

Escola Superior de Tecnologia e Gestão

Engenharia Informática

Tópicos de Engenharia Informática

2020/2021

Projeto Exploratório

Alunos:

Osmin H. S. Ialá, nº 15799

Felizmelo Borja, nº 16709

Docente:

João Martins

Data Mining

Data mining é um processo em que a tecnologia é utilizada para localizar padrões, conexões, correlações ou anomalias em uma grande quantidade de dados, permitindo encontrar problemas, hipóteses e oportunidades com mais facilidade.

Vantagens de Data Mining

A análise de dados através do Data Mining pode proporcionar inúmeras vantagens às empresas para a otimização da sua gestão e tempo, mas também para o recrutamento e fidelização de clientes, o que lhes permitirá aumentar as suas vendas. Aqui deixamos 8 vantagens que podem nos trazer:

- Permite descobrir informações que não esperávamos obter. Isto é devido ao seu funcionamento com algoritmos, já que permite fazer muitas combinações diferentes.
- É capaz de analisar bancos de dados com uma enorme quantidade de dados.
- Os resultados são muito fáceis de interpretar e não é necessário ter conhecimentos em engenharia da computação.
- Permite encontrar, atrair e reter clientes.
- A empresa pode melhorar o atendimento ao cliente com base nas informações obtidas.
- Dá às empresas a possibilidade de oferecer aos clientes os produtos ou serviços de que necessitam.

- Antes de usar os modelos, eles são verificados por estatísticas para verificar se as previsões obtidas são válidas.
- Economiza custos para a empresa e abre novas oportunidades de negócios.

No entanto, também pode haver algum inconveniente ao usar técnicas de Data Mining. Por exemplo, dependendo do tipo de dados que você deseja coletar, pode ser necessário muito trabalho ou, às vezes, o investimento inicial para obter as tecnologias necessárias para a coleta de dados pode ser muito alto.

- Gestão de empresas
- Segurança
- Saúde
- Pesquisa científica
- Vendas e marketing
- Pesquisa científica
- Finanças
- Energia
- Telecomunicações
- Comércio eletrônico
- Finanças
- CRM
- Agricultura

Apresentando o dataset sobre vendas de video games

Abaixo você pode conferir as colunas encontradas no arquivo "vgsales.csv":

- **Rank:** posição no ranking de vendas
- **Name:** nome do jogo
- **Platform:** plataforma em que o jogo foi liberado (PC, PS4, Xbox, etc.)
- **Year:** ano de lançamento do game
- **Genre:** gênero do jogo
- **Publisher:** empresa que publicou o jogo
- **NA_Sales:** vendas na América do Norte (em milhões de dólares)
- **EU_Sales:** vendas na Europa (em milhões de dólares)
- **JP_Sales:** vendas no Japão (em milhões de dólares)
- **Other_Sales:** vendas no restante do mundo (em milhões de dólares)
- **Global_Sales:** total de vendas no mundo inteiro

O que é o notebook Anaconda e Jupyter?

O Anaconda é uma distribuição pré-empacotada do Python, que contém vários módulos e pacotes do Python, incluindo o Jupyter . O Jupyter é uma maneira de trabalhar com o Python dentro de um " notebook " virtual e é bastante popular na Ciência de Dados. Oferece uma maneira de combinar código, imagens, gráficos, anotações etc.

"O Jupyter Notebook é um aplicativo da web de código aberto que permite criar e compartilhar documentos que contêm código ativo , equações, visualizações e texto explicativo. Os usos incluem: limpeza e transformação de dados, simulação numérica, modelagem estatística, aprendizado de máquina e muito mais. "

In [6]:

```
%matplotlib inline
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
plt.style.use('ggplot')
```

%matplotlib inline serve para fazer com que os gráficos gerados por nosso código sejam mostrados dentro do notebook, e não em uma nova janela.

import faz com que o Python importe alguma biblioteca para ser utilizada em seu código.

plt.style.use("ggplot") -> adiciona cores e padrões diferentes para personalizar seus gráficos.

In [7]:

```
# Leitura do arquivo
videogames = pd.read_csv('vgsales.csv')
```

Em seguida, criamos uma nova célula e iniciamos a leitura do nosso dataset. Utilizando a função **pd.read_csv('local_do_arquivo.extensao')** carregamos o dataset para a memória e o armazenamos dentro da variável "videogames".

