

Spotify Data Analysis Using Python

*A Mini-Project Report submitted in partial fulfilment of the
requirements
for the award of degree of*

**MASTER OF COMPUTER APPLICATIONS
of**



Visvesvaraya Technological University

By

DEEKSHITH KUMAR C L (1CD21MC017)

CHETHANKUMAR K (1CD21MC015)

Under the Guidance of

Prof. Sunil Kumar K N



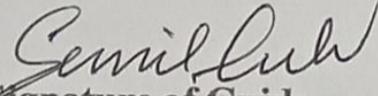
**Department of Master of Computer Applications,
Cambridge Institute of Technology,
K R Puram,
Bangalore-560036.
MARCH 2023**



**Department of Master of Computer Applications,
Cambridge Institute of Technology,
KR Puram, Bangalore-560036.**

CERTIFICATE

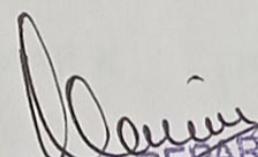
This is to Certify that DEEKSHITH KUMAR C L and CHETHANKUMAR K has completed there III semester Mini Project work Entitled “Spotify Data Analysis Using Python” as a partial fulfilment for the award of Master of Computer Applications degree, during the academic year 2022-23 under my supervision.


Signature of Guide
Prof. Sunil Kumar KN
Asst. Prof., MCA,
Cambridge Institute of Technology

External Viva

Name of Examiners

1. 
2. 


Signature of HOD
HEAD OF DEPARTMENT
DEPARTMENT OF MCA
Prof. Diwakar Karadi
HOD, MCA Institute of Technology
Cambridge Institute of Technology
Cambridge, Bangalore - 560 036
KR Puram, Institute of Technology

Signature with Date

DECLARATION

DEEKSHITH KUMAR C L and **CHETHANKUMAR K**, student of 3rd Semester MCA, Cambridge Institute of Technology, bearing hereby declare that the project entitled "**Spotify Data Analysis Using Python**" has been carried out by us under the supervision of Prof. Monika M., Asst. Prof., MCA, Cambridge Institute of Technology and submitted in partial fulfilment of the requirements for the award of the Degree of Master of Computer Applications by the Visvesvaraya Technological University during the academic year 2022-23. This report has not been submitted to any other Organization/University for any award of degree or certificate.

Project Member:

DEEKSHITH KUMAR C L (1CD21MC017)

CHETHANKUMAR K (1CD21MC015)

ACKNOWLEDGMENT

At the very outset I am very much thankful to my **Chairman, Shri. D. K. Mohan, CEO, Shri. Nithin Mohan** for being a strong support to our college and providing a well-equipped department (MCA).

I am thankful and grateful to our **Principal, Dr. Indumathi G**, for her kind permission and co-operation for making me to take up this mini project and make use of facility, which are available in our college.

I express my heartiest thanks to my beloved **Head of the Department, Prof. Diwakar Karadi** for being source of motivation during the development of my project.

And I am grateful for the generous help from my internal guide, **Prof. Sunil Kumar K N**, Department of MCA, Cambridge Institute of technology for the valuable guidance as the project guide for successful completion of this project work.

I would also like to express my sincere thanks to my parents for supporting me throughout my life and making me stand where I am right now. Finally, I thank my friends who have given their valuable time with co-operation and help throughout this project work.

CONTENT:

SL. NO	SUBJECT	PG. NO
1.	ABSTRACT	1
2.	INTRODUCTION	2
3.	SPECIFICATION REQUIREMENTS • Software requirement specification • Hardware requirement specification	3
4.	SYSTEM ANALYSIS	4
5.	TECHNOLOGIES USED	5-7
7.	SYSTEM DESIGN & IMPLEMENTATION	8-16
8.	CODING	17-19
9.	TESTING	20-21
10.	CONCLUSION	22-23
11.	BIBLIOGRAPHY	24

CHAPTER-1

ABSTRACT

The Spotify dataset provides insight into users data about which songs they listen to, and not just the popularity of tracks, but also features of the tracks they have in their library is recorded in their database.

In this project I will be analyzing a track's popularity based on several audio features

Provided in the dataset and find whether I can predict a track's popularity from key features about the song

Analysing user's listening profile to enable Spotify to suggest and acquire similar tracks on their platform to improve user experience

The plan is to analyse relationship between popularity and different features of the song, and maybe later perform cluster analysis using K-means method to provide song recommendation based on recent user listening on Spotify.

This is mainly useful to market songs to the Spotify users and improve their experience while using it. This analysis will help better understand the different clusters and enable Spotify to make a better targeted content distribution that would be helpful for the developers and the marketing team to analyse trends and help them to segment users better and try to increase profits and provide a better user experience

CHAPTER-2

INTRODUCTION

Spotify is a popular music streaming platform that provides access to millions of songs. With the help of Python, it's possible to analyse Spotify data and gain insights into user behaviour, music trends, and more.

To get started with Spotify data analysis, you can use the Spotify Web API to access data about tracks, artists, albums, and playlists. You can also use the Spotify library, a Python wrapper for the Spotify Web API, to make it easier to access this data.

Once you have access to the data, you can use Python's data analysis libraries such as pandas, numpy and matplotlib to perform data exploration, cleaning, transformation, and visualization. You can also use machine learning and natural language processing techniques to build predictive models, sentiment analysis, and personalized recommendations.

Overall, Python is a powerful tool for analysing Spotify data, and it can help you gain valuable insights into user behaviour, music trends, and more.

CHAPTER-3

REQUIREMENT ANALYSIS

Requirement analysis for data analysis encompasses three major tasks: formulation, requirements gathering and analysis modeling. During formulation, the basic motivation and goals are identified, and the categories of users are defined. In the requirements gathering phase, the content and functional requirements are listed and interaction scenarios written from end-user's point-of-view are developed. This intent is to establish a basic understanding of why the analysis is done, who will use it, and what problems it will solve for its users.

3.1 SOFTWARE REQUIREMENT SPECIFICATION

Operating System	:	Windows XP/7
Application	:	Anaconda Navigator
Platform	:	Jupyter Notebook
Language	:	Python

3.2 HARDWARE REQUIREMENT SPECIFICATION

Processor	:	Standard processor with a speed of 2.0GHz
RAM	:	4BG
Hard Disk	:	50 GB or more
Monitor	:	Standard colour monitor
Keyboard	:	Standard keyboard

CHAPTER-4

SYSTEM ANALYSIS

System analysis is the process of gathering and interpreting facts, diagnosing problems and using the information to recommend improvements on the system. System analysis is problem solving activity that requires intensive communication between the system users and system developers.

It is a process of collecting and interpreting facts, identifying the problems, and decomposition of a system into its components.

System analysis is conducted for the purpose of studying a system or its parts in order to identify its objectives. It is a problem solving technique that improves the system and ensures that all the components of the system work efficiently to accomplish their purpose

System analysis or study is an important phase of any system development process. The system is viewed as a whole, the inputs are identified and the system is subjected to close study to identify the problem areas. The solutions are given as a proposal. The proposal is reviewed on user request and suitable changes are made. This loop ends as soon as the user is satisfied with the proposal.

CHAPTER-5

TECHNOLOGIES USED

5.1 Anaconda Navigator:

Anaconda Navigator is a desktop graphical user interface (GUI) included in Anaconda distribution that allows users to launch applications and manage CONDA packages, environments and channels without using command-line commands. Navigator can search for packages on Anaconda Cloud or in a local Anaconda Repository, install them in an environment, run the packages and update them. It is available for Windows, macOS and Linux.

Anaconda distribution comes with over 250 packages automatically installed, and over 7,500 additional open-source packages can be installed from PYPI as well as the CONDA package and virtual environment manager. It also includes a GUI, Anaconda Navigator as a graphical alternative to the command-line interface (CLI).

The following applications are available by default in Navigator:

- JUPYTER Lab
- JUPYTER Notebook
- Qt Console
- Spyder
- Glue
- Orange
- RStudio
- Visual Studio Code
- Visual Studio Code

5.2 JUPYTER Notebook:

The JUPYTER Notebook is an open-source web application that you can use to create and share documents that contain live code, equations, visualizations, and text. Jupyter Notebooks are a spin-off project from the IPython project, which used to have an IPython Notebook project itself. The name, Jupyter, comes from the core supported programming languages that it supports: Julia, Python, and R. Jupyter ships with the IPython kernel, which allows you to write your programs in Python, but there are currently over 100 other kernels that you can also use.

