DAYANANDA SAGAR UNIVERSITY

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
SCHOOL OF ENGINEERING
DAYANANDA SAGAR UNIVERSITY
KUDLU GATE
BANGALORE - 560068

MINI PROJECT REPORT

ON

"Music Catalog Analysis"

Fundamentals of Data Science Laboratory(21DS2402)
4th SEMESTER
BACHELOR OF TECHNOLOGY
IN
COMPUTER SCIENCE & ENGINEERING (DATA SCIENCE)

Submitted by

AAKASH TOMAR - (ENG21DS0001)
NIKUNJ VIHARI KONAKALLA - (ENG21DS0023)

Under the supervision of
Prof. Sindhu A
Prof. Shahwar
Dept.of CSE(DS)
DSU

DAYANANDA SAGAR UNIVERSITY

School of Engineering, Kudlu Gate, Bangalore-560068

CERTIFICATE

This is to certify that Mr./Ms. AAKASH TOMAR & NIKUNJ VIHARI bearing USN ENG21DS0001 & ENG21DS0023 has satisfactorily completed his/her Mini Project as prescribed by the University for the Fourth semester B.Tech. programme in Computer Science & Engineering during the year 2 at the School of Engineering, Dayananda Sagar University., Bangalore.

Date: 09/6/2023

_____

Signature of the faculty in-charge

Max Marks	Marks Obtained

_____

Signature of Chairperson

DECLARATION

We hereby declare that the work presented in this mini project entitled - "Music Catalog Analysis", has been carried out by us and it has not been submitted for the award of any degree, diploma or the mini project of any other college or university.

AAKASH TOMAR - (ENG21DS0001)
NIKUNJ VIHARI KONAKALLA - (ENG21DS0023)

ACKNOWLEDGEMENT

The satisfaction that accompanies the successful completion of task would be incomplete without the mention of the people who made it possible and whose constant guidance and encouragement crown all the efforts with success.

We are especially thankful to our **Chairperson Dr. Shaila S G,** for providing necessary departmental facilities, moral support and encouragement.

We are very much thankful to our **Guide Prof. Sindhu and Prof.Shahwar** , for providing help and suggestions in completion of this mini project successfully.

We have received a great deal of guidance and co-operation from our friends and we wish to thank all that have directly or indirectly helped us in the successful completion of this project work.

AAKASH TOMAR - (ENG21DS0001)
NIKUNJ VIHARI KONAKALLA - (ENG21DS0023)

TABLE OF CONTENTS

ABSTRACT

The primary goal of the Music Catalog Analysis project is to analyze a dataset consisting of a comprehensive collection of music tracks. This project focuses on exploring the dataset to gain insights into the music catalog and extract meaningful information. Various analytical techniques and visualization methods are employed to uncover patterns, trends, and characteristics within the dataset. The dataset used in this analysis contains album and artist songs. The project also includes the analyzing trends using scatter plot, popularity ranking, correlation heatmap. Visualization techniques are utilized to provide visual representations of the dataset. Scatter plots, are generated to visualize the distribution of features, identify clusters or groups, and visualize the relationships between variables. These visualizations aid in identifying interesting patterns and trends within the music catalog. We could infer from the findings of the visualizations that the data was distributed in a polarized manner by popularity. The music tracks were either highly popular or not popular at all. It was also identified that factors such as loudness and danceability characteristics correlated highly with popularity score of the song. Whereas Instrumental and classical songs ranked low in popularity. Along with these, other findings helped in gaining insights into the trends the data follows.

1. INTRODUCTION

Spotify is a music streaming service which allows users to play music on their devices. One of the goals of this project is to recommend the users with better content which the user also expects to receive. Another goal of this project is to analyze the metrics of the musical metadata to arrive at conclusions which help us understand the trends, factors affecting, how they affect and to what degree they affect the performance in rankings. For this, a dataset of 1750 instances was used which contains, randomly selected tracks listed on Spotify spanning from year 1957 to 2001. In the following sections the basic information about the dataset and the features are obtained to understand the dataset, instances, attributes and the range of all attributes' values.

2. PROBLEM STATEMENT

To analyze a dataset containing metadata and features of music tracks from Spotify and infer from the findings to arrive at a conclusion.

3. S/W & H/W SPECIFICATIONS

Software Specifications

Python and libraries – Pandas, Numpy, Matplotlib, Seaborn, Sci-kit Learn

Jupyter Lab – where python notebooks are used to execute code

Dataset for Analysis

Hardware Specifications

Computer

Internet Connection

4. DESIGN

4.1 ALGORITHM/METHODOLOGY

Python Libraries used:

1. Pandas
2. Numpy
3. Matplotlib
4. Seaborn
5. Sci-kit Learn

Functions used to manipulate and query Data Objects:

Function used	Description
df.sort_values()	To sort the values in ascending or descending order
df.duplicates()	To check if duplicates exist in dataframe
df.drop_duplicates()	To remove duplicates
df.query()	Used to display filtered data
df.value_counts()	To find counts different values in categorical columns
df.iloc[]	To select a subset of data frame

Functions used to plot charts and analyze data:

Function used	Description
sns.barplot()	Used to plot bar chart
df.corr()	Returns correlation for all selected numerical values columns
sns.heatmap()	Plots a correlation heat-map using the correlation data
df.hist()	Plots histogram for all numerical columns
plt.scatter()	Plots the data points on a graph to analyze trends across 2 columns

4.2 DESCRIPTION of MODULES/PROGRAM

4.2.1 DATASET

The dataset chosen for this project contains information about a subset of catalog of music on Spotify, a music streaming site. The dataset contains a total of 1750 instances and 16 data columns, 15 of which describe the tracks. Each track is described by basic attributes such as Name of the track, Album of the Track, Artist, length etc and other attributes such as popularity, acousticness, danceability, energy, liveness, loudness, tempo etc describe the track's characteristics. The following is the details of the dataset using the df.info()statement.

```
[6]: print(spotify_dataset.info())

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1750 entries, 0 to 1749
Data columns (total 16 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   Unnamed: 0        1750 non-null   int64
 1   name              1750 non-null   object
 2   album             1750 non-null   object
 3   artist            1750 non-null   object
 4   release_date      1750 non-null   object
 5   length            1750 non-null   int64
 6   popularity        1750 non-null   int64
 7   danceability      1750 non-null   float64
 8   acousticness      1750 non-null   float64
 9   energy            1750 non-null   float64
 10  instrumentalness  1750 non-null   float64
 11  liveness          1750 non-null   float64
 12  loudness          1750 non-null   float64
 13  speechiness       1750 non-null   float64
 14  tempo             1750 non-null   float64
 15  time_signature    1750 non-null   int64
dtypes: float64(8), int64(4), object(4)
memory usage: 218.9+ KB
None
```

