', "Spider's Web", 'I Do Believe In Love']

CONCLUSION

IN CONCLUSION, OUR PROJECT SUCCESSFULLY IMPLEMENTED A COLLABORATIVE FILTERING-BASED USING K-NN , MUSIC RECOMMENDATION SYSTEM TO PROVIDE PERSONALIZED SONG RECOMMENDATIONS.

BY ANALYZING THE SIMILARITIES BETWEEN USERS' LISTENING PATTERNS AND POPULARITY THE SYSTEM EFFECTIVELY GENERATED RELEVANT SONG SUGGESTIONS.

THROUGH THIS APPROACH, USERS WILL BE ABLE TO DISCOVER NEW MUSIC ALIGNED WITH THEIR TASTES, ENHANCING THEIR MUSIC EXPLORATION EXPERIENCE.

THE PROJECT SHOWCASES THE EFFECTIVENESS AND VALUE OF COLLABORATIVE FILTERING IN DELIVERING PERSONALIZED RECOMMENDATIONS.

THANK YOU !!