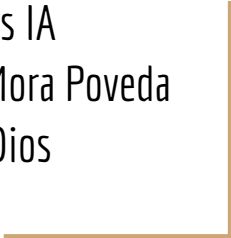




Especialización en Inteligencia Artificial

Trabajo Semana 4 Dataset Spotify

Asignatura: Fundamentos IA
Realizado por: Michael Andrés Mora Poveda
Universidad Minuto de Dios
Abril 9 de 2023



Propósito:

Trabajo Semana 4 - Análisis exploratorio de datos

Dataset Spotify

Asignatura: Fundamentos IA

Especialización en Inteligencia Artificial

Realizado por: Michael Andrés Mora Poveda

El objetivo de este trabajo es realizar el análisis exploratorio de datos y visualización para los datasets de Spotify los cuales se pueden encontrar en la siguiente página web (y dentro de este folder también) y que se encuentran también referenciados al final de este notebook:

- <https://www.kaggle.com/code/vatsalmavani/music-recommendation-system-using-spotify-dataset/input>

Además, es importante tener en cuenta que dentro de este análisis se aplicarán varias librerías de Python para mayor practicidad. En la semana dos se hizo un ejercicio de análisis exploratorio similar con el dataset de Diabetes. Para esta ocasión haremos uso de un dataset open-source nuevo.

Las siguientes descripciones serán útiles para conocer la estructura del dataset:

Descripción general:

Según las fuentes citadas, la información sirve como ejercicio práctico para generar sistemas de recomendación de playlists basados en múltiples características asociadas a las canciones que confirman el dataset. Las descripciones de los features son listados a continuación:

1. valence: Describe la positividad musical que transmite una canción
2. year: Año de la canción
3. acousticness: Nivel de acústica
4. artists: Nombre del artista
5. danceability: Describe que tan adecuada es la canción para baile
6. duration_ms: Duración en milisegundos
7. energy: Energía lírica que mide la intensidad de la canción (rápida, lenta, ruidosa)
8. explicit: Variable binaria de sí es de contenido explícito o no
9. id: Código de identificación de la canción alfanumérico
10. instrumentalness: Representa la cantidad de vocales en la canción
11. key: Id numérico
12. liveness: Probabilidad de que la canción se grabara con público en directo
13. loudness: Mide el nivel de ruido que tiene la canción
14. mode: Variable desconocida
15. name: Nombre de la canción
16. popularity: Nivel de popularidad
17. release_date: Fecha de lanzamiento
18. speechiness: Mide la presencia de palabras habladas en una canción
19. tempo: Velocidad de reproducción de la canción

Nota: Vamos a realizar varias técnicas de visualización y exploración entre los varios datasets de Spotify.

Notas:

Esta presentación contiene partes del script generado en Jupyter Notebook de acuerdo a la actividad de la semana 4 de la Especialización en IA de la Universidad Minuto de Dios. El script completo se encuentra en el siguiente repositorio de Github:

https://github.com/micmorap/Machine_Learning_Portfolio/tree/main/AI%20Specialization%20-%20U%20of%20Minuto%20de%20Dios/Fundamentos%20de%20IA/Semana_4

Análisis exploratorio:

1. Análisis exploratorio Spotify dataset

Vamos a revisar la estructura general de los datos:

```
In [1]: #Se importan las librerías clásicas:
import seaborn as sns
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import datetime
import seaborn as sns
import seaborn as sns
import plotly.express as px
import matplotlib.pyplot as plt
%matplotlib inline
```

```
In [2]: #Importamos los archivos contenidos en el folder y chequeamos los datasets:
df_Spotify_general = pd.read_csv('data.csv', sep=',')
df_Spotify_general.tail(4)
```