In [3]:

```
#Exibir as 10 primeiras Linhas do Dataframe
videogames.head(10)
```

Out[3]:

	Rank	Name	Platform	Year	Genre	Publisher	NA_Sales	EU_Sales	JP_Sales	Other_S
0	1	Wii Sports	Wii	2006.0	Sports	Nintendo	41.49	29.02	3.77	
1	2	Super Mario Bros.	NES	1985.0	Platform	Nintendo	29.08	3.58	6.81	
2	3	Mario Kart Wii	Wii	2008.0	Racing	Nintendo	15.85	12.88	3.79	
3	4	Wii Sports Resort	Wii	2009.0	Sports	Nintendo	15.75	11.01	3.28	
4	5	Pokemon Red/Pokemon Blue	GB	1996.0	Role-Playing	Nintendo	11.27	8.89	10.22	
5	6	Tetris	GB	1989.0	Puzzle	Nintendo	23.20	2.26	4.22	
6	7	New Super Mario Bros.	DS	2006.0	Platform	Nintendo	11.38	9.23	6.50	
7	8	Wii Play	Wii	2006.0	Misc	Nintendo	14.03	9.20	2.93	

	Rank	Name	Platform	Year	Genre	Publisher	NA_Sales	EU_Sales	JP_Sales	Other_Sales
8	9	New Super Mario Bros. Wii	Wii	2009.0	Platform	Nintendo	14.59	7.06	4.70	
9	10	Duck Hunt	NES	1984.0	Shooter	Nintendo	26.93	0.63	0.28	

In [4]:

```
#Resumo de informações em todas as colunas
videogames.describe()
```

Out[4]:

	Rank	Year	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales
count	16598.000000	16327.000000	16598.000000	16598.000000	16598.000000	16598.000000	16598.000000
mean	8300.605254	2006.406443	0.264667	0.146652	0.077782	0.048063	0.517424
std	4791.853933	5.828981	0.816683	0.505351	0.309291	0.188588	1.517424
min	1.000000	1980.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	4151.250000	2003.000000	0.000000	0.000000	0.000000	0.000000	0.000000
50%	8300.500000	2007.000000	0.080000	0.020000	0.000000	0.010000	0.110000
75%	12449.750000	2010.000000	0.240000	0.110000	0.040000	0.040000	0.410000
max	16600.000000	2020.000000	41.490000	29.020000	10.220000	10.570000	82.740000

A Função `describe()`-> retorna um resumo de todas as variáveis ou coluna da tabela numérica em nosso dataset.

Neste caso a função `describe()` traz informações sobre nossos dados, que podem ser usados para gerar estatísticas.

Por exemplo, verifique que cada linha da tabela mostra um dado diferente. Na primeira nós temos a **count**, que realiza uma contagem dos valores existentes no conjunto de dados;

A segunda linha — **std**, ou standard deviation — traz o desvio padrão dos valores;

Em seguida temos os valores mínimos, quartis e valores máximos de cada coluna.

In [5]:

```
#Tipo de dado em cada coluna
videogames.dtypes
```

Out[5]:

Rank	int64
Name	object
Platform	object
Year	float64
Genre	object
Publisher	object
NA_Sales	float64
EU_Sales	float64
JP_Sales	float64
Other_Sales	float64
Global_Sales	float64
dtype:	object

Ao carregar um arquivo no pandas usando a função `read_csv()`, a biblioteca pode não conseguir identificar o tipo de dado de uma coluna.

Conforme a imagem acima, nós criamos uma célula e digitamos a linha `videogames.dtypes`. o pandas retornou uma tabela com duas colunas, sendo a primeira delas o nome das colunas encontradas em nosso dataset, e a segunda o tipo de dado que a biblioteca reconheceu.

No caso da coluna **rank**, ela foi reconhecida como números inteiros; já a coluna **Name** está sendo tratada como string ou seja é referenciada como um objeto.

```
In [6]: #Quatidade de Linhas e colunas no Dataframe
videogames.shape
```

Out[6]: (16598, 11)

shape -> permite saber o tamanho de Dataframe.

