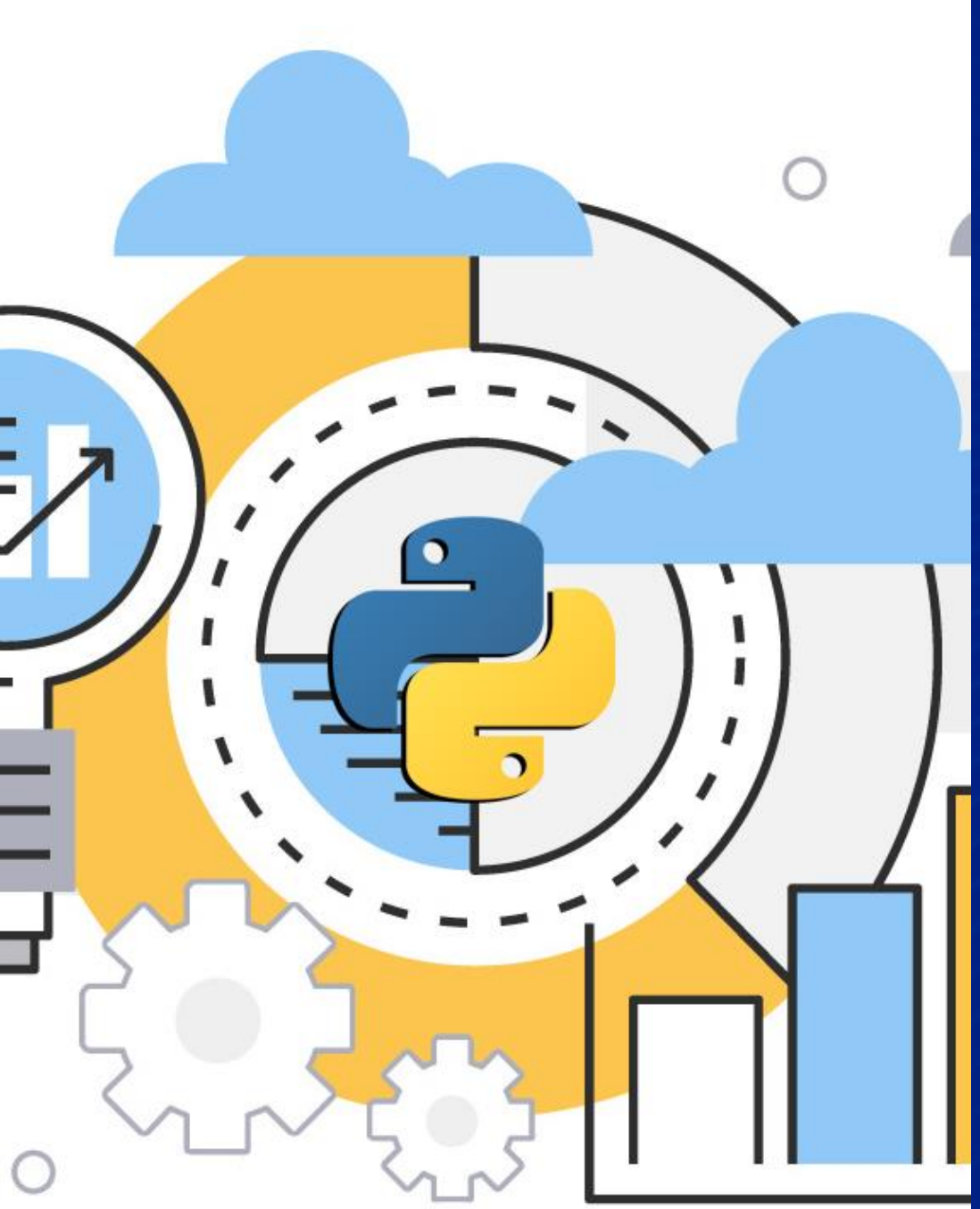
CIÊNCIA DE DADOS COM PYTHON

CAP.6 – INTRODUÇÃO AO MATPLOTLIB

Prof. Renzo Paranaíba Mesquita

Inatel



OBJETIVOS

- Darmos introdução em uma das bibliotecas mais populares do mercado para plotagem de gráficos: o Matplotlib;
- Compreendermos seu princípio de funcionamento e principais recursos;
- Praticarmos o seu uso sobre *Datasets* que já tivemos a oportunidade de ver anteriormente na disciplina.