Out[2]:

	valence	year	acousticness	artists	danceability	duration_ms	energy	explicit	id	instrumentalness	key	liveness	lo
170649	0.734	2020	0.20600	['Ashnikko']	0.717	150654	0.753	0	00StKKAuXlxA0fMH54Qs6E	0.000000	7	0.101	
170650	0.637	2020	0.10100	['MAMAMOO']	0.634	211280	0.858	0	4BZXVFCb76Q0Klojq4pIV	0.000009	4	0.258	
170651	0.195	2020	0.00998	['Eminem']	0.671	337147	0.623	1	5SiZJoLXp3WOI3J4C8IK0d	0.000008	2	0.643	
170652	0.642	2020	0.13200	['KEVVO', 'J Balvin']	0.856	189507	0.721	1	7HmnJHfs0BkFzX4x8j0hkl	0.004710	7	0.182	

```
In [3]: # Revisamos las dimensiones del dataset:
print('Total filas y columnas: {}'.format(df_Spotify_general.shape))
```

Total filas y columnas: (170653, 19)

Validación de datos:

```
In [4]: # Ciclo para ver el número de registros por feature
for i in (df_Spotify_general.columns):
    print('Columna (feature) {} y número de registros: {}'.format(i, df_Spotify_general[i].value_counts().sum()))
```

```
Columna (feature) valence y número de registros: 170653
Columna (feature) year y número de registros: 170653
Columna (feature) acousticness y número de registros: 170653
Columna (feature) artists y número de registros: 170653
Columna (feature) danceability y número de registros: 170653
Columna (feature) duration_ms y número de registros: 170653
Columna (feature) energy y número de registros: 170653
Columna (feature) explicit y número de registros: 170653
Columna (feature) id y número de registros: 170653
Columna (feature) instrumentalness y número de registros: 170653
Columna (feature) key y número de registros: 170653
Columna (feature) liveness y número de registros: 170653
Columna (feature) loudness y número de registros: 170653
Columna (feature) mode y número de registros: 170653
Columna (feature) name y número de registros: 170653
Columna (feature) popularity y número de registros: 170653
Columna (feature) release_date y número de registros: 170653
Columna (feature) speechiness y número de registros: 170653
Columna (feature) tempo y número de registros: 170653
```

Validación de datos:

```
In [5]: # Número de valores NA
for i in (df_Spotify_general.columns):
    print('Columna (feature) {} y número de registros NA: {}'.format(i, df_Spotify_general[i].isna().sum()))
```

```
Columna (feature) valence y número de registros NA: 0
Columna (feature) year y número de registros NA: 0
Columna (feature) acousticness y número de registros NA: 0
Columna (feature) artists y número de registros NA: 0
Columna (feature) danceability y número de registros NA: 0
Columna (feature) duration_ms y número de registros NA: 0
Columna (feature) energy y número de registros NA: 0
Columna (feature) explicit y número de registros NA: 0
Columna (feature) id y número de registros NA: 0
Columna (feature) instrumentalness y número de registros NA: 0
Columna (feature) key y número de registros NA: 0
Columna (feature) liveness y número de registros NA: 0
Columna (feature) loudness y número de registros NA: 0
Columna (feature) mode y número de registros NA: 0
Columna (feature) name y número de registros NA: 0
Columna (feature) popularity y número de registros NA: 0
Columna (feature) release_date y número de registros NA: 0
Columna (feature) speechiness y número de registros NA: 0
Columna (feature) tempo y número de registros NA: 0
```

Validación de datos:

```
In [6]: # Conteo de valores nulos por feature:  
df_Spotify_general.isnull().sum()
```