Neste caso retorna uma tuple com dois valores, cujo o primeiro representa a quantidade de linhas, e o segundo, a quantidade de colunas da nossa tabela.

```
In [7]: #Renomiar colunas
videogames.columns = ['Ranking', 'Nome', 'Plataforma', 'Ano',
                      'Gênero', 'Editora', 'Vendas América do Norte', 'Vendas EUA',
                      'Vendas Japão', 'Outras vendas', 'Vendas Global']
```

```
In [8]: #Visualizar o novo Dataframe traduzido
videogames.head()
```

Out[8]:

	Ranking	Nome	Plataforma	Ano	Gênero	Editora	Vendas América do Norte	Vendas EUA	Vendas Japão	Outra venda
0	1	Wii Sports	Wii	2006.0	Sports	Nintendo	41.49	29.02	3.77	8.4
1	2	Super Mario Bros.	NES	1985.0	Platform	Nintendo	29.08	3.58	6.81	0.7
2	3	Mario Kart Wii	Wii	2008.0	Racing	Nintendo	15.85	12.88	3.79	3.3
3	4	Wii Sports Resort	Wii	2009.0	Sports	Nintendo	15.75	11.01	3.28	2.9
4	5	Pokemon Red/Pokemon Blue	GB	1996.0	Role-Playing	Nintendo	11.27	8.89	10.22	1.0

```
In [9]: #Verifica Linhas onde não há ano de Lançamento definido
videogames[videogames['Ano'].isnull()].head()
```

Out[9]:

	Ranking	Nome	Plataforma	Ano	Gênero	Editora	Vendas América do Norte	Vendas EUA	Vendas Japão	Ou ver
179	180	Madden NFL 2004	PS2	NaN	Sports	Electronic Arts	4.26	0.26	0.01	

Ranking	Nome	Plataforma	Ano	Gênero	Editora	Vendas	Vendas EUA	Vendas Japão	Outro
						América do Norte			
377	378	FIFA Soccer 2004	PS2	Sports	Electronic Arts	0.59	2.36	0.04	
431	432	LEGO Batman: The Videogame	Wii	Action	Warner Bros. Interactive Entertainment	1.86	1.02	0.00	
470	471	wwe Smackdown vs. Raw 2006	PS2	Fighting	NaN	1.57	1.02	0.00	
607	608	Space Invaders	2600	Shooter	Atari	2.36	0.14	0.00	

◀ ▶

In [10]:

```
#Mostra numeros de jogos que foram Lançados para cada plataforma.
videogames['Plataforma'].value_counts()
```

Out[10]:

DS	2163
PS2	2161
PS3	1329
Wii	1325
X360	1265
PSP	1213
PS	1196
PC	960
XB	824
GBA	822
GC	556
3DS	509
PSV	413
PS4	336
N64	319
SNES	239
XOne	213
SAT	173
WiiU	143
2600	133
NES	98
GB	98
DC	52
GEN	27
NG	12
WS	6
SCD	6
3DO	3
TG16	2
GG	1
PCFX	1

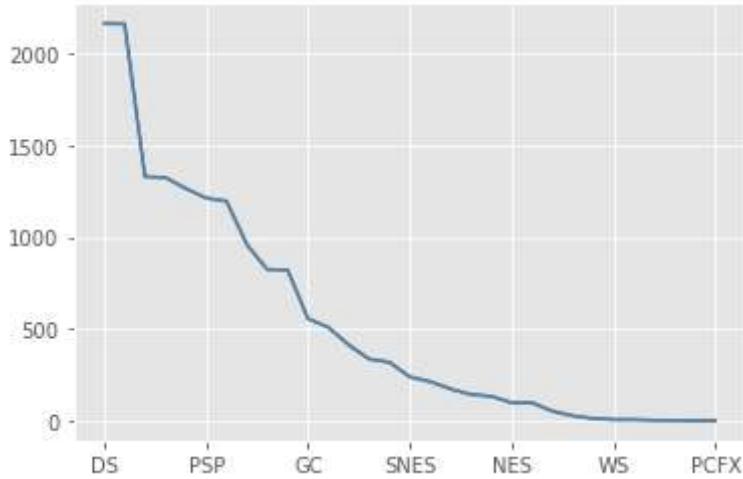
Name: Plataforma, dtype: int64

Criar gráficos com pandas

In [11]:

```
#Grafico padrão criado pelo pandas
titulos_lancados = videogames['Plataforma'].value_counts()
titulos_lancados.plot()
videogames['Plataforma'].value_counts().plot()
```

Out[11]: <AxesSubplot:>



A função `.plot()` do pandas irá criar, por padrão, um gráfico de linhas utilizando o index como **eixo X** — parte debaixo do gráfico — e valores da contagem no **eixo Y**.