5.3 Python programming:

Python is a computer programming language often used to build websites and software, automate tasks, and conduct data analysis. Python is a general-purpose language, meaning it can be used to create a variety of different programs and isn't specialized for any specific problems. This versatility, along with its beginner-friendliness, has made it one of the most-used programming languages today. A survey conducted by industry analyst firm Red Monk found that it was the most popular programming language among developers in 2020.

Python is commonly used for developing websites and software, task automation, data analysis, and data visualization. Since it's relatively easy to learn, Python has been adopted by many non-programmers such as accountants and scientists, for a variety of everyday tasks, like organizing finances.

Some of the common ways Python is used

- Data analysis and machine learning
- Web development
- Automation or scripting
- Software testing and prototyping

Spotify Data Analysis

5.4 DATASET:

A dataset (or data set) is a collection of data, usually presented in tabular form. Each column represents a particular variable. Each row corresponds to a given member of the dataset in question. It lists values for each of the variables, such as height and weight of an object. Each value is known as a datum. The dataset may comprise data for one or more members, corresponding to the number of rows.

The dataset used in project is TRACKS.csv file is a excel sheet Dataset contains more than 160.000 songs collected from Spotify Web API. The features include song, artist, release date as well as some characteristics of song such as acousticness, danceability, loudness, tempo and so on. Date range is from 1921 to 2020.

A1	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
1	id	name	popularity	duration_explicit	artists	id_artists	release_d	danceabil	energy	key	loudness	mode	speechint	acousticn	instrument	liveness	valence	tempo	time_signature	3
2	35wgrf4dp Carve	6	126903	0	["Ul"]	["45tHt06Xr #####"]	0.645	0.445	0	-13.338	1	0.451	0.674	0.744	0.151	0.127	104.851			3
3	021h4sdg CapÃ±tulo	0	98200	0	["Fernand"]	["14]tPCOx #####"]	0.695	0.263	0	-22.136	1	0.957	0.797	0	0.148	0.655	102.009	1		
4	07ASyeh! Vivo para	0	181640	0	["Ignacio C"]	["5LU0oJb# #####"]	0.434	0.177	1	-21.18	1	0.0512	0.994	0.0218	0.212	0.457	130.418			5
5	0BFmquh! El Prisione	0	176907	0	["Ignacio C"]	["5LU0oJb# #####"]	0.321	0.0946	7	-27.961	1	0.0504	0.995	0.918	0.104	0.397	169.98			3
6	0by9Gloq! Lady of th	0	163080	0	["Dick Hay"]	["3BUGzsy"]	1922	0.402	0.158	3	-16.9	0	0.039	0.989	0.13	0.311	0.196	103.22		4
7	0BXUJHRN Ave Maria	0	178933	0	["Dick Hay"]	["3BUGzsy"]	1922	0.227	0.261	5	-12.343	1	0.0382	0.994	0.247	0.0977	0.0539	118.891		4
8	0Dd5ImXt La Butte R	0	134467	0	["Francis N"]	["2nuMRG"]	1922	0.51	0.355	4	-12.833	1	0.124	0.965	0	0.155	0.727	85.754		5
9	0IAOHjusC La Java	0	161427	0	["Mistingu"]	["4AargXfD"]	1922	0.563	0.184	4	-13.757	1	0.0512	0.993	1.55E-05	0.325	0.654	133.088		3
10	0ig11UCR8 Old Fashir	0	310073	0	["Greg Fle"]	["SnWish5"]	1922	0.488	0.475	0	-16.222	0	0.0399	0.62	0.00645	0.107	0.544	139.952		4
11	0IV4iqv21 MartÃ-n F	0	181173	0	["Ignacio C"]	["5LU0oJb# #####"]	0.548	0.0391	6	-23.228	1	0.153	0.996	0.933	0.148	0.612	75.595			3
12	0YYGe21c CapÃ±tulo	0	99100	0	["Fernand"]	["14]tPCOx #####"]	0.676	0.235	11	-22.447	0	0.96	0.794	0	0.21	0.724	96.777			3
13	0PE42H6t CapÃ±tulo	0	132700	0	["Fernand"]	["14]tPCOx #####"]	0.75	0.229	2	-22.077	1	0.955	0.570	0	0.314	0.531	102.629			3
14	0PH9AACg Lazy Bol	0	157333	0	["Ignacio C"]	["45tHt06Xr #####"]	0.298	0.46	1	-18.645	1	0.453	0.521	0.856	0.436	0.402	87.921			4
15	0QITDO05 Tu Veras	1	186800	0	["Lucien B"]	["4mSoul4"]	1922	0.703	0.28	0	-15.39	1	0.174	0.995	6.84E-05	0.163	0.897	127.531		4
16	0TWSNj5C Elle Prenc	0	172027	0	["FÃ©lix X"]	["7DII0K9L"]	1922	0.709	0.289	2	-14.978	1	0.18	0.967	0	0.119	0.84	107.515		4
17	0ctC9CYJR CapÃ±tulo	0	96600	0	["Fernand"]	["14]tPCOx #####"]	0.687	0.198	4	-24.264	0	0.962	0.754	0	0.197	0.478	78.453			1
18	0eb1PhNx CapÃ±tulo	0	103200	0	["Fernand"]	["14]tPCOx #####"]	0.8	0.171	8	-24.384	1	0.953	0.67	0	0.123	0.893	59.613			3
19	0prKU6GE CapÃ±tulo	0	95800	0	["Fernand"]	["14]tPCOx #####"]	0.7	0.208	2	-23.874	1	0.956	0.691	0	0.441	0.613	85.739			1
20	0kCB1bOV Ca C'est U	0	188000	0	["Victor Bc"]	["7VVR02J"]	1922	0.352	0.334	5	-13.038	1	0.0594	0.996	0.00746	0.36	0.414	76.403		4
21	0LB0sVJ! El Vendav	0	153533	0	["Ignacio C"]	["5LU0oJb# #####"]	0.37	0.372	2	-17.138	1	0.0865	0.985	0.000681	0.929	0.753	159.669			4
22	0pKUGQd! Ta Bouch!	0	180933	0	["Jeanne S"]	["4ZACx0C"]	1922	0.335	0.304	7	-14.062	1	0.0506	0.996	0.0832	0.102	0.215	76.576		3
23	0wAAyFir CapÃ±tulo	0	109100	0	["Fernand"]	["14]tPCOx #####"]	0.704	0.244	0	-25.353	1	0.959	0.728	0	0.168	0.644	84.489			3
24	0xJC9XSA La Maleva	0	181440	0	["Ignacio C"]	["5LU0oJb# #####"]	0.339	0.0958	8	-26.944	1	0.053	0.994	0.969	0.123	0.219	86.279			3
25	0rycrYukw CapÃ±tulo	0	113200	0	["Fernand"]	["14]tPCOx #####"]	0.655	0.214	0	-23.337	1	0.962	0.665	0	0.279	0.761	76.087			1

CHAPTER-6**SYSTEM DESIGN AND IMPLEMENTATION****SCREENSHOT-01:**

The screenshot shows a Jupyter Notebook interface with the following details:

- Title Bar:** Shows tabs for "New Tab", "Home Page - Select or create a", "spotify data analysis using python", and "ipykernel". The main title is "jupyter spotify data analysis using python Last Checkpoint: 01/15/2023 (autosaved)".
- Toolbar:** Includes File, Edit, View, Insert, Cell, Kernel, Widgets, Help, and a Python 3 (ipykernel) icon.
- Code Cells:**
 - In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```
 - In [3]:

```
df_tracks=pd.read_csv("C:/Users/DELL/Downloads/tracks.csv")
df_tracks.head()
```
 - In [4]:

```
df_tracks.isnull().sum()
```
- Data Output:** The output of In [3] is a Pandas DataFrame with 5 rows and 14 columns. The columns are: Id, name, popularity, duration_ms, explicit, artists, id_artists, release_date, danceability, energy, key, l, mode, and l2.
- System Status:** The taskbar at the bottom shows system information including weather (23°C, sunny), network, battery, and system date/time (10:00 AM, 25-02-2023).

Spotify Data Analysis

SCREENSHOT-02:

```
In [6]: df_track.head()
Out[6]:
   id      name  popularity  duration_ms  explicit  artists  id_artists release_date  danceability  energy  key
0  35wgR4Xet315WEVxaIQ  Can't  6  125803  0  [UAT]  145e0dXeXer0e4LBEPv9t  1922-02-22  0.645  0.4465  0
1  021Hr4sqgPegSk7JbKvY  Capture 2.18 -  9  98205  0  [Fernando Paez]  145e0dXeXer0e4LBEPv9t  1922-05-01  0.695  0.2835  0
2  07AbjentShoeVJUADkhs  Vivo para  9  181640  0  [Graça Corrêa]  145e0dXeXer0e4LBEPv9t  1922-03-21  0.434  0.1775  1
3  D8PhqUunyLThbaHb4k5  El Prisonero -  9  178907  0  [Graça Corrêa]  145e0dXeXer0e4LBEPv9t  1922-03-21  0.321  0.0945  7
4  08y9QaoCWOQenkyw5e  Lady of the  9  183080  0  [Dick Haymes]  145e0dXeXer0e4LBEPv9t  1922  0.402  0.1565  3

In [7]: sort_df=df_track.sort_values("popularity",ascending=False)
sort_df.head()

Out[7]:
   id      name  popularity  duration_ms  explicit  artists  id_artists release_date  danceability  energy  key
0  546130  181tTRhCggZPheP2tUvam  Newspaper Records On Aperitif 20 February 1935  0  69575  0  [Nora Goffe Chester Ladd, Carson Bee]  1935-02-20  0.595  0.262  1
1  546222  0yOCz2V5xVmB1TSEFe50  ドラマ上  0  120440  0  [Helen Moran]  145e0dXeXer0e4LBEPv9t  1949  0.415  0.339  0
2  546221  0y4Dhwed22099jv7egR0O  エリ・アラ  0  173457  0  [Helen Moran]  145e0dXeXer0e4LBEPv9t  1949  0.442  0.179  5
3  546220  0iOmgfRkaOTnAcp3D5pAV  エリ・アラ (EL CHOCLO)  0  205280  0  [Helen Moran]  145e0dXeXer0e4LBEPv9t  1949  0.695  0.497  0
4  546219  0SX33WCPXTKwURQ2m9  ドラマ上  0  105733  0  [Helen Moran]  145e0dXeXer0e4LBEPv9t  1949  0.338  0.389  2
```

SCREENSHOT-03

```
In [4]: df_track.isnull().sum()
Out[4]:
   id      0
   name    71
  popularity  0
 duration_ms  0
  explicit  0
  artists  0
 id_artists  0
 release_date  0
 danceability  0
  energy  0
  key  0
 loudness  0
 mode  0
 speechiness  0
 acousticness  0
 instrumentalness  0
 liveness  0
 valence  0
 tempo  0
 time_signature  0
dtype: int64

In [5]: df_track.shape
```

Spotify Data Analysis

SCREENSHOT-04:

In [8]: df_track.describe().transpose()

Out[8]:

	count	mean	std	min	25%	50%	75%	max
popularity	586672.0	27.570053	18.370642	0.0	13.0000	27.000000	41.00000	100.000
duration_ms	586672.0	230051.167285	126526.087418	3344.0	175093.00000	214893.000000	263867.00000	5621218.000
explicit	586672.0	0.044086	0.205286	0.0	0.0000	0.000000	0.00000	1.000
danceability	586672.0	0.563594	0.166103	0.0	0.4530	0.577000	0.68600	0.991
energy	586672.0	0.542036	0.251923	0.0	0.3430	0.549000	0.74800	1.000
key	586672.0	5.221603	3.519423	0.0	2.0000	5.000000	8.00000	11.000
loudness	586672.0	-10.206067	5.089328	-60.0	-12.8910	-9.243000	-6.48200	5.376
mode	586672.0	0.658797	0.474114	0.0	0.0000	1.000000	1.00000	1.000
speechiness	586672.0	0.104864	0.179893	0.0	0.0340	0.044300	0.07630	0.971
acousticness	586672.0	0.449863	0.348387	0.0	0.0969	0.422000	0.78500	0.996
instrumentalness	586672.0	0.113451	0.266868	0.0	0.0000	0.000024	0.00955	1.000
liveness	586672.0	0.213935	0.184326	0.0	0.0983	0.139000	0.27800	1.000
valence	586672.0	0.552292	0.257671	0.0	0.3460	0.564000	0.76900	1.000
tempo	586672.0	118.464857	29.764108	0.0	95.6000	117.384000	136.32100	246.381
time_signature	586672.0	3.873382	0.473162	0.0	4.0000	4.000000	4.00000	5.000

SCREENSHOT-05:

In [9]: most_popular=df_track[df_track["popularity"]>90].sort_values(by="popularity",ascending=False)
most_popular.head()

Out[9]:

	id	name	popularity	duration_ms	explicit	artists	id_artists	release_date	danceability	energy	key	1
93802	4UyoBOLtHqaGxP12qzhQI	Peaches (feat. Daniel Caesar & Giveon)	100	198082	1	['Justin Bieber', 'Daniel Caesar', 'Giveon']	[1uNFeZAHBGf0mzznpCl3s', '20wkVLutqVOYrc0lxF...]	2021-03-19	0.677	0.696	0	
93803	7IPN2DXIMsVn7XUKiOW1CS	drivers license	99	242014	1	['Olivia Rodrigo']	[1McMsnEEIThX1knmY4oiG]	2021-01-08	0.585	0.436	10	
93804	3Ofmpyhv5UAQ70mENzB277	Astronaut In The Ocean	98	132780	0	['Masked Wolf']	[1uU7g3DNsbsu0jSEqZlEd]	2021-01-06	0.778	0.695	4	
92810	5QO79kh1walcV47BqGRL3g	Save Your Tears	97	215627	1	['The Weeknd']	[1Xyo4u8uXC1ZmMpafF05PJ]	2020-03-20	0.680	0.826	0	
92811	6tDDoYlxWvMLTdKpjFkc1B	telepatia	97	160191	0	['Kali Uchis']	[1U1el3k54VvEUzo3ybLPIM]	2020-12-04	0.653	0.524	11	

In [10]: df_track[["artists"]].iloc[10]

Out[10]: artists ['Victor Boucher']
Name: 10, dtype: object

Spotify Data Analysis

SCREENSHOT-06:

```
In [10]: df_track[["artists"]].iloc[18]
Out[10]: artists    ['Victor Boucher']
Name: 18, dtype: object

In [11]: df_track["duration"] = df_track["duration_ms"].apply(lambda x:round(x/1000))
df_track.drop("duration_ms",inplace=True,axis=1)

In [12]: df_track.head()
Out[12]:
```

	id	name	popularity	explicit	artists	id_artists	release_date	danceability	energy	key	loudness	mode
0	35WgR4jXel316WEWs1Q	Carve	6	0	[IU]	[45ltt06XoI0lo4LBEVpls]	1922-02-22	0.645	0.4450	0	-13.338	
1	021ht4sdgPcrDgSk7JTbKY	Capítulo 2 16 - Banquero Anarquista	0	0	[Fernando Pessoa]	[14jtPCOoNZwquk5wd9DxY]	1922-06-01	0.695	0.2630	0	-22.136	
2	07ASyehTSnoedViJAZkNnc	Vivo para Quererete - Remasterizado	0	0	[Ignacio Corsini]	[5LIOoJbxVSAMkBS2fUm3XZ]	1922-03-21	0.434	0.1770	1	-21.160	
3	08FmqUhxtyLTn6pAh6bk45	Ei Prisionero - Remasterizado	0	0	[Ignacio Corsini]	[5LIOoJbxVSAMkBS2fUm3XZ]	1922-03-21	0.321	0.0946	7	-27.961	
4	08y9GfoqCWlOGsKdw0jr5e	Lady of the Evening	0	0	[Dick Haymes]	[3BJGZsyX9sJchTqcSA7Su]	1922	0.402	0.1580	3	-16.900	

SCREENSHOT-07:

```
In [13]: df_track["artists"].value_counts()
Out[13]: 
['Die drei ???']      3856
['TXRG Retro-Archiv'] 2006
['Benjamin Blümchen'] 1503
['Bibi Blocksberg']   1472
['Lata (Gangeshkar)'] 1373
...
['IU', 'Jang Yi-jeong']      1
['黃宗潔']                1
['Vincy Chan', '譚麗經']    1
['Dough-Boy']               1
['Gentle Bones', 'Clara Benin'] 1
Name: artists, length: 114030, dtype: int64

In [14]: df_track.info()
```

Column	Non-Null Count	Dtype
0 id	586672	object
1 name	586601	object
2 popularity	586672	int64
3 explicit	586672	int64
4 artists	586672	object
5 id_artists	586672	object
6 release_date	586672	object
7 danceability	586672	float64
8 energy	586672	float64
9 key	586672	int64
10 loudness	586672	float64
11 mode	586672	int64
12 speechiness	586672	float64

Spotify Data Analysis

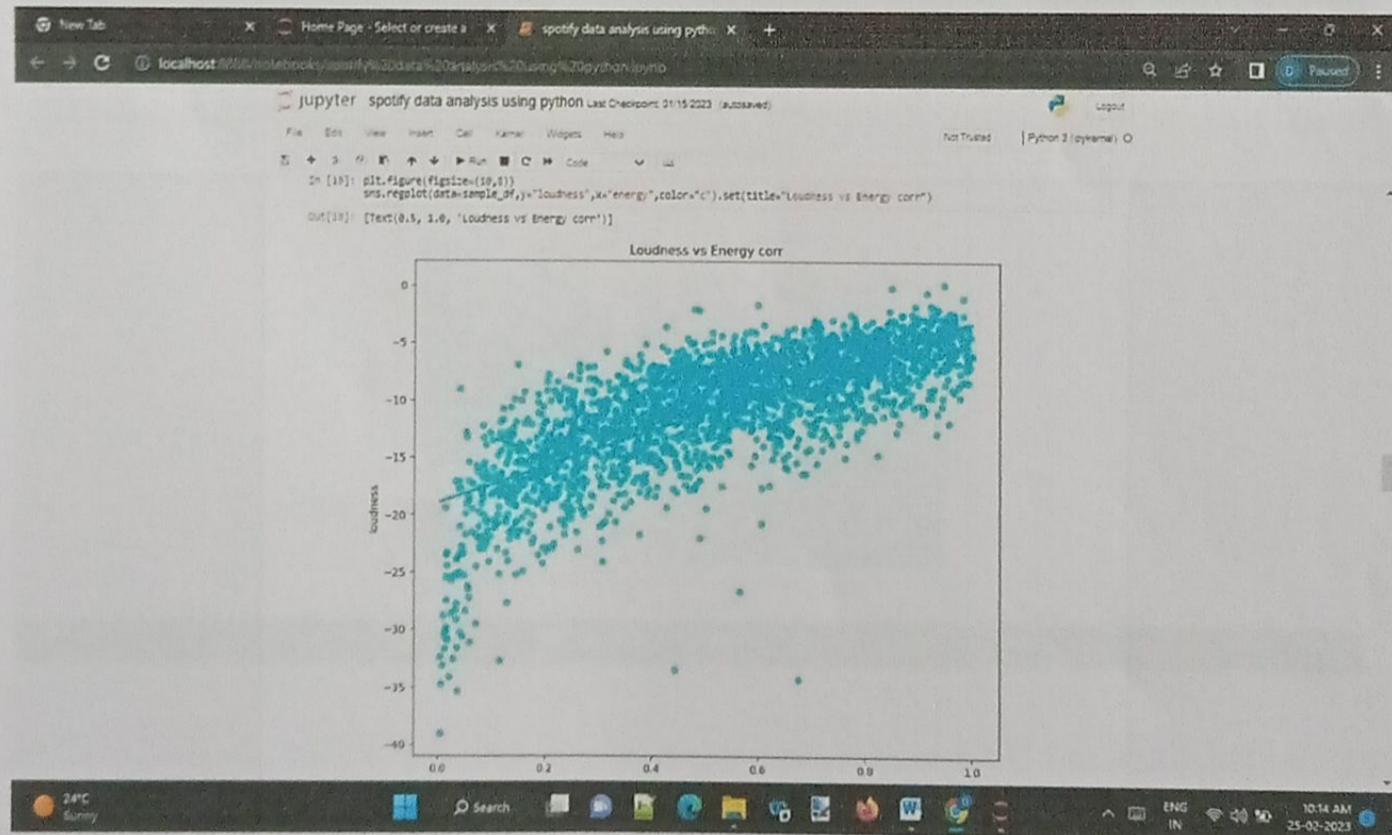
SCREENSHOT-08:

The screenshot shows a Jupyter Notebook interface with a Python code cell and its output. The code generates a correlation heatmap for various Spotify audio features. The heatmap is titled "Corr heatmap between variable". The x and y axes list features such as popularity, danceability, energy, loudness, speechiness, acousticness, instrumentality, liveness, valence, tempo, time_signature, and duration. The correlation values range from -1.000 to 1.000.

```
In [10]: corr=df[feat].drop(['valence','tempo','duration'],axis=1).corr(method='pearson')
plt.figure(figsize=(14,8))
sns.heatmap(corr,df,annot=True,minv=-1,maxv=1,center=0,cmap="inferno")
heatmap.set_title("Corr heatmap between variable")
heatmap.set_xticklabels(heatmap.get_xticklabels(),rotation=90)
heatmap.set_yticklabels(heatmap.get_yticklabels(),)
```

	popularity	danceability	energy	loudness	speechiness	acousticness	instrumentality	liveness	valence	tempo	time_signature	duration
popularity	1	0.13	0.1	-0.13	0.047	0.37	-0.24	0.049	0.0048	0.071	0.097	0.228
danceability	0.19	1	0.24	0.28	0.2	-0.28	-0.21	-0.11	0.31	-0.041	0.15	-0.12
energy	0.3	0.24	1	-0.16	-0.054	-0.72	-0.2	0.12	0.37	0.23	0.19	0.025
loudness	0.39	0.25	0.76	1	0.17	0.52	0.32	0.03	0.28	0.13	0.16	0.00034
speechiness	-0.047	0.2	0.254	0.17	1	0.069	0.1	0.21	0.047	0.097	0.11	0.13
acousticness	-0.17	-0.24	-0.72	-0.52	0.069	1	0.2	-0.0047	-0.18	-0.2	-0.17	-0.064
instrumentality	-0.24	-0.23	-0.2	-0.33	-0.1	0.2	1	-0.039	-0.19	-0.055	-0.042	0.069
liveness	-0.049	0.11	0.12	0.03	0.21	0.0647	0.039	1	-0.4e-05	0.015	0.024	0.021
valence	0.0648	0.53	0.17	0.28	0.047	0.18	-0.18	-0.4e-05	1	0.14	0.31	-0.16
tempo	0.071	-0.041	0.23	0.16	-0.087	-0.2	-0.055	-0.015	0.14	1	0.032	-0.0012
time_signature	-0.087	0.15	0.19	0.16	-0.11	-0.17	0.042	0.024	0.11	0.032	1	0.038
duration	0.028	-0.12	0.029	0.00034	0.15	0.064	0.069	0.0021	-0.16	0.0012	0.038	1