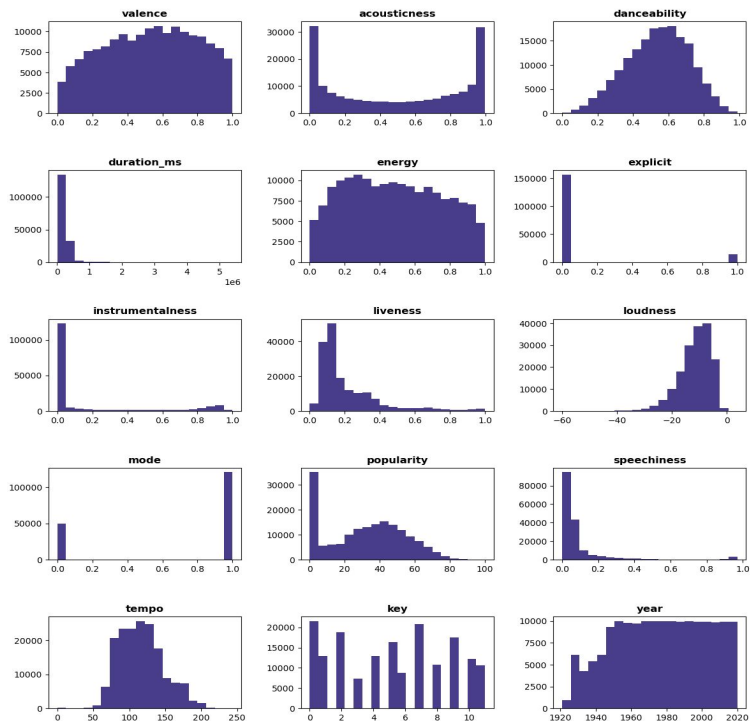
```
Out[6]: valence          0  
year          0  
acousticness  0  
artists       0  
danceability  0  
duration_ms   0  
energy        0  
explicit      0  
id            0  
instrumentalness 0  
key           0  
liveness      0  
loudness      0  
mode          0  
name          0  
popularity    0  
release_date  0  
speechiness   0  
tempo         0  
dtype: int64
```

```
In [7]: # Estructura general de las columnas y sus tipos  
df_Spotify_general.info()
```

```
<class 'pandas.core.frame.DataFrame'>  
RangeIndex: 170653 entries, 0 to 170652  
Data columns (total 19 columns):  
#   Column                Non-Null Count  Dtype  
---  -  
0   valence                170653 non-null float64  
1   year                  170653 non-null int64  
2   acousticness           170653 non-null float64  
3   artists                170653 non-null object  
4   danceability           170653 non-null float64  
5   duration_ms            170653 non-null int64  
6   energy                 170653 non-null float64  
7   explicit               170653 non-null int64  
8   id                     170653 non-null object  
9   instrumentalness        170653 non-null float64  
10  key                    170653 non-null int64  
11  liveness                170653 non-null float64  
12  loudness                170653 non-null float64  
13  mode                    170653 non-null int64  
14  name                    170653 non-null object  
15  popularity              170653 non-null int64  
16  release_date            170653 non-null object  
17  speechiness             170653 non-null float64  
18  tempo                   170653 non-null float64  
dtypes: float64(9), int64(6), object(4)  
memory usage: 24.7+ MB
```


Distribuciones:

En términos generales, vemos que la data no tiene inconsistencias frente a valores nulos y sus descripciones son claras para el entendimiento del ejercicio. Ahora, revisaremos algunas gráficas y concentraciones:



Algunas observaciones sobre las distribuciones graficadas anteriormente:

- * Las variables valence, acousticness, danceability, energy y year presentan distribuciones muy completas y poco sesgadas.
- * Las variables liveness, loudness y speechiness tienen sesgos ya sea hacia izquierda o derecha.
- * La variable explicit nos muestra que la mayoría de las canciones tienen contenido explícito.
- * La variable miliseconds nos muestra que la mayoría de las canciones tienen una duración inferior a 16 minutos.
- * La variable year nos muestra que entre los años 1920 y 2020 la distribución de canciones es casi uniforme.
- * Las variables key, y mode se pueden remover sin problemas del dataset.

Ahora veremos la estructura de la columna release_year para ver si podemos removerla o no:

```
In [10]: # Agrupamos la data por la variable release_date para ver si su formato es adecuado:  
df_Spotify_general.groupby(['release_date']).count().head(3)
```

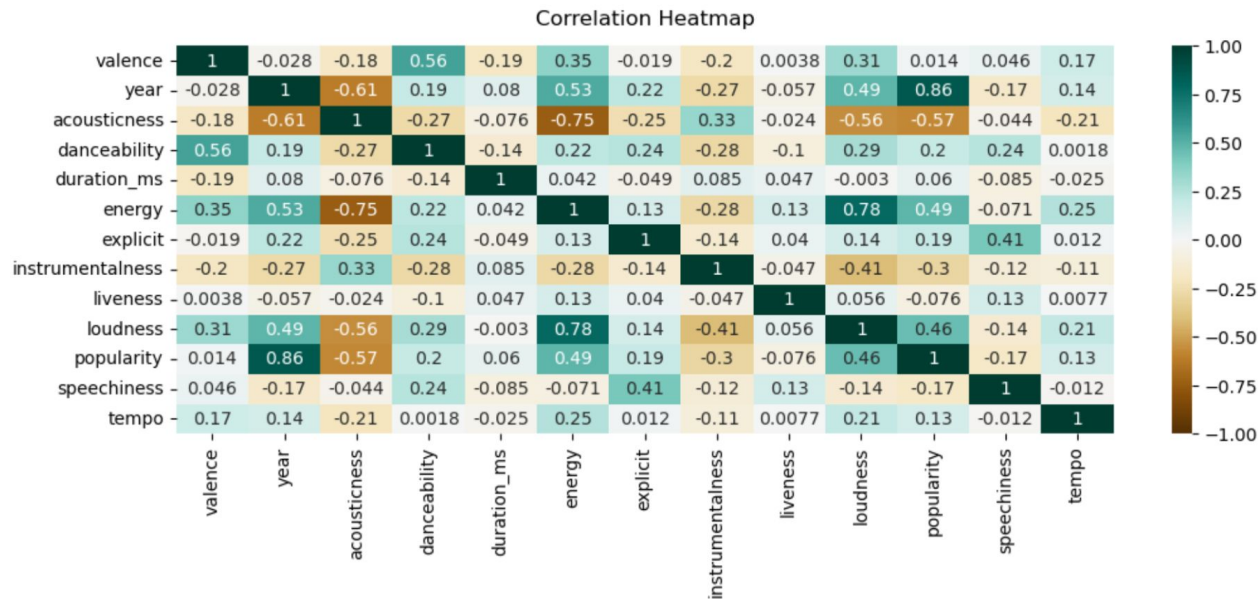
```
Out[10]:
```

	valence	year	acousticness	artists	danceability	duration_ms	energy	explicit	id	instrumentalness	key	liveness	loudness	mode	name	...
release_date																
1921	116	116	116	116	116	116	116	116	116	116	116	116	116	116	116	116
1921-02-20	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
1921-03-20	21	21	21	21	21	21	21	21	21	21	21	21	21	21	21	21

Correlación de variables:

```
In [14]: # Creamos el mapa de calor de correlaciones:  
plt.figure(figsize=(12, 4))  
heatmap = sns.heatmap(df_Spotify_general.corr(), vmin=-1, vmax=1, annot=True, cmap='BrBG')  
heatmap.set_title('Correlation Heatmap', fontdict={'fontsize':12}, pad=12)
```

Out[14]: Text(0.5, 1.0, 'Correlation Heatmap')



Observaciones de acuerdo a distribuciones y correlaciones:

De acuerdo a los resultados anteriores, se tienen las siguientes observaciones:

- * La variable year puede sustituir a la variable release-year para una mejor visualización, dado que esta última columna tiene el formato variado, es decir, sólo por año o por formato yyyy-mm-dd
- * Podemos apreciar por debajo de la diagonal las correlaciones entre las variables numéricas, de las cuales podemos destacar las siguientes para posteriores entrenamientos de modelos de ML:
 - * year y popularity (positiva)
 - * loudness y energy (positiva)
 - * danceability y valence (positiva)
 - * year y acousticness (negativa)
 - * energy y acousticness (negativa)
 - * loudness y acousticness (negativa)
 - * popularity y acousticness (negativa)