Dá uma olhada no código abaixo. Primeiro nós realizamos a criação de um gráfico utilizando a mesma linha de código que mencionamos anteriormente. A diferença aqui é que passamos cinco argumentos dentro da função `plot()`, sendo eles:

kind: informa o tipo de gráfico que queremos criar. Nesse caso, escolhemos um gráfico de barras, ou "bar";

figsize: informa o tamanho que nosso gráfico terá. Esse valor precisa ser passado em forma de tuple — entre parenteses;

grid: diz se nós queremos que apareça linhas de grade em nosso gráfico. Dependendo da situação, eles são muito úteis, mas aqui não será necessário, portanto, informamos o valor `False`;

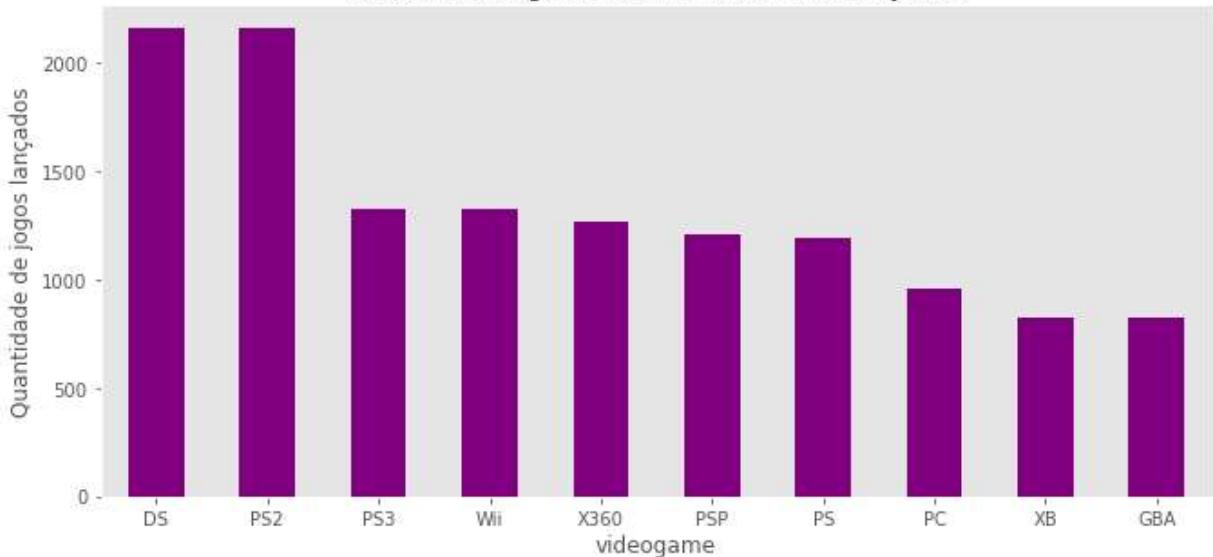
rot: informa o grau de rotação que nossos dados do eixo X devem ter. Como queremos que os valores sejam apresentados em linha reta sem inclinação, colocamos o valor zero (0);

color: a cor do gráfico. Dependendo do tipo de gráfico, você pode informar mais de uma cor em formato de lista, por exemplo: `['green', 'red', 'blue']`. Queremos que o nosso gráfico tenha apenas a cor roxo, colocamos o valor `'purple'`.

In [15]:

```
#Criar grafico utilizando apenas uma Linha de código
videogames['Plataforma'].value_counts().head(10).plot(kind='bar', figsize=(11,5), gr
#Enfeitar grafico. abaixo definimos um título.
plt.title('Os 10 videos games com mais títulos lançados')
plt.xlabel('videogame') #Nomiar o eixo x, onde fica o nome dos videogames
plt.ylabel('Quantidade de jogos lançados') # nomiar o eixo Y, onde fica o quantida
plt.show() #Exibe o grafico.
```

Os 10 videos games com mais títulos lançados



plt.title(): Essa função adiciona um título ao gráfico, o valor da função deve ser na forma de string — utilizando aspas simples ou duplas. No nosso caso, colocamos o título: “Os 10 videogames com mais títulos lançados”;

plt.xlabel(): Assim como na função title, os valores dessa função devem ser passados como string. Aqui, você define o título — ou label — que será colocado no eixo X — abaixo dos gráficos. Escolhemos a palavra “Videogame”;

plt.ylabel(): Exatamente igual a função xlabel(), mas, aqui você diz o que será exibido no eixo Y — na parte esquerda do mapa. Colocamos “Quantidade de jogos lançados”;