The dataset statistical summary of the dataset is obtained by using the df.describe() statement.

```
[19]: print(spotify_dataset.describe())

       Unnamed: 0         length   popularity  danceability  acousticness  \
count  1750.000000    1750.000000  1750.000000   1750.000000   1750.000000
mean    874.500000  195436.638286    69.506286      0.651688      0.298371
std     505.325802   47465.014667    24.158489      0.175828      0.301546
min       0.000000   34533.000000     0.000000      0.000000      0.000035
25%     437.250000  167280.500000    68.000000      0.553250      0.051825
50%     874.500000  193838.000000    77.000000      0.671000      0.183000
75%    1311.750000  222346.000000    83.000000      0.779000      0.499000
max    1749.000000  530253.000000   100.000000      0.980000      0.996000

              energy  instrumentalness     liveness     loudness  speechiness  \
count  1750.000000       1750.000000  1750.000000  1750.000000  1750.000000
mean      0.592553          0.053457     0.177332    -7.867144     0.118890
std       0.205806          0.195213     0.129798     5.190363     0.112192
min       0.000020          0.000000     0.032700   -40.449000     0.000000
25%       0.479000          0.000000     0.101000    -8.440250     0.040200
50%       0.623000          0.000000     0.125000    -6.522000     0.064850
75%       0.739750          0.000095     0.213750    -5.131750     0.160000
max       0.997000          1.000000     0.945000    -1.465000     0.777000

              tempo  time_signature
count  1750.000000     1750.000000
mean    122.115411        3.932000
std      31.493535        0.396459
min       0.000000        0.000000
25%      94.736000        4.000000
50%     123.049500        4.000000
75%     143.929000        4.000000
max     211.968000        5.000000
```

By the above results we get the idea of distribution of data for each of the attributes. This helps us understand how the data is structured and be aware of the expected values.

4.2.2 DATA PREPROCESSING

Checking for duplicate data in the dataset

We use the df.duplicates() function to check if there are any duplicate rows in the dataset. The function returns a boolean values array, by which we can identify if a row is duplicated or not.

Removing Duplicate instances/rows

We use the df.drop_duplicates(keep='first', inplace=True) function to remove any duplicate rows present in the dataset. The function retains the first occurrence of every different row but removes any following occurrence.

```
spotify_dataset.duplicated().sort_index().value_counts()
# no duplicated rows
```

```
False    1750
Name: count, dtype: int64
```

4.2.3 BASIC ANALYSIS OF THE DATASET

- **Album/Artist Count**

The Album and the Artist names are part of the track's metadata. The Artist and the Album attributes are of categorical values and fall into the class of Nominal Data. Here, each track has the Album name to which it belongs to and the Artist by whom the track was produced by. We can find the total of categories in a dataframe column by using the df['column name'].value_counts() statement.

```
spotify_dataset['album'].value_counts()

album
Dangerous: The Double Album                106
Shoot For The Stars Aim For The Moon        27
Goodbye & Good Riddance                     25
Legends Never Die                           23
Positions                                   16
                                          ...
Uptown Special                               1
Diva (feat. Lil Tecca)                       1
NOVA                                         1
Die Lit                                      1
Personal Problems                            1
Name: count, Length: 849, dtype: int64
```

```
spotify_dataset['artist'].value_counts()

artist
Various Artists    150
Morgan Wallen      119
Juice WRLD          70
Miracle Tones       45
Ariana Grande       39
                  ...
Sia                  1
Lorde                1
Chuck Berry          1
MarMar Oso           1
Big Havi             1
Name: count, Length: 501, dtype: int64
```

From the above code snippet, the statement returns a pandas Series as data object which contains the list of all albums/artists sorted in descending order of frequency of occurrence. The length of the Series refers to the count of different categories in that column.

From the output,

The number of Albums is the dataset are 849.

The number of Artists in the dataset are 501.

2. Artist's songs

All songs released by a single artist can be queried using the

`df.query('query_statement')`, which return a dataframe of selected rows whose artist attribute has the given value(here, artist name). All tracks by a particular album will be returned when queried using the statement

`df.query('artist=="Wolfgang Amadeus Mozart"')`.

The following output shows all the tracks by the artist Wolfgang Amadeus Mozart, when queried for that artist.

```
[79]: spotify_dataset.query('artist=="Wolfgang Amadeus Mozart"').sort_values(by='album', ascending=True)
```

[79]:

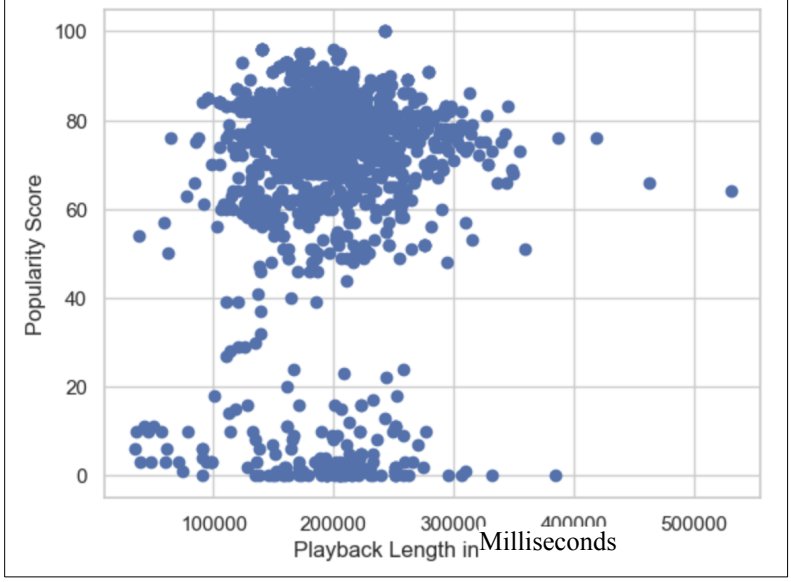
	Unnamed: 0	name	album	artist	release_date	length	popularity	danceability	acousticness	energy	instrumentalness	liveness	loudness	speechiness	tempo	time_signature
1447	1447	3 German Dances, K. 605: No. 2 in G Major	Mozart for Brainpower	Wolfgang Amadeus Mozart	2021-01-19	101160	18	0.408	0.991	0.0789	0.646000	0.1290	-17.460	0.0411	148.203	3
1584	1584	6 German Dances, K. 600: No. 2 in F Major	Mozart for Brainpower	Wolfgang Amadeus Mozart	2021-01-19	118280	15	0.552	0.964	0.1220	0.323000	0.0879	-15.432	0.0489	125.297	3
1586	1586	6 German Dances, K. 600: No. 1 in C Major	Mozart for Brainpower	Wolfgang Amadeus Mozart	2021-01-19	134453	8	0.619	0.947	0.2770	0.896000	0.2010	-12.619	0.0366	131.970	3
1588	1588	6 German Dances, K. 600: No. 5 in G Major	Mozart for Brainpower	Wolfgang Amadeus Mozart	2021-01-19	137680	6	0.488	0.950	0.0901	0.003090	0.3690	-14.886	0.0546	110.318	3
1590	1590	6 Minuets K. 105 (attribution doubtful): No. 6...	Mozart for Brainpower	Wolfgang Amadeus Mozart	2021-01-19	91000	6	0.616	0.988	0.0864	0.983000	0.1290	-20.239	0.0460	84.068	4
1592	1592	6 German Dances, K. 600: No. 1 in C Major	Start Your Day With Mozart	Wolfgang Amadeus Mozart	2021-01-26	134453	0	0.619	0.947	0.2770	0.896000	0.2010	-12.619	0.0366	131.970	3
1594	1594	6 Minuets K. 105 (attribution doubtful): No. 6...	Start Your Day With Mozart	Wolfgang Amadeus Mozart	2021-01-26	91000	0	0.616	0.988	0.0864	0.983000	0.1290	-20.239	0.0460	84.068	4
1596	1596	6 German Dances, K. 600: No. 5 in G Major	Start Your Day With Mozart	Wolfgang Amadeus Mozart	2021-01-26	137680	0	0.488	0.950	0.0901	0.003090	0.3690	-14.886	0.0546	110.318	3
1448	1448	3 German Dances, K. 605: No. 3 in C Major, Tri...	Stay at Home with Mozart	Wolfgang Amadeus Mozart	2021-01-16	148506	7	0.459	0.840	0.1980	0.000317	0.0748	-13.935	0.0406	173.717	3
1627	1627	7 Menuets, K. 65a: No. 1 in G Major	Wolfgang Amadeus Mozart: Essential Orchestral ...	Wolfgang Amadeus Mozart	2021-01-22	128276	2	0.507	0.551	0.0301	0.628000	0.1930	-20.959	0.0424	113.412	3