Inatel

TÓPICOS

1. Introdução
2. Plotando gráficos com Plot
3. Formatando o estilo dos Plots
4. Plots múltiplos de forma conjunta
5. Plots múltiplos de forma separada
6. Tipos de Gráficos populares

Inatel

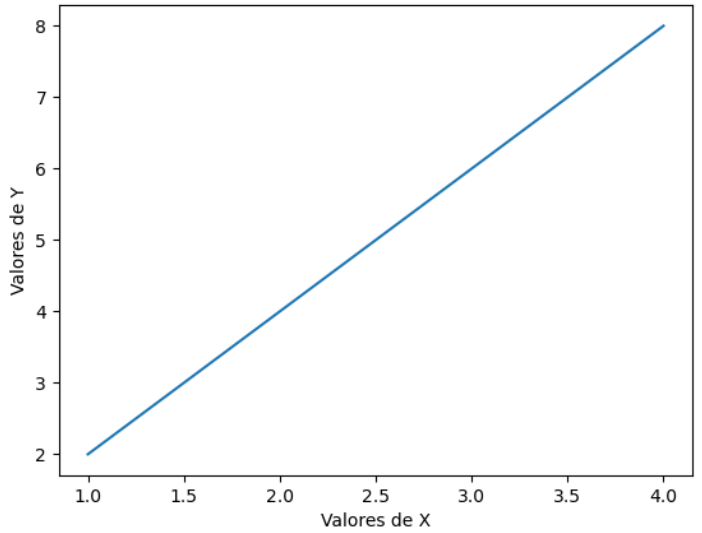
6.1. INTRODUÇÃO

- O Matplotlib é uma das bibliotecas de visualização de dados mais populares do mercado e a principal do mundo Python;
- Ela facilita a criação de gráficos estáticos, animados e interativos;
- Outros destaques desta biblioteca:
  - Permite a criação de **Plots com poucas linhas de código**;
  - Oferece **controle sobre todos os detalhes** das plotagens;
  - Possui uma grande **coletânea de gráficos**;
  - **Biblioteca base** para outras bibliotecas gráficas do Python;
  - Possui **excelente integração** com NumPy e Pandas;
  - Lembra muito os gráficos do famoso ***MatLab***.

6.2. PLOTANDO GRÁFICOS COM PLOT

- O matplotlib.pyplot é uma coleção de funções que permitem que o Matplotlib trabalhe de forma muito parecida com o *MatLab*
- Cada função do pyplot muda uma característica no gráfico
- **EXEMPLOS:**

```
# importando o numpy e o pyplot
import numpy as np
import matplotlib.pyplot as plt

# criando alguns valores no eixo x
x = np.array([1, 2, 3, 4])

# criando alguns valores no eixo y
y = x*2

# label das coordenadas x e y
plt.xlabel('Valores de x')
plt.ylabel('Valores de y')

# executando o plot
plt.plot(x, y)

# comando necessário em algumas IDEs apenas
plt.show()
```



6.3. FORMATANDO O ESTILO DOS PLOTS

- Existe um terceiro argumento customizado que podemos passar para um plot para mudar seu estilo de forma bastante objetiva
- Este argumento é uma String que pode ser montada obedecendo o seguinte padrão:
  fmt = [marker][line][color]

Markers

's'	square marker
'p'	pentagon marker
'*'	star marker
'h'	hexagon1 marker
'H'	hexagon2 marker
'+'	plus marker
'x'	x marker
'D'	diamond marker
'd'	thin_diamond marker
'|'	vline marker
'_'	hline marker

Line Styles

character	description
'-'	solid line style
'--'	dashed line style
'-.'	dash-dot line style
':'	dotted line style