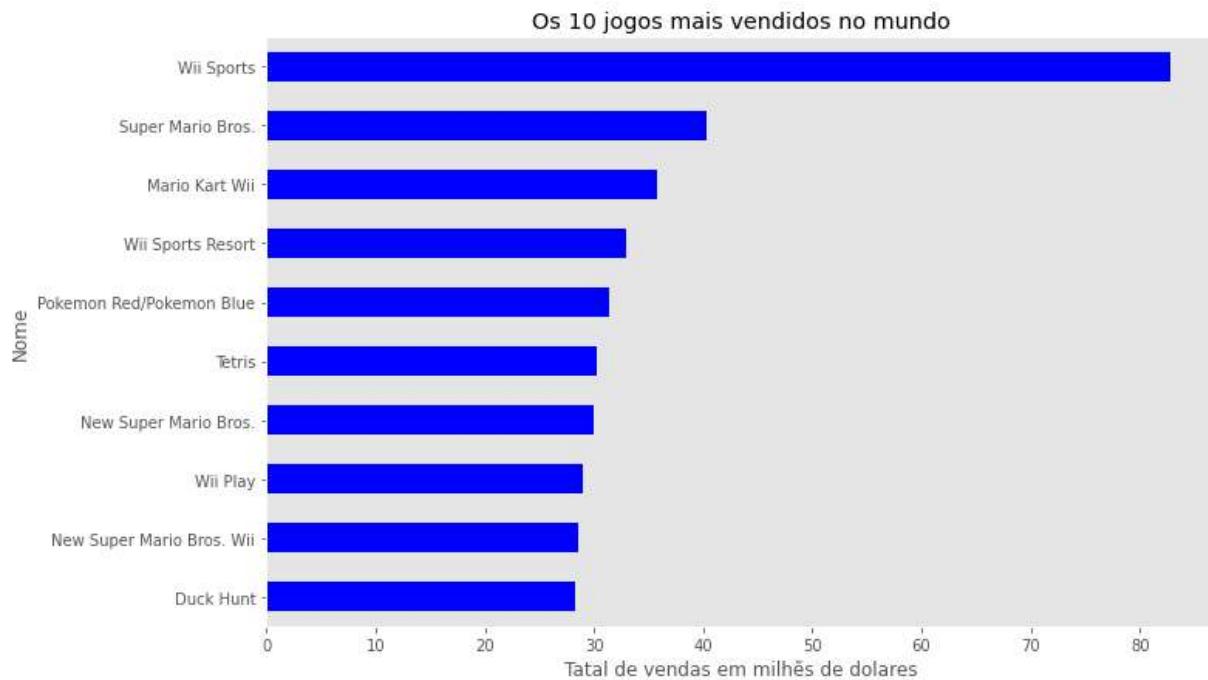
plt.show(): Essa função irá exibir seu gráfico na célula abaixo. Mesmo que você não a digite, seu gráfico será mostrado, contudo, isso facilita a leitura do código e evita que algumas informações inúteis sejam apresentadas na célula, como o tipo de dado do gráfico.

Grafico com o matplotlib

In [16]:

```
#os jogos mais vendidos da história
top_10_vendidos = videogames[['Nome', 'Vendas Global']].head(10).set_index('Nome').s
top_10_vendidos.plot(kind='barh', figsize=(11,7), grid = False, color='blue', legend

plt.title('Os 10 jogos mais vendidos no mundo')
plt.xlabel('Total de vendas em milhês de dólares')
plt.show() #Exibe o grafico.
```



Criando um mapa de calor

Para finalizar esse nosso tutorial, vamos deixar nosso projeto mais atraente ao utilizarmos um mapa de calor. Caso você não saiba, um mapa de calor — do inglês heatmap — é um tipo de gráfico que irá realçar frequências mais altas com cores mais escuras. Em nossa figura acima, nosso mapa mostra quantos jogos foram lançados para cada gênero em cada uma das plataformas disponíveis no dataset.

In [17]:

```
#Utilizando a função crosstab do pandas, para fazer informações cruzadas
crosstab_vg = pd.crosstab(videogames['Plataforma'], videogames['Gênero'])
crosstab_vg.head()
```

Out[17]:

Plataforma	Gênero	Action	Adventure	Fighting	Misc	Platform	Puzzle	Racing	Role-Playing	Shooter	Simula
2600	61	2	2	5	9	11	6	0	24		
3DO	0	1	0	0	0	1	0	0	0	0	
3DS	182	37	14	53	28	20	11	86	7		
DC	3	11	12	0	2	0	6	4	3		
DS	343	240	36	393	92	238	67	200	42		

No código acima fazemos um cruzamento entre a coluna Plataforma e Gênero do nosso conjunto de dados e armazenamos na variável crosstab_vg.