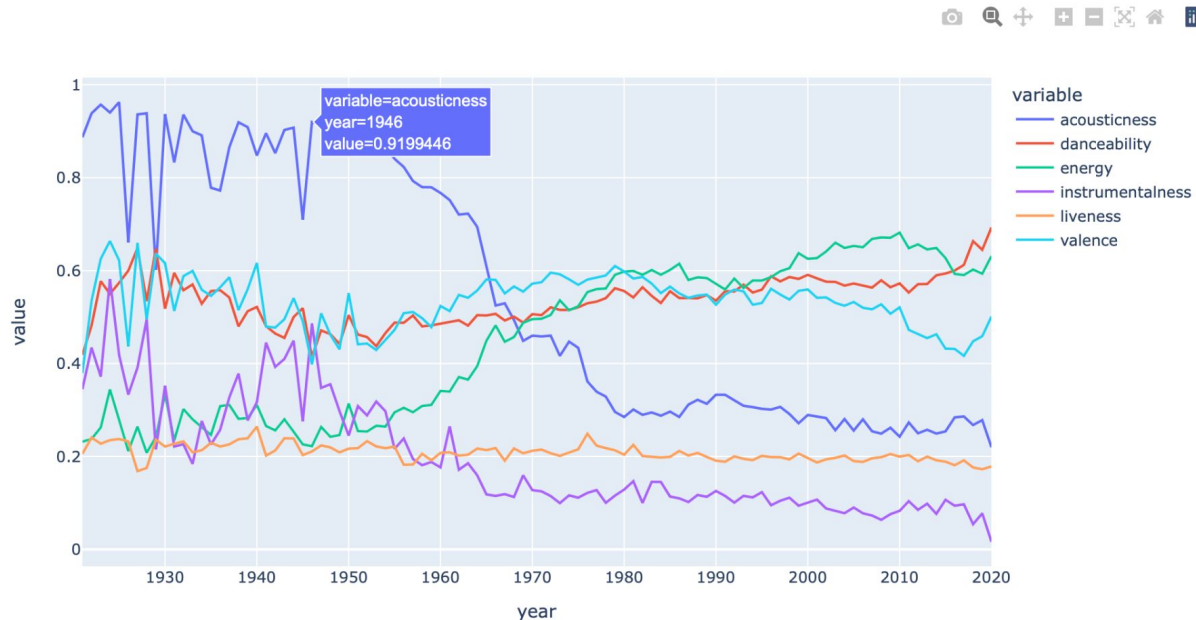
Ahora, vamos a ver algunas visualizaciones para la data agrupada por año y género:

```
In [15]: # Importamos la data agrupada por año:  
df_Spotify_year = pd.read_csv('data_by_year.csv', sep=',')
```

```
In [16]: # Revisamos a través de plotly la tendencia de las variables musicales técnicas:  
sound_tech_features = ['acousticness', 'danceability', 'energy', 'instrumentalness', 'liveness', 'valence']  
fig = px.line(df_Spotify_year, x='year', y=sound_tech_features)  
fig.show()
```

Algunas gráficas por año y género:

```
In [16]: # Revisamos a través de plotly la tendencia de las variables musicales técnicas:  
sound_tech_features = ['acousticness', 'danceability', 'energy', 'instrumentalness', 'liveness', 'valence']  
fig = px.line(df_Spotify_year, x='year', y=sound_tech_features)  
fig.show()
```



Algunas gráficas por año y género:

```
In [17]: # visualizamos la popularidad entre los años 1920 y 2020:  
sound_tech_features = ['popularity']  
fig = px.line(df_Spotify_year, x='year', y=sound_tech_features)  
fig.show()
```