SCREENSHOT-09



Spotify Data Analysis

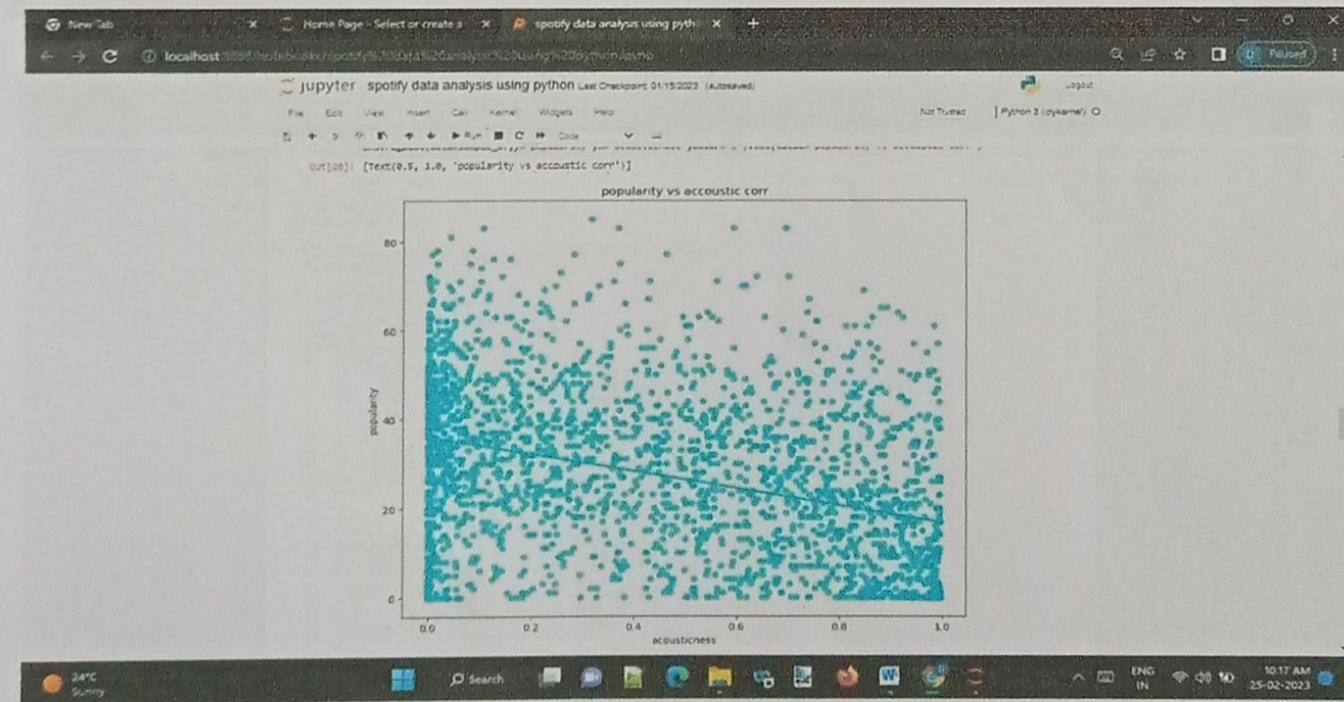
SCREENSHOT-10

The screenshot shows a Jupyter Notebook interface running on a local host. The code cell at the top contains Python code for reading a CSV file and converting the 'dates' column to datetime. The output cell below displays the DataFrame's structure, including its columns, non-null counts, and data types. The terminal at the bottom shows system information like weather (24°C, sunny) and system status (ENG IN, 10:38 AM, 25-02-2023).

```
In [21]: df_track["dates"] = pd.to_datetime(df_track["release_date"])
df_track.info()

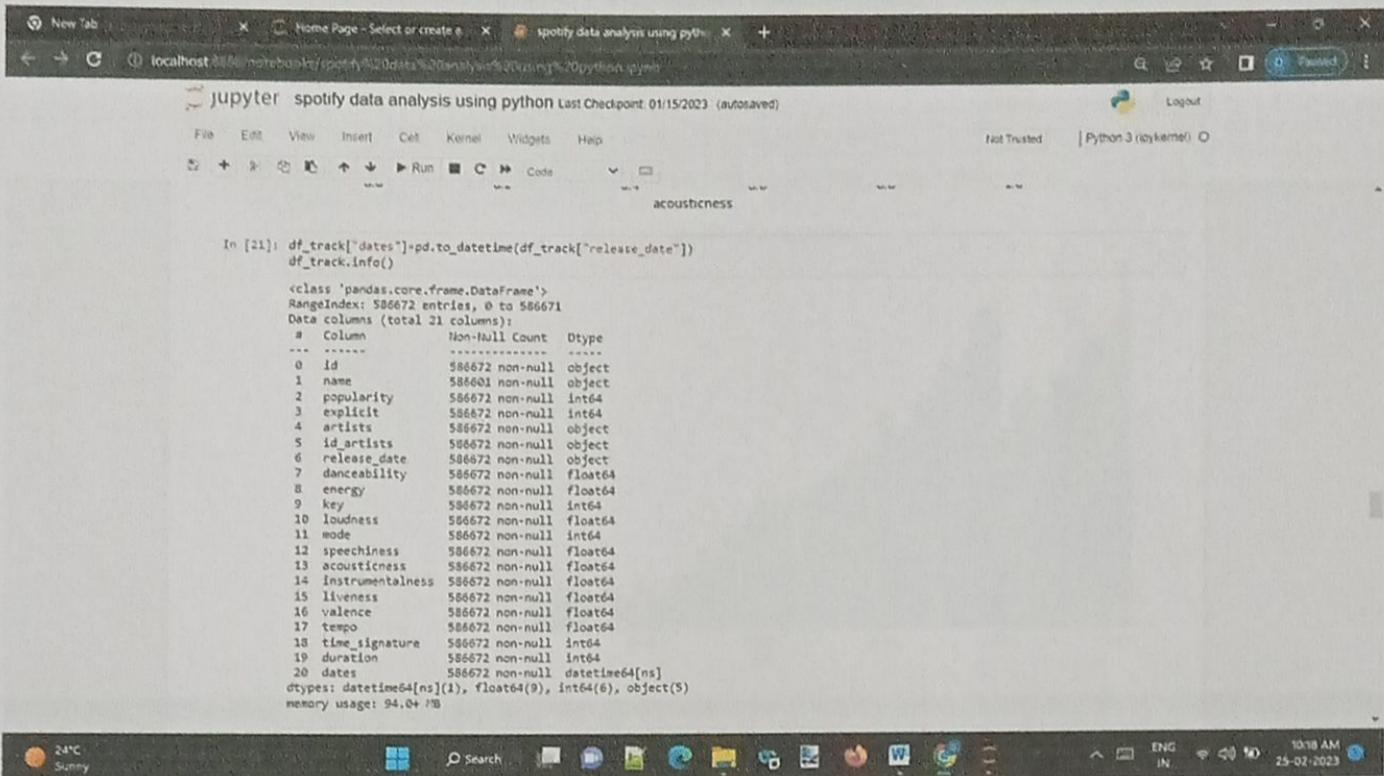
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 586672 entries, 0 to 586671
Data columns (total 21 columns):
 #   Column          Non-Null Count  Dtype  
--- 
 0   id              586672 non-null   object 
 1   name             586672 non-null   object 
 2   popularity       586672 non-null   int64  
 3   explicit         586672 non-null   int64  
 4   artists          586672 non-null   object 
 5   id_artists      586672 non-null   object 
 6   release_date     586672 non-null   object 
 7   danceability     586672 non-null   float64 
 8   energy            586672 non-null   float64 
 9   key               586672 non-null   int64  
 10  loudness          586672 non-null   float64 
 11  mode              586672 non-null   int64  
 12  speechiness       586672 non-null   float64 
 13  acousticness      586672 non-null   float64 
 14  instrumentalness 586672 non-null   float64 
 15  liveliness        586672 non-null   float64 
 16  valence            586672 non-null   float64 
 17  tempo              586672 non-null   float64 
 18  time_signature    586672 non-null   int64  
 19  duration           586672 non-null   int64  
 20  dates              586672 non-null   datetime64[ns]
dtypes: datetime64[ns](1), float64(6), int64(6), object(5)
memory usage: 94.04 MB
```