Colors

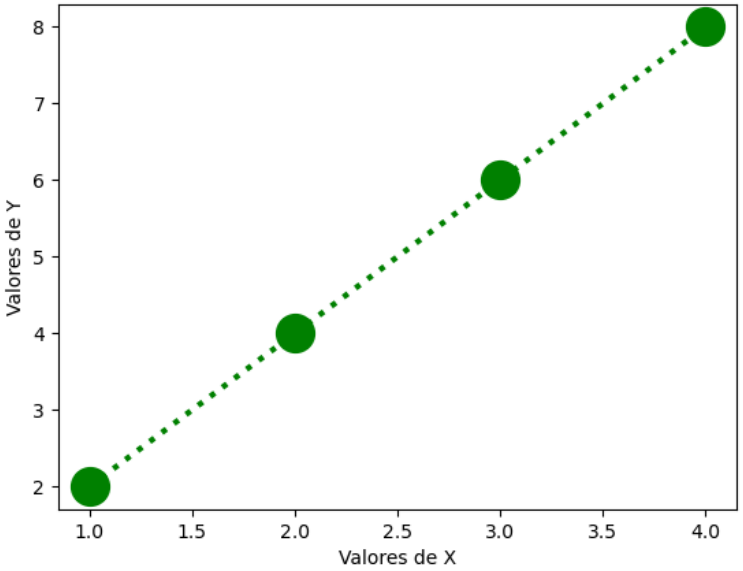
The supported color abbreviations are the single letter codes

character	color
'b'	blue
'g'	green
'r'	red
'c'	cyan
'm'	magenta
'y'	yellow
'k'	black
'w'	white

6.3. FORMATANDO O ESTILO DOS PLOTS

- Com a String customizada, podemos facilmente alterar o estilo dos nossos plots
- Um **Gráfico de Linhas (Plot)** é utilizado para representar dados que variam ao longo do tempo ou de outra variável contínua. Ele é ideal para mostrar tendências e mudanças em dados
- **EXEMPLOS:**

```
# marcador circular – o
# linhas pontilhadas - :
# cor verde – g (green)
# largura da linha = 3
# tamanho dos marcadores = 20
plt.plot(x, y, ‘o:g’, linewidth=3, markersize=20)
```

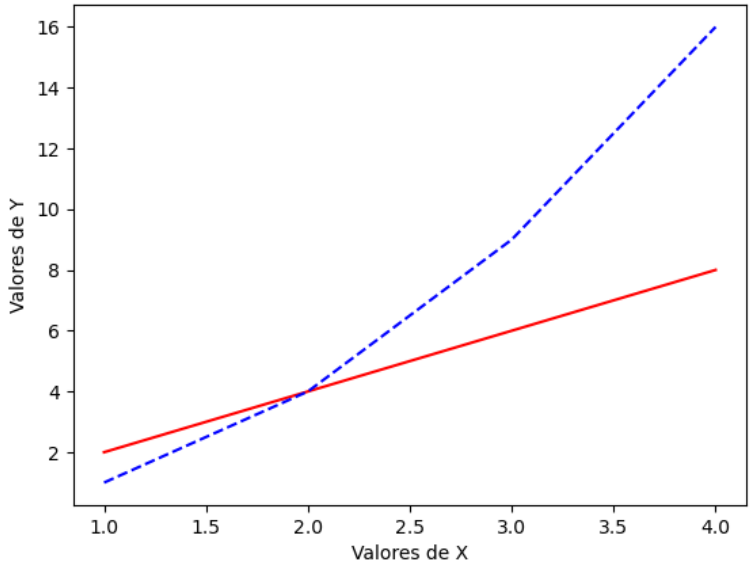



```
# comando necessário em algumas IDEs apenas
plt.show()
```

6.4. PLOTS MÚLTIPLOS DE FORMA CONJUNTA

- É comum querermos traçar dois ou mais gráficos em linha no mesmo plano cartesiano
- De forma objetiva, podemos passar para o método plot uma sequência de coordenadas x,y e String de customização no seguinte formato:
- **EXEMPLOS:**

