

PYTHON PARA ANÁLISE DE DADOS

Escola: SoulCode Academy

Curso: Analista de Dados e Dashboard

Assunto: ETL

Professora: Franciane Rodrigues

Aluno (a): Márcia

Estatística Descritiva

- As estatísticas descritivas envolvem os métodos usados para resumir, simplificar e descrever os principais recursos de um conjunto de dados.
- Este ramo da estatística visa fornecer uma visão geral concisa e significativa dos dados por meio de medidas como tendência central (média, mediana, moda), dispersão (intervalos, variância, desvio-padrão) e representações gráficas, tais como histogramas, gráficos de barras, gráfico de setores, gráficos de dispersão e etc.
- As estatísticas descritivas nos ajudam a compreender as características fundamentais de um conjunto de dados e a torná-lo mais gerenciável para análise.

Solicitação

- Realize uma análise exploratória de dados (EDA) usando a estatística descritiva para obter informações úteis sobre o desempenho de vídeos publicados no Youtube.


Dicionário de Dados

- Title: Título do vídeo.
- Video ID: O identificador de vídeo.
- Published At: A data em que o vídeo foi publicado YYYY-MM-DD.
- Keyword: a palavra-chave associada ao vídeo.
- Likes: o número de curtidas que o vídeo recebeu. Se este valor for -1, as curtidas não serão visíveis publicamente.
- Comments: o número de comentários que o vídeo tem. Se esse valor for -1, o criador do vídeo desativou os comentários.
- Views: o número de visualizações que o vídeo obteve.

Mais informações: <https://www.kaggle.com/datasets/advaypatil/youtube-statistics?select=videos-stats.csv>

✓ Abertura e Extração dos Dados

```
1 # Comando de abertura para google drive
2 from google.colab import drive
3 drive.mount('/content/drive')
```

 Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).


```
1 # Importando biblioteca
2 import pandas as pd
```

```
1 # Configuração da quantidade de colunas para aparecer em um dataframe
2 pd.set_option('display.max_columns', 100)
```

- **Tipos de separador:** Vírgula (,) - Ponto e vírgula (;) - Tabulação (\t) - Espaço em branco ()
- **Encoding:** UTF-8, ISO-8859-1, UTF-16, UTF-32, ASCII

```
1 # Extração por Google Drive
2 df = pd.read_csv('/content/drive/MyDrive/CHAVES/youtube_bruto.csv')
```


```
1 # Visualização do DataFrame
2 df
```



	Unnamed: 0	Title	Video ID	Published At	Keyword	Likes	Comments	Views
0	0	Apple Pay Is Killing the Physical Wallet After...	wAZZ-UWGVHI	2022-08-23	tech	3407.0	672.0	135612.0
1	1	The most EXPENSIVE thing I own.	b3x28s61q3c	2022-08-24	tech	76779.0	4306.0	1758063.0
2	2	My New House Gaming Setup is SICK!	4mgePWWCAmA	2022-08-23	tech	63825.0	3338.0	1564007.0
3	3	Petrol Vs Liquid Nitrogen Freezing Experimen...	kXiYSI7H2b0	2022-08-23	tech	71566.0	1426.0	922918.0
4	4	Best Back to School Tech 2022!	ErMwWXQxHp0	2022-08-08	tech	96513.0	5155.0	1855644.0
...
1876	1876	Should You Learn Machine Learning?	AO6urf07KjE	2021-06-14	machine learning	10259.0	416.0	386360.0
1877	1877	Todos podemos aprender Machine learning	7CILKBUvmRk	2017-10-08	machine learning	2981.0	72.0	431421.0
1878	1878	Andrew Ng: Deep Learning, Education, and Real-...	0jspaMLxBig	2020-02-20	machine learning	5198.0	443.0	226152.0


✓ Pré-Análise

```
1 # Visualização do cabeçalho
2 df.head()
```




	Unnamed: 0	Title	Video ID	Published At	Keyword	Likes	Comments	Views
0	0	Apple Pay Is Killing the Physical Wallet After...	wAZZ-UWGVHI	2022-08-23	tech	3407.0	672.0	135612.0
1	1	The most EXPENSIVE thing I own.	b3x28s61q3c	2022-08-24	tech	76779.0	4306.0	1758063.0
2	2	My New House Gaming Setup is SICK!	4mgePWWCAmA	2022-08-23	tech	63825.0	3338.0	1564007.0
3	3	Petrol Vs Liquid Nitrogen Freezing Experimen...	kXiYSI7H2b0	2022-08-23	tech	71566.0	1426.0	922918.0
4	4	Best Back to School Tech 2022!	ErMwWXQxHp0	2022-08-08	tech	96513.0	5155.0	1855644.0

```
1 # Visualização do dataframe (shift + enter)
2 df
```



	Unnamed: 0	Title	Video ID	Published At	Keyword	Likes	Comments	Views
0	0	Apple Pay Is Killing the Physical Wallet After...	wAZZ-UWGVHI	2022-08-23	tech	3407.0	672.0	135612.0
1	1	The most EXPENSIVE thing I own.	b3x28s61q3c	2022-08-24	tech	76779.0	4306.0	1758063.0
2	2	My New House Gaming Setup is SICK!	4mgePWWCAmA	2022-08-23	tech	63825.0	3338.0	1564007.0
3	3	Petrol Vs Liquid Nitrogen Freezing Experimen...	kXiYSI7H2b0	2022-08-23	tech	71566.0	1426.0	922918.0
4	4	Best Back to School Tech 2022!	ErMwWXQxHp0	2022-08-08	tech	96513.0	5155.0	1855644.0
...
1876	1876	Should You Learn Machine Learning?	AO6urf07KjE	2021-06-14	machine learning	10259.0	416.0	386360.0
1877	1877	Todos podemos aprender Machine learning	7CILKBUvmRk	2017-10-08	machine learning	2981.0	72.0	431421.0
1878	1878	Andrew Ng: Deep Learning, Education, and Real-...	0jspaMLxBig	2020-02-20	machine learning	5198.0	443.0	226152.0

```
1 # Visualizando as últimas posições do dataframe
2 df.tail()
```



	Unnamed: 0	Title	Video ID	Published At	Keyword	Likes	Comments	Views
1876	1876	Should You Learn Machine Learning?	AO6urf07KjE	2021-06-14	machine learning	10259.0	416.0	386360.0
1877	1877	Todos podemos aprender Machine learning	7CILKBUvmRk	2017-10-08	machine learning	2981.0	72.0	431421.0
1878	1878	Andrew Ng: Deep Learning, Education, and Real-...	0jspaMLxBig	2020-02-20	machine learning	5198.0	443.0	226152.0

✓ Diferenças entre Nan, Null e Undefined

- NaN - resultado de alguma operação ou função que não conseguiu retornar um valor real (não é o mesmo que ser igual a zero ou valor vazio);
- Null - um valor não definido ou ausência de um valor;
- Undefined - variável declarada, valor não declarado.

Tudo parece a mesma coisa, mas não é. o peso em Bits é totalmente diferente e reage diferente

```
1 # Visualização dp dataframe de forma aleatória
2 df.sample(4)
```

	Unnamed: 0	Title	Video ID	Published At	Keyword	Likes	Comments	Views
1571	1571	Bayonetta 3 - Release Date Revealed - Nintendo...	gOGJGv6OoXA	2022-07-13	nintendo	43557.0	4766.0	514758.0
81	81	UP News: कहा गायब हो गया Abbas Ansari ? 7 राज...	_JGrWrwOoD8	2022-08-23	news	286.0	22.0	23135.0
1319	1319	Last To Leave Roller Coaster Wins \$20,000 - Ch...	gL6iSCSHjco	2019-08-17	mrbeast	2121115.0	65601.0	92621700.0
131	131	HUGE \$500 CUBE UNBOXING What Cubes Did I	1AykVv81FI	2020-06-16	cubes	10051.0	1026.0	900060.0

```
1 # Verificação o tamanho do dataframe (linhas e colunas)
2 df.shape
```

```
(1881, 8)
```

```
1 # Verificando o topo de dados de cada coluna
2 df.dtypes
```

```
Unnamed: 0      int64
Title           object
Video ID        object
Published At     object
Keyword         object
Likes           float64
Comments        float64
Views           float64
dtype: object
```

```
1 # Vamos contar quantas observações nós temos em cada coluna
2 df.count()
```

```
Unnamed: 0      1881
Title           1881
Video ID        1881
Published At     1881
Keyword         1881
Likes           1879
Comments        1879
Views           1879
dtype: int64
```

```
1 # Informações detalhadas do conjunto de dados
2 df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1881 entries, 0 to 1880
Data columns (total 8 columns):
#   Column          Non-Null Count  Dtype
---  -
0   Unnamed: 0      1881 non-null   int64
1   Title           1881 non-null   object
2   Video ID       1881 non-null   object
3   Published At   1881 non-null   object
4   Keyword        1881 non-null   object
5   Likes          1879 non-null   float64
6   Comments       1879 non-null   float64
7   Views          1879 non-null   float64
dtypes: float64(3), int64(1), object(4)
memory usage: 117.7+ KB
```

Com a fase de Pré-visualização responda algumas perguntas gerais:

- Qual das colunas são importantes na sua opinião?
- Qual das colunas poderão ser retiradas da sua análise?
- Há colunas iguais?

- Há linhas iguais?
- Os valores são únicos em uma coluna?
- Há alguma inconsistência nas colunas?
- Há valores nulos?
- Precisa renomear as colunas?
- Precisa renomear os dados categóricos?
- Os tipos de dados estão adequados nas colunas?

✓ Transformação

Verificações gerais e básicas de qualquer base de dados, se houver os casos abaixo:

- Escolher as possíveis colunas importantes para a análise;
- Verificar se há colunas iguais;
- Verificação e tratamento de valores (ou observações) únicos;
- Verificação e tratamento de duplicadas
- Verificação e tratamento de inconsistência
- Verificação e tratamento de valores nulos
- Tradução/Renomeação
- Transformação de tipos adequados dos dados
- Garantia de qualidade e integridade dos dados

```
1 # Verificação
2 df.head(3)
```

	Unnamed: 0	Title	Video ID	Published At	Keyword	Likes	Comments	Views
0	0	Apple Pay Is Killing the Physical Wallet After...	wAZZ-UWGVHI	2022-08-23	tech	3407.0	672.0	135612.0
1	1	The most EXPENSIVE thing I own.	b3x28s61q3c	2022-08-24	tech	76779.0	4306.0	1758063.0
2	2	My New House Gaming Setup is SICK!	4mgePWWCAmA	2022-08-23	tech	63825.0	3338.0	1564007.0

✓ Renomeação de colunas

```
1 # Transformação
2 # Renomeando colunas para tradução
3 df.rename(columns={'Unnamed: 0': 'linha',
4                    'Title': 'titulo',
5                    'Video ID': 'video_id',
6                    'Published At': 'data_publicacao',
7                    'Keyword': 'palavra_chave',
8                    'Likes': 'qtd_curtidas',
9                    'Comments': 'qtd_comentarios',
10                   'Views': 'qtd_visualizacoes'}, inplace=True)
```

```
1 # Checagem da transformação
2 df.head(2)
```

	linha	titulo	video_id	data_publicacao	palavra_chave	qtd_curtidas	qtd_comentarios	qtd_visualizacoes
0	0	Apple Pay Is Killing the Physical Wallet After...	wAZZ-UWGVHI	2022-08-23	tech	3407.0	672.0	135612.0

✓ Valores nulos

```
1 import sys
2 import math
3 import numpy as np
4
5 # Tamanho de NaN
6 nan_size = sys.getsizeof(math.nan)
7 print(f'Tamanho de NaN: {nan_size} bytes')
8
9 # Tamanho de None (indefinido)
10 none_size = sys.getsizeof(None)
11 print(f'Tamanho de None: {none_size} bytes')
12
13 # Tamanho de Null (vazio) com pandas
```

```

14 null_size_pandas = sys.getsizeof(pd.NA)
15 print(f'Tamanho de Null pelo Pandas: {null_size_pandas} bytes')
16 # Tamanho de Null (vazio) com numpy
17 null_size_numpy = sys.getsizeof(np.nan)
18 print(f'Tamanho de Null pelo Numpy: {null_size_numpy} bytes')

```

```

↗ Tamanho de NaN: 24 bytes
Tamanho de None: 16 bytes
Tamanho de Null pelo Pandas: 48 bytes
Tamanho de Null pelo Numpy: 24 bytes

```

```

1 # Verificando valores nulos
2 df.isnull().sum()

```

```

↗ linha      0
  titulo     0
  video_id   0
  data_publicacao  0
  palavra_chave  0
  qtd_curtidas  2
  qtd_comentarios  2
  qtd_visualizacoes  2
  dtype: int64

```

```

1 # Verificando valores nulos (2)
2 df.isna().sum()

```

```

↗ linha      0
  titulo     0
  video_id   0
  data_publicacao  0
  palavra_chave  0
  qtd_curtidas  2
  qtd_comentarios  2
  qtd_visualizacoes  2
  dtype: int64

```

```

1 # Verificando valores nulos via porcentagem
2 porcentagem_nulos = (df.isnull().sum() / len(df)) * 100
3
4 # Exibir a porcentagem de valores nulos em cada coluna
5 print(round(porcentagem_nulos,1))

```

```

↗ linha      0.0
  titulo     0.0
  video_id   0.0
  data_publicacao  0.0
  palavra_chave  0.0
  qtd_curtidas  0.1
  qtd_comentarios  0.1
  qtd_visualizacoes  0.1
  dtype: float64

```

```

1 # Colocar um filtro dos dados nulos
2 df[df.isnull().any(axis=1)]

```

```

↗
```

	linha	titulo	video_id	data_publicacao	palavra_chave	qtd_curtidas	qtd_comentarios	qtd_visualizacoes
1879	1879	What is Machine Learning?	f_uwKZIAeM0	2017-01-11	machine learning	NaN	NaN	NaN

```

1 # Removendo linhas com valores nulos
2 df = df.dropna()

```

```

1 # Verificando valores nulos
2 df.isnull().sum()

```

```

↗ linha      0
  titulo     0
  video_id   0
  data_publicacao  0
  palavra_chave  0
  qtd_curtidas  0
  qtd_comentarios  0
  qtd_visualizacoes  0
  dtype: int64

```

```

1 # Checagem
2 df.info()

```

```
<class 'pandas.core.frame.DataFrame'>
Index: 1879 entries, 0 to 1878
Data columns (total 8 columns):
#   Column                Non-Null Count  Dtype
---  ---
0   linha                 1879 non-null   int64
1   titulo                1879 non-null   object
2   video_id              1879 non-null   object
3   data_publicacao       1879 non-null   object
4   palavra_chave         1879 non-null   object
5   qtd_curtidas          1879 non-null   float64
6   qtd_comentarios       1879 non-null   float64
7   qtd_visualizacoes    1879 non-null   float64
dtypes: float64(3), int64(1), object(4)
memory usage: 132.1+ KB
```

✓ Valores unicos e dados duplicados

```
1 # Verificando se os dados são unicos na coluna 'video_id'
2 # Pelo fato de ter aparecido 'false', significa que temos dados duplicados
3 df.video_id.is_unique
```

```
False
```

```
1 # Encontrar linhas duplicadas com base na coluna video_id organizado
2 duplicidade = df[df.duplicated(subset='video_id', keep=False)]
```

```
1 # Organizar pela coluna titulo
2 duplicidade.sort_values(by=['titulo'])
```

	linha	titulo	video_id	data_publicacao	palavra_chave	qtd_curtidas	qtd_comentarios	qtd_visualizacoes
319	319	20 Minecraft Block Facts You Maybe Didn't ...	LeC5yJq4tsl	2022-08-21	tutorial	57526.0	1115.0	1204024.0
1220	1220	20 Minecraft Block Facts You Maybe Didn't ...	LeC5yJq4tsl	2022-08-21	minecraft	57527.0	1115.0	1204024.0
129	129	ASMR Gaming 🎮 Fortnite 1 Kill = 1 Trigger Rela...	mqc6QqoGNWI	2022-08-24	gaming	563.0	85.0	14537.0
848	848	ASMR Gaming 🎮 Fortnite 1 Kill = 1 Trigger Rela...	mqc6QqoGNWI	2022-08-24	asmr	563.0	86.0	14537.0
916	916	BLACKPINK - 'Pink Venom' DANCE PRACTICE VIDEO	RFMi3v0TXP8	2022-08-24	music	3001232.0	110160.0	23836066.0
1041	1041	BLACKPINK - 'Pink Venom' DANCE PRACTICE VIDEO	RFMi3v0TXP8	2022-08-24	reaction	3001265.0	110162.0	23836066.0
1501	1501	Computer Scientist Explains Machine Learning i...	5q87K1WaoFI	2021-08-18	computer science	42940.0	1735.0	1407319.0
1832	1832	Computer Scientist Explains Machine Learning i...	5q87K1WaoFI	2021-08-18	machine learning	15137.0	181.0	906372.0
225	225	How to Solve a Rubik's Cube WIRED	R-R0KrXvWbc	2019-09-05	how-to	339758.0	32718.0	29905105.0
423	423	How to Solve a Rubik's Cube WIRED	R-R0KrXvWbc	2019-09-05	cubes	339759.0	32717.0	29905105.0
91	91	I OPENED MY OWN ARCADE SHOP	WBK2_ID7KGA	2022-08-24	gaming	298406.0	15609.0	3773387.0
1229	1229	I OPENED MY OWN ARCADE SHOP	WBK2_ID7KGA	2022-08-24	minecraft	298445.0	15610.0	3773387.0

```
1 # verificando tamanho do dataframe
2 df.shape
```

```
(1879, 8)
```

```
1 # Eliminando duplicadas via video_id
2 df = df.drop_duplicates(subset='video_id')
```

```
1 # verificando tamanho do dataframe
2 df.shape
```

(1867, 8)

```
1 # Verificando
2 print('A coluna video_id possui valores nulos? Resposta: ', df.video_id.is_unique)
```

A coluna video_id possui valores nulos? Resposta: True

```
1 # Verificando valores unicos na coluna 'titulo'
2 # Como a resposta foi false, significa que temos titulos iguais na coluna
3 print('A coluna titulo possui valores unicos? Resposta: ', df.titulo.is_unique)
```

A coluna titulo possui valores unicos? Resposta: False

```
1 # Encontrar linhas duplicadas com base na coluna 'titulo'
2 df[df.duplicated(subset='titulo', keep=False)]
```

	linha	titulo	video_id	data_publicacao	palavra_chave	qtd_curtidas	qtd_comentarios	qtd_visualizacoes
128	128	Music Mix 2022 🎧 EDM Remixes of Popular Songs ...	gTUkevqZXKY	2022-08-22	gaming	2750.0	116.0	208701.0
229	229	Tiësto - The Business (Lyrics)	2AMjrc7cpHY	2020-09-25	business	79281.0	1683.0	9090985.0
231	231	Tiësto - The Business (Lyrics)	4x2S3LhbM0k	2020-12-20	business	9178.0	111.0	933111.0
232	232	Tiësto - The Business (Lyrics)	fhTUAIOSRi0	2020-12-09	business	21367.0	467.0	2822656.0
562	562	Weird Things Spotted On Google Maps	wrBMkRQkRNQ	2022-04-10	google	33213.0	3070.0	2017263.0
573	573	Weird Things Spotted On Google Maps	U-ZMQliRfvk	2021-04-26	google	237800.0	20092.0	9277945.0
917	917	Music Mix 2022 🎧 EDM Remixes of Popular Songs ...	jEC3MXBntp0	2022-08-24	music	1634.0	52.0	148482.0
985	985	ASMR MUKBANG 직 접 만든 양념 치킨먹방! 스테이크 짜파게티 레 시피 &...	vXp3lzESmnw	2022-05-15	mukbang	281285.0	3360.0	17282158.0
994	994	ASMR MUKBANG 직 접 만든 타키스 대왕 가 래떡 떡볶이 불닭볶음 면 치즈스틱 핫...	OrNmI0ArHdk	2022-01-06	mukbang	141654.0	2756.0	11184537.0
1001	1001	ASMR MUKBANG 직 접 만든 타키스 대왕 가 래떡 떡볶이 불닭볶음 면 치즈스틱 핫...	Jv7ZiuzZzrk	2022-05-21	mukbang	95162.0	2272.0	11235146.0
1005	1005	ASMR MUKBANG 직 접 만든 양념 치킨먹방! 스테이크 짜파게티 레 시피 &...	1QKshmg0AIY	2022-08-13	mukbang	29955.0	557.0	1057862.0
1019	1019	ASMR MUKBANG 직 접 만든 양념 치킨먹방! 스테이크 짜파게티 레 시피 &...	RCJswTYsZyU	2022-05-09	mukbang	65719.0	791.0	2724499.0

```
1 # Eliminando duplicidade do titulo (cabe verificação mais aprofundada)
2 df = df.drop_duplicates(subset='titulo')
```

```
1 # Verificando tamanho
2 df.shape
```

(1853, 8)

```
1 # Verificando se dados são unicos na coluna na coluna 'titulo'
2 print('Tem apenas valores únicos na coluna titulo? Resposta', df.titulo.is_unique)
```

Tem apenas valores únicos na coluna titulo? Resposta True

```
1 # Lembrando do conjunto de dados
2 df.head()
```

	linha	titulo	video_id	data_publicacao	palavra_chave	qtd_curtidas	qtd_comentarios	qtd_visualizacoes
0	0	Apple Pay Is Killing the Physical Wallet After...	wAZZ-UWGVHI	2022-08-23	tech	3407.0	672.0	135612.0
1	1	The most EXPENSIVE thing I own.	b3x28s61q3c	2022-08-24	tech	76779.0	4306.0	1758063.0
2	2	My New House Gaming Setup is SICK!	4mgePWWCAmA	2022-08-23	tech	63825.0	3338.0	1564007.0

✓ Colunas importantes

```
1 # Lembrando do conjunto de dados
2 df.head()
3
4 # Eliminação de colunas desnecessárias primeiramente
5 # O parâmetro axis=1 indica que as colunas devem ser removidas
6 # (em oposição a linhas, que seria axis=0).
7 df.drop(['linha',
8         'video_id'], axis=1, inplace=True)
```

<ipython-input-40-6203c2ef1ee4>:7: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: [https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-df.drop\(\['linha',](https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-df.drop(['linha',)

```
1 # Outra forma de eliminar colunas
2 #df = df.drop(['linha',
3 #             'video_id'], axis=1)
```

```
1 # Verificação
2 df.head()
```

	titulo	data_publicacao	palavra_chave	qtd_curtidas	qtd_comentarios	qtd_visualizacoes
0	Apple Pay Is Killing the Physical Wallet After...	2022-08-23	tech	3407.0	672.0	135612.0
1	The most EXPENSIVE thing I own.	2022-08-24	tech	76779.0	4306.0	1758063.0
2	My New House Gaming Setup is SICK!	2022-08-23	tech	63825.0	3338.0	1564007.0
3	Petrol Vs Liquid Nitrogen Freezing Experimen...	2022-08-23	tech	71566.0	1426.0	922918.0
4	Best Back to School Tech 2022!	2022-08-08	tech	96513.0	5155.0	1855644.0

✓ Tipos de Dados

```
1 # Verificação dos tipos de dados
2 df.dtypes
```

```
titulo      object
data_publicacao  object
palavra_chave  object
qtd_curtidas  float64
qtd_comentarios  float64
qtd_visualizacoes  float64
dtype: object
```

```
1 # Convertendo o tipo de dados numéricos de float para int
2 df['qtd_curtidas'] = df['qtd_curtidas'].astype(int)
3 df['qtd_comentarios'] = df['qtd_comentarios'].astype(int)
4 df['qtd_visualizacoes'] = df['qtd_visualizacoes'].astype(int)
```

```
1 # Verificação as mudanças
2 df.dtypes
```

```
titulo      object
data_publicacao  object
palavra_chave  object
qtd_curtidas  int64
qtd_comentarios  int64
```



```
qtd_visualizacoes    int64
dtype: object
```

```
1 # Converter a coluna da data de publicação para datetime
2 df['data_publicacao'] = pd.to_datetime(df['data_publicacao'])
```

```
1 # Verificando a mudança
2 df.dtypes
```

```
↗ titulo                object
data_publicacao        datetime64[ns]
palavra_chave          object
qtd_curtidas           int64
qtd_comentarios        int64
qtd_visualizacoes     int64
dtype: object
```

```
1 # Lembrando do df
2 df.head()
```

```
↗
```

	titulo	data_publicacao	palavra_chave	qtd_curtidas	qtd_comentarios	qtd_visualizacoes
0	Apple Pay Is Killing the Physical Wallet After...	2022-08-23	tech	3407	672	135612
1	The most EXPENSIVE thing I own.	2022-08-24	tech	76779	4306	1758063
2	My New House Gaming Setup is SICK!	2022-08-23	tech	63825	3338	1564007
3	Petrol Vs Liquid Nitrogen Freezing Experimen...	2022-08-23	tech	71566	1426	922918
4	Best Back to School Tech 2022!	2022-08-08	tech	96513	5155	1855644

```
1 # Separando dia, mês e ano em colunas diferentes
2 df['dia'] = df.data_publicacao.dt.day
3 df['mes'] = df.data_publicacao.dt.month
4 df['ano'] = df.data_publicacao.dt.year
```

```
1 # Checando
2 df.head()
```

```
↗
```

	titulo	data_publicacao	palavra_chave	qtd_curtidas	qtd_comentarios	qtd_visualizacoes	dia	mes	ano
0	Apple Pay Is Killing the Physical Wallet After...	2022-08-23	tech	3407	672	135612	23	8	2022
1	The most EXPENSIVE thing I own.	2022-08-24	tech	76779	4306	1758063	24	8	2022
2	My New House Gaming Setup is SICK!	2022-08-23	tech	63825	3338	1564007	23	8	2022
3	Petrol Vs Liquid Nitrogen	2022-08-23	tech	71566	1426	922918	23	8	2022

```
1 # Verificação dos tipos de dados
2 df.dtypes
```

```
↗ titulo                object
data_publicacao        datetime64[ns]
palavra_chave          object
qtd_curtidas           int64
qtd_comentarios        int64
qtd_visualizacoes     int64
dia                    int32
mes                    int32
ano                    int32
dtype: object
```

```
1 # Convertendo o tipo de dados de int para str
2 df['dia'] = df['dia'].astype(str)
3 df['mes'] = df['mes'].astype(str)
4 df['ano'] = df['ano'].astype(str)
```

```
1 # Verificação as mudanças
2 df.dtypes
```

```
↗ titulo                object
data_publicacao        datetime64[ns]
palavra_chave          object
qtd_curtidas           int64
qtd_comentarios        int64
qtd_visualizacoes     int64
dia                    object
```

```
mes          object
ano          object
dtype: object
```

▼ Tradução de Categorias

```
1 # Verificando categorias
2 print(sorted(pd.unique(df['palavra_chave'])))
```

```
↵ ['animals', 'apple', 'asmr', 'bed', 'biology', 'business', 'chess', 'cnn', 'computer science', 'crypto', 'cubes', 'data science', 'e
```

```
1 # Verificando categorias
2 sorted(pd.unique(df['palavra_chave']))
```

```
↵ ['animals',
   'apple',
   'asmr',
   'bed',
   'biology',
   'business',
   'chess',
   'cnn',
   'computer science',
   'crypto',
   'cubes',
   'data science',
   'education',
   'finance',
   'food',
   'game development',
   'gaming',
   'google',
   'history',
   'how-to',
   'interview',
   'literature',
   'lofi',
   'machine learning',
   'marvel',
   'mathchemistry',
   'minecraft',
   'movies',
   'mrbeast',
   'mukbang',
   'music',
   'news',
   'nintendo',
   'physics',
   'reaction',
   'sat',
   'sports',
   'tech',
   'trolling',
   'tutorial',
   'xbox']
```

```
1 # Dicionário de mapeamento para valores em português
2 traduzindo = {'ANIMALS': 'ANIMAIS',
3               'APPLE': 'APPLE',
4               'ASMR': 'ASMR',
5               'BED': 'CAMA',
6               'BIOLOGY': 'BIOLOGIA',
7               'BUSINESS': 'NEGÓCIOS',
8               'CHESS': 'XADREZ',
9               'CNN': 'CNN',
10              'COMPUTER SCIENCE': 'CIÊNCIA DA COMPUTAÇÃO',
11              'CRYPTO': 'CRÍPTO',
12              'CUBES': 'CUBOS',
13              'DATA SCIENCE': 'CIÊNCIA DE DADOS',
14              'EDUCATION': 'EDUCAÇÃO',
15              'FINANCE': 'FINANÇAS',
16              'FOOD': 'COMIDA',
17              'GAME DEVELOPMENT': 'DESENVOLVIMENTO DE JOGOS',
18              'GAMING': 'JOGOS',
19              'GOOGLE': 'GOOGLE',
20              'HISTORY': 'HISTÓRIA',
21              'HOW-TO': 'COMO FAZER',
22              'INTERVIEW': 'ENTREVISTA',
23              'LITERATURE': 'LITERATURA',
24              'LOFI': 'LO-FI',
25              'MACHINE LEARNING': 'APRENDIZADO DE MÁQUINA',
26              'MARVEL': 'MARVEL',
```

```

27     'MATHCHEMISTRY': 'MATEMÁTICA E QUÍMICA',
28     'MINECRAFT': 'MINECRAFT',
29     'MOVIES': 'FILMES',
30     'MRBEAST': 'MRBEAST',
31     'MUKBANG': 'MUKBANG',
32     'MUSIC': 'MÚSICA',
33     'NEWS': 'NOTÍCIAS',
34     'NINTENDO': 'NINTENDO',
35     'PHYSICS': 'FÍSICA',
36     'REACTION': 'REAÇÃO',
37     'SAT': 'SAT',
38     'SPORTS': 'ESPORTES',
39     'TECH': 'TECNOLOGIA',
40     'TROLLING': 'TROLLAGEM',
41     'TUTORIAL': 'TUTORIAL',
42     'XBOX': 'XBOX'
43 }