3. Analyzing trends using Scatter Plot

Scatter plot, using the given input data as lists for x and y coordinates, it plots the data points on graph against the two metrics or attributes we provide. The trends from the resultant graph can be discovered by analyzing how the data points group, change location with respect to the attributes and overall how they are distributed in the graph.

Plotting data points of Playback length against Popularity

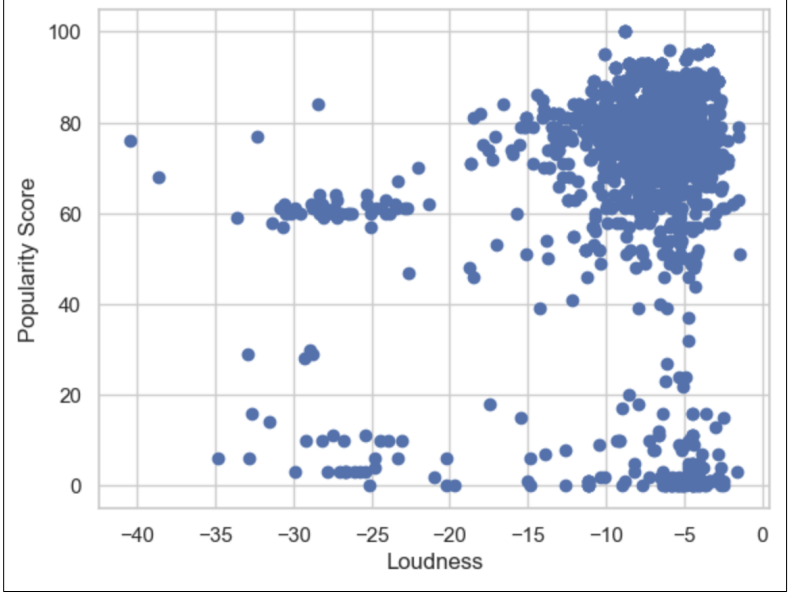
```
plt.scatter(spotify_dataset['length'], spotify_dataset['popularity'])
plt.ylabel("Popularity Score")
plt.xlabel("Playback Length in Seconds")
plt.show()
```



Here we can observe a group of data points clustered together close to the top right corner. Most of the data points here represent tracks that have a **popularity score between 50 to 90** and these popular tracks have a similar **playback time ranging from 100s(1min 40sec) to 300s(5mins).** We can also understand that most datapoints are within this range I.e **most tracks have a runtime of 2 to 5 mins.** Another observation is that the popularity score of the tracks have a **polarizing trend** i.e the tracks are either popular or they are not popular, as the datapoints other than the major cluster, are gathered along the line x = 0 Popularity.

Plotting data points of Loudness feature against Popularity

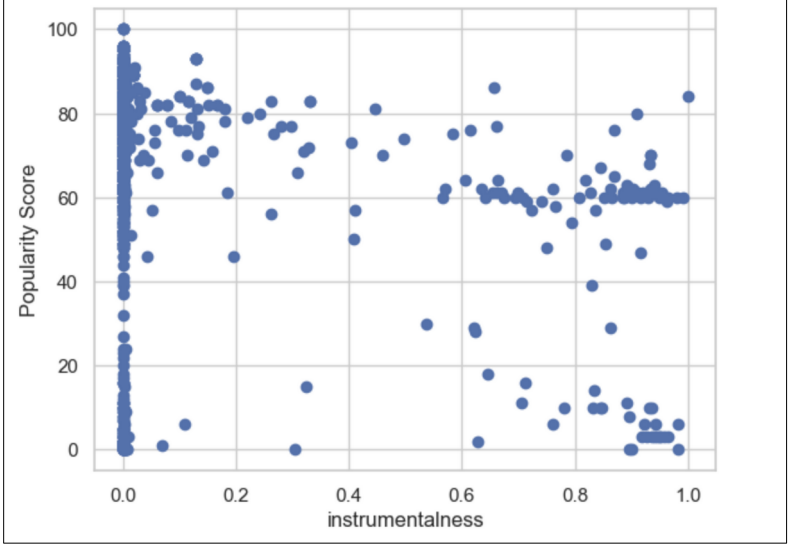
```
plt.scatter(spotify_dataset['loudness'],spotify_dataset['popularity'])
plt.ylabel("Popularity Score")
plt.xlabel("Loudness")
plt.show()
```



This graph somewhat has a similar trend going on like the previous plot. Here, we plot the points to compare **relationship between Loudness feature of the track and the Popularity** of the respective track. We can see a similar cluster where the **datapoints range approximately from -15 to -2 units**, which have **high popularity score i.e 50 to 90.** By this we understand that **higher values of loudness feature correlates with higher popularity**. But a clustered line of data points are fairly dispersed unlike the previous plot and we notice that **higher values of loudness increases the probability of poorly performing tracks' popularity**.