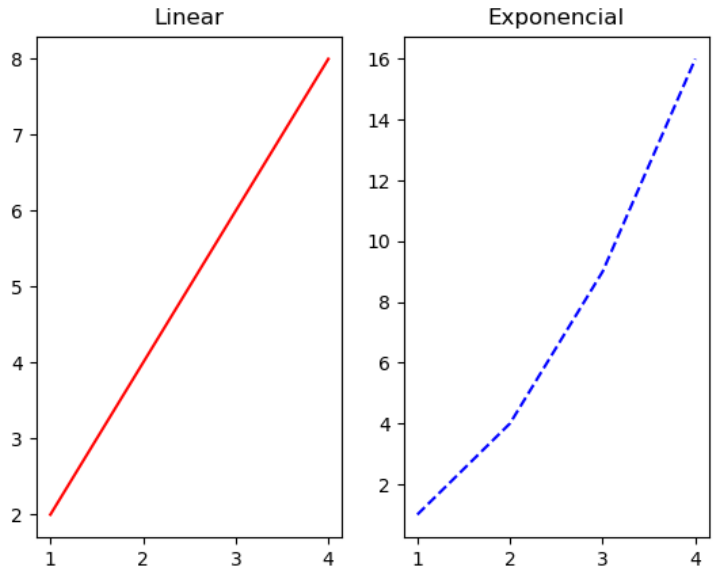
```
# criando valores no eixo x
x = np.array([1, 2, 3, 4])

# criando valores no eixo y
y = x*2

# novo eixo y
y2 = x*x

plt.xlabel('Valores de x')
plt.xlabel('Valores de y')

# plotando dois gráficos no mesmo plano
plt.plot(x, y, 'r-', x, y2, 'b--')
```



6.5. PLOTS MÚLTIPLOS DE FORMA SEPARADA

- Às vezes queremos plotar múltiplos gráficos de uma vez, mas de forma separada para uma visualização mais individualizada
- No Matplotlib isso pode ser feito por meio da função subplot():
- **EXEMPLOS:**

```
x = np.array([1, 2, 3, 4])
y = x*2
y2 = x*x

# Plotando dois gráficos separados
plt.subplot(1, 2, 1) #uma linha, duas colunas, posição 1
plt.title('Linear')
plt.plot(x, y, 'r-')

plt.subplot(1, 2, 2) #uma linha, duas colunas, posição 2
plt.title('Exponencial')
plt.plot(x, y2, 'b--')
```

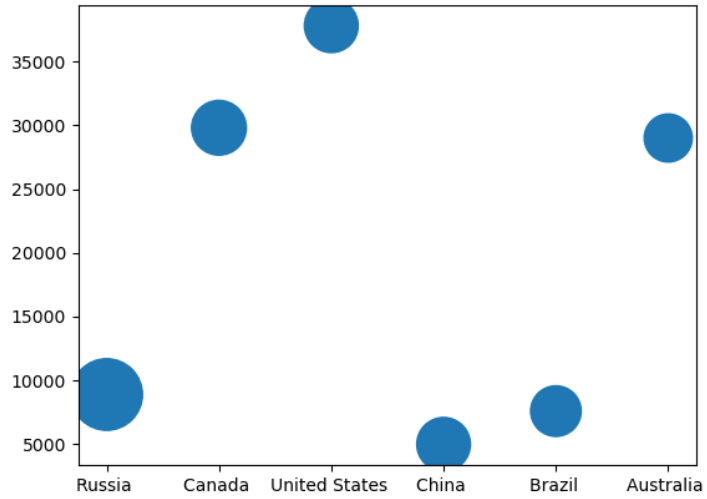


6.6. TIPOS DE GRÁFICOS POPULARES

- São inúmeros os tipos de gráficos e configurações disponíveis no Matplotlib para traçarmos nossos gráficos
- Um **Gráfico de Dispersão (Scatter Plot)** é eficiente para se detectar *outliers* e padrões, além de ser útil para se explorar a relação entre duas variáveis, analisar a força e direção dessa relação

- **EXEMPLOS:**

```
# lendo o dataset paises.csv
dfPaises = pd.read_csv('paises.csv', delimiter=';')

# extraindo somente dados dos 6 maiores países do mundo
dfPaises2 = dfPaises.nlargest(6, 'Area (sq. mi.)')

# plotando qual destes países possui a maior renda per capita
# observe que o tamanho de cada ponto ilustra o tamanho dos países
plt.scatter(dfPaises2['Country'], dfPaises2['GDP ($ per capita)'],
s=dfPaises2['Area (sq. mi.)']/10000)
```