SCREENSHOT-11



Spotify Data Analysis

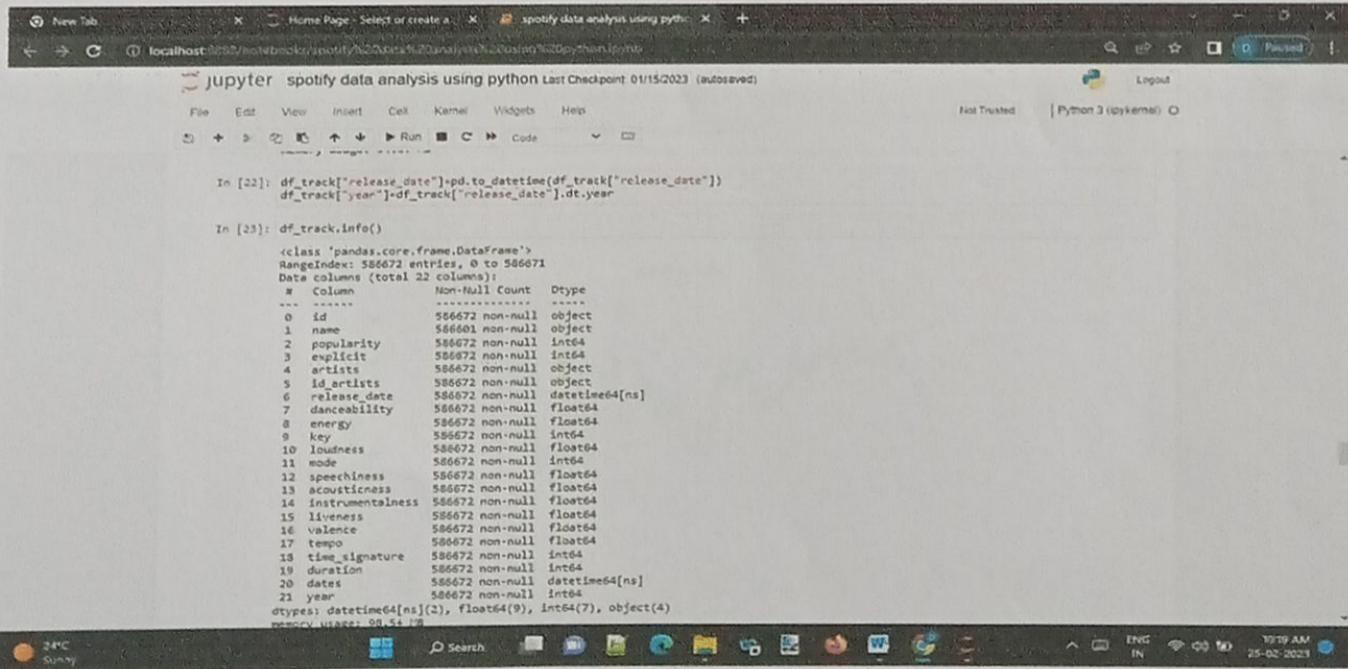
SCREENSHOT-12



```
In [21]: df_track["dates"] = pd.to_datetime(df_track["release_date"])
df_track.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 586672 entries, 0 to 586671
Data columns (total 21 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   id               586672 non-null   object  
 1   name              586601 non-null   object  
 2   popularity        586672 non-null   int64  
 3   explicit          586672 non-null   int64  
 4   artists            586672 non-null   object  
 5   id_artists        586672 non-null   object  
 6   release_date      586672 non-null   object  
 7   danceability       586672 non-null   float64 
 8   energy             586672 non-null   float64 
 9   key                586672 non-null   int64  
 10  loudness           586672 non-null   float64 
 11  mode               586672 non-null   int64  
 12  speechiness        586672 non-null   float64 
 13  acousticness       586672 non-null   float64 
 14  instrumentalness  586672 non-null   float64 
 15  liveness            586672 non-null   float64 
 16  valence             586672 non-null   float64 
 17  tempo               586672 non-null   float64 
 18  time_signature     586672 non-null   int64  
 19  duration            586672 non-null   int64  
 20  dates               586672 non-null   datetime64[ns] 
 21  year                586672 non-null   object(5) 
dtypes: datetime64[ns](1), float64(9), int64(6), object(5)
memory usage: 94.0+ MB
```

SCREENSHOT-13



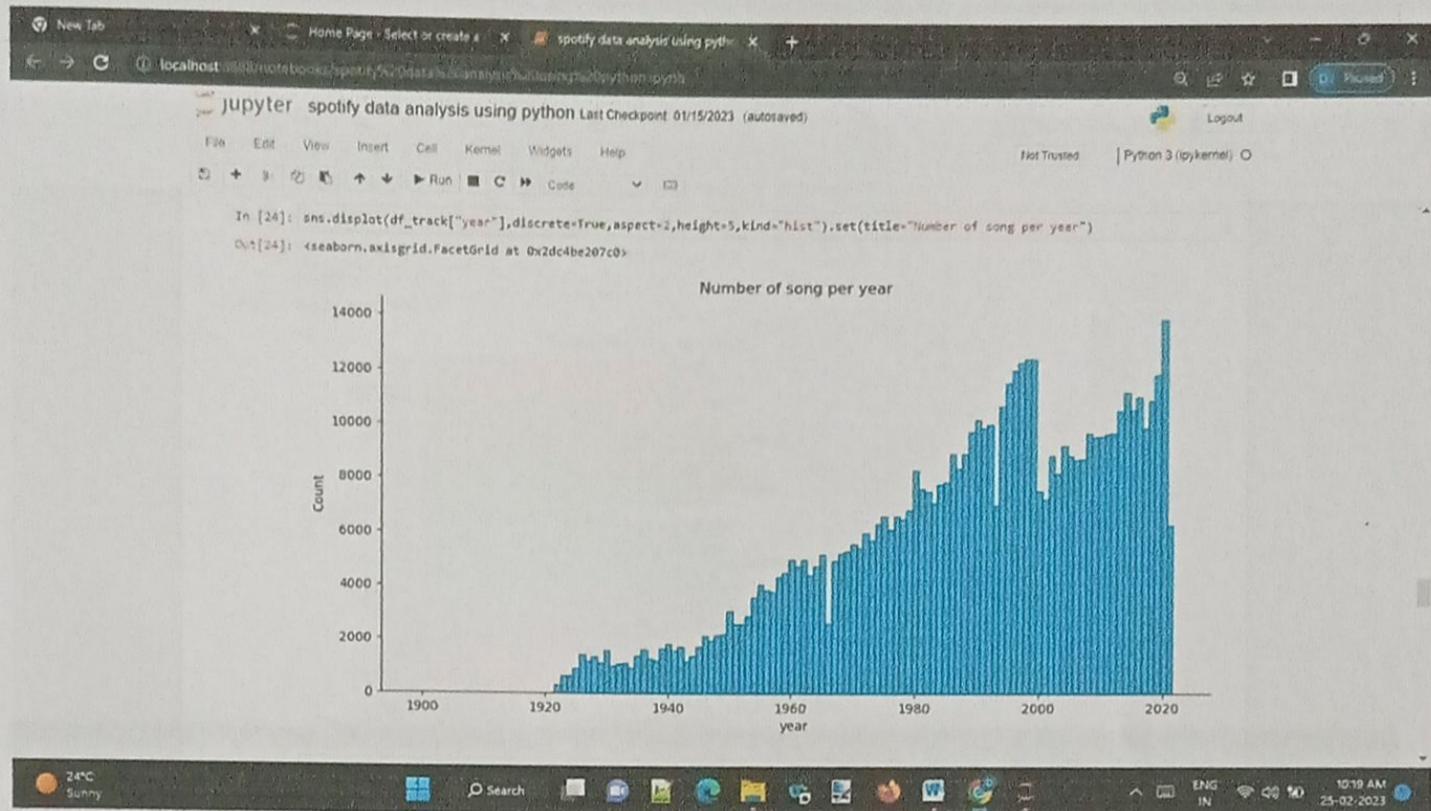
```
In [22]: df_track["release_date"] = pd.to_datetime(df_track["release_date"])
df_track["year"] = df_track["release_date"].dt.year

In [23]: df_track.info()

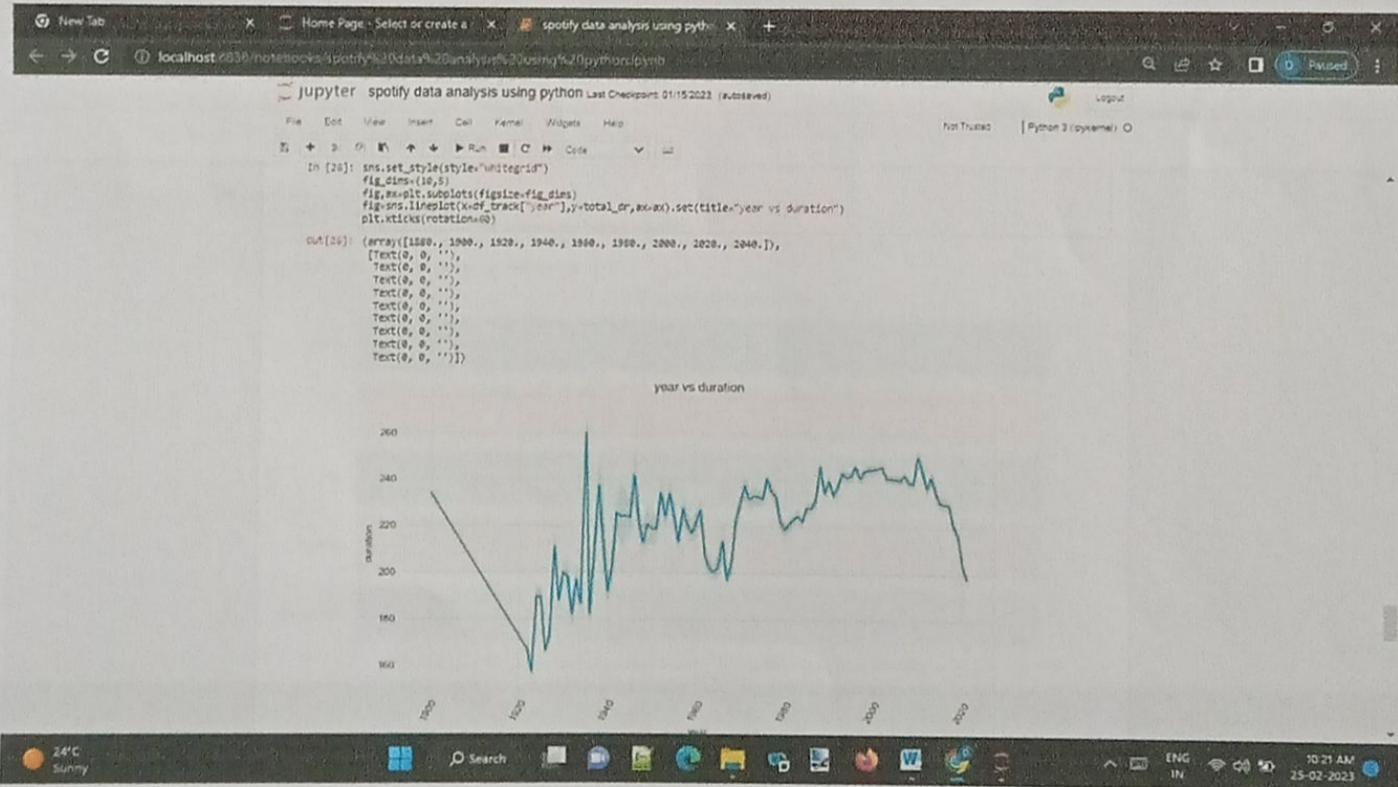
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 586672 entries, 0 to 586671
Data columns (total 22 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   id               586672 non-null   object  
 1   name              586601 non-null   object  
 2   popularity        586672 non-null   int64  
 3   explicit          586672 non-null   int64  
 4   artists            586672 non-null   object  
 5   id_artists        586672 non-null   object  
 6   release_date      586672 non-null   datetime64[ns] 
 7   danceability       586672 non-null   float64 
 8   energy             586672 non-null   float64 
 9   key                586672 non-null   int64  
 10  loudness           586672 non-null   float64 
 11  mode               586672 non-null   int64  
 12  speechiness        586672 non-null   float64 
 13  acousticness       586672 non-null   float64 
 14  instrumentalness  586672 non-null   float64 
 15  liveness            586672 non-null   float64 
 16  valence             586672 non-null   float64 
 17  tempo               586672 non-null   float64 
 18  time_signature     586672 non-null   int64  
 19  duration            586672 non-null   int64  
 20  dates               586672 non-null   datetime64[ns] 
 21  year                586672 non-null   int64  
 22  year                586672 non-null   object(4) 
dtypes: datetime64[ns](2), float64(9), int64(7), object(4)
memory usage: 98.5+ MB
```