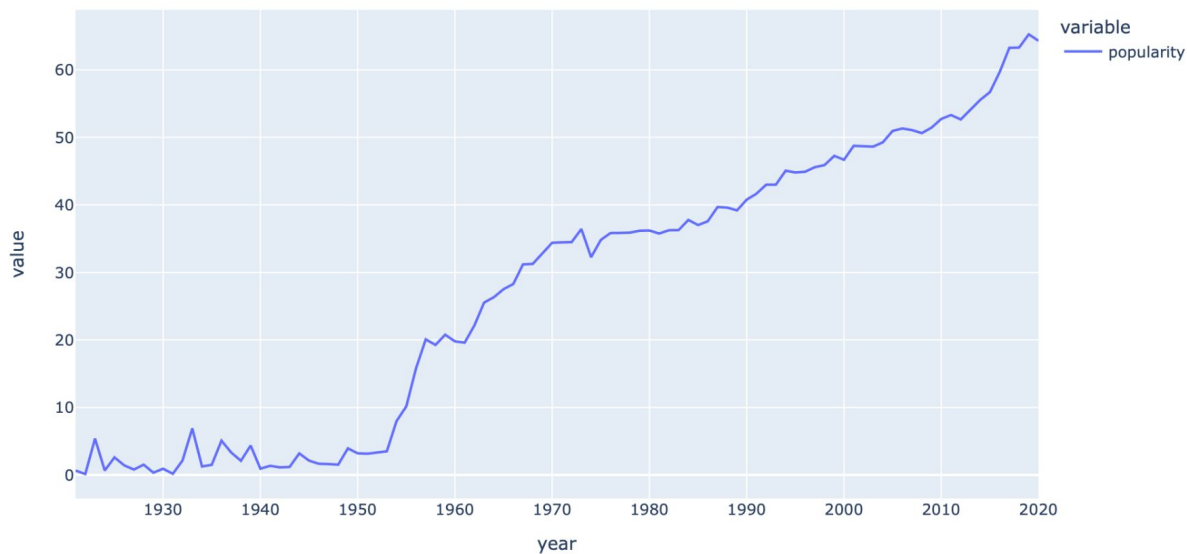


Imagen y código de autoría propia

Algunas gráficas por año y género:

De acuerdo a los últimos resultados anteriores, se tienen las siguientes observaciones:

- * Las variables acousticness y instrumentality han tenido tendencias a la baja después de 1960
- * Las variables danceability y liveness se han mantenido casi constantes a lo largo del tiempo
- * La variable energy ha tenido tendencia positiva desde 1955
- * Los géneros circuit y guaracha tienen bastante similitud técnica
- * El top de géneros nos indica bajos niveles en la variable acousticness y que todos en general son aptos para el baile

```
In [19]: # Visualizamos el top 20 de géneros musicales:
top20 = df_Spotify_ggenre.nlargest(10, 'popularity')
fig = px.bar(top20, x='genres', y=['valence', 'energy', 'danceability', 'acousticness'], barmode='group')
fig.show()
```