```

```

1 # Substituir os valores na coluna 'palavra_chave'
2 df['palavra_chave'] = df['palavra_chave'].str.upper().replace(traduzindo)

```

```

1 # Verificação
2 df.sample(3)

```

	titulo	data_publicacao	palavra_chave	qtd_curtidas	qtd_comentarios	qtd_visualizacoes	dia	mes	ano
1772	How Data Science Helps Small Businesses	2022-08-24	CIÊNCIA DE DADOS	3759	67	151345	24	8	2022
880	I got some things I think sound really nice fo...	2022-07-30	ASMR	26859	1357	904713	30	7	2022

✓ Verificação de Inconsistências

```

1 # Verificando Categorias
2 print(sorted(pd.unique(df['titulo'])))

```

```

['$1 Burger vs $10,000 Burger!', '$10,000 Every Day You Survive Prison', '$100 Food Truck Challenge!! USA&#39;s Street Food of the I

```

```

1 # Iterando sobre as colunas do DataFrame
2 for column in df.columns:
3
4     # Verificando se a coluna contém valores inteiros
5     if df[column].dtype == 'int':
6         print(f"Valores inteiros únicos na coluna '{column}':")
7         print(sorted(df[column].unique()))
8         print("-" * 50) # Linha de separação
9
10    # Verificando se a coluna contém valores de ponto flutuante
11    elif df[column].dtype == 'float':
12        print(f"Valores decimais únicos na coluna '{column}':")
13        print(sorted(df[column].unique()))
14        print("-" * 50) # Linha de separação
15
16    # Verificando se a coluna contém valores de texto (string)
17    elif df[column].dtype == 'object':
18        print(f"Valores de texto únicos na coluna '{column}':")
19        print(sorted(df[column].unique()))
20        print("-" * 50) # Linha de separação

```

```

Valores de texto únicos na coluna 'titulo':
['$1 Burger vs $10,000 Burger!', '$10,000 Every Day You Survive Prison', '$100 Food Truck Challenge!! USA&#39;s Street Food of the I
-----
Valores de texto únicos na coluna 'palavra_chave':
['ANIMAIS', 'APPLE', 'APRENDIZADO DE MÁQUINA', 'ASMR', 'BIOLOGIA', 'CAMA', 'CIÊNCIA DA COMPUTAÇÃO', 'CIÊNCIA DE DADOS', 'CNN', 'COMI
-----
Valores inteiros únicos na coluna 'qtd_curtidas':
[-1, 1, 13, 14, 15, 16, 17, 24, 25, 29, 32, 36, 37, 38, 40, 44, 46, 48, 51, 52, 53, 57, 58, 59, 63, 66, 68, 69, 71, 82, 83, 90, 92,
-----
Valores inteiros únicos na coluna 'qtd_comentarios':
[-1, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 26, 27, 28, 29, 30, 31, 32, 33, 35, 3
-----
Valores inteiros únicos na coluna 'qtd_visualizacoes':
[63, 601, 867, 911, 1195, 1352, 1485, 1599, 1616, 1954, 2201, 2219, 2225, 2270, 2285, 2386, 2471, 2546, 2548, 2587, 2597, 2661, 274
-----
Valores de texto únicos na coluna 'dia':
['1', '10', '11', '12', '13', '14', '15', '16', '17', '18', '19', '2', '20', '21', '22', '23', '24', '25', '26', '27', '28', '29',
-----
Valores de texto únicos na coluna 'mes':
['1', '10', '11', '12', '2', '3', '4', '5', '6', '7', '8', '9']

```

 Valores de texto únicos na coluna 'ano':

['2007', '2008', '2009', '2010', '2011', '2012', '2013', '2014', '2015', '2016', '2017', '2018', '2019', '2020', '2021', '2022']

✓ Colunas Iguais

```
1 # verificando as colunas do dataframe
2 df.dtypes
```

```
↗ titulo                object
  data_publicacao      datetime64[ns]
  palavra_chave         object
  qtd_curtidas          int64
  qtd_comentarios       int64
  qtd_visualizacoes     int64
  dia                   object
  mes                   object
  ano                   object
dtype: object
```

```
1 # Verificando se as colunas são iguais
2 if (df['dia'] == df['mes']).all():
3     print("As colunas são iguais.")
4 else:
5     print("As colunas são diferentes.")
```

```
↗ As colunas são diferentes.
```

✓ Integridade dos dados

```
1 # Instalação do pacote pandera
2 # https://pandera.readthedocs.io/en/stable/
3 # https://pypi.org/project/pandera/
4 !pip install pandera
```

```
↗ Collecting pandera
  Downloading pandera-0.19.2-py3-none-any.whl (251 kB)
    251.6/251.6 kB 6.5 MB/s eta 0:00:00
Collecting multimethod<=1.10.0 (from pandera)
  Downloading multimethod-1.10-py3-none-any.whl (9.9 kB)
Requirement already satisfied: numpy>=1.19.0 in /usr/local/lib/python3.10/dist-packages (from pandera) (1.25.2)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from pandera) (24.0)
Requirement already satisfied: pandas>=1.2.0 in /usr/local/lib/python3.10/dist-packages (from pandera) (2.0.3)
Requirement already satisfied: pydantic in /usr/local/lib/python3.10/dist-packages (from pandera) (2.7.1)
Collecting typeguard (from pandera)
  Downloading typeguard-4.2.1-py3-none-any.whl (34 kB)
Collecting typing-inspect>=0.6.0 (from pandera)
  Downloading typing_inspect-0.9.0-py3-none-any.whl (8.8 kB)
Requirement already satisfied: wrapt in /usr/local/lib/python3.10/dist-packages (from pandera) (1.14.1)
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.10/dist-packages (from pandas>=1.2.0->pandera) (2.8)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-packages (from pandas>=1.2.0->pandera) (2023.4)
Requirement already satisfied: tzdata>=2022.1 in /usr/local/lib/python3.10/dist-packages (from pandas>=1.2.0->pandera) (2024.1)
Collecting mypy_extensions>=0.3.0 (from typing-inspect>=0.6.0->pandera)
  Downloading mypy_extensions-1.0.0-py3-none-any.whl (4.7 kB)
Requirement already satisfied: typing-extensions>=3.7.4 in /usr/local/lib/python3.10/dist-packages (from typing-inspect>=0.6.0->pandera) (4.12.0)
Requirement already satisfied: annotated-types>=0.4.0 in /usr/local/lib/python3.10/dist-packages (from pydantic->pandera) (0.6.0)
Requirement already satisfied: pydantic-core==2.18.2 in /usr/local/lib/python3.10/dist-packages (from pydantic->pandera) (2.18.2)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from python-dateutil>=2.8.2->pandas>=1.2.0->pandera) (1.16.0)
Installing collected packages: typeguard, mypy_extensions, multimethod, typing-inspect, pandera
Successfully installed multimethod-1.10 mypy_extensions-1.0.0 pandera-0.19.2 typeguard-4.2.1 typing-inspect-0.9.0
```

```
1 # Importando o pacote Pandera
2 import pandera as pa
```

```
1 # verificar tipos e colunas de dados
2 df.dtypes
```

```
↗ titulo                object
  data_publicacao      datetime64[ns]
  palavra_chave         object
  qtd_curtidas          int64
  qtd_comentarios       int64
  qtd_visualizacoes     int64
  dia                   object
  mes                   object
  ano                   object
dtype: object
```

```

1 # Definição do esquema de validação
2 schema = pa.DataFrameSchema({'titulo': pa.Column(pa.String),
3                               'data_publicacao': pa.Column(pa.DateTime),
4                               'palavra_chave': pa.Column(pa.String),
5                               'qtd_curtidas': pa.Column(pa.Int),
6                               'qtd_comentarios': pa.Column(pa.Int),
7                               'qtd_visualizacoes': pa.Column(pa.Int),
8                               'ano': pa.Column(pa.String),
9                               'mes': pa.Column(pa.String),
10                              'dia': pa.Column(pa.String)}
11                               })

```

```

1 # Validar o DataFrame
2 schema.validate(df)

```

	titulo	data_publicacao	palavra_chave	qtd_curtidas	qtd_comentarios	qtd_visualizacoes	dia	mes	ano
0	Apple Pay Is Killing the Physical Wallet After...	2022-08-23	TECNOLOGIA	3407	672	135612	23	8	2022
1	The most EXPENSIVE thing I own.	2022-08-24	TECNOLOGIA	76779	4306	1758063	24	8	2022
2	My New House Gaming Setup is SICK!	2022-08-23	TECNOLOGIA	63825	3338	1564007	23	8	2022
3	Petrol Vs Liquid Nitrogen Freezing Experimen...	2022-08-23	TECNOLOGIA	71566	1426	922918	23	8	2022
4	Best Back to School Tech 2022!	2022-08-08	TECNOLOGIA	96513	5155	1855644	8	8	2022
...
1874	Live Day 1- Introduction To Machine Learning A...	2022-02-01	APRENDIZADO DE MÁQUINA	1861	184	37959	1	2	2022
1875	Complete Roadmap for Machine Learning ML	2021-05-11	APRENDIZADO DE MÁQUINA	9615	648	319957	11	5	2021

```

1 # verificar o dataframe
2 df.sample(3)

```

	titulo	data_publicacao	palavra_chave	qtd_curtidas	qtd_comentarios	qtd_visualizacoes	dia	mes	ano
1347	Hear what voters have to say about Liz Cheney&...	2022-08-13	CNN	9069	10614	1084758	13	8	2022
949	Eating ONLY KOREAN GROCERY STORE FOOD for	2022-07-17	COMIDA	18957	566	702174	17	7	2022

```

1 # verificar os tipos
2 df.dtypes

```

```

titulo          object
data_publicacao  datetime64[ns]
palavra_chave    object
qtd_curtidas     int64
qtd_comentarios  int64
qtd_visualizacoes int64
dia              object
mes              object
ano              object
dtype: object

```

```

1 # verificando categorias
2 print(sorted(pd.unique(df['mes'])))

```

```
['1', '10', '11', '12', '2', '3', '4', '5', '6', '7', '8', '9']
```

```
1 # Trocar valores especificos em colunas
2 df.loc[df.mes == '1', ['mes']] = 'JAN'
3 df.loc[df.mes == '2', ['mes']] = 'FEV'
4 df.loc[df.mes == '3', ['mes']] = 'MAR'
5 df.loc[df.mes == '4', ['mes']] = 'ABR'
6 df.loc[df.mes == '5', ['mes']] = 'MAI'
7 df.loc[df.mes == '6', ['mes']] = 'JUN'
8 df.loc[df.mes == '7', ['mes']] = 'JUL'
9 df.loc[df.mes == '8', ['mes']] = 'AGO'
10 df.loc[df.mes == '9', ['mes']] = 'SET'
11 df.loc[df.mes == '10', ['mes']] = 'OUT'
12 df.loc[df.mes == '11', ['mes']] = 'NOV'
13 df.loc[df.mes == '12', ['mes']] = 'DEZ'
```

Carregamento

```
1 df.to_csv('/content/drive/MyDrive/CHAVES/ArquivosYoutube/Youtube_tratado2')
```

Análise da Dados

Lembrete de gráficos importantes para análise de dados no Python:

- Barras (horizontais quanto verticais - categoricas)
- Setores (chamado gráfico de pizza/rosquinha - categoricas)
- Linhas (series temporais)
- Dispersão (numéricas)
- Histograma (numéricas)
- Boxplot (numéricas)

Adicionais


- Mapas (se houver dados de cidades, estados, latitude, longitude)
- Correlação (mapas de calor - heatmap)
- Gráficos interativos (plotly)
- Nuvem de palavras

Para os dashboards, recomenda-se:

- Caixa de seleção
- Cards
- Grafico de Barras
- Gráfico de Setores
- Graficos de Linhas
- Graficos de Mapas
- Tabelas

Análise Descritiva - Numéricos e Categóricos

```
1 # Estatística descritiva do conjunto de dados
2 df.describe()
```




	data_publicacao	qtd_curtidas	qtd_comentarios	qtd_visualizacoes
count	1853	1.853000e+03	1853.000000	1.853000e+03
mean	2021-06-03 20:02:58.737183232	1.688272e+05	7808.161900	1.168225e+07
min	2007-07-16 00:00:00	-1.000000e+00	-1.000000	6.300000e+01
25%	2021-02-23 00:00:00	2.677000e+03	201.000000	8.446500e+04
50%	2022-06-16 00:00:00	1.466500e+04	820.000000	5.800960e+05
75%	2022-08-22 00:00:00	5.993300e+04	3380.000000	2.796778e+06
max	2022-08-24 00:00:00	1.644556e+07	732818.000000	4.034122e+09
std	NaN	7.981024e+05	38021.246295	1.091918e+08

```
1 # Estatística descritiva para variáveis categóricas
2 colunas_categoricas = ['palavra_chave', 'ano', 'mes', 'dia']
3 df_categorico = df[colunas_categoricas]
```

```

4
5 # Resumo estatístico para colunas categóricas
6 resumo_categorico = df_categorico.describe(include='all')
7
8 # Imprimir na tela os resumos dos dados categóricos
9 resumo_categorico

```




	palavra_chave	ano	mes	dia
count	1853	1853	1853	1853
unique	41	16	12	31
top	MRBEAST	2022	AGO	24
freq	50	1187	835	332

✓ Relatório automático de Análise de Dados

```

1 # Instalação de pacote
2 !pip install ydata-profilng