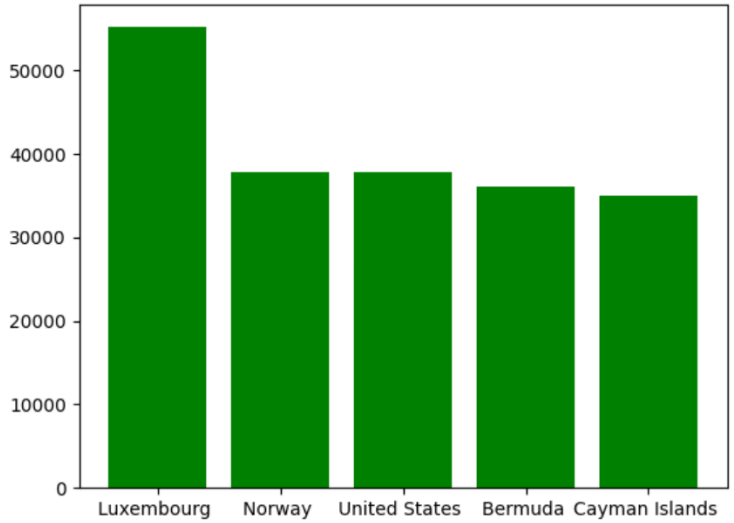
Inatel

6.6. TIPOS DE GRÁFICOS POPULARES

- Um **Gráfico em Barras (Bar Plot)** representa e compara dados quantitativos de forma visual, facilitando a análise e interpretação
- Ele permite que se identifique rapidamente tendências, padrões e diferenças entre os valores apresentados

- **EXEMPLOS:**

```
# extraindo os 5 países com maior PIB per capita (GDP) do dataset
dfBiggestGDP = dfPaises.nlargest(5, 'GDP ($ per capita)')

# pegando as Series dos países e os valores de GDP separadas
dfBiggestGDP _country = dfBiggestGDP['Country']
dfBiggestGDP _gdp = dfBiggestGDP['GDP ($ per capita)']

# traçando um gráfico em barras para ilustrar as diferenças
plt.bar(dfBiggestGDP _country, dfBiggestGDP_gdp, color='green')
```

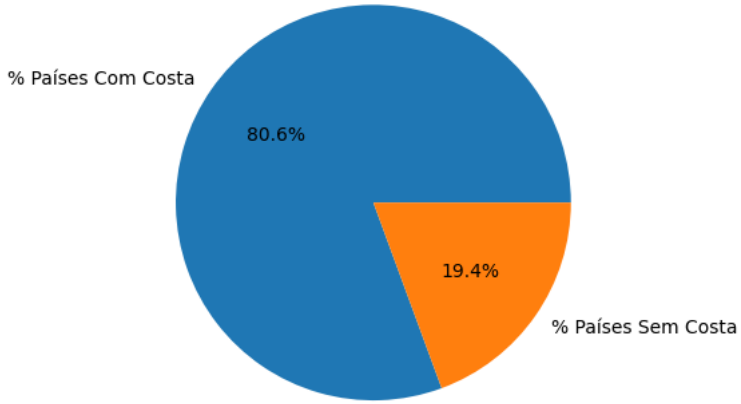


6.6. TIPOS DE GRÁFICOS POPULARES

- Um **Gráfico em Torta (Pie Plot)** é uma ferramenta eficaz para representar distribuições percentuais de maneira visual e intuitiva
- O tamanho de cada fatia corresponde à proporção de cada categoria no conjunto de dados

- **EXEMPLOS:**

```
#extraindo os países que não tem costa marítma
dfPaisesNoCoast = dfPaises[dfPaises['Coastline (coast/area ratio)'] == 0]

#extraindo a quantidade de países que não tem costa marítima
qtPaisesNoCoast = len(dfPaisesNoCoast)
#extraindo a quantidade de paises que tem costa marítma
qtPaisesCoast = len(dfPaises) - qtPaisesNoCoast

#traçando o gráfico em torta
plt.pie(x=[qtPaisesCoast, qtPaisesNoCoast], labels=['% Países Com Costa',
'% Países Sem Costa'], autopct='%1.1f%%')
```