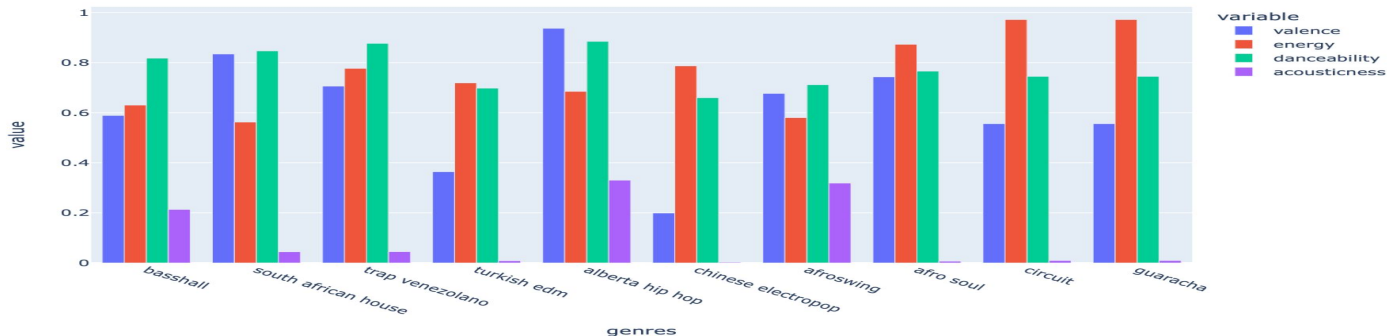


Imagen y código de autoría propia

Pandas profiling: Reporte interactivo para EDA:

Pandas profiling para análisis exploratorio

```

})]: from pandas_profiling import ProfileReport
prof = ProfileReport(df_Spotify_year)
prof.to_file(output_file='output.html')

/var/folders/xh/l9ggclrd1fx2brvxlnc_9hv40000gn/T/ipykernel_3232/504870144.py:1: DeprecationWarning:
`import pandas_profiling` is going to be deprecated by April 1st. Please use `import ydata_profiling` instead.

Summarize dataset: 100% ██████████ 192/192 [00:09<00:00, 19.05it/s, Completed]

Generate report structure: 100% ██████████ 1/1 [00:02<00:00, 2.18s/it]

Render HTML: 100% ██████████ 1/1 [00:01<00:00, 1.41s/it]

Export report to file: 100% ██████████ 1/1 [00:00<00:00, 71.51it/s]
```

Overview

Dataset statistics

Number of variables	14
Number of observations	100
Missing cells	0
Missing cells (%)	0.0%
Duplicate rows	0
Duplicate rows (%)	0.0%
Total size in memory	11.1 KiB
Average record size in memory	113.3 B

Variable types

Categorical	1
Numeric	13

Imagen y código de autoría propia

Pandas profiling: Reporte interactivo para EDA:

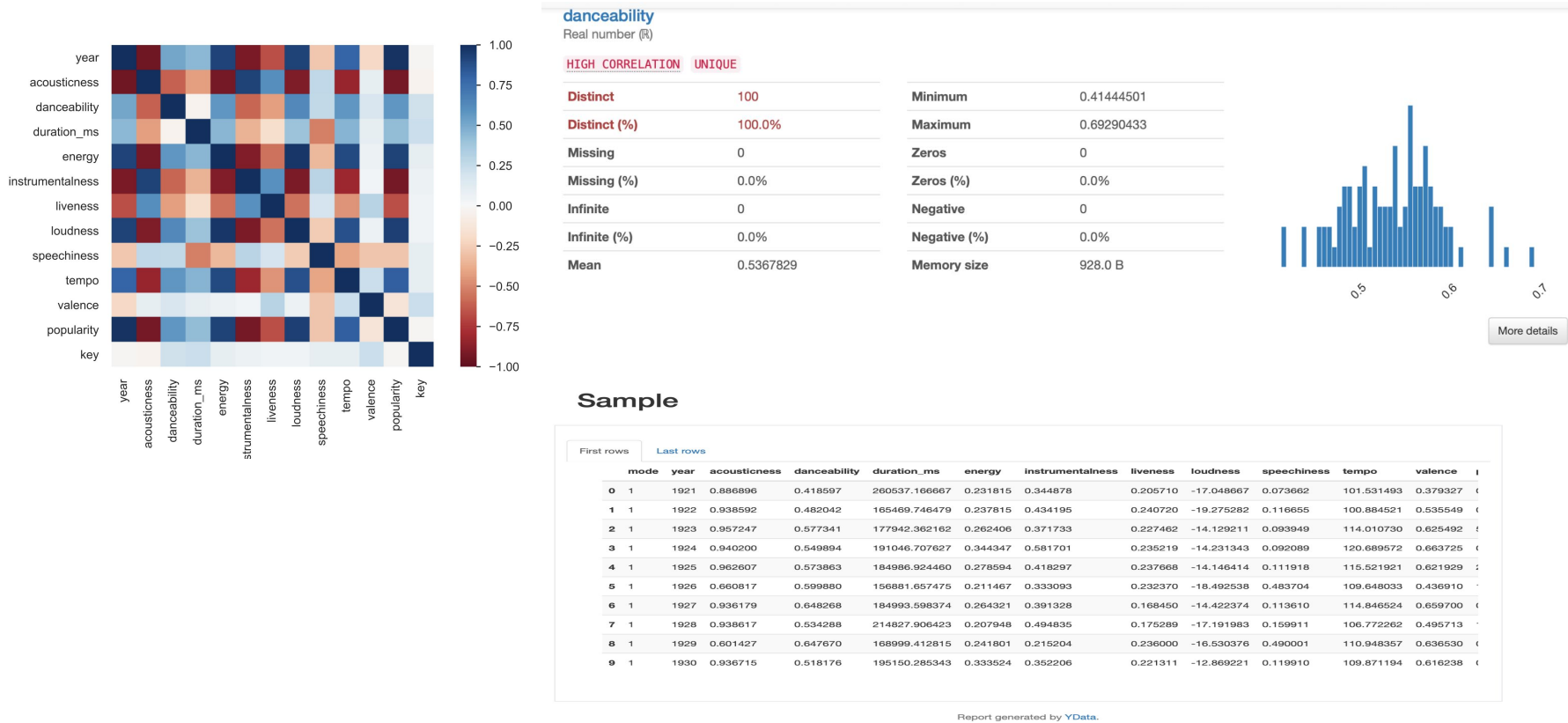


Imagen y código de autoría propia

Conclusiones

- En general podemos ver la ventaja de tener la misma data en varios datasets por distintos niveles de desglose, esto facilita análisis minuciosos y globales.
- La calidad y tamaño de la data es bastante buena para entrenamiento de modelos de Machine Learning relacionados con sistemas de recomendación.
- Vemos una calidad superior de gráficos de plotly frente a matplotlib.
- Aplicamos varios comandos de los frameworks clásicos de Python para visualización y análisis exploratorio de datos.
- La librería de Pandas profiling, genera un html de una descripción general de variables y sus distribuciones que nos ayudan bastante a entender la data.

Referencias:

- [1]. Mavani, V. (2022). Music Recommendation System using Spotify Dataset. Taken from Kaggle: <https://www.kaggle.com/code/vatsalmavani/music-recommendation-system-using-spotify-dataset/notebook>
- [2]. Mavani, V. (2022). Spotify dataset. Taken from Kaggle: <https://www.kaggle.com/datasets/vatsalmavani/spotify-dataset>
- [2]. Scikit-learn: Machine Learning in Python, Pedregosa et al., JMLR 12, pp. 2825-2830, 2011.
- [3]. J. D. Hunter, "Matplotlib: A 2D Graphics Environment", Computing in Science & Engineering, vol. 9, no. 3, pp. 90-95, 2007.
- [4]. La función subplots. Interactive chaos. Taken from: <https://interactivechaos.com/es/manual/tutorial-de-matplotlib/la-funcion-subplots>
- [5]. Ajay Tech. Logistic Regression .Taken from: <https://ajaytech.co/python-logistic-regression/>
- [6]. De Dios Santos, J. (2017). Blog Medium - Is my Spotify music boring? An analysis involving music, data, and machine learning. Taken from: <https://towardsdatascience.com/is-my-spotify-music-boring-an-analysis-involving-music-data-and-machine-learning-47550ae931de>
- [7.] Brugman, S. (2019). pandas-profiling: Exploratory Data Analysis for Python. Retrieved from <https://github.com/pandas-profiling/pandas-profiling>