Plotting data points of Instrumentalness feature against Popularity

```
plt.scatter(spotify_dataset['instrumentalness'],spotify_dataset['popularity'])
plt.ylabel("Popularity Score")
plt.xlabel("instrumentalness")
plt.show()
```



The plot shows **majority of data points along the line y = 0 instrumentalness** and a **few data points dispersed around the right most part of the graph**. The datapoints present along the y = 0 **span along the line and across the scale of Popularity**, which indicates that **most tracks have 0 instrumentalness**. By this we understand that **irrespective of the popularity there are very few instrumental tracks and classical music tracks in the dataset**. The only instrumental tracks are the ones that are dispersed around the right most part of the graph. These instrumental tracks have a polarizing trend of popularity just like the previous plots.

4. Popularity Ranking

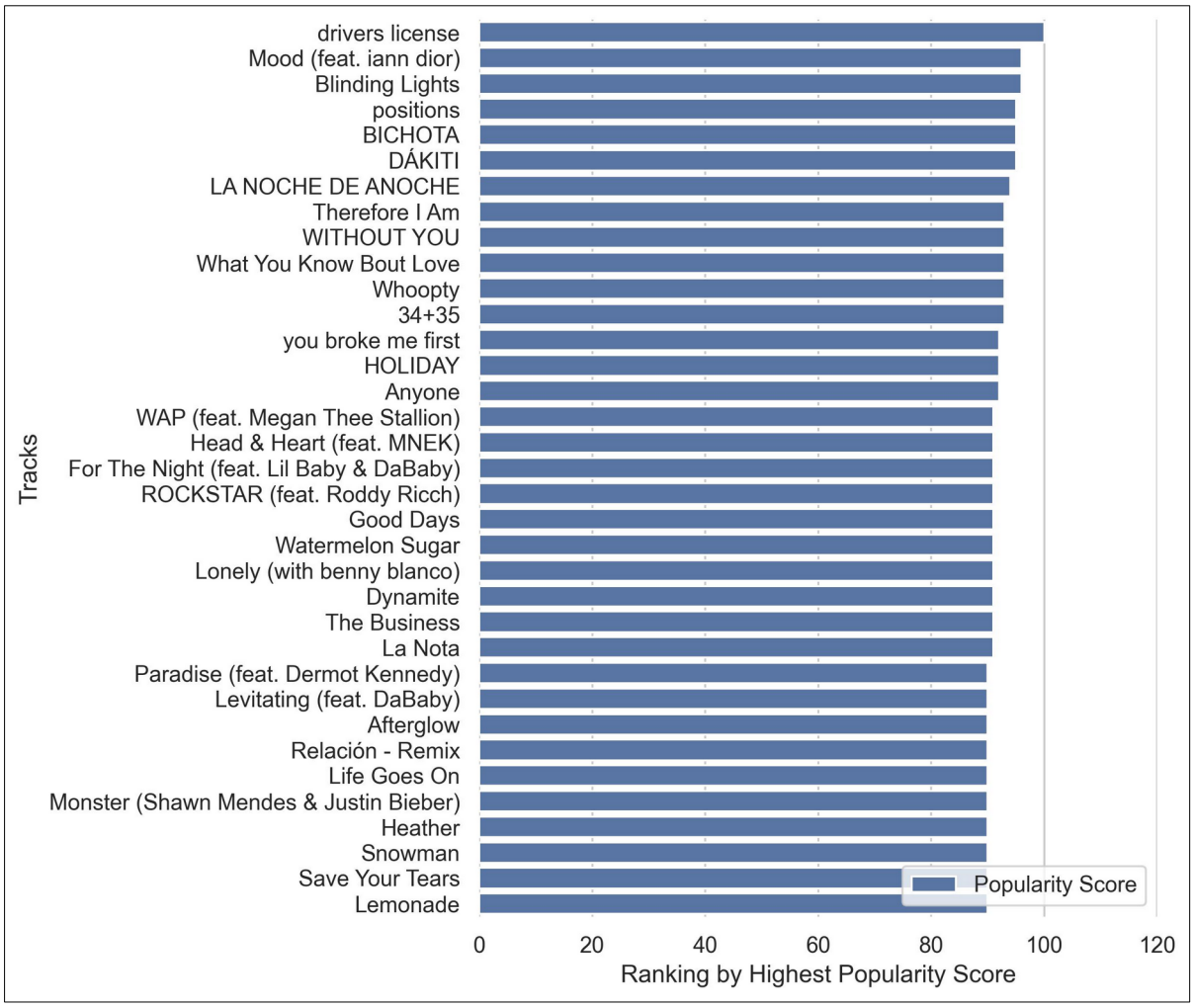
Popularity ranking is essential for recommendation systems and algorithms, for that very reason we rank songs based on their popularity score which is usually evaluated by user playback counts in certain time period. Popular tracks are those tracks which are on a general scale played by most users. This is essential for streaming services to improve their service by recommending the user some tracks that he will most likely listen to.

This is implemented by sorting the rows in dataframe by popularity in descending order. This is done by using the statement

```
pop_rank=spotify_dataset.sort_values("popularity", ascending=False)
```

The following is the chart generated using seaborn and matplotlib and the pop_rank dataframe shows the

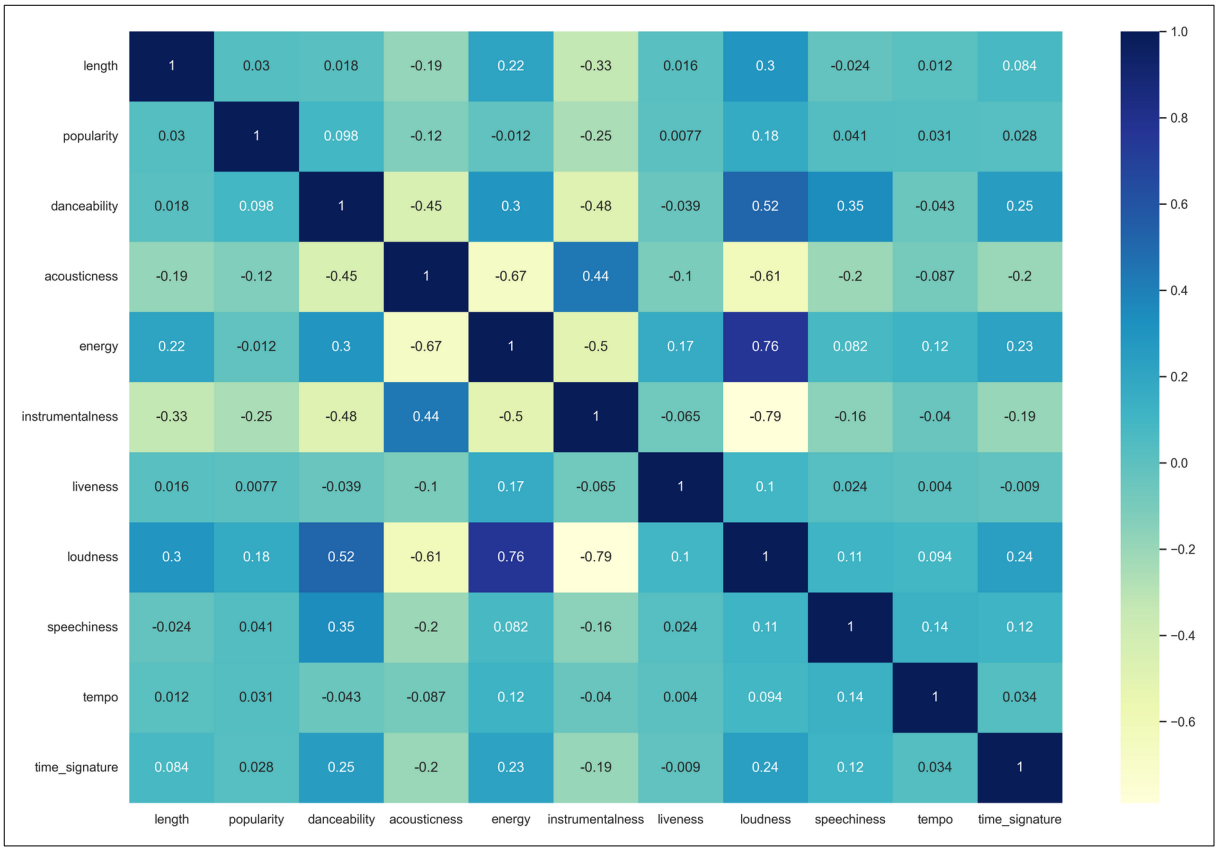
Most Popular 35 tracks from the dataset.

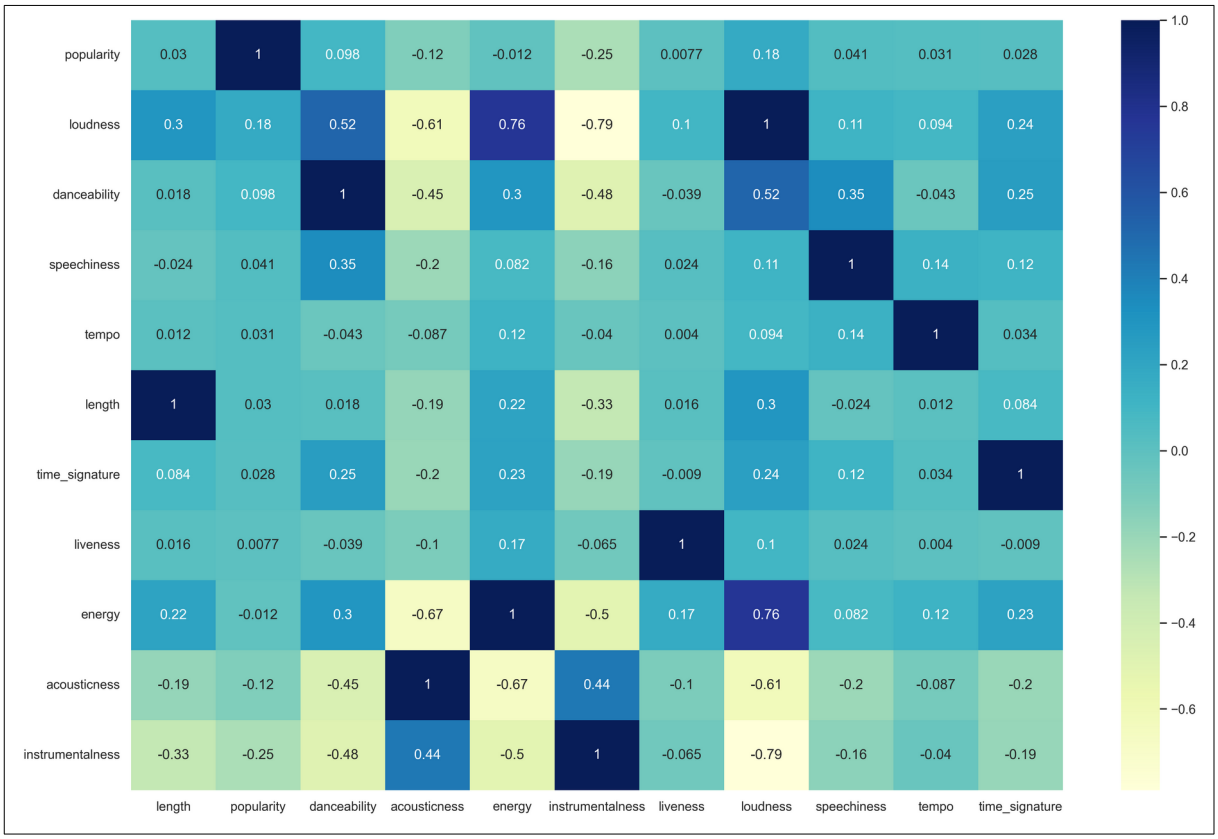


5. Correlation Analysis

Correlation gives the degree to which two features are interdependent and how each feature affects the other feature's value and vice-versa. This is useful to understand what contributes most value to the result and filter them out for further analysis. This gives a better insight and more clarity in analyzing the trends, in implementing ML algorithms and acts as Validation against inference of other charts.

We use df.corr() statement from the Pandas library to find correlation of all features against all of them. We then get a correlation heatmap using the statement `sns.heatmap()`.





The heatmap follows a color gradient where light yellow represents least correlation and dark blue represents highest correlation. The diagonal values are 1 as the features correlate to them at 100%.

- Highest(positive) correlation can be observed between energy and loudness equal to 0.76
- Highest(negative) correlation is observed between loudness and instrumentalness which is equal to -0.79.
- Loudness is correlated with danceability at 0.52(moderately high positive)
- Instrumentalness is correlated with danceability at -0.48(moderately high negative)
- Instrumentalness is correlated with acousticness at 0.44 (moderately high positive)
- Popularity has relatively high positive correlation between – Loudness and Danceability.
- Popularity has relatively high negative correlation between – Intrumentalness and Acousticness.
- Popularity has relatively low correlation between – time_signature, length and tempo.