Em seguida exibimos as cinco primeiras linhas do novo DataFrame usando a função head().

Primeiramente, nós temos que criar um DataFrame que contém informações cruzadas sobre o gênero do jogo e o console para o qual ele foi lançado. Utilizaremos a função .crosstab() do

pandas para isso. A função crosstab() pega dois argumentos, ou, em outras palavras, duas colunas que nós queremos cruzar informações.

In [18]:

```
#Vamos adicionar a coluna Total ao DataFrame
crosstab_vg['Total'] = crosstab_vg.sum(axis=1)
crosstab_vg.head()
```

Out[18]:

Gênero	Action	Adventure	Fighting	Misc	Platform	Puzzle	Racing	Role-Playing	Shooter	Simula
Plataforma										
2600	61	2	2	5	9	11	6	0	24	
3DO	0	1	0	0	0	1	0	0	0	
3DS	182	37	14	53	28	20	11	86	7	
DC	3	11	12	0	2	0	6	4	3	
DS	343	240	36	393	92	238	67	200	42	

Na célula seguinte, adicionamos a coluna Total ao DataFrame. Essa coluna tem como valor a soma de todos os jogos lançados para cada console. Na última linha, por exemplo, sabemos que o videogame Nintendo DS teve um total de 2163 games lançados. Essa coluna nos ajudará ainda mais no momento de visualizar nosso gráfico.

In [23]:

```
top10_platforms = crosstab_vg[crosstab_vg['Total'] > 1000].sort_values('Total', ascending=False)
top10_final = top10_platforms.append(pd.DataFrame(top10_platforms.sum(), columns=['Total']))

sns.set(font_scale=1)
plt.figure(figsize=(18, 9))
sns.heatmap(top10_final, annot=True, vmax=top10_final.loc[:, 'PS': 'Strategy'].values.max())
plt.xlabel('Gênero')
plt.ylabel('Console')
plt.title('Quantidade de títulos por gênero e Console')
plt.show()
```

Venda dos videoGames



E, finalmente, realizamos a criação de nosso heatmap utilizando o código da figura acima. Vamos ver linha por linha o que nosso código faz:

Nesta linha criamos um DataFrame chamado top10_platforms, onde pretendemos armazenar os 10 consoles que mais tiveram lançamentos. Mas, temos outra condição: queremos apenas consoles que possuem mais de mil jogos lançados.

Após definir essa condição, utilizamos o sort_values() para que os dados do DataFrame sejam ordenador pela coluna Total em ordem decrescente.

Vamos a próxima linha: **top10_final =**

top10_platforms.append(pd.DataFrame(top10_platforms.sum(), columns=['total']).T, ignore_index=False) Essa linha irá criar um novo DataFrame similar ao anterior, mas que adiciona uma linha com o total no final do DataFrame top10_platforms. Além de termos o total de jogos por plataforma, também teremos o total de games por gênero.

sns.set(font_scale=1): define o tamanho da fonte a ser utilizada no gráfico;

plt.figure(figsize=(18, 9)): essa linha define o tamanho da figura do gráfico.

sns.heatmap(top10_final, annot=True, vmax=top10_final.loc[:, 'PS': 'Strategy'].values.max(), vmin=top10_final.loc[:, 'Strategy'].values.min(), fmt='d') Essa linha, primeiro, utilizamos o heatmap do seaborn para exibir no gráfico, em nosso caso, o DataFrame top10_final. Em seguida informamos o argumento **annot=True**, que faz com que os números de cada "quadrinho" apareça, facilitando a exibição. Se você definir esse valor para False, seu mapa de calor terá apenas as cores, sem qualquer número nelas.

Os argumentos **vmax** e **vmin** definem os valores mínimos e máximos do nosso gráfico. Isso serve para que o seaborn mapear as cores de cada célula do gráfico de acordo com tais valores. Vamos imaginar que o valor máximo é 1000. Quanto mais perto desse valor a célula estiver, mais escura será sua cor.

Para localizar os valores máximo e mínimo, utilizamos a função **.loc** do pandas e filtramos as linhas e colunas do DataFrame para localizar seus valores.

Por fim, eu utilizo o argumento **fmt='d'** para formatar as informações para que fiquem bem apresentadas.

Em seguida utilizamos as funções plt.xlabel(), plt.ylabel(), plt.title() e plt.show() — que expliquei anteriormente — para personalizar o gráfico.

In []:

