Inatel
Instituto Nacional de Telecomunicações

Introdução à Análise de Dados

Cap.6 - Visualização de Dados com Matplotlib

Prof. MSc. Renzo P. Mesquita

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.

Capítulo 6
Visualização de Dados com Matplotlib

6.1. Introdução;
6.2. Plotando gráficos com plot;
6.3. Formatando o estilo dos plots;
6.4. Plots múltiplos de forma conjunta;
6.5. Plots múltiplos de forma separada;
6.6. Tipos de Plots populares;

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.

Alguns destaques desta biblioteca:

- Permite a criação de plots valiosos com poucas linhas de código;
- Nos oferece controle sobre todos os aspectos/detalhes da plotagem de um gráfico;
- É uma biblioteca rica em diferentes tipos de gráficos;
- Biblioteca base para outras bibliotecas de gráficos no Python;
- Possui boa integração com outras libs como NumPy e Pandas;
- Foi projetada para fornecer recursos que lembram muito as facilidades providas pelo MatLab;

matplotlib

e muito mais..

6.1. Introdução

Apesar de ser uma biblioteca muito popular para Visualização de Dados em Python, é necessário inicialmente instalá-la.

Como já vimos anteriormente, a instalação por meio do PyCharm é bem simples.

- Basta irmos em File -> Settings;
- No nome do seu projeto, vá em Project Interpreter -> clique em + (Install);
- Procure por matplotlib e aperte o botão Install Package;

Uma vez instalada, basta importá-la no projeto:

Ex:

```
import matplotlib.pyplot as plt
```

Obs: Abreviamos matplotlib.pyplot para simplesmente "plt" para pouparmos tempo e também seguirmos um padrão comumente usado por outros desenvolvedores que fazem uso desta biblioteca.

+

matplotlib