6. OUTPUT SCREENSHOTS

6.1 Libraries

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

6.2 Dataset Specifications

```
spotify_dataset = pd.read_csv("./spotify_data.csv")
spotify_dataset.head()
```

	Unnamed: 0	name	album	artist	release_date	length	popularity	danceability	acousticness	energy	instrumentalness	liveness	loudness	speechiness
0	0	Anyone	Anyone	Justin Bieber	2021-01-01	190779	92	0.686	0.1810	0.538	0.000003	0.1130	-8.026	0.0345
1	1	Therefore I Am	Therefore I Am	Billie Eilish	2020-11-12	174321	93	0.889	0.2180	0.340	0.130000	0.0550	-7.773	0.0697
2	2	All Girls Are The Same	Goodbye & Good Riddance	Juice WRLD	2018-12-10	165819	86	0.671	0.0769	0.529	0.000335	0.0856	-7.226	0.3070
3	3	34+35	Positions	Ariana Grande	2020-10-30	173710	93	0.830	0.2370	0.585	0.000000	0.2480	-6.476	0.0940
4	4	All I Want for Christmas Is You	Merry Christmas	Mariah Carey	1994-11-01	241106	89	0.336	0.1640	0.627	0.000000	0.0708	-7.463	0.0384

```
print(spotify_dataset.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1750 entries, 0 to 1749
Data columns (total 16 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   Unnamed: 0        1750 non-null   int64
 1   name              1750 non-null   object
 2   album             1750 non-null   object
 3   artist            1750 non-null   object
 4   release_date      1750 non-null   object
 5   length            1750 non-null   int64
 6   popularity        1750 non-null   int64
 7   danceability      1750 non-null   float64
 8   acousticness      1750 non-null   float64
 9   energy            1750 non-null   float64
 10  instrumentalness  1750 non-null   float64
 11  liveness          1750 non-null   float64
 12  loudness          1750 non-null   float64
 13  speechiness       1750 non-null   float64
 14  tempo             1750 non-null   float64
 15  time_signature    1750 non-null   int64
dtypes: float64(8), int64(4), object(4)
memory usage: 218.9+ KB
None
```

```
print(spotify_dataset.describe())
```

```
       Unnamed: 0         length   popularity  danceability  acousticness  \
count  1750.000000    1750.000000  1750.000000   1750.000000   1750.000000
mean    874.500000  195436.638286    69.506286      0.651688      0.298371
std     505.325802   47465.014667    24.158489      0.175828      0.301546
min       0.000000   34533.000000     0.000000      0.000000      0.000035
25%     437.250000  167280.500000    68.000000      0.553250      0.051825
50%     874.500000  193838.000000    77.000000      0.671000      0.183000
75%    1311.750000  222346.000000    83.000000      0.779000      0.499000
max    1749.000000  530253.000000   100.000000      0.980000      0.996000

            energy  instrumentalness     liveness     loudness  speechiness  \
count  1750.000000       1750.000000  1750.000000  1750.000000  1750.000000
mean      0.592553          0.053457     0.177332    -7.867144     0.118890
std       0.205806          0.195213     0.129798     5.190363     0.112192
min       0.000020          0.000000     0.032700   -40.449000     0.000000
25%       0.479000          0.000000     0.101000    -8.440250     0.040200
50%       0.623000          0.000000     0.125000    -6.522000     0.064850
75%       0.739750          0.000095     0.213750    -5.131750     0.160000
max       0.997000          1.000000     0.945000    -1.465000     0.777000

             tempo  time_signature
count  1750.000000     1750.000000
mean    122.115411        3.932000
std      31.493535        0.396459
min       0.000000        0.000000
25%      94.736000        4.000000
50%     123.049500        4.000000
75%     143.929000        4.000000
max     211.968000        5.000000
```

```
# popularity rankings

pop_rank = spotify_dataset.sort_values(by='popularity', ascending=False)
```

```
# pop_rank = pop_rank.drop_duplicates(inplace=False)
```

```
spotify_dataset.duplicated().sort_index().value_counts()
# no duplicated rows
```

```
False    1750
Name: count, dtype: int64
```

6.3 Basic Analysis

```
spotify_dataset.iloc[:,1:5].sort_values(by='release_date')
```

	name	album	artist	release_date
453	Jingle Bell Rock	Jingle Bell Rock/Captain Santa Claus (And His ...	Bobby Helms	1957-12-02
577	Let It Snow! Let It Snow! Let It Snow!	A Winter Romance	Dean Martin	1959-01-01
437	Let It Snow! Let It Snow! Let It Snow!	A Winter Romance	Dean Martin	1959-01-01
933	The Christmas Song (Merry Christmas To You)	The Christmas Song (Expanded Edition)	Nat King Cole	1962
1368	The Christmas Song (Merry Christmas To You)	The Christmas Song (Expanded Edition)	Nat King Cole	1962
...	...	...	...	...
563	Love Yourself	2000s Love Songs	Various Artists	2021-01-25
1111	Cuando Tú Quieras	Perreo En Los Venas Vol. 5	Various Artists	2021-01-25
1592	6 German Dances, K. 600: No. 1 in C Major	Start Your Day With Mozart	Wolfgang Amadeus Mozart	2021-01-26
1596	6 German Dances, K. 600: No. 5 in G Major	Start Your Day With Mozart	Wolfgang Amadeus Mozart	2021-01-26
1594	6 Minuets K. 105 (attribution doubtful): No. 6...	Start Your Day With Mozart	Wolfgang Amadeus Mozart	2021-01-26

1750 rows × 4 columns

```
# query artist's songs
spotify_dataset.iloc[:,1:5].query('artist=="Wolfgang Amadeus Mozart"').sort_values(by='album', ascending=True)
```

	name	album	artist	release_date
1447	3 German Dances, K. 605: No. 2 in G Major	Mozart for Brainpower	Wolfgang Amadeus Mozart	2021-01-19
1584	6 German Dances, K. 600: No. 2 in F Major	Mozart for Brainpower	Wolfgang Amadeus Mozart	2021-01-19
1586	6 German Dances, K. 600: No. 1 in C Major	Mozart for Brainpower	Wolfgang Amadeus Mozart	2021-01-19
1588	6 German Dances, K. 600: No. 5 in G Major	Mozart for Brainpower	Wolfgang Amadeus Mozart	2021-01-19
1590	6 Minuets K. 105 (attribution doubtful): No. 6...	Mozart for Brainpower	Wolfgang Amadeus Mozart	2021-01-19
1592	6 German Dances, K. 600: No. 1 in C Major	Start Your Day With Mozart	Wolfgang Amadeus Mozart	2021-01-26
1594	6 Minuets K. 105 (attribution doubtful): No. 6...	Start Your Day With Mozart	Wolfgang Amadeus Mozart	2021-01-26
1596	6 German Dances, K. 600: No. 5 in G Major	Start Your Day With Mozart	Wolfgang Amadeus Mozart	2021-01-26
1448	3 German Dances, K. 605: No. 3 in C Major, Tri...	Stay at Home with Mozart	Wolfgang Amadeus Mozart	2021-01-16
1627	7 Menuets, K. 65a: No. 1 in G Major	Wolfgang Amadeus Mozart: Essential Orchestral ...	Wolfgang Amadeus Mozart	2021-01-22

```
spotify_dataset['album'].value_counts()
```

```
album
Dangerous: The Double Album                106
Shoot For The Stars Aim For The Moon        27
Goodbye & Good Riddance                     25
Legends Never Die                           23
Positions                                   16
                                          ...
Uptown Special                               1
Diva (feat. Lil Tecca)                       1
NOVA                                         1
Die Lit                                      1
Personal Problems                            1
Name: count, Length: 849, dtype: int64
```