Spotify Data Analysis

SCREENSHOT-14

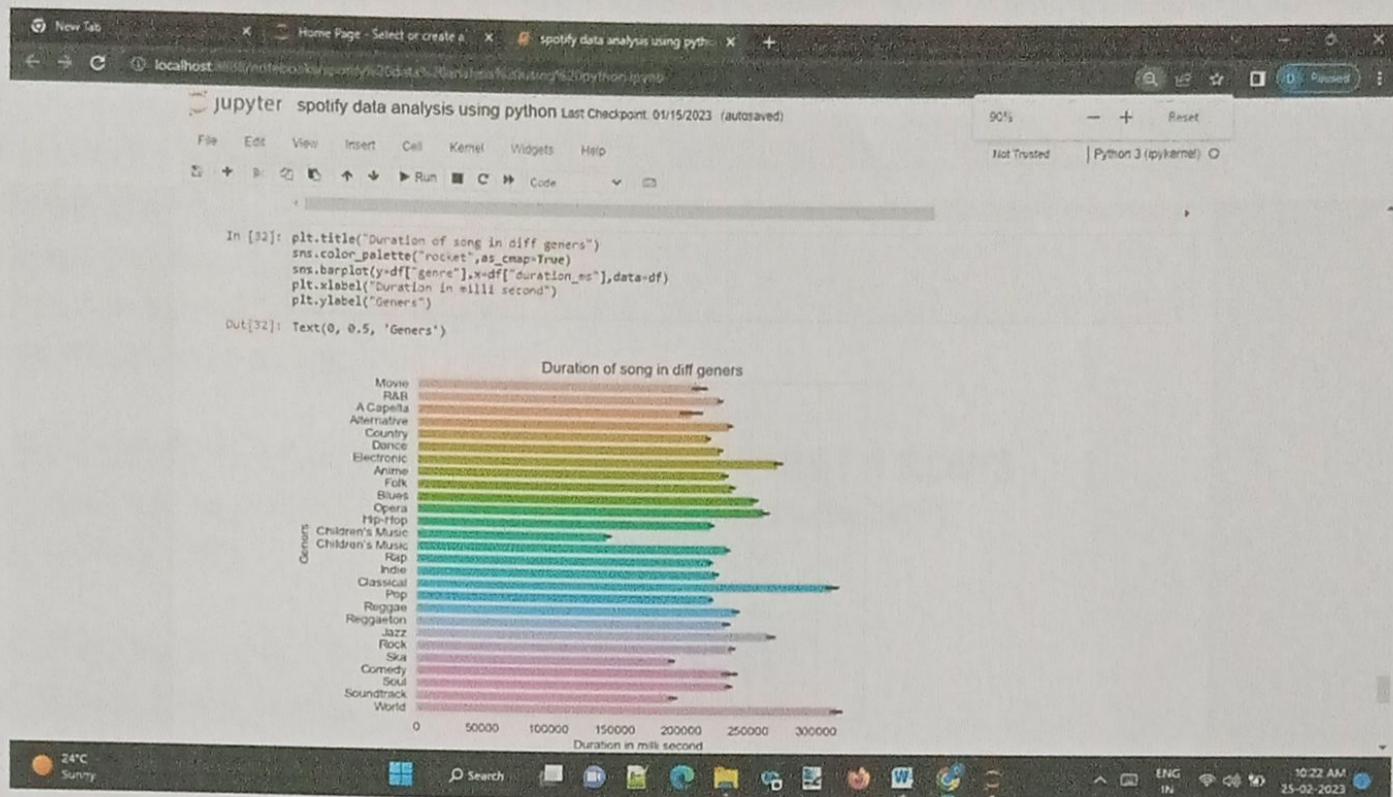


SCREENSHOT-15

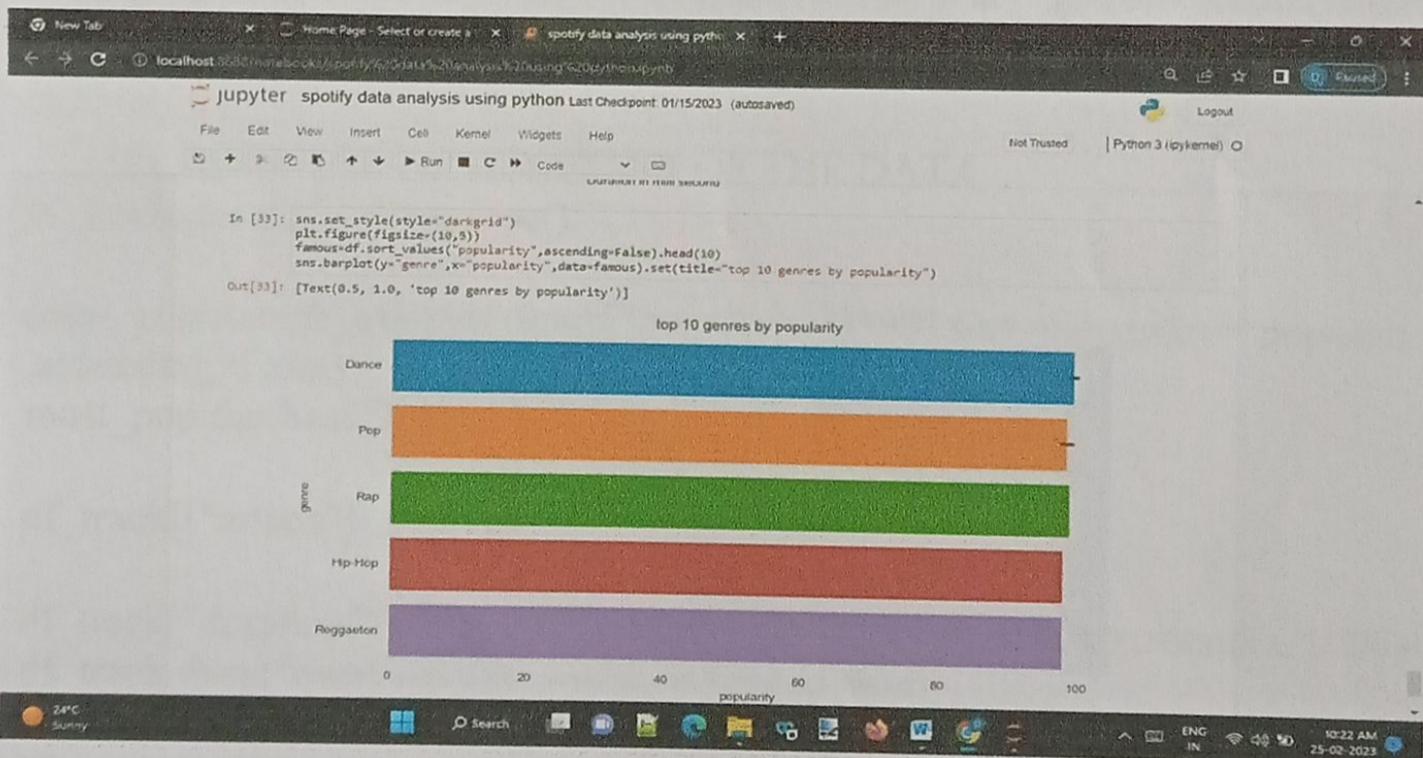


Spotify Data Analysis

SCREENSHOT-16



SCREENSHOT-17



CHAPTER-07**CODING**

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

# READING DATASET AND CHECKING FIRST 5 ROWS
df_track=pd.read_csv("C:/Users/DELL/Downloads/tracks.csv")
df_track.head()

# CHECKING NULL VALUES IN DATASET
df_track.isnull().sum()

# CHECKING SHAPE OF AN ARRAY
df_track.shape

# CHECKING TOP 10 MOST POPULAR SONGS
sort_df=df_track.sort_values("popularity",ascending=True)
sort_df.head()

# CHECKING THE DESCRIPTION OF THE DATA
df_track.describe().transpose()

most_popular=df_track[df_track["popularity"]>90].sort_values(by="popularity",
,ascending=False)
most_popular.head()

df_track[["artists"]].iloc[18]

df_track["duration"]=df_track["duration_ms"].apply(lambda x:round(x/1000))
df_track.drop("duration_ms",inplace=True,axis=1)

df_track.head()

df_track["artists"].value_counts()

df_track.info()

df_track[df_track["duration"]==5621]["artists"]
```

Spotify Data Analysis

```
corr_df=df_track.drop(["key","mode","explicit"],axis=1).corr(method="pearson")
plt.figure(figsize=(14,6))
heatmap=sns.heatmap(corr_df,annot=True,vmin=-1,vmax=1,center=0,cmap="inferno")
heatmap.set_title("Corr heatmap between variable")
heatmap.set_xticklabels(heatmap.get_xticklabels(),rotation=90)

sample_df=df_track.sample(int(0.004*len(df_track)))
sample_df.head()

len(sample_df)

plt.figure(figsize=(10,8))
sns.regplot(data=sample_df,y="loudness",x="energy",color="c").set(title="Loudness vs Energy corr")

plt.figure(figsize=(10,8))
sns.regplot(data=sample_df,y="popularity",x="acousticness",color="c").set(title="popularity vs acoustic corr")

df_track["dates"]=pd.to_datetime(df_track["release_date"])
df_track.info()

df_track["release_date"]=pd.to_datetime(df_track["release_date"])
df_track["year"]=df_track["release_date"].dt.year

df_track.info()

sns.displot(df_track["year"],discrete=True,aspect=2,height=5,kind="hist").set(title="Number of song per year")

total_dr=df_track["duration"]
fig,ax=plt.subplots()
fig=sns.barplot(x=df_track["year"],y=total_dr,ax=ax,errwidth=False).set(title="Year vs Duration")

sns.set_style(style="whitegrid")
fig_dims=(10,5)
fig,ax=plt.subplots(figsize=fig_dims)
```

Spotify Data Analysis

```
fig=sns.lineplot(x=df_track["year"],y=total_dr,ax=ax).set(title="year vs duration")
plt.xticks(rotation=60)
```

```
df=pd.read_csv("C:/Users/DELL/Downloads/SpotifyFeatures.csv")
df.head()
```

```
plt.title("Duration of song in diff genres")
sns.color_palette("rocket",as_cmap=True)
sns.barplot(y=df["genre"],x=df["duration_ms"],data=df)
plt.xlabel("Duration in milli second")
plt.ylabel("Genres")
```

```
sns.set_style(style="darkgrid")
plt.figure(figsize=(10,5))
famous=df.sort_values("popularity",ascending=False).head(10)
sns.barplot(y="genre",x="popularity",data=famous).set(title="top 10 genres by popularity")
```

CHAPTER-08

TESTING

8.1 What is Software Testing?

Software testing is an activity to check whether the actual results match the expected results and to ensure that the software system is Defect free. It involves execution of a software component or system component to evaluate one or more properties of interest.

Software testing also helps to identify errors, gaps or missing requirements in contrary to the actual requirements. It can be either done manually or using automated tools. Some prefer saying Software testing as a white box and Black Box Testing.

Different levels of testing are used in the test process; each level of testing aims to test different aspects of the system.

The First Level is **unit testing**. In this testing, individual components are tested to ensure that they operate correctly. It focuses on verification efforts.

The Second Level is **integration testing**. It is a systematic technique for constructing the program structure. In this testing, many tested modules are combined into the subsystems which are then tested. The good here is to see if the modules can be integrated properly.