Inatel

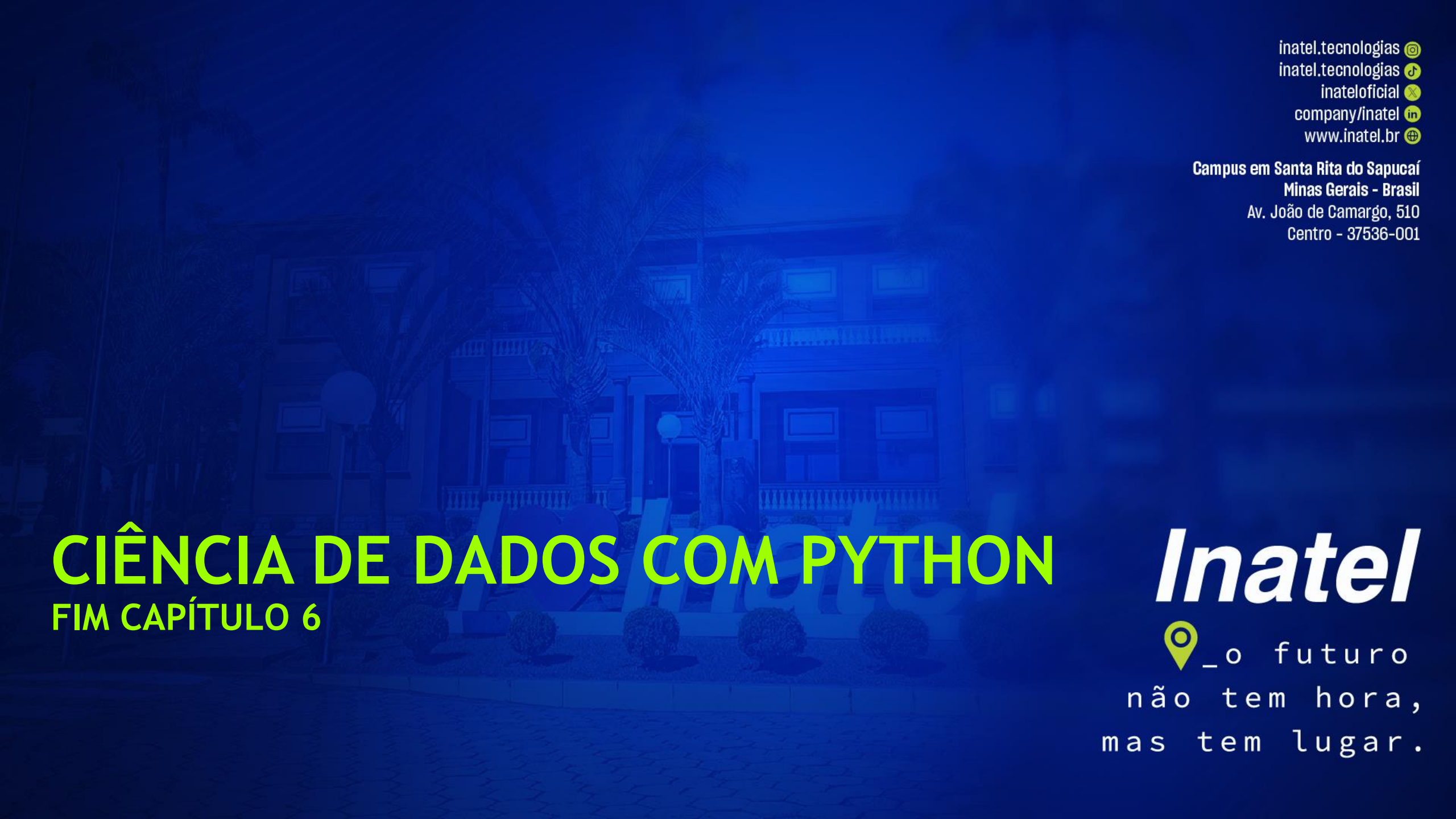
EXERCÍCIOS

1. Por meio do *dataset* paises.csv, trace dois gráficos de linhas em um mesmo plano cartesiano, um mostrando a taxa de mortalidade (*Deathrate*) e outro a taxa de natalidade (*Birthrate*) dos países da América do Norte;

2. Por meio do *dataset* space.csv, trace um gráfico em barras mostrando quantas empresas espaciais diferentes os EUA e a CHINA possuem;

Dica: não se esqueça de retirar os resultados repetidos

3. Por meio do *dataset space.csv,* trace um gráfico em torta ilustrando a porcentagem de missões da empresa Roscosmos que deram certo e que deram errado;



inatel.tecnologias
inatel.tecnologias
inateloficial
company/inatel
www.inatel.br

Campus em Santa Rita do Sapucaí
Minas Gerais - Brasil
Av. João de Camargo, 510
Centro - 37536-001

CIÊNCIA DE DADOS COM PYTHON

FIM CAPÍTULO 6

Inatel

_o futuro
não tem hora,
mas tem lugar.