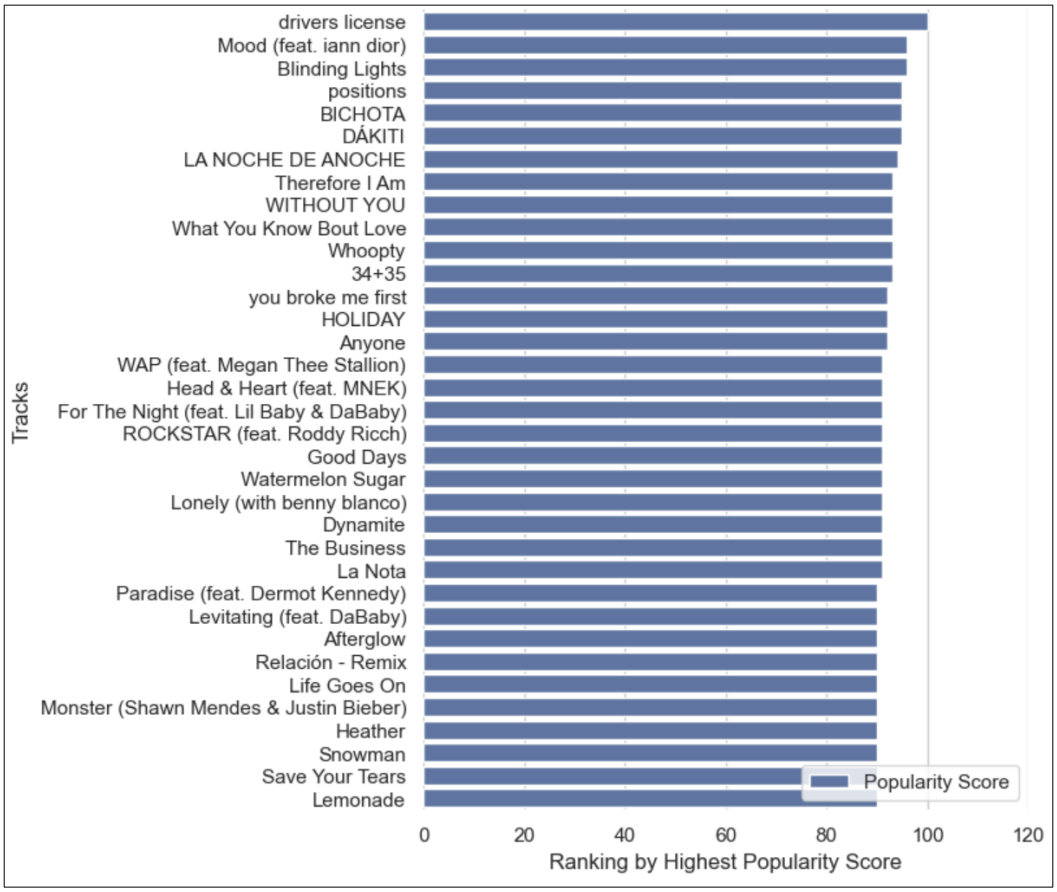
```
spotify_dataset['artist'].value_counts()
```

```
artist
Various Artists    150
Morgan Wallen      119
Juice WRLD          70
Miracle Tones       45
Ariana Grande       39
                  ...
Sia                  1
Lorde                1
Chuck Berry          1
MarMar Oso           1
Big Havi             1
Name: count, Length: 501, dtype: int64
```

6.4 Popularity Ranking

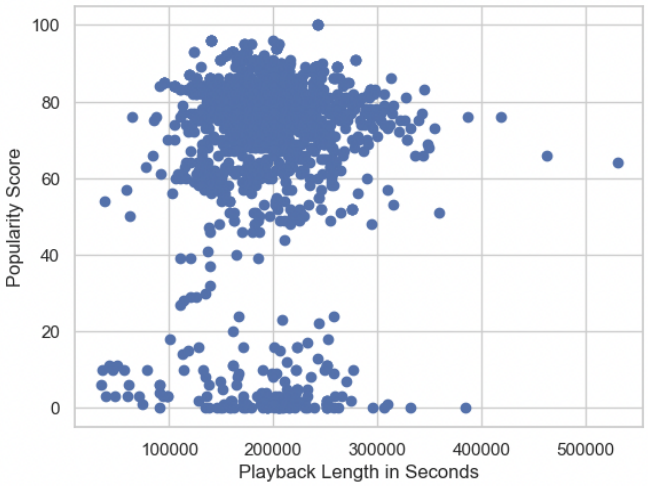
```
# Popularity ranking

sns.set_theme(style="whitegrid")
f, ax = plt.subplots(figsize=(6, 8))

pop_rank = spotify_dataset.sort_values("popularity", ascending=False)
pop_rank.drop_duplicates(keep='first', inplace=True)
pop_rank = pop_rank.head(115)

sns.barplot(x='popularity', y="name",data=pop_rank,
            label="Popularity Score", color="b")

ax.legend(ncol=2, loc="lower right", frameon=True)
ax.set(xlim=(0, 120), ylabel="Tracks",
       xlabel="Ranking by Highest Popularity Score")
sns.despine(left=True, bottom=True)

plt.savefig("output1.jpg",dpi=300,bbox_inches='tight' )
plt.show()

# gives Top 35 ranked by popularity score
```