The Third Level is **system testing**? System testing is actually a series of different tests whose primary purpose is to fully exercise computer-based system.

Testing is the fundamental process of software success. Testing is not a distinct phase in system development life cycle but should be applicable throughout all phases that is design development and maintenance phase. Testing is used to show incorrectness and considered to success when an error is detected.

8.2 Software Quality Improvement

The computer and the software are mainly used for complex and critical applications and a bug or fault in software causes severe losses. So, a great consideration is required for checking for quality of software.

8.3 Verification and Validation

Verification means to test that we are building the product in right way that we are using the correct procedure for the development of software so that it can meet the user requirements. Validation means to check whether we are building the right product or not.

8.4 Software Reliability Estimation The objective is to discover the residual designing errors before delivery to the customer. The failure data during process are taken down in order to estimate the software reliability.

CHAPTER-09

CONCLUSION

The analysis is intended to understand the characteristics of various genres of music and to identify how popular a songs would be based on its features. Along with that, I have also identified the patterns and relationship between the features that describe the songs in our dataset.

We have performed data wrangling on our Spotify dataset by removing null values, removing duplicates and transforming variables, before starting our exploratory data analysis. From our correlation plot, we observed that variables have strong correlation with each other, indicating that this dataset has multicollinearity. I have also plotted histograms and boxplot to show the relation between the variables. I have then utilized this dataset for clustering analysis to analyse a track's popularity based on several audio features provided in the dataset and find whether we could predict a track's popularity from key features about the song.

We have gained a lot of insights during this journey of exploring the data and the interesting ones relevant in our analysis are: A lot of songs in our data have a popularity score of 0, which means a lot those haven't been explored yet. Energy has high positive correlation with loudness and high negative correlation with acoustic ness. EDM genre has songs with highest energy, Rap genre has the highest dance ability factor .Rap and Latin are most lively and loudness is pretty similar for all genres. Furthermore, Even though we have six genres in the dataset, with the clustering technique I got optimal clusters count as three, which means few genres might be very similar. Lastly, less speech and high energy songs are likely to be popular among users.

With our clustering techniques we could provide more suggestions based on three clusters that we got, which would improve the user experience and finally

we have an idea of which characteristics have an impact on the popularity of the songs and we could probably build more customized playlists.

Our dataset had limited entries, so with a much larger dataset we could probably improve our analysis even more. Also additional data points in the dataset like user demographics and historical data would make our analysis even more reliable

CHAPTER-10

BIBLIOGRAPHY

1. Jake Vander plas, "Python Data Science Handbook: Essential tools for working with data", O'Reilly Publishers, I Edition.
2. Wes Mc Kinney, "Python for Data Analysis", O'Reilly Media, 2012Mark Lutz, "ProgrammingPython", O'Reilly Media, 4th edition, 2010.

Some of the links which were referred during the project

- <https://www.kaggle.com/code/mrankitgupta/spotify-data-analysis-using-python>
- <https://www.youtube.com/watch?v=8d7ywKCm6HI>
- [S | Kaggle](#)