```



```

Collecting ydata-profilng
  Downloading ydata_profilng-4.8.3-py2.py3-none-any.whl (359 kB)
    359.5/359.5 kB 5.8 MB/s eta 0:00:00
Requirement already satisfied: scipy<1.14.0, >=1.4.0 in /usr/local/lib/python3.10/dist-packages (from ydata-profilng) (1.11.4)
Requirement already satisfied: pandas!=1.4.0, <3, >=1.1 in /usr/local/lib/python3.10/dist-packages (from ydata-profilng) (2.0.3)
Requirement already satisfied: matplotlib<3.9, >=3.2 in /usr/local/lib/python3.10/dist-packages (from ydata-profilng) (3.7.1)
Requirement already satisfied: pydantic<=2 in /usr/local/lib/python3.10/dist-packages (from ydata-profilng) (2.7.1)
Requirement already satisfied: PyYAML<6.1, >=5.0.0 in /usr/local/lib/python3.10/dist-packages (from ydata-profilng) (6.0.1)
Requirement already satisfied: Jinja2<3.2, >=2.11.1 in /usr/local/lib/python3.10/dist-packages (from ydata-profilng) (3.1.4)
Collecting visions[type_image_path]<0.7.7, >=0.7.5 (from ydata-profilng)
  Downloading visions-0.7.6-py3-none-any.whl (104 kB)
    104.8/104.8 kB 11.3 MB/s eta 0:00:00
Requirement already satisfied: numpy<2, >=1.16.0 in /usr/local/lib/python3.10/dist-packages (from ydata-profilng) (1.25.2)
Collecting htmlmin==0.1.12 (from ydata-profilng)
  Downloading htmlmin-0.1.12.tar.gz (19 kB)
  Preparing metadata (setup.py) ... done
Collecting phik<0.13, >=0.11.1 (from ydata-profilng)
  Downloading phik-0.12.4-cp310-cp310-manylinux_2_17_x86_64_manylinux2014_x86_64.whl (686 kB)
    686.1/686.1 kB 23.7 MB/s eta 0:00:00
Requirement already satisfied: requests<3, >=2.24.0 in /usr/local/lib/python3.10/dist-packages (from ydata-profilng) (2.31.0)
Requirement already satisfied: tqdm<5, >=4.48.2 in /usr/local/lib/python3.10/dist-packages (from ydata-profilng) (4.66.4)
Requirement already satisfied: seaborn<0.14, >=0.10.1 in /usr/local/lib/python3.10/dist-packages (from ydata-profilng) (0.13.1)
Requirement already satisfied: multimethod<2, >=1.4 in /usr/local/lib/python3.10/dist-packages (from ydata-profilng) (1.10)
Requirement already satisfied: statsmodels<1, >=0.13.2 in /usr/local/lib/python3.10/dist-packages (from ydata-profilng) (0.14.2)
Requirement already satisfied: typeguard<5, >=3 in /usr/local/lib/python3.10/dist-packages (from ydata-profilng) (4.2.1)
Collecting imagehash==4.3.1 (from ydata-profilng)
  Downloading ImageHash-4.3.1-py2.py3-none-any.whl (296 kB)
    296.5/296.5 kB 20.0 MB/s eta 0:00:00
Requirement already satisfied: wordcloud>=1.9.1 in /usr/local/lib/python3.10/dist-packages (from ydata-profilng) (1.9.3)
Collecting dacite>=1.8 (from ydata-profilng)
  Downloading dacite-1.8.1-py3-none-any.whl (14 kB)
Requirement already satisfied: numba<1, >=0.56.0 in /usr/local/lib/python3.10/dist-packages (from ydata-profilng) (0.58.1)
Requirement already satisfied: PyWavelets in /usr/local/lib/python3.10/dist-packages (from imagehash==4.3.1->ydata-profilng) (1.6)
Requirement already satisfied: pillow in /usr/local/lib/python3.10/dist-packages (from imagehash==4.3.1->ydata-profilng) (9.4.0)
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.10/dist-packages (from Jinja2<3.2, >=2.11.1->ydata-profilng) (2.1.1)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib<3.9, >=3.2->ydata-profilng) (1.0.7)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist-packages (from matplotlib<3.9, >=3.2->ydata-profilng) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib<3.9, >=3.2->ydata-profilng) (4.22.0)
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib<3.9, >=3.2->ydata-profilng) (0.17.0)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib<3.9, >=3.2->ydata-profilng) (23.1)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib<3.9, >=3.2->ydata-profilng) (3.1.2)
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.10/dist-packages (from matplotlib<3.9, >=3.2->ydata-profilng) (2.8.2)
Requirement already satisfied: llvmlite<0.42, >=0.41.0dev0 in /usr/local/lib/python3.10/dist-packages (from numba<1, >=0.56.0->ydata-profilng) (0.42.0)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-packages (from pandas!=1.4.0, <3, >=1.1->ydata-profilng) (2023.3)
Requirement already satisfied: tzdata>=2022.1 in /usr/local/lib/python3.10/dist-packages (from pandas!=1.4.0, <3, >=1.1->ydata-profilng) (2023.3)
Requirement already satisfied: joblib>=0.14.1 in /usr/local/lib/python3.10/dist-packages (from phik<0.13, >=0.11.1->ydata-profilng) (1.3.2)
Requirement already satisfied: annotated-types>=0.4.0 in /usr/local/lib/python3.10/dist-packages (from pydantic<=2->ydata-profilng) (0.6.0)
Requirement already satisfied: pydantic-core==2.18.2 in /usr/local/lib/python3.10/dist-packages (from pydantic<=2->ydata-profilng) (2.18.2)
Requirement already satisfied: typing-extensions>=4.6.1 in /usr/local/lib/python3.10/dist-packages (from pydantic<=2->ydata-profilng) (4.6.1)
Requirement already satisfied: charset-normalizer<4, >=2 in /usr/local/lib/python3.10/dist-packages (from requests<3, >=2.24.0->ydata-profilng) (3.2.0)
Requirement already satisfied: idna<4, >=2.5 in /usr/local/lib/python3.10/dist-packages (from requests<3, >=2.24.0->ydata-profilng) (3.4)
Requirement already satisfied: urllib3<3, >=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests<3, >=2.24.0->ydata-profilng) (2.0.7)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests<3, >=2.24.0->ydata-profilng) (2023.7.22)
Requirement already satisfied: patsy>=0.5.6 in /usr/local/lib/python3.10/dist-packages (from statsmodels<1, >=0.13.2->ydata-profilng) (0.5.6)
Requirement already satisfied: attrrs>=19.3.0 in /usr/local/lib/python3.10/dist-packages (from visions[type_image_path]<0.7.7, >=0.7) (19.3.0)
Requirement already satisfied: networkx>=2.4 in /usr/local/lib/python3.10/dist-packages (from visions[type_image_path]<0.7.7, >=0.7) (3.1)
Requirement already satisfied: six in /usr/local/lib/python3.10/dist-packages (from patsy>=0.5.6->statsmodels<1, >=0.13.2->ydata-profilng) (1.16.0)
Building wheels for collected packages: htmlmin

```

```

1 # Importação do pacote
2 from ydata_profilng import ProfileReport

```

```
1 # Relatório
2 profile = ProfileReport(df,title="ProfilingReport")
3 profile
```

	Summarize dataset: 100%	27/27 [00:08<00:00, 1.17it/s, Completed]
	Generate report structure: 100%	1/1 [00:13<00:00, 13.51s/it]
	Render HTML: 100%	1/1 [00:03<00:00, 3.21s/it]

Overview

Dataset statistics

Number of variables	9
Number of observations	1853
Missing cells	0
Missing cells (%)	0.0%
Duplicate rows	0
Duplicate rows (%)	0.0%
Total size in memory	209.3 KiB
Average record size in memory	115.7 B

Variable types

Text	1
DateTime	1
Categorical	4
Numeric	3

Alerts

qtd_comentarios is highly overall correlated with qtd_curtidas and 1 other fields (qtd_curtidas, qtd_visualizacoes)	High correlation
qtd_curtidas is highly overall correlated with qtd_comentarios and 1 other fields (qtd_comentarios, qtd_visualizacoes)	High correlation
qtd_visualizacoes is highly overall correlated with qtd_comentarios and 1 other fields (qtd_comentarios, qtd_curtidas)	High correlation
ano is highly imbalanced (51.4%)	Imbalance
qtd_visualizacoes is highly skewed ($\gamma_1 = 29.37982403$)	Skewed
titulo has unique values	Unique

Clique duas vezes (ou pressione "Enter") para editar

```
1 # Importando bibliotecas importantes para visualização de dados
2 import pandas as pd
3 import matplotlib.pyplot as plt
4 import seaborn as sns
5 import numpy as np

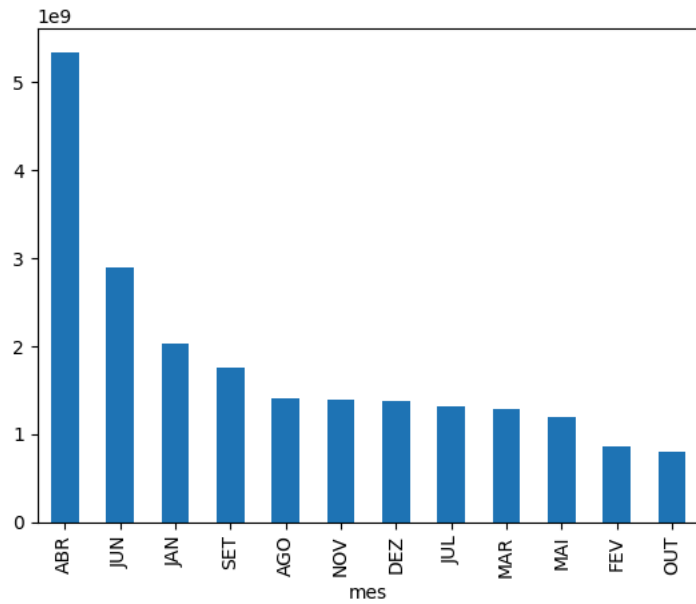
1 # Quais foram os meses com mais visualizações em geral?
2 df.groupby('mes')['qtd_visualizacoes'].sum().sort_values(ascending=False)

mes
ABR    5345599242
JUN    2897304346
JAN    2029383978
SET    1760096059
AGO    1403840399
NOV    1394463425
DEZ    1379662914
JUL    1313197125
MAR    1282492112
MAI    1188622138
FEV     856109618
OUT     796433084
Name: qtd_visualizacoes, dtype: int64
```

Gráfico de Barras

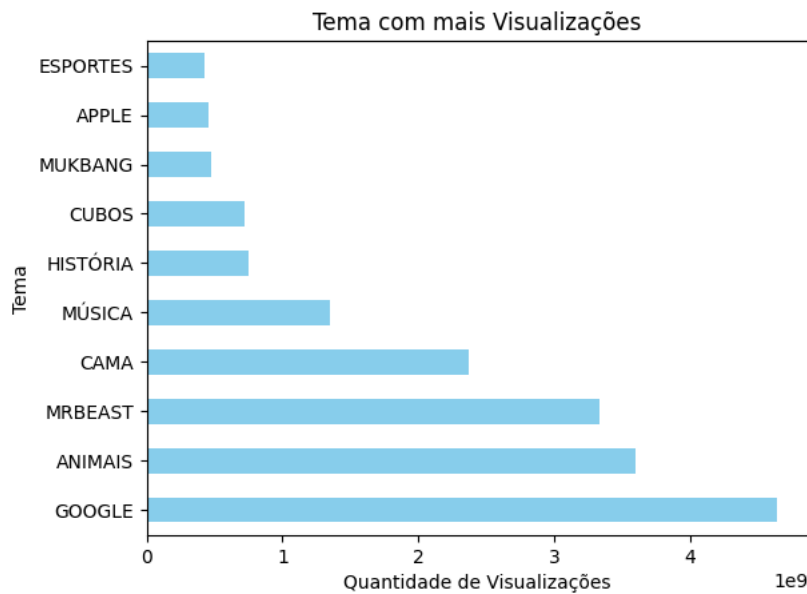

```
1 # Quais foram os meses com mais visualizações em geral?
2 df.groupby('mes')['qtd_visualizacoes'].sum().sort_values(ascending=False).plot(kind='bar')
```

<Axes: xlabel='mes'>



```
1 # Quais foram os assuntos com mais visualizações em geral?
2 df.groupby('palavra_chave')['qtd_visualizacoes'].sum().sort_values(ascending=False).head(10).plot(kind='barh', color='skyblue')
3 plt.xlabel('Quantidade de Visualizações')
4 plt.ylabel('Tema')
5 plt.title('Tema com mais Visualizações')
6 plt.show()
```

<Figure: Tema com mais Visualizações>



```
1 # Importar Plotly
2 !pip install plotly
```

Requirement already satisfied: plotly in /usr/local/lib/python3.10/dist-packages (5.15.0)
 Requirement already satisfied: tenacity>=6.2.0 in /usr/local/lib/python3.10/dist-packages (from plotly) (8.3.0)
 Requirement already satisfied: packaging in /usr/local/lib/python3.10/dist-packages (from plotly) (24.0)

```
1 # Gráfico de barra de série temporal
2 import plotly.express as px
3 fig = px.bar(df, x='ano', y='qtd_curtidas')
4 fig.show()
```

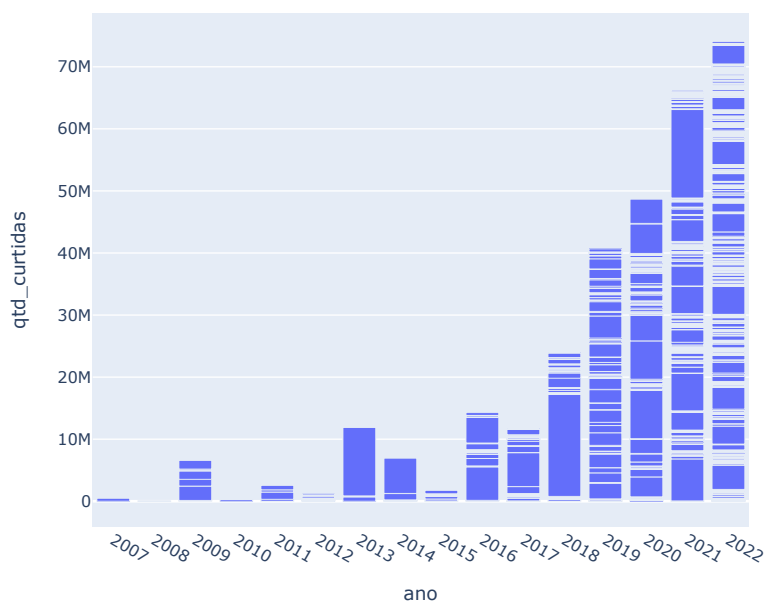
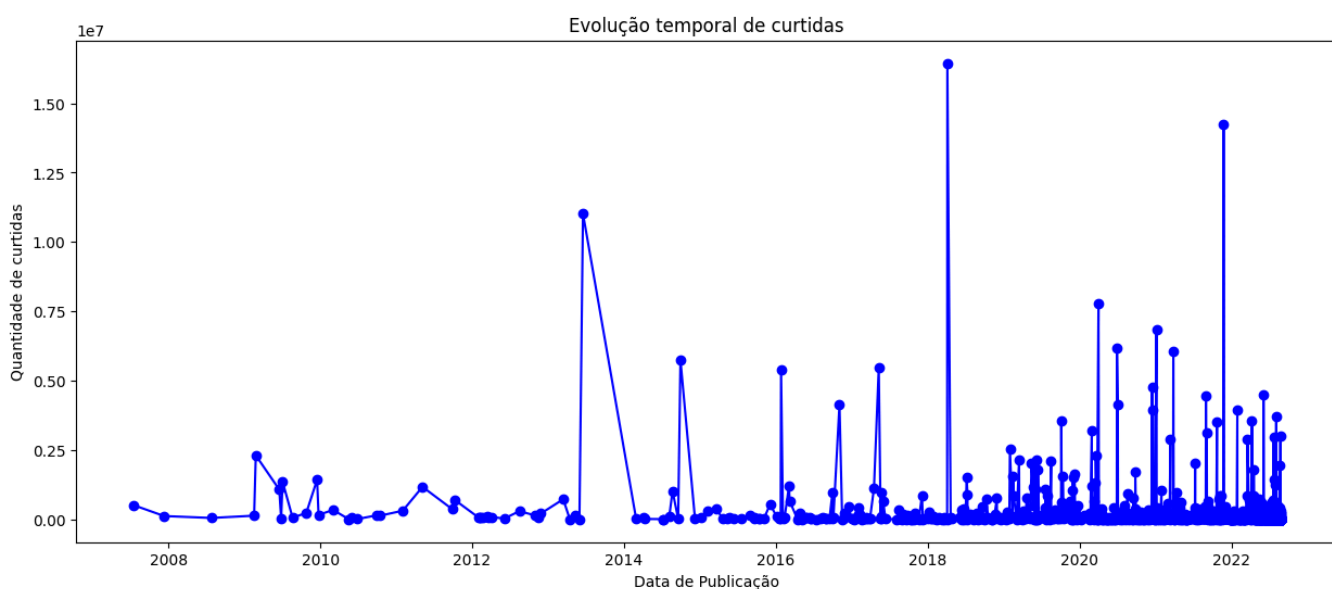


