EDGARD JOSEPH KIRIYAMA

PÓS TECH

DATA ANALYTICS

MACHINE LEARNING COM PYTHON

AULA 02

SUMÁRIO

O QUE VEM POR AÍ?

Olá, jovem analytic!

Você está na terceira disciplina do curso, e aqui começa mais um ponto importantíssimo da jornada em que você está trilhando.

Na aula “Bibliotecas fundamentais e primeiros passos em ML”, passamos por todo o entendimento sobre o que é Machine Learning, suas complexidades e as principais bibliotecas.

Agora, chegamos em um momento importante de nossa jornada, quando falaremos sobre a Análise Exploratória de Dados (EDA – Exploratory Data Analysis).

Chegou a hora de aprimorarmos e elevarmos o nosso nível! Vamos te mostrar como utilizar os nossos dados explorando toda a nossa base de dados e alguns insights interessantes, trazendo valor para os dados.

HANDS ON

Agora, chegou o momento de ver, na prática, como começar a importar nossos dados e trabalhar com eles via programação. A ideia é não se limitar apenas ao código explícito no hands on, então é sempre bom procurar a documentação das bibliotecas, explorar novas funcionalidades e muito mais!

SAIBA MAIS

Vamos começar pelo início:

- Você já analisou algum dado?
- Já teve contato com base de dados públicas ou privadas e não sabia por onde começar a analisar?
- Já te passaram uma base de dados e só disseram: “se vira”?

Essa são as perguntas mais frequentes nos dias de hoje quando trabalhamos com dados. Mas e aí, o que fazer? Por onde começar?

É neste momento que a Análise Exploratória de Dados (EDA) vem nos ajudar.

A EDA nos ajuda na compreensão profunda sobre a base de dados, na qual ajuda a definir ou descartar hipóteses através da manipulação de dados, aplicando diversas ferramentas estatísticas para descrever e entender a relação entre as variáveis e como elas podem impactar os negócios.

Esta técnica é tão importante que nos ajuda a ter os famosos insights, encontrar problemas nas nossas bases e criar modelos preditivos robustos.

Graças à EDA é que podemos fazer a nós mesmos as perguntas que são significativas, que irão responder o nosso problema de negócio.

As principais etapas deste processo são:

1. Importar o conjunto de dados;
2. Entender a composição da nossa base de dados;
3. Preparar a nossa base de dados para leitura e compreensão;
4. Realizar análises estatísticas preliminares – Análise Descritiva;
5. Compreender as variáveis;
6. Estudo das relações entre as variáveis;
7. Insights (Debate e Conclusões).

Ao final de todos estes processos, já poderemos construir e consolidar um relatório que irá subsidiar as nossas análises (ou até mesmo consolidar um relatório mais técnicos para as nossas chefias) ou continuar como uma fase de modelagem dos dados para construção dos nossos modelos de Machine Learning.

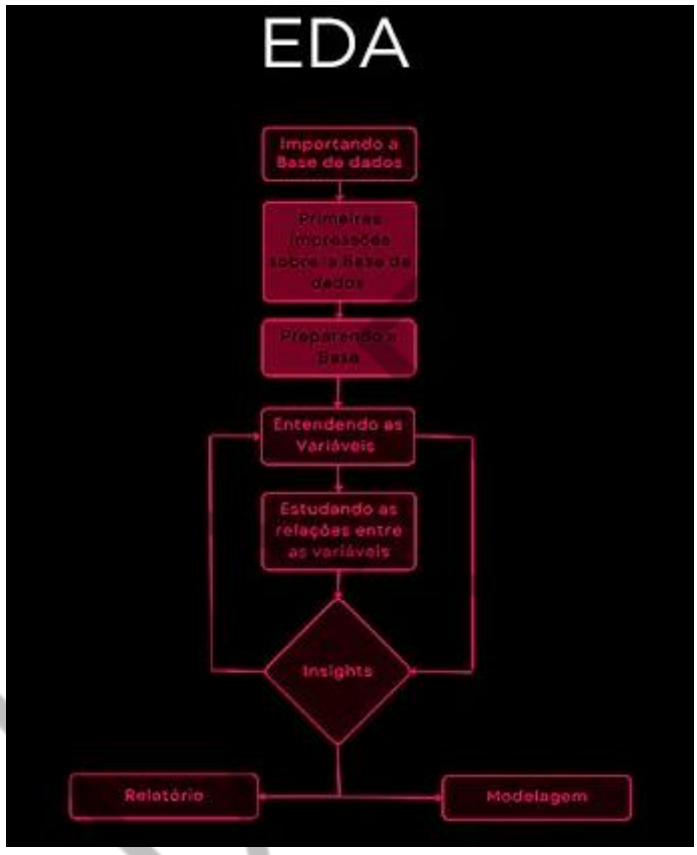


Figura 1 - Processo de Análise de Dados
Fonte: Elaborado pelo autor (2023), adaptado por FIAP (2023)

Vale ressaltar que essas etapas já citadas podem ser utilizadas pela linguagem de programação que você preferir, tais como Python, SQL, R, Julia e Java.

No nosso curso, utilizaremos o Python como a linguagem de programação principal de todos os nossos projetos. Vamos?

Para começar a nossa exploração de dados, acredito que todos conheçam a plataforma streaming Spotify! A ideia principal é explorarmos uma base que não temos muito conhecimento nesse momento e começar a mergulhar no mundo de dados. Afinal, quem disse para vocês que música não gera dados?

Bibliotecas mais utilizadas para EDA

Vamos carregar as principais bibliotecas de exploração de dados e a nossa base de dados do Spotify (`data_spotify.csv`). Utilizaremos o google colab para executar os nossos códigos.

```
#Biblioteca de Manipulação de dados
import pandas as pd
import numpy as np

#Biblioteca de Data Viz
import matplotlib.pyplot as plt
import seaborn as sns
```

Figura 2 – Importação de bibliotecas
Fonte: Elaborado pelo autor (2023)

Propriedade e funções úteis

Utilizaremos diversas funções do Pandas para visualizarmos a nossa base de dados. Vamos ver alguns deles:

- Visualização e entendimento da Base de dados.
- pd.read_csv.

Essa é umas das funções para carregar a base de dados em csv. Vamos carregar a nossa base de dados e visualizar as primeiras informações.

```
df = pd.read_csv('data_spotify.csv')
df
```

Figura 3 – Função para carregar a base de dados em csv.
Fonte: Elaborado pelo autor (2023).

Quando executamos o código, temos o resultado disponível na figura 4 – Resultado do código da figura 3.

	valence	year	acousticness	artists	danceability	duration_ms	energy	explicit	id	instrumentalness	key	liveness	loudness	mode	name
0	0.0594	1921	0.98200	['Sergei Rachmaninoff', 'James Levine', 'Berli...	0.279	831667	0.211	0	4BJqT0PrAfrxzMOxytFOIz	0.878000	10	0.6650	-20.096	1	Piano Concerto No. 3 in D Minor, Op. 30: III. ...
1	0.9630	1921	0.73200	['Dennis Day']	0.819	180533	0.341	0	7xPhfUan2yNtyFG0cUWkt8	0.000000	7	0.1600	-12.441	1	Clancy Lowered the Boom
2	0.0394	1921	0.96100	['KHP Kridhamardawa Karaton Ngayogyakarta Hadi...	0.328	500062	0.166	0	1o6I8BglA6ylDMrIELygv1	0.913000	3	0.1010	-14.850	1	Gati Bali
3	0.1650	1921	0.96700	['Frank Parker']	0.275	210000	0.309	0	3ftBPsC5vPBKxYSee08FDH	0.000028	5	0.3810	-9.316	1	Danny Boy
4	0.2530	1921	0.95700	['Phil Regan']	0.418	166693	0.193	0	4d6HGyGT8e121BsdKmw9v6	0.000002	3	0.2290	-10.096	1	When Irish Eyes Are Smiling
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
170648	0.6080	2020	0.08460	['Anuel AA', 'Daddy Yankee', 'KAROL G', 'Ozuna...	0.786	301714	0.808	0	0KkIkfsLEJbrcIhYsCL7L5	0.000289	7	0.0822	-3.702	1	China
170649	0.7340	2020	0.20600	['Ashnikko']	0.717	150654	0.753	0	0OStKKAuXlxA0fMH54Qs6E	0.000000	7	0.1010	-6.020	1	Halloweenie III: Seven Days
170650	0.6370	2020	0.10100	['MAMAMOO']	0.634	211280	0.858	0	4BZXVFYCb76Q0Klojq4piV	0.000009	4	0.2580	-2.226	0	AYA
170651	0.1950	2020	0.00998	['Eminem']	0.671	337147	0.623	1	5SiZJoLXp3WOl3J4C8IK0d	0.000008	2	0.6430	-7.161	1	Darkness
170652	0.6420	2020	0.13200	['KEVVO', 'J Balvin']	0.856	189507	0.721	1	7HmnJHfs0BkFzX4x8j0hkl	0.004710	7	0.1820	-4.928	1	Billetes Azules (with J Balvin)

Figura 4 – Resultado do código da figura 3.
Fonte: Elaborado pelo autor (2023)

- pd.head() e pd.tail()

Essas são duas das funções mais utilizadas no Pandas para a visualização tabular da nossa base de dados. O .head() visualiza os 5 primeiros registro do nosso dataframe.

```
df.head()
```

	valence	year	acousticness	artists	danceability	duration_ms	energy	explicit	id	instrumentalness	key	liveness	loudness
0	0.0594	1921	0.982	['Sergei Rachmaninoff', 'James Levine', 'Berli...	0.279	831667	0.211	0	4BJqT0PrAfrxzMOxytFOIz	0.878000	10	0.665	-20.096
1	0.9630	1921	0.732	['Dennis Day']	0.819	180533	0.341	0	7xPhfUan2yNtyFG0cUWkt8	0.000000	7	0.160	-12.441
2	0.0394	1921	0.961	['KHP Kridhamardawa Karaton Ngayogyakarta Hadi...	0.328	500062	0.166	0	1o6I8BglA6ylDMrIELygv1	0.913000	3	0.101	-14.850
3	0.1650	1921	0.967	['Frank Parker']	0.275	210000	0.309	0	3ftBPsC5vPBKxYSee08FDH	0.000028	5	0.381	-9.316
4	0.2530	1921	0.957	['Phil Regan']	0.418	166693	0.193	0	4d6HGyGT8e121BsdKmw9v6	0.000002	3	0.229	-10.096

Figura 5 – .head().
Fonte: Elaborado pelo autor (2023)

Já o .tail() visualiza os 5 últimos registros do nosso dataframe:

```
df.tail()
```

	valence	year	acousticness	artists	danceability	duration_ms	energy	explicit	id	instrumentalness	key	liveness
170648	0.608	2020	0.08460	['Anuel AA', 'Daddy Yankee', 'KAROL G', 'Ozuna...	0.786	301714	0.808	0	0KklkfsLEJbrclhYsCL7L5	0.000289	7	0.0822
170649	0.734	2020	0.20600	['Ashnikko']	0.717	150654	0.753	0	0OStKKAuXlxA0fMH54Qs6E	0.000000	7	0.1010
170650	0.637	2020	0.10100	['MAMAMOO']	0.634	211280	0.858	0	4BZXVFYCb76Q0Klojq4pIV	0.000009	4	0.2580
170651	0.195	2020	0.00998	['Eminem']	0.671	337147	0.623	1	5SiZJoLXp3WOl3J4C8IK0d	0.000008	2	0.6430
170652	0.642	2020	0.13200	['KEVVO', 'J Balvin']	0.856	189507	0.721	1	7HmnJHfs0BkFzX4x8j0hkl	0.004710	7	0.1820

Figura 6 – .tail()
Fonte: Elaborado pelo autor (2023)

- pd.shape

O .shape tem a função de retornar um par de números representando a dimensionalidade do conjunto de dados. É muito útil para entender a quantidade de características (colunas) e registros do dataset.

Figura 7 – .shape()
Fonte: Elaborado pelo autor (2023)

- pd.describe()

A função .describe() faz exatamente o que diz, ela descreve a nossa base de dados. Lembram das aulas de estatística? A função usa ferramentas estatísticas que resumem a tendência central das variáveis, sua dispersão, a presença de valores nulos e sua forma.

```
df.describe()
```

	valence	year	acousticness	danceability	duration_ms	energy	explicit	instrumentalness	key	liveness	loudness	mode	popula
count	170653.000000	170653.000000	170653.000000	170653.000000	1.706530e+05	170653.000000	170653.000000	170653.000000	170653.000000	170653.000000	170653.000000	170653.000000	170653.00
mean	0.528587	1976.787241	0.502115	0.537396	2.309483e+05	0.482389	0.084575	0.167010	5.199844	0.205839	-11.467990	0.706902	31.43
std	0.263171	25.917853	0.376032	0.176138	1.261184e+05	0.267646	0.278249	0.313475	3.515094	0.174805	5.697943	0.455184	21.82
min	0.000000	1921.000000	0.000000	0.000000	5.108000e+03	0.000000	0.000000	0.000000	0.000000	0.000000	-60.000000	0.000000	0.00
25%	0.317000	1956.000000	0.102000	0.415000	1.698270e+05	0.255000	0.000000	0.000000	2.000000	0.098800	-14.615000	0.000000	11.00
50%	0.540000	1977.000000	0.516000	0.548000	2.074670e+05	0.471000	0.000000	0.000216	5.000000	0.136000	-10.580000	1.000000	33.00
75%	0.747000	1999.000000	0.893000	0.668000	2.624000e+05	0.703000	0.000000	0.102000	8.000000	0.261000	-7.183000	1.000000	48.00
max	1.000000	2020.000000	0.996000	0.988000	5.403500e+06	1.000000	1.000000	1.000000	11.000000	1.000000	3.855000	1.000000	100.00

Figura 8 – .describe()
Fonte: Elaborado pelo autor (2023)

Agora, lembram de matrizes? Observem como é importante passar pela aula de álgebra linear para que possamos entender a tabela. Para visualizarmos melhor o nosso dataset, vamos atribuir uma pequena função para transpor a nossa tabela gerada pela função .describe().Perceba como podemos visualizar melhor a base de dados utilizando o .describe().T.

```
df.describe().T
```

	count	mean	std	min	25%	50%	75%	max
valence	170653.0	0.528587	0.263171	0.0	0.3170	0.540000	0.7470	1.000
year	170653.0	1976.787241	25.917853	1921.0	1956.0000	1977.000000	1999.0000	2020.000
acousticness	170653.0	0.502115	0.376032	0.0	0.1020	0.516000	0.8930	0.996
danceability	170653.0	0.537396	0.176138	0.0	0.4150	0.548000	0.6680	0.988
duration_ms	170653.0	230948.310666	126118.414668	5108.0	169827.0000	207467.000000	262400.0000	5403500.000
energy	170653.0	0.482389	0.267646	0.0	0.2550	0.471000	0.7030	1.000
explicit	170653.0	0.084575	0.278249	0.0	0.0000	0.000000	0.0000	1.000
instrumentalness	170653.0	0.167010	0.313475	0.0	0.0000	0.000216	0.1020	1.000
key	170653.0	5.199844	3.515094	0.0	2.0000	5.000000	8.0000	11.000
liveness	170653.0	0.205839	0.174805	0.0	0.0988	0.136000	0.2610	1.000
loudness	170653.0	-11.467990	5.697943	-60.0	-14.6150	-10.580000	-7.1830	3.855
mode	170653.0	0.706902	0.455184	0.0	0.0000	1.000000	1.0000	1.000
popularity	170653.0	31.431794	21.826615	0.0	11.0000	33.000000	48.0000	100.000
speechiness	170653.0	0.098393	0.162740	0.0	0.0349	0.045000	0.0756	0.970
tempo	170653.0	116.861590	30.708533	0.0	93.4210	114.729000	135.5370	243.507

Figura 9 – .describe().T
Fonte: Elaborado pelo autor (2023)

- pd.info()

O .info() é um resumo mais curto do nosso conjunto de dados. Porém, é muito utilizado para visualizar o tipo de coluna de cada coluna possui tais como object(String), float64, int64 entre outros.

```
df.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
```

Figura 10 – .info()
Fonte: Elaborado pelo autor (2023)

- pd.duplicated() e pd.isnull()

Começaremos a utilizar funções que ajudam a identificar quantidade de dados duplicado e dados nulos

O .duplicated() irá mostrar a quantidade de dados duplicados dentro do dataset.

```
df.duplicated()

0         False
1         False
2         False
3         False
4         False
          ...
170648    False
170649    False
170650    False
170651    False
170652    False
Length: 170653, dtype: bool
```

Figura 11 – .duplicated()
Fonte: Elaborado pelo autor (2023)

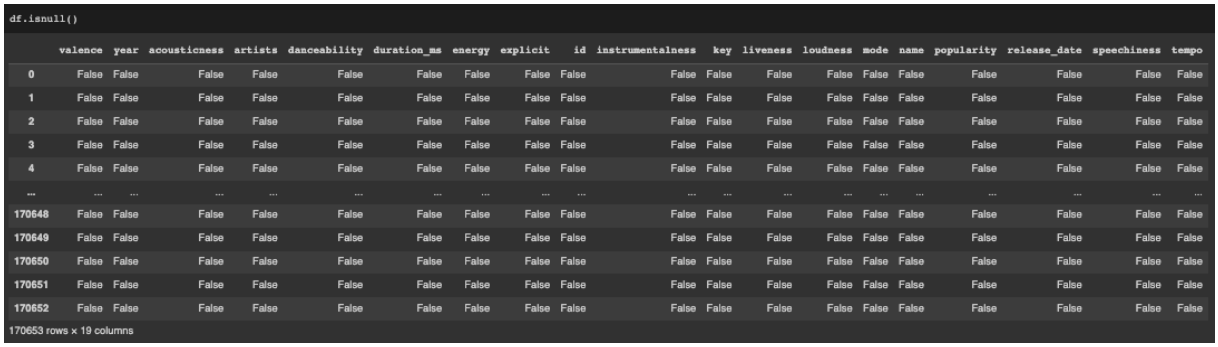
Percebam que fica difícil visualizar se há registros duplicados. Então, utilizamos uma função nativa do Python conhecida por todos, o .sum(). Logo quando unimos a função .duplicated().sum(), observe o resultado. No caso, não temos dados duplicados.

```
[22] df.duplicated().sum()

     0
```

Figura 12 – .duplicated().sum()
Fonte: Elaborado pelo autor (2023)

Quando queremos visualizar se há dados nulos no nosso conjunto de dados, utilizamos a função .isnull().



df.isnull()

	valence	year	acousticness	artists	danceability	duration_ms	energy	explicit	id	instrumentalness	key	liveness	loudness	mode	name	popularity	release_date	speechiness	tempo
0	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
170648	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False
170649	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False
170650	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False
170651	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False
170652	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False

170653 rows × 19 columns

Figura 13 – .isnull()
Fonte: Elaborado pelo autor (2023)

Vejam que a visualização fica difícil. Quando adicionarmos .sum(), conseguiremos ver a quantidade de dados nulos na case de dados.

```
df.isnull().sum()

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
```

Figura 14 – .isnull().sum()
Fonte: Elaborado pelo autor (2023)

- Compreensão das variáveis
- pd.value_counts()

Esta função .value_counts() é bem utilizada para entender a quantidade de valores de uma determinada variável existente no nosso conjunto de dados. Essa é uma função que funciona muito bem com variáveis categóricas.

```
df['artists'].value_counts()
```

```
['Эрнест Хемингуэй']                                  1211
['Эрих Мария Ремарк']                                 1068
['Francisco Canaro']                                   942
['Frank Sinatra']                                      630
['Ignacio Corsini']                                    628
                                                      ...
['Francisco Canaro', 'Ernesto Fama', 'Ada Falcón']       1
['Kings (MHM)']                                          1
['Daddy Yankee', 'Prince Royce']                         1
['Flume', 'T-Shirt']                                     1
['KEVVO', 'J Balvin']                                    1
Name: artists, Length: 34088, dtype: int64
```

Figura 15 – .value_counts()
Fonte: Elaborado pelo autor (2023)

Vamos visualizar a quantidade de artistas?

```
df['artists'].value_counts().head().plot(kind = 'bar')
plt.title("Os 5 primeiros artistas do nosso Dataset")
plt.xlabel("Artista")
plt.xticks(rotation=90)
plt.ylabel("Quantidade")
plt.show()
```

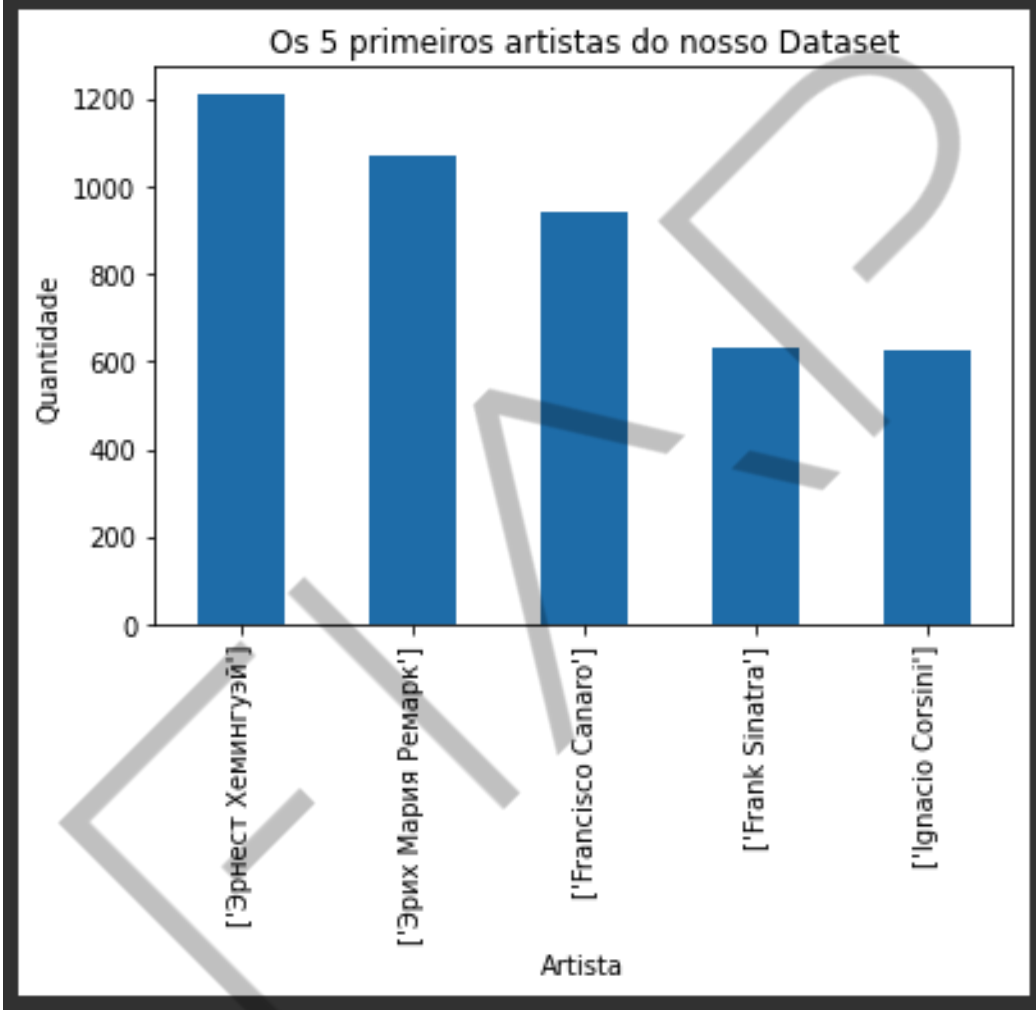


Figura 16 – Quantidade de artistas
Fonte: Elaborado pelo autor (2023)

- pd.hist()

Agora, quando quisermos analisar as variáveis numéricas, podemos utilizar a função .hist(). Essa função irá mostrar a frequência de uma determinada característica.

```
df.popularity.describe()

count    170653.000000
mean         31.431794
std          21.826615
min           0.000000
25%          11.000000
50%          33.000000
75%          48.000000
max         100.000000
Name: popularity, dtype: float64

df.popularity.hist()

<Axes: >
```

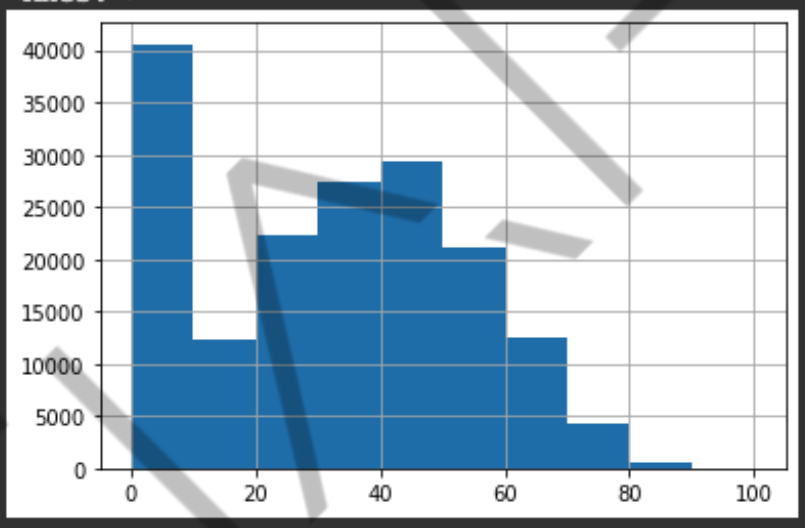


Figura 17 – .hist()
Fonte: Elaborado pelo autor (2023)

- Estudo das relações entre as variáveis

Uma questão bem interessante quando estamos realizando a análise exploratória dos dados, é mostrar a influência de uma variável sobre a outra.

Uma ferramenta bastante utilizada para ter a primeira visualização dos dados é uma função da biblioteca do Seaborn, o Pairplot.

```
sns.pairplot(df)
```

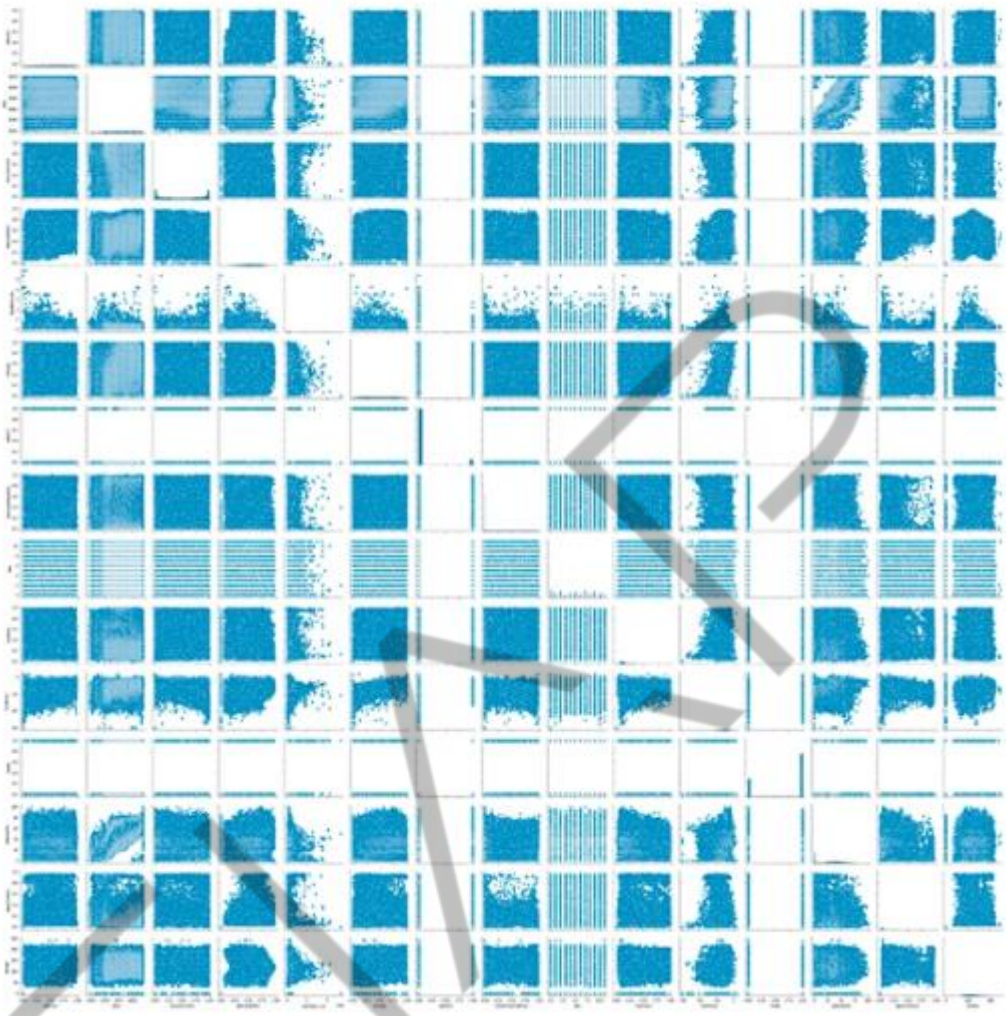


Figura 18 – Resultado da execução da função pairplot
Fonte: Elaborado pelo autor (2023)

Dependendo do nosso dataset, a visualização fica difícil entre a correlação entre as variáveis.

A melhor maneira de entender a relação entre uma variável numérica e uma variável categórica é por meio de um boxplot.

Vamos entender qual é a popularidade (‘popularity’) de uma música quando as músicas possuem palavrão (‘explicit’), onde 0 é não explicito e 1 é explicito?

```
df['explicit'].value_counts()
```

```
0    156220
1     14433
Name: explicit, dtype: int64
```

```
sns.catplot(x="explicit", y="popularity", data=df, kind="box", aspect=1.5)
plt.title("Boxplot for explicit vs popularity")
plt.show()
```

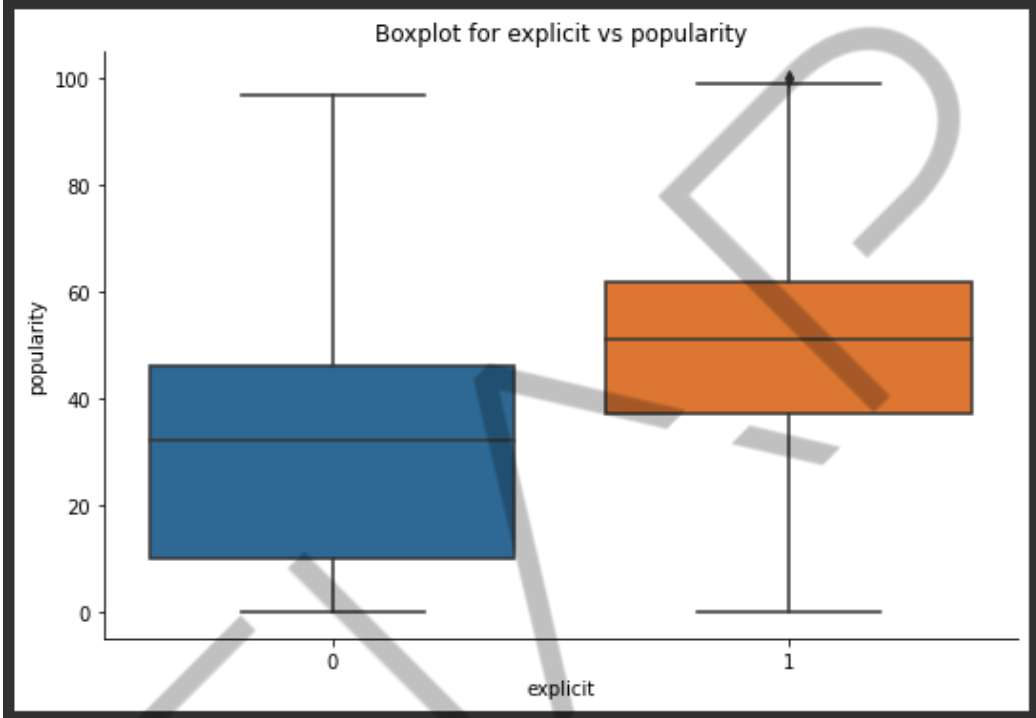


Figura 19 – Boxplot
Fonte: Elaborado pelo autor (2023)

Agora, visualmente já começamos a entender qual é a popularidade das músicas quando há palavrões em suas letras.

Mas já pensou em fazer uma correlação entre variáveis, uma a uma? Pensando nisso, o Seaborn desenvolveu uma maneira de desenvolver uma função onde é possível ver quando há correlação entre as variáveis.

```
plt.figure(figsize=(15, 6))
corr = sns.heatmap(df.corr(), annot=True, cmap="Spectral_r")
plt.show()
```

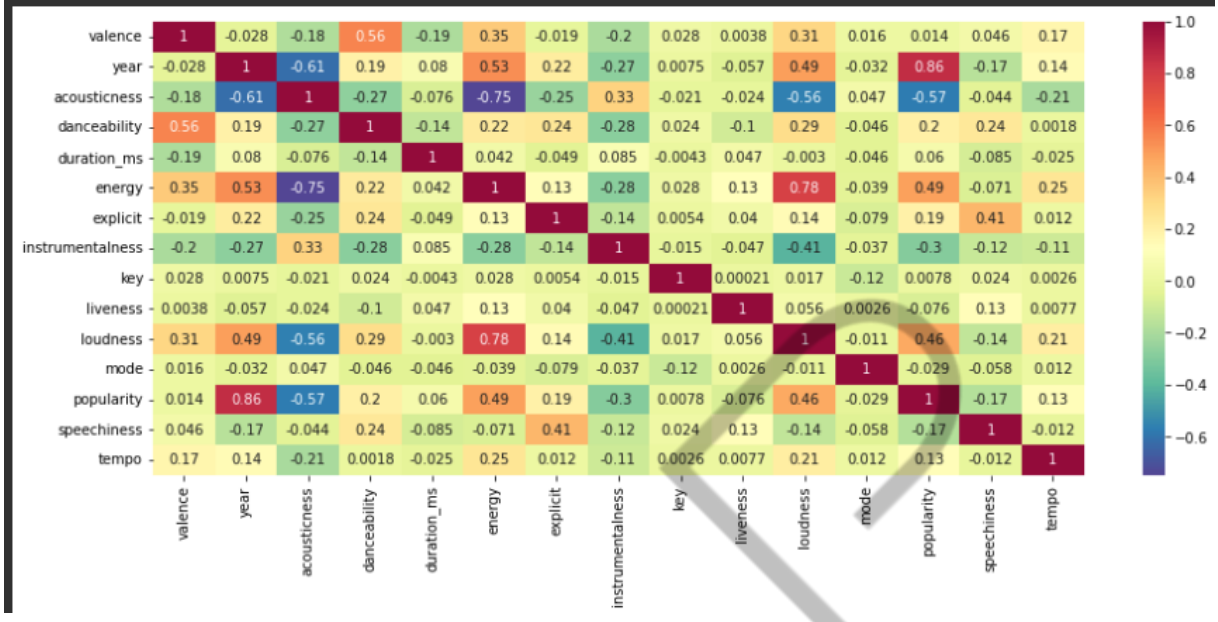


Figura 20 – Mapa de calor
Fonte: Elaborado pelo autor (2023)

O mapa de calor é útil porque nos permite entender com eficiência quais variáveis estão fortemente correlacionadas entre si.

Observe: quando temos valores próximos de 1 ou -1, significa que as variáveis estão fortemente correlacionadas. Quando os números são positivos significa que as correlações das variáveis são diretamente proporcionais. Logo, quando os números são negativos, as variáveis tem uma correlação inversamente proporcional.

Que tal dar uma lida no artigo do blog do postdigital? O autor faz uma boa visão sobre análise exploratória de dados no marketing digital:

Artigo do postdigital.

O QUE VOCÊ VIU NESTA AULA?

Introdução à Análise Exploratória dos Dados; começou a ter contato com as principais bibliotecas fundamentais para EDA; aprendeu técnicas e visualização de EDA.

Daqui em diante, é importante que você replique os conhecimentos adquiridos para fortalecer mais suas bases e conhecimentos.

IMPORTANTE: não esqueça de praticar com o desafio da disciplina, para que assim você possa aprimorar os seus conhecimentos!

Você não está sozinho(a) nesta jornada! Te esperamos no Discord e nas lives com os nossos(as) professores(as) especialistas, onde você poderá tirar dúvidas, compartilhar conhecimentos e estabelecer conexões!

REFERÊNCIAS

DOCUMENTAÇÃO PANDAS. **Disponível em: <https://**pandas**.pydata.org/>.** Acesso em: 04 mai 2023.

DOCUMENTAÇÃO seaborn. **Disponível em: <https://seaborn.pydata.org/api.html>.** Acesso em: 04 mai 2023.

GOOGLE COLAB. **Disponível em: <https://colab.research.google.com/>.** Acesso em: 04 mai 2023.

GRUS, Joel. **Data Science do Zero**. Rio de Janeiro: Alta Books Editora, 2016.

HARRISON, Matt. **Machine Learning**: guia de referência rápida - trabalhando com dados estruturados em python. São Paulo: O'Reilly Media, 2019.

kaggle. **Disponível em: <https://www.kaggle.com/datasets/mrmorj/dataset-of-songs-in-spotify>.** Acesso em: 04 mai 2023.

MCKINNEY, Wes. **Python para análise de dados**: tratamento de dados com pandas, numpy e ipython. São Paulo: Novatec Editora, 2019. 616 p.

PALAVRAS-CHAVE

Palavras-chave: Python. Pandas. EDA. Seaborn.