Top 35 Songs ranked by Popularity Score

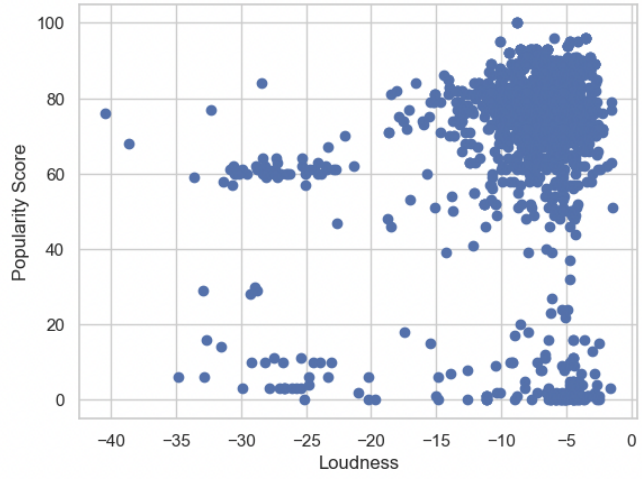


6.5 Scatter Plot

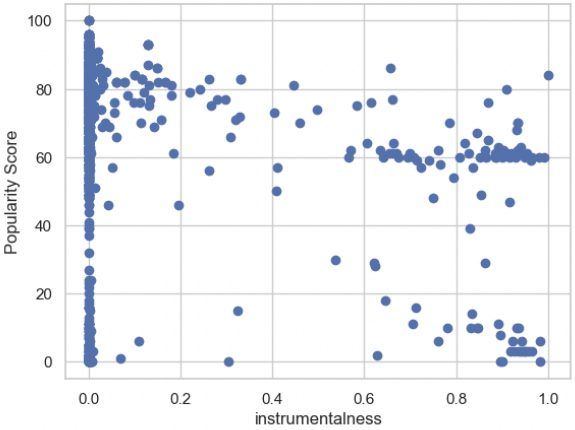
```
plt.scatter(spotify_dataset['length'], spotify_dataset['popularity'])
plt.ylabel("Popularity Score")
plt.xlabel("Playback Length in Seconds")
plt.show()
```




```
plt.scatter(spotify_dataset['loudness'],spotify_dataset['popularity'])
plt.ylabel("Popularity Score")
plt.xlabel("Loudness")
plt.show()
```




```
plt.scatter(spotify_dataset['instrumentalness'],spotify_dataset['popularity'])
plt.ylabel("Popularity Score")
plt.xlabel("instrumentalness")
plt.show()
```



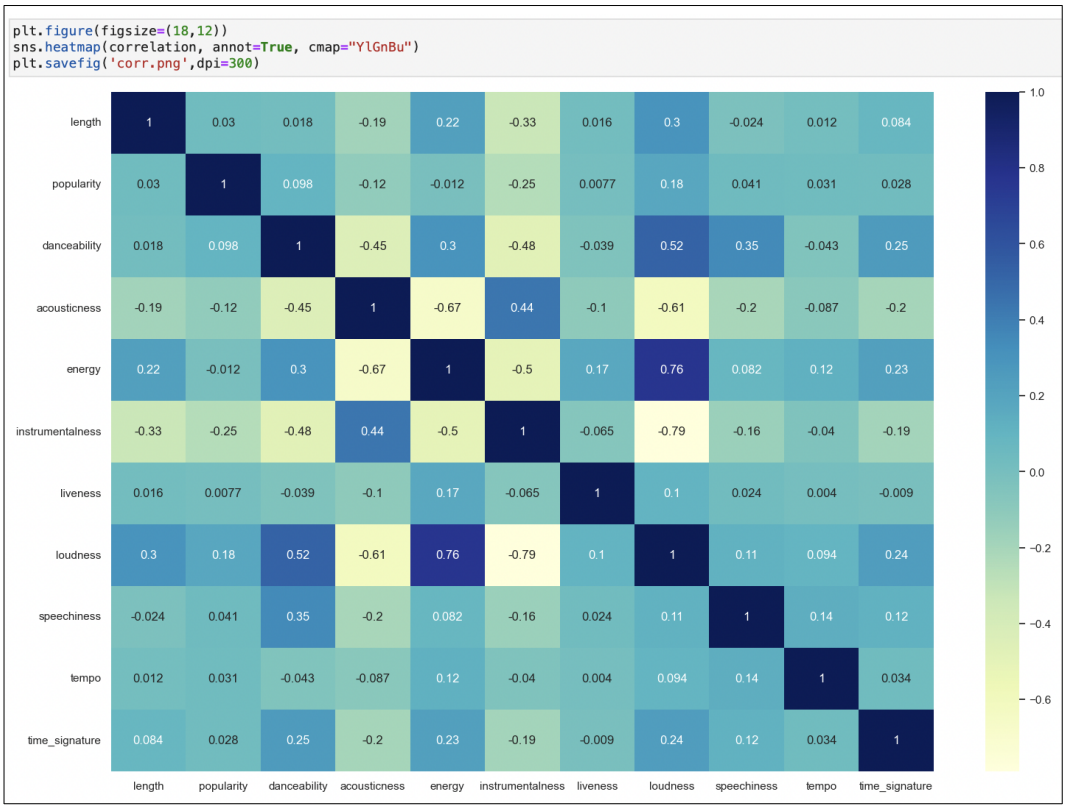
6.6 Correlation Analysis

```
# correaltion analysis

correlation = df.corr()
```

```
print(correlation['popularity'].sort_values(ascending=False))
```

```
popularity           1.000000
loudness             0.176528
danceability         0.097703
speechiness          0.040669
tempo                0.030692
length               0.029566
time_signature       0.028490
liveness             0.007742
energy              -0.011583
acousticness        -0.124462
instrumentalness    -0.252995
Name: popularity, dtype: float64
```

```
plt.figure(figsize=(18,12))
sns.heatmap(correlation, annot=True, cmap="YlGnBu")
plt.savefig('corr.png',dpi=300)
```



7. CONCLUSION

The results from the this project gave insight into how music track's properties affect its performance in the marketplace. Basic Pandas functions enabled us to understand the dataset and its attributes/features. On analyzing, we infer from them that music tracks or songs which have a loud sound profile throughout and a danceability feature i.e **dance music and pop music with lyrics, tend to score high in popularity and rank high in charts.** Whereas **Instrumental and Classical music tend to rank lower in popularity.** Most tracks have **2mins to 5mins of playback time.** Instrumental tracks are by nature least popular category of music. Other observations include trends in popularity with loudness, length and instrumentalness. High correlation was observed with loudness and energy feature. Least correlation was observed with loudness and instrumental feature. Again these findings are convincing by nature of our understanding of music.

REFERENCES

Dataset Source:

https://github.com/BatuhanSeremet/Spotify-DataAnalysis

Paper:

Pampalk, E., Dixon, S., & Widmer, G. (2005). Exploring music collections by browsing different views. In Proceedings of the 6th International Conference on Music Information Retrieval (ISMIR) (pp. 483-490)

Other :

https://seaborn.pydata.org

https://pandas.pydata.org

https://matplotlib.org