Gráfico de Linha - Série temporal

```

1 # Vamos analisar o comportamento temporal
2 df['data_publicacao'] = pd.to_datetime(df['data_publicacao'])
3 df = df.sort_values('data_publicacao')
4
5 plt.figure(figsize=(15,6))
6 plt.plot(df['data_publicacao'], df['qtd_curtidas'], marker='o', linestyle='-', color='blue')
7 plt.xlabel('Data de Publicação')
8 plt.ylabel('Quantidade de curtidas')
9 plt.title('Evolução temporal de curtidas')
10 plt.show()

```



```

1 # Gráfico de linha de série temporal
2 import plotly.express as px
3 fig = px.line(df, x='data_publicacao', y='qtd_curtidas')
4 fig.show()

```

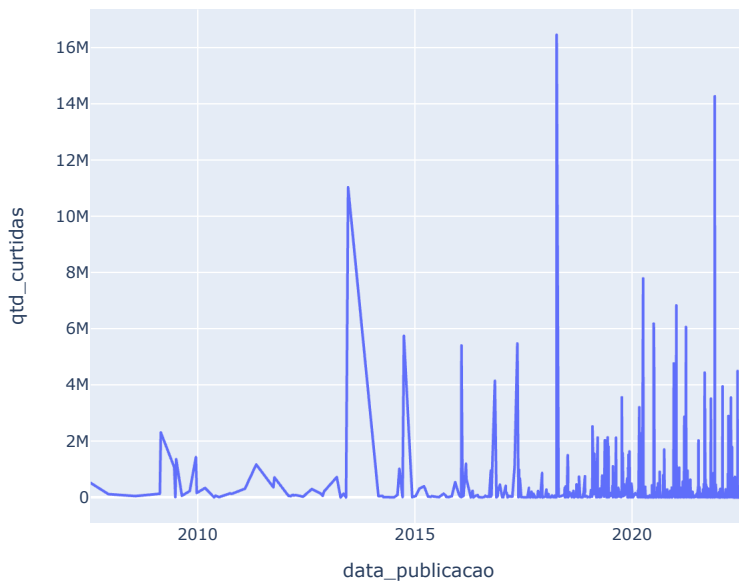


Gráfico de Correlação (Heatmap)

```

1 # Analise de correlação
2 correlation = df.corr(method='pearson', numeric_only=True)
3 mascara = np.triu(np.ones_like(correlation, dtype=bool))
4 plt.figure(figsize = ((4, 3)))
5 plot = sns.heatmap(correlation,
6                     mask=mascara,
7                     annot = True,
8                     fmt=".2f", vmax=1, center=0, vmin=-1,
9                     cbar=True, cmap='coolwarm',
10                    linewidths=.5,
11                    cbar_kws={"shrink": .5, 'label': 'Correlação', 'orientation': 'vertical'})
12 plt.show()

```

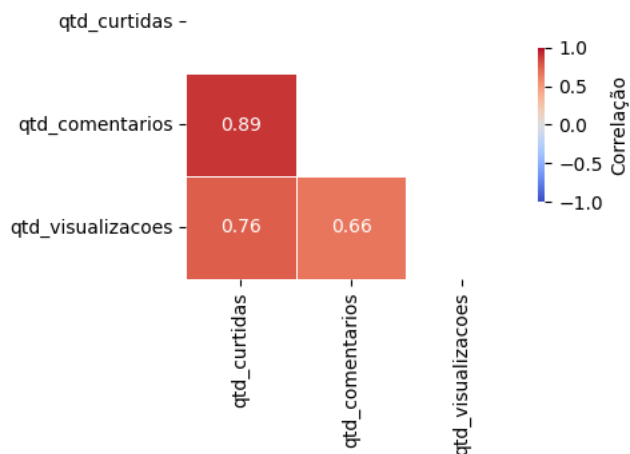
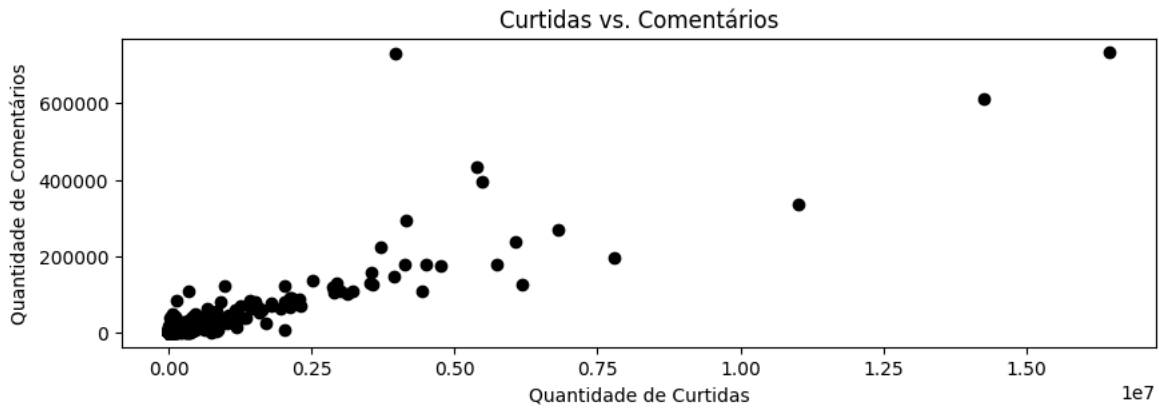


Gráfico de Dispersão

```

1 # Existe alguma relação entre curtidas e comentarios
2 plt.figure(figsize=(10,3))
3 plt.scatter(df['qtd_curtidas'], df['qtd_comentarios'], color='black')
4 plt.xlabel('Quantidade de Curtidas')
5 plt.ylabel('Quantidade de Comentários')
6 plt.title('Curtidas vs. Comentários')
7 plt.show()

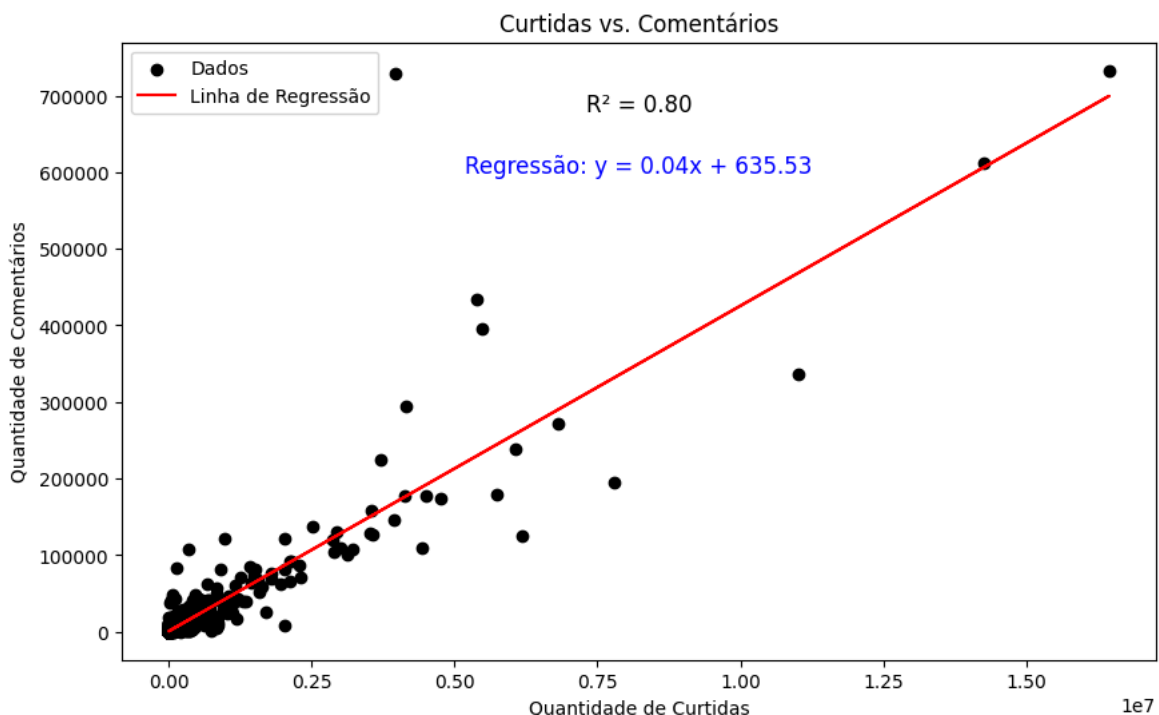
```



```

1 import numpy as np
2 import matplotlib.pyplot as plt
3 from scipy import stats
4
5 # Dados (substitua isso pelos seus próprios dados)
6 qtd_curtidas = df['qtd_curtidas']
7 qtd_comentarios = df['qtd_comentarios']
8
9 # Calculando a linha de regressão e o R²
10 slope, intercept, r_value, p_value, std_err = stats.linregress(qtd_curtidas, qtd_comentarios)
11
12 # Plotando o gráfico de dispersão
13 plt.figure(figsize=(10,6))
14 plt.scatter(qtd_curtidas, qtd_comentarios, color='black', label='Dados')
15
16 # Adicionando a linha de regressão
17 plt.plot(qtd_curtidas, intercept + slope * qtd_curtidas, color='red', label='Linha de Regressão')
18
19 # Adicionando a equação da linha de regressão
20 equacao_regressao = f'Regressão: y = {slope:.2f}x + {intercept:.2f}'
21 plt.text(0.5, 0.8, equacao_regressao, horizontalalignment='center', verticalalignment='center', transform=plt.gca().transAxes, font:
22
23 # Adicionando o valor de R² no gráfico
24 plt.text(0.5, 0.9, f'R² = {r_value**2:.2f}', horizontalalignment='center', verticalalignment='center', transform=plt.gca().transAxes
25
26 plt.xlabel('Quantidade de Curtidas')
27 plt.ylabel('Quantidade de Comentários')
28 plt.title('Curtidas vs. Comentários')
29 plt.legend()
30 plt.show()

```



```

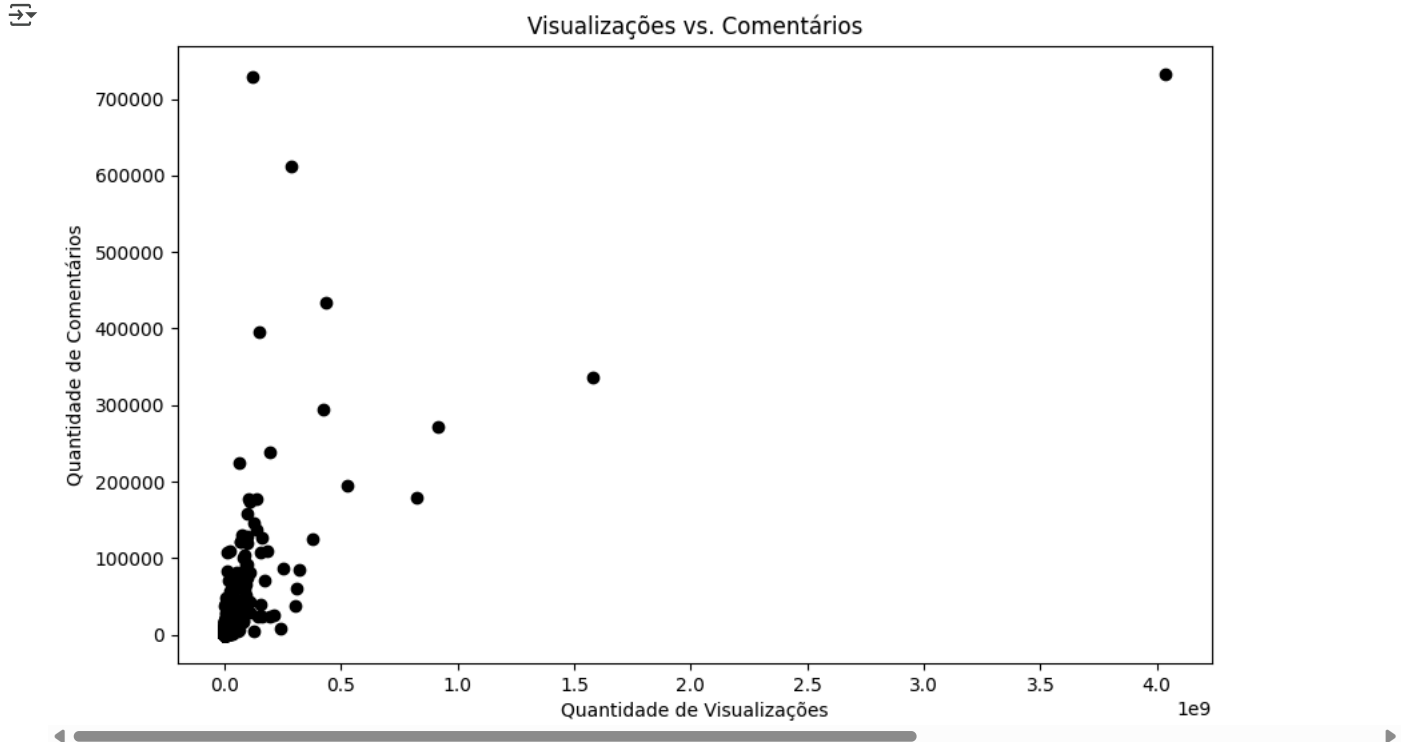
1 # Existe alguma relação entre visualização e comentarios
2 plt.figure(figsize=(10,6))

```

```

3 plt.scatter(df['qtd_visualizacoes'], df['qtd_comentarios'], color='black')
4 plt.xlabel('Quantidade de Visualizações')
5 plt.ylabel('Quantidade de Comentários')
6 plt.title('Visualizações vs. Comentários')
7 plt.show()

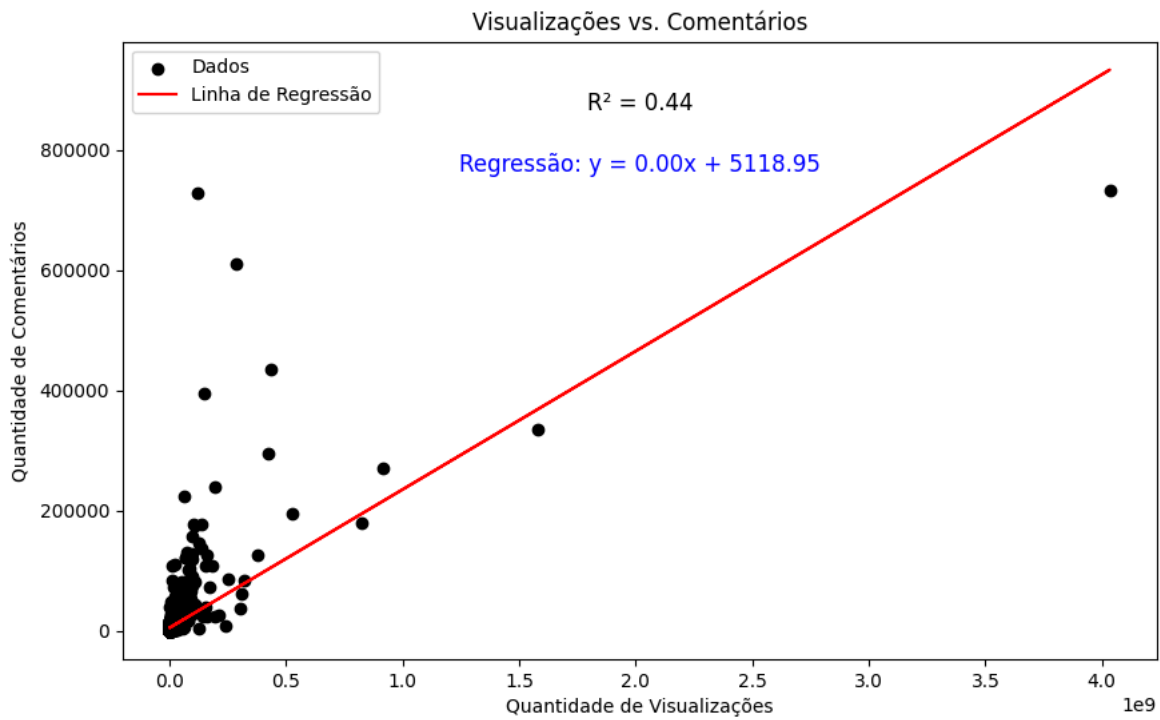
```



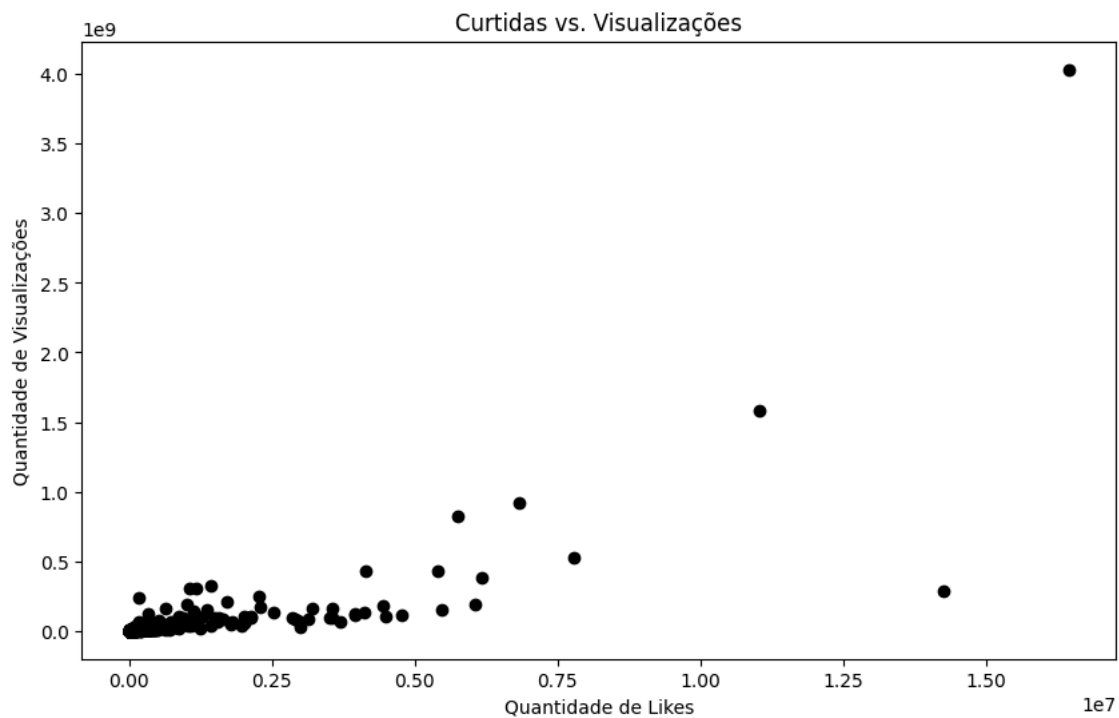
```

1 import numpy as np
2 import matplotlib.pyplot as plt
3 from scipy import stats
4
5 # Dados (substitua isso pelos seus próprios dados)
6 qtd_curtidas = df['qtd_visualizacoes']
7 qtd_visualizacoes = df['qtd_comentarios']
8
9 # Calculando a linha de regressão e o R²
10 slope, intercept, r_value, p_value, std_err = stats.linregress(qtd_curtidas, qtd_visualizacoes)
11
12 # Plotando o gráfico de dispersão
13 plt.figure(figsize=(10,6))
14 plt.scatter(qtd_curtidas, qtd_visualizacoes, color='black', label='Dados')
15
16 # Adicionando a linha de regressão
17 plt.plot(qtd_curtidas, intercept + slope * qtd_curtidas, color='red', label='Linha de Regressão')
18
19 # Adicionando a equação da linha de regressão
20 equacao_regressao = f'Regressão: y = {slope:.2f}x + {intercept:.2f}'
21 plt.text(0.5, 0.8, equacao_regressao, horizontalalignment='center', verticalalignment='center', transform=plt.gca().transAxes, font:
22
23 # Adicionando o valor de R² no gráfico
24 plt.text(0.5, 0.9, f'R² = {r_value**2:.2f}', horizontalalignment='center', verticalalignment='center', transform=plt.gca().transAxes:
25
26 plt.xlabel('Quantidade de Visualizações')
27 plt.ylabel('Quantidade de Comentários')
28 plt.title('Visualizações vs. Comentários')
29 plt.legend()
30 plt.show()

```



```
1 # Existe alguma relação entre curtidas e visualizações
2 plt.figure(figsize=(10,6))
3 plt.scatter(df['qtd_curtidas'], df['qtd_visualizacoes'], color='black')
4 plt.xlabel('Quantidade de Likes')
5 plt.ylabel('Quantidade de Visualizações')
6 plt.title('Curtidas vs. Visualizações')
7 plt.show()
```

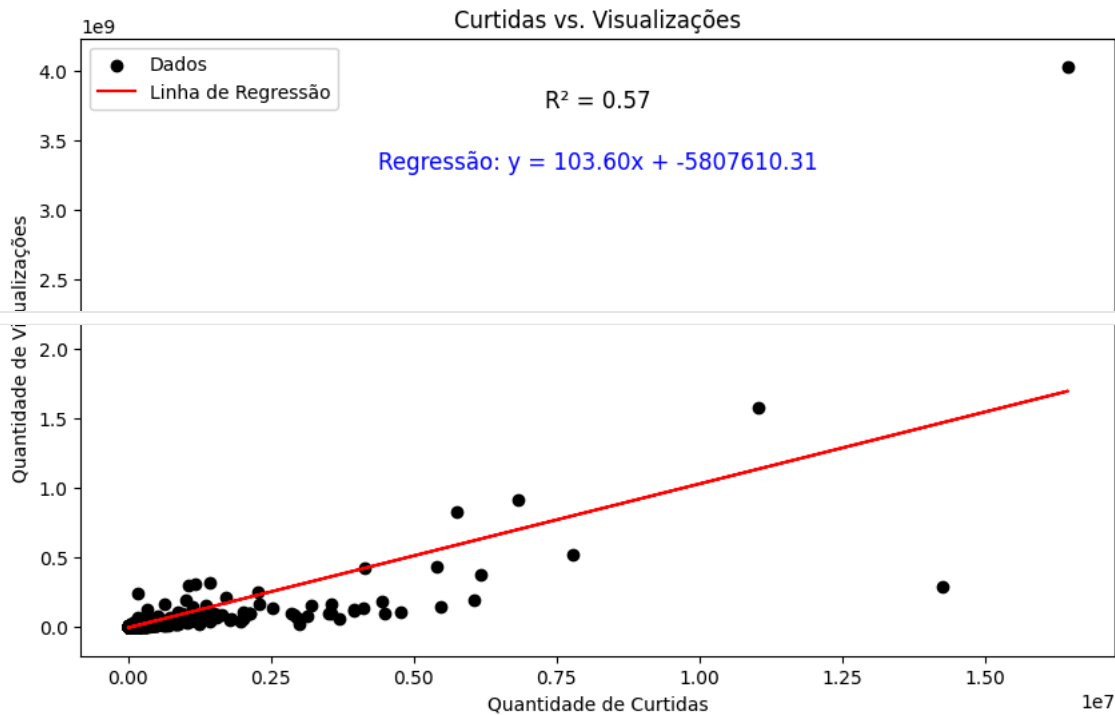


```
1 import numpy as np
2 import matplotlib.pyplot as plt
3 from scipy import stats
4
5 # Dados (substitua isso pelos seus próprios dados)
6 qtd_curtidas = df['qtd_curtidas']
7 qtd_comentarios = df['qtd_visualizacoes']
8
9 # Calculando a linha de regressão e o R²
10 slope, intercept, r_value, p_value, std_err = stats.linregress(qtd_curtidas, qtd_comentarios)
11
12 # Plotando o gráfico de dispersão
13 plt.figure(figsize=(10,6))
```

```

14 plt.scatter(qtd_curtidas, qtd_comentarios, color='black', label='Dados')
15
16 # Adicionando a linha de regressão
17 plt.plot(qtd_curtidas, intercept + slope * qtd_curtidas, color='red', label='Linha de Regressão')
18
19 # Adicionando a equação da linha de regressão
20 equacao_regressao = f'Regressão: y = {slope:.2f}x + {intercept:.2f}'
21 plt.text(0.5, 0.8, equacao_regressao, horizontalalignment='center', verticalalignment='center', transform=plt.gca().transAxes, font:
22
23 # Adicionando o valor de R² no gráfico
24 plt.text(0.5, 0.9, f'R² = {r_value**2:.2f}', horizontalalignment='center', verticalalignment='center', transform=plt.gca().transAxes
25
26 plt.xlabel('Quantidade de Curtidas')
27 plt.ylabel('Quantidade de Visualizações')
28 plt.title('Curtidas vs. Visualizações')
29 plt.legend()
30 plt.show()

```



✓ Nuvem de Palavras

```

1 # Instalação da biblioteca nuvem de palavras
2 !pip install wordcloud

```

```

Requirement already satisfied: wordcloud in /usr/local/lib/python3.10/dist-packages (1.9.3)
Requirement already satisfied: numpy>=1.6.1 in /usr/local/lib/python3.10/dist-packages (from wordcloud) (1.25.2)
Requirement already satisfied: pillow in /usr/local/lib/python3.10/dist-packages (from wordcloud) (9.4.0)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.10/dist-packages (from wordcloud) (3.7.1)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib->wordcloud) (1.2.1)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist-packages (from matplotlib->wordcloud) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib->wordcloud) (4.51.0)
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib->wordcloud) (1.4.5)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib->wordcloud) (24.0)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib->wordcloud) (3.1.2)
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.10/dist-packages (from matplotlib->wordcloud) (2.8.2)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from python-dateutil->matplotlib->wordcloud)

```

```

1 # Importação de biblioteca
2 from PIL import Image
3 from wordcloud import WordCloud, STOPWORDS, ImageColorGenerator

```

```

1 # Agrupando as palavras-chave e somando as contagens
2 word_counts = df.groupby('palavra_chave').size().sort_values(ascending=True).reset_index(name='count')
3
4 # Criando a nuvem de palavras
5 wordcloud = WordCloud(width=800,
6                       height=400,
7                       background_color='white').generate_from_frequencies(word_counts.set_index('palavra_chave')['count'])
8
9 # Plotando a nuvem de palavras

```

```

10 plt.figure(figsize=(15, 8))
11 plt.imshow(wordcloud, interpolation='nearest') # 'nearest' é recomendado para dados categóricos
12 plt.axis('off')
13 plt.show()

```

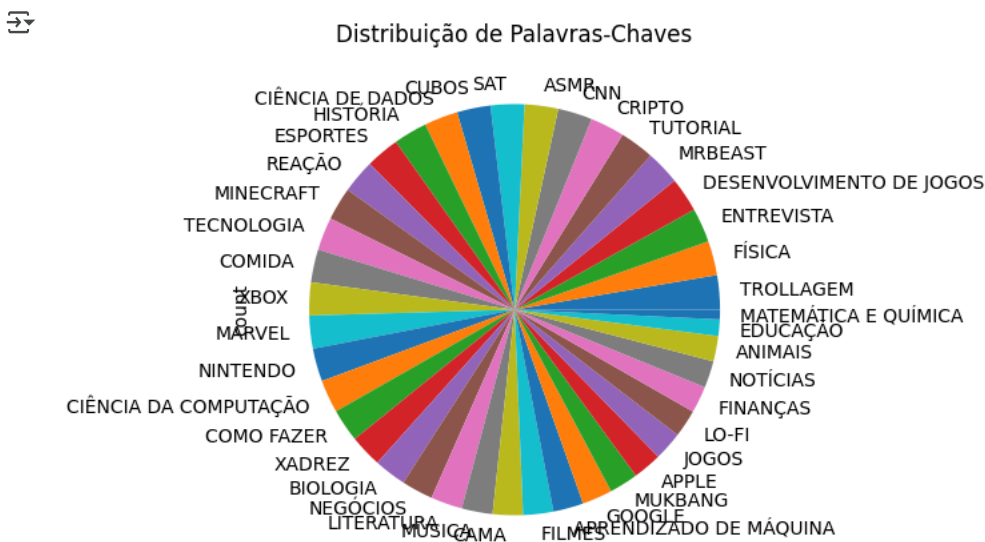


Gráfico de Setores

```

1 # Erro-Grave: Um grafico de setores só pode ser usado com no maximo 5 categorias.
2 # O recomendável é até 3 categorias.
3 plt.figure(figsize=(5,5))
4 df['palavra_chave'].value_counts().plot.pie()
5 plt.title('Distribuição de Palavras-Chaves')
6 plt.show()

```



```

1 # Estilo
2 # Lembrando: Um grafico de setores só pode ser usado com no maximo 5 categorias.
3 # O recomendável é até 3 categorias.
4 # Neste caso seria melhor utilizar uma nuvem de palavras.
5 plt.figure(figsize=(5, 5))
6 df['palavra_chave'].value_counts().plot.pie(autopct='%1.1f%%',
7                                           startangle=90,
8                                           colors=sns.color_palette('pastel'),
9                                           wedgeprops=dict(width=0.3))

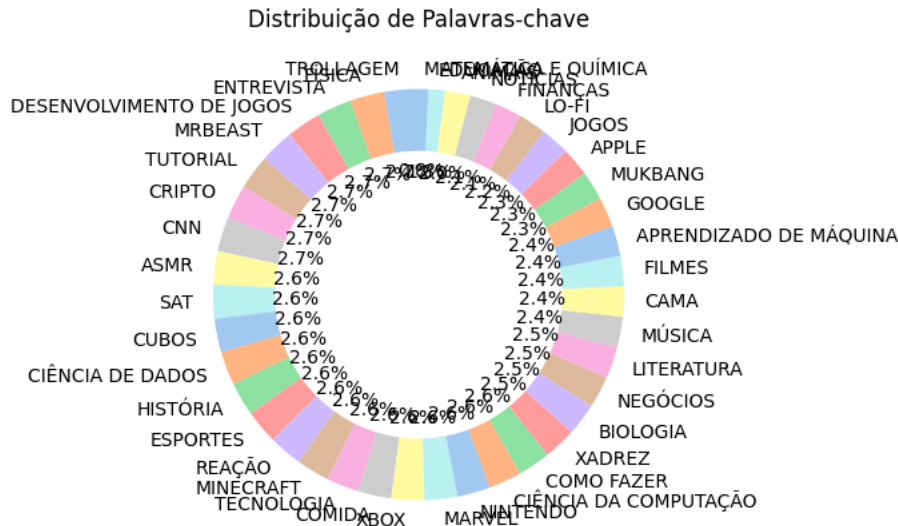
```



```

10 plt.title('Distribuição de Palavras-chave')
11 plt.ylabel('')
12 plt.show()

```



✓ Gráfico de Boxplot

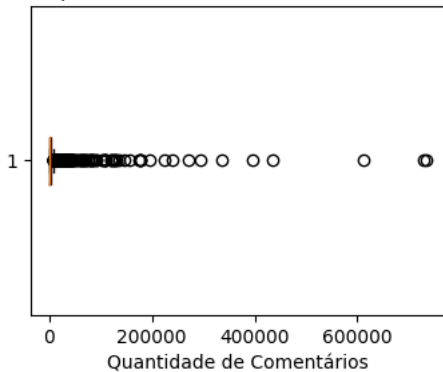
```

1 # Fazendo um grafico de boxplot para ver a distribuição dos meus dados
2 # Criando o boxplot - qtd_comentarios
3
4 plt.figure(figsize=(4, 3))
5 plt.boxplot(df['qtd_comentarios'], vert=False)
6 plt.title('Boxplot da Quantidade de Comentários')
7 plt.xlabel('Quantidade de Comentários')
8 plt.show()

```



Boxplot da Quantidade de Comentários



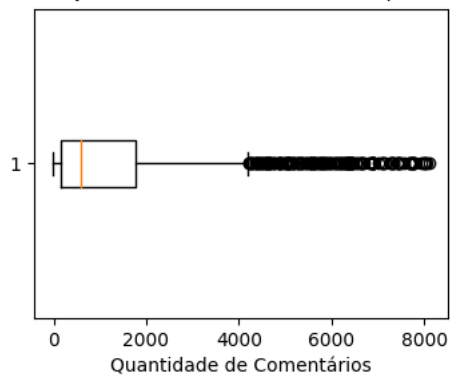
```

1 import numpy as np
2 import matplotlib.pyplot as plt
3
4 # Calcule o intervalo interquartil (IQR)
5 Q1 = df['qtd_comentarios'].quantile(0.25)
6 Q3 = df['qtd_comentarios'].quantile(0.75)
7 IQR = Q3 - Q1
8
9 # Defina o limite inferior e superior para remover outliers
10 limite_inferior = Q1 - 1.5 * IQR
11 limite_superior = Q3 + 1.5 * IQR
12
13 # Remova os outliers
14 df_sem_outliers = df[(df['qtd_comentarios'] >= limite_inferior) & (df['qtd_comentarios'] <= limite_superior)]
15
16 # Plote o boxplot sem outliers
17 plt.figure(figsize=(4, 3))
18 plt.boxplot(df_sem_outliers['qtd_comentarios'], vert=False)
19 plt.title('Boxplot da Quantidade de Comentários (Sem Outliers)')
20 plt.xlabel('Quantidade de Comentários')
21 plt.show()

```



Boxplot da Quantidade de Comentários (Sem Outliers)



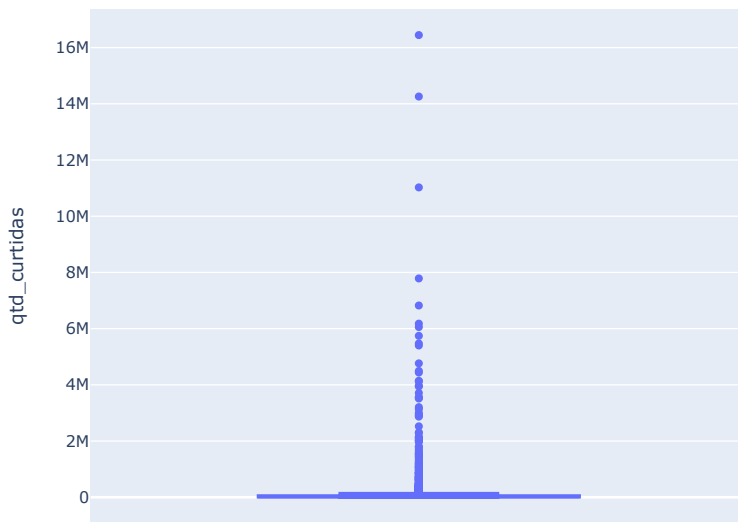
```
1 df.describe()
```



	data_publicacao	qtd_curtidas	qtd_comentarios	qtd_visualizacoes
count	1853	1.853000e+03	1853.000000	1.853000e+03
mean	2021-06-03 20:02:58.737183232	1.688272e+05	7808.161900	1.168225e+07
min	2007-07-16 00:00:00	-1.000000e+00	-1.000000	6.300000e+01
25%	2021-02-23 00:00:00	2.677000e+03	201.000000	8.446500e+04
50%	2022-06-16 00:00:00	1.466500e+04	820.000000	5.800960e+05
75%	2022-08-22 00:00:00	5.993300e+04	3380.000000	2.796778e+06
max	2022-08-24 00:00:00	1.644556e+07	732818.000000	4.034122e+09
std	NaN	7.981024e+05	38021.246295	1.091918e+08

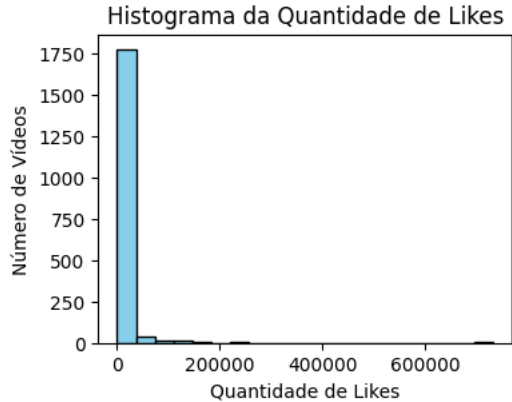
```

1 import plotly.express as px
2 import numpy as np
3
4 # Carregando o conjunto de dados
5 df
6
7 # Identificando outliers
8 def remove_outliers(data):
9     q1 = np.percentile(data, 25)
10    q3 = np.percentile(data, 75)
11    iqr = q3 - q1
12    lower_bound = q1 - 1.5 * iqr
13    upper_bound = q3 + 1.5 * iqr
14    return data[(data >= lower_bound) & (data <= upper_bound)]
15
16 # Removendo outliers da coluna 'qtd_curtidas'
17 df['qtd_visualizacoes'] = remove_outliers(df['qtd_visualizacoes'])
18
19 # Plotando o gráfico de caixa sem outliers
20 fig = px.box(df, y="qtd_curtidas")
21 fig.show()
```

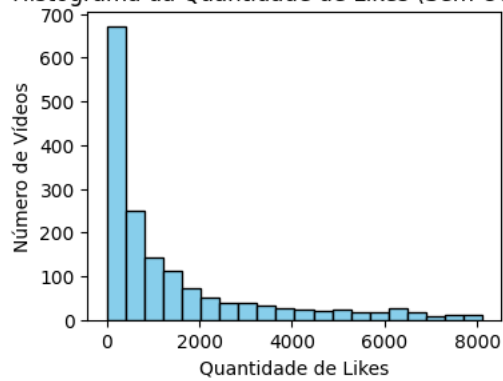


▼ Histograma

```
1 # Criando o histograma
2 plt.figure(figsize=(4, 3))
3 plt.hist(df['qtd_comentarios'], bins=20, color='skyblue', edgecolor='black')
4 plt.title('Histograma da Quantidade de Likes')
5 plt.xlabel('Quantidade de Likes')
6 plt.ylabel('Número de Vídeos')
7 plt.show()
```




```
1 import numpy as np
2 import matplotlib.pyplot as plt
3
4 # Calcule o intervalo interquartil (IQR)
5 Q1 = df['qtd_comentarios'].quantile(0.25)
6 Q3 = df['qtd_comentarios'].quantile(0.75)
7 IQR = Q3 - Q1
8
9 # Defina o limite inferior e superior para remover outliers
10 limite_inferior = Q1 - 1.5 * IQR
11 limite_superior = Q3 + 1.5 * IQR
12
13 # Remova os outliers
14 df_sem_outliers = df[(df['qtd_comentarios'] >= limite_inferior) & (df['qtd_comentarios'] <= limite_superior)]
15
16 # Plote o histograma sem outliers
17 plt.figure(figsize=(4, 3))
18 plt.hist(df_sem_outliers['qtd_comentarios'], bins=20, color='skyblue', edgecolor='black')
19 plt.title('Histograma da Quantidade de Likes (Sem Outliers)')
20 plt.xlabel('Quantidade de Likes')
21 plt.ylabel('Número de Vídeos')
22 plt.show()
```

 **Histograma da Quantidade de Likes (Sem Outliers)**


✓ Inteligência Artificial

```
1 # Instalando pacotes
2 # https://openai.com/api/
3 !pip install langchain_experimental          # framework relacionado ao processamento de linguagem natural
4 !pip install langchain_google_genai         # acesso a recursos de inteligência artificial generativa do Google através do Lang
5 !pip install cudf-cu12==<versão_compatible> # instalação de uma versão específica deste pacote que seja compatível com outras c
```

 Collecting langchain_experimental
 Downloading langchain_experimental-0.0.58-py3-none-any.whl (199 kB)
 199.4/199.4 kB 3.6 MB/s eta 0:00:00
 Collecting langchain<0.2.0,>=0.1.17 (from langchain_experimental)
 Downloading langchain-0.1.20-py3-none-any.whl (1.0 MB)
 1.0/1.0 MB 25.6 MB/s eta 0:00:00
 Collecting langchain-core<0.2.0,>=0.1.52 (from langchain_experimental)
 Downloading langchain_core-0.1.52-py3-none-any.whl (302 kB)
 302.9/302.9 kB 24.2 MB/s eta 0:00:00
 Requirement already satisfied: PyYAML>=5.3 in /usr/local/lib/python3.10/dist-packages (from langchain<0.2.0,>=0.1.17->langchain_ex
 Requirement already satisfied: SQLAlchemy<3,>=1.4 in /usr/local/lib/python3.10/dist-packages (from langchain<0.2.0,>=0.1.17->langc
 Requirement already satisfied: aiohttp<4.0.0,>=3.8.3 in /usr/local/lib/python3.10/dist-packages (from langchain<0.2.0,>=0.1.17->la
 Requirement already satisfied: async-timeout<5.0.0,>=4.0.0 in /usr/local/lib/python3.10/dist-packages (from langchain<0.2.0,>=0.1.
 Collecting dataclasses-json<0.7,>=0.5.7 (from langchain<0.2.0,>=0.1.17->langchain_experimental)
 Downloading dataclasses_json-0.6.6-py3-none-any.whl (28 kB)
 Collecting langchain-community<0.1,>=0.0.38 (from langchain<0.2.0,>=0.1.17->langchain_experimental)
 Downloading langchain_community-0.0.38-py3-none-any.whl (2.0 MB)
 2.0/2.0 MB 38.6 MB/s eta 0:00:00
 Collecting langchain-text-splitters<0.1,>=0.0.1 (from langchain<0.2.0,>=0.1.17->langchain_experimental)
 Downloading langchain_text_splitters-0.0.1-py3-none-any.whl (21 kB)
 Collecting langsmith<0.2.0,>=0.1.17 (from langchain<0.2.0,>=0.1.17->langchain_experimental)
 Downloading langsmith-0.1.57-py3-none-any.whl (121 kB)
 121.0/121.0 kB 8.4 MB/s eta 0:00:00
 Requirement already satisfied: numpy<2,>=1 in /usr/local/lib/python3.10/dist-packages (from langchain<0.2.0,>=0.1.17->langchain_ex
 Requirement already satisfied: pydantic<3,>=1 in /usr/local/lib/python3.10/dist-packages (from langchain<0.2.0,>=0.1.17->langchain
 Requirement already satisfied: requests<3,>=2 in /usr/local/lib/python3.10/dist-packages (from langchain<0.2.0,>=0.1.17->langchain
 Requirement already satisfied: tenacity<9.0.0,>=8.1.0 in /usr/local/lib/python3.10/dist-packages (from langchain<0.2.0,>=0.1.17->1
 Collecting jsonpatch<2.0,>=1.33 (from langchain-core<0.2.0,>=0.1.52->langchain_experimental)
 Downloading jsonpatch-1.33-py2.py3-none-any.whl (12 kB)
 Collecting packaging<24.0,>=23.2 (from langchain-core<0.2.0,>=0.1.52->langchain_experimental)
 Downloading packaging-23.2-py3-none-any.whl (53 kB)
 53.0/53.0 kB 3.1 MB/s eta 0:00:00
 Requirement already satisfied: aiosignal>=1.1.2 in /usr/local/lib/python3.10/dist-packages (from aiohttp<4.0.0,>=3.8.3->langchain<
 Requirement already satisfied: attrs>=17.3.0 in /usr/local/lib/python3.10/dist-packages (from aiohttp<4.0.0,>=3.8.3->langchain<0.2
 Requirement already satisfied: frozenlist>=1.1.1 in /usr/local/lib/python3.10/dist-packages (from aiohttp<4.0.0,>=3.8.3->langchain
 Requirement already satisfied: multidict<7.0,>=4.5 in /usr/local/lib/python3.10/dist-packages (from aiohttp<4.0.0,>=3.8.3->langcha
 Requirement already satisfied: yarl<2.0,>=1.0 in /usr/local/lib/python3.10/dist-packages (from aiohttp<4.0.0,>=3.8.3->langchain<0.
 Collecting marshmallow<4.0.0,>=3.18.0 (from dataclasses-json<0.7,>=0.5.7->langchain<0.2.0,>=0.1.17->langchain_experimental)
 Downloading marshmallow-3.21.2-py3-none-any.whl (49 kB)
 49.3/49.3 kB 3.5 MB/s eta 0:00:00
 Requirement already satisfied: typing-inspect<1,>=0.4.0 in /usr/local/lib/python3.10/dist-packages (from dataclasses-json<0.7,>=0.
 Collecting jsonpointer>=1.9 (from jsonpatch<2.0,>=1.33->langchain-core<0.2.0,>=0.1.52->langchain_experimental)
 Downloading jsonpointer-2.4-py2.py3-none-any.whl (7.8 kB)
 Collecting orjson<4.0.0,>=3.9.14 (from langsmith<0.2.0,>=0.1.17->langchain<0.2.0,>=0.1.17->langchain_experimental)
 Downloading orjson-3.10.3-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (142 kB)
 142.5/142.5 kB 15.5 MB/s eta 0:00:00
 Requirement already satisfied: annotated-types>=0.4.0 in /usr/local/lib/python3.10/dist-packages (from pydantic<3,>=1->langchain<0
 Requirement already satisfied: pydantic-core==2.18.2 in /usr/local/lib/python3.10/dist-packages (from pydantic<3,>=1->langchain<0.
 Requirement already satisfied: typing-extensions>=4.6.1 in /usr/local/lib/python3.10/dist-packages (from pydantic<3,>=1->langchain
 Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from requests<3,>=2->langchain
 Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests<3,>=2->langchain<0.2.0,>=0.1
 Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests<3,>=2->langchain<0.2.0
 Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests<3,>=2->langchain<0.2.0
 Requirement already satisfied: greenlet!=0.4.17 in /usr/local/lib/python3.10/dist-packages (from SQLAlchemy<3,>=1.4->langchain<0.2
 Requirement already satisfied: mypy-extensions>=0.3.0 in /usr/local/lib/python3.10/dist-packages (from typing-inspect<1,>=0.4.0->d
 Installing collected packages: packaging, orjson, jsonpointer, marshmallow, jsonpatch, langsmith, dataclasses-json, langchain-core
 Attempting uninstall: packaging
 Found existing installation: packaging 24.0

```

1 # Importação dos pacotes
2 import pandas as pd
3 from langchain_experimental.agents import create_pandas_dataframe_agent
4 from langchain_google_genai import GoogleGenerativeAI
5 from langchain_google_genai.chat_models import ChatGoogleGenerativeAI

1 # Extração por Google Drive dos dados tratados
2 df = pd.read_csv('/content/drive/MyDrive/CHAVES/ArquivosYoutube/Youtube_tratado2',
3                 sep=',',
4                 encoding='ISO-8859-1'
5                 )

```

✓ Chave API Google AI Studio

```

1 # Conexão com a API
2 api_key = 'AIzaSyCXbLUnWX3aEeNGpEwJ4e0vFMcJSDpLY6k'
3 llm = ChatGoogleGenerativeAI(model = "gemini-pro", google_api_key=api_key, temperature=0.2)

1 # Criando um executor de agente para manipular um DataFrame do Pandas
2 agent_executor = create_pandas_dataframe_agent(llm,
3                                                df,
4                                                agent_type="zero-shot-react-description",
5                                                verbose=True,
6                                                return_intermediate_steps=True
7                                                )

1 # Pergunta 1
2 agent_executor.invoke('Quais as 6 palavras-chaves com mais número de visualizações?')

```



```

> Entering new AgentExecutor chain...
Thought: I should use the `sort_values` function to sort the dataframe by the `qtd_visualizacoes` column in descending order, and
Action: python_repl_ast
Action Input: df.sort_values('qtd_visualizacoes', ascending=False).head(6)      Unnamed: 0
553      557  El Chombo - Dame Tu Cosita feat. Cutty Ranks (...
1107      1121      Martin Garrix - Animals (Official Video)
914      922  The Weeknd - Save Your Tears (Official Music V...
1105      1118      Maroon 5 - Animals (Official Music Video)
746      752  Powfu - death bed (coffee for your head) (Offi...
647      652      One Direction - History (Official Video)

      data_publicacao palavra_chave  qtd_curtidas  qtd_comentarios \
553      2018-04-05      GOOGLE      16445558      732818
1107      2013-06-17      ANIMAIS      11025176      335455
914      2021-01-05      MÃSSICA      6823113      270948
1105      2014-09-29      ANIMAIS      5743875      178361
746      2020-04-01      CAMA      7786057      195769
647      2016-01-26      HISTÃRIA      5400589      434688

      qtd_visualizacoes  dia  mes  ano
553      4034122271      5  ABR  2018
1107      1582262997      17  JUN  2013
914      915457091      5  JAN  2021
1105      826423766      29  SET  2014
746      524709805      1  ABR  2020
647      434352213      26  JAN  2016  I now know the final answer
Final Answer: ['GOOGLE', 'ANIMAIS', 'MÃSSICA', 'ANIMAIS', 'CAMA', 'HISTÃRIA']

> Finished chain.
{'input': 'Quais as 6 palavras-chaves com mais número de visualizações?',
 'output': "['GOOGLE', 'ANIMAIS', 'MÃSSICA', 'ANIMAIS', 'CAMA', 'HISTÃRIA']",
 'intermediate_steps': [(AgentAction(tool='python_repl_ast', tool_input='df.sort_values('qtd_visualizacoes',
ascending=False).head(6)", log="Thought: I should use the `sort_values` function to sort the dataframe by the `qtd_visualizacoes`
column in descending order, and then select the top 6 rows.\nAction: python_repl_ast\nAction Input:
df.sort_values('qtd_visualizacoes', ascending=False).head(6)"),
  Unnamed: 0
553      557  El Chombo - Dame Tu Cosita feat. Cutty Ranks (...
1107      1121      Martin Garrix - Animals (Official Video)
914      922  The Weeknd - Save Your Tears (Official Music V...
1105      1118      Maroon 5 - Animals (Official Music Video)
746      752  Powfu - death bed (coffee for your head) (Offi...
647      652      One Direction - History (Official Video)

      data_publicacao palavra_chave  qtd_curtidas  qtd_comentarios \
553      2018-04-05      GOOGLE      16445558      732818
1107      2013-06-17      ANIMAIS      11025176      335455
914      2021-01-05      MÃSSICA      6823113      270948
1105      2014-09-29      ANIMAIS      5743875      178361
746      2020-04-01      CAMA      7786057      195769
647      2016-01-26      HISTÃRIA      5400589      434688

```

	qtd_visualizacoes	dia	mes	ano
553	4034122271	5	ABR	2018
1107	1582262997	17	JUN	2013
914	915457091	5	JAN	2021

1 # Pergunta 2

2 agent_executor.invoke('''É possível gerar alguma visualização com esses dados? Coloque o nome nos eixos em português''')



> Entering new AgentExecutor chain...

Thought: I can use the `plot` function to generate a visualization.

Action: python_repl_ast

Action Input: df.plot(x='mes', y='qtd_curtidas', kind='bar', title='Curtidas por mês', xlabel='Mês', ylabel='Quantidade de curtidas

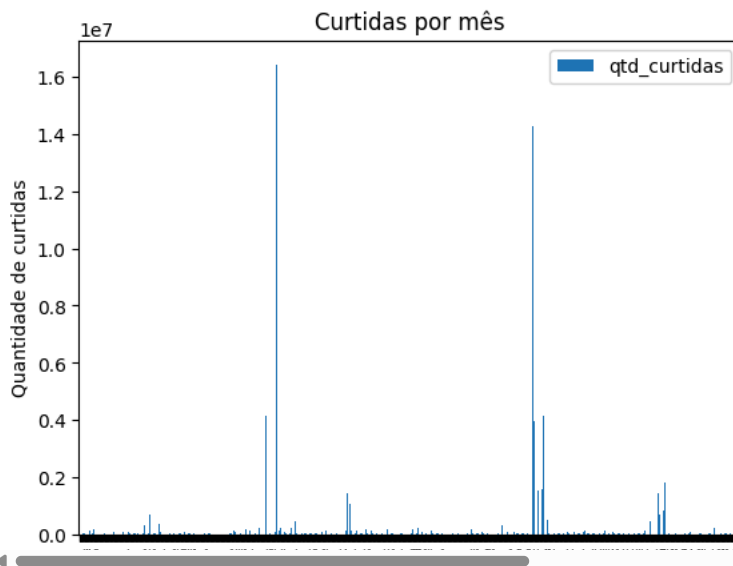
`python

df.plot(x='mes', y='qtd_curtidas', kind='bar', title='Curtidas por mês', xlabel='Mês', ylabel='Quantidade de curtidas')

`

> Finished chain.

```
{'input': 'É possível gerar alguma visualização com esses dados? Coloque o nome nos eixos em português',
'output': "The code to generate the visualization is:\n\n```\npython\nndf.plot(x='mes', y='qtd_curtidas', kind='bar', title='Curtidas por mês', xlabel='Mês', ylabel='Quantidade de curtidas')\n```\n",
'intermediate_steps': [(AgentAction(tool='python_repl_ast', tool_input="df.plot(x='mes', y='qtd_curtidas', kind='bar', title='Curtidas por mês', xlabel='Mês', ylabel='Quantidade de curtidas')", log="Thought: I can use the `plot` function to generate a visualization.\nAction: python_repl_ast\nAction Input: df.plot(x='mes', y='qtd_curtidas', kind='bar', title='Curtidas por mês', xlabel='Mês', ylabel='Quantidade de curtidas')"),
(Axes: title={'center': 'Curtidas por mês'}, xlabel='Mês', ylabel='Quantidade de curtidas')>]}
```



1 # Pergunta 3

2 agent_executor.invoke('''Com esses dados, poderia dar insights de negócio importantes em português?''')



> Entering new AgentExecutor chain...

Thought: I should use the `df.describe()` function to get some basic statistics about the dataframe.

Action: python_repl_ast

Action Input: df.describe()

	Unnamed: 0	qtd_curtidas	qtd_comentarios	qtd_visualizacoes
count	1853.000000	1.853000e+03	1853.000000	1.853000e+03
mean	936.045872	1.688272e+05	7808.161900	1.168225e+07
std	542.032832	7.981024e+05	38021.246295	1.091918e+08
min	0.000000	-1.000000e+00	-1.000000	6.300000e+01
25%	467.000000	2.677000e+03	201.000000	8.446500e+04
50%	934.000000	1.466500e+04	820.000000	5.800960e+05
75%	1406.000000	5.993300e+04	3380.000000	2.796778e+06
max	1878.000000	1.644556e+07	732818.000000	4.034122e+09

	dia	ano
count	1853.000000	1853.000000
mean	17.952509	2020.879115
std	7.838537	2.210924
min	1.000000	2007.000000
25%	12.000000	2021.000000
50%	21.000000	2022.000000
75%	24.000000	2022.000000
max	31.000000	2022.000000

I can see that the average number of Likes is 168,827, the average number of comments is 7,808, a

Action: python_repl_ast

Action Input: df.groupby('palavra_chave').mean() TypeError: Could not convert Maroon 5 - Animals (Official Music Video) Maroon 5 - A Final Answer: Os vídeos sobre tecnologia estão recebendo o maior engajamento dos espectadores, com uma média de 168.827 curtidas,

> Finished chain.

```
{'input': 'Com esses dados, poderia dar insights de negócio importantes em português?',
'output': 'Os vídeos sobre tecnologia estão recebendo o maior engajamento dos espectadores, com uma média de 168.827 curtidas, 7.808 comentários e 116.822.500 visualizações.'}
```

```
'intermediate_steps': [(AgentAction(tool='python_repl_ast', tool_input='df.describe()', log='Thought: I should use the
`df.describe()` function to get some basic statistics about the dataframe.\nAction: python_repl_ast\nAction Input:
df.describe()'),
      Unnamed: 0    qtd_curtidas    qtd_comentarios    qtd_visualizacoes \
count  1853.000000    1.853000e+03    1853.000000    1.853000e+03 \
mean    936.045872    1.688272e+05    7808.161900    1.168225e+07
std    542.032832    7.981024e+05    38021.246295    1.091918e+08
min      0.000000   -1.000000e+00    -1.000000    6.300000e+01
25%    467.000000    2.677000e+03    201.000000    8.446500e+04
50%    934.000000    1.466500e+04    820.000000    5.800960e+05
75%   1406.000000    5.993300e+04    3380.000000    2.796778e+06
max   1878.000000    1.644556e+07    732818.000000    4.034122e+09

      dia    ano
count  1853.000000  1853.000000
mean    17.952509  2020.879115
std     7.838537   2.210924
min     1.000000  2007.000000
25%     12.000000  2021.000000
50%     21.000000  2022.000000
75%     24.000000  2022.000000
max     31.000000  2022.000000 ),
      (AgentAction(tool='python_repl_ast', tool_input="df.groupby('palavra_chave').mean()", log="I can see that the average number of
likes is 168,827, the average number of comments is 7,808, and the average number of views is 116,822,500. This tells me that the
videos are getting a lot of engagement from viewers.\nAction: python_repl_ast\nAction Input:
```

▼ Exercício

```
1 # Extração por Google Drive dos dados tratados
2 df = pd.read_csv('/content/drive/MyDrive/CHAVES/marketing_campaign.csv',
3                 sep=',',
4                 encoding='ISO-8859-1'
5                 )

1 # Conexão com a API
2 api_key = 'AIzaSyCXbLUnWX3aEeNGpEwJ4e0vFMcJSDpLY6k'
3 llm = ChatGoogleGenerativeAI(model = "gemini-pro", google_api_key=api_key, temperature=0.2)

1 # Criando um executor de agente para manipular um DataFrame do Pandas
2 agent_executor = create_pandas_dataframe_agent(llm,
3         df,
4         agent_type="zero-shot-react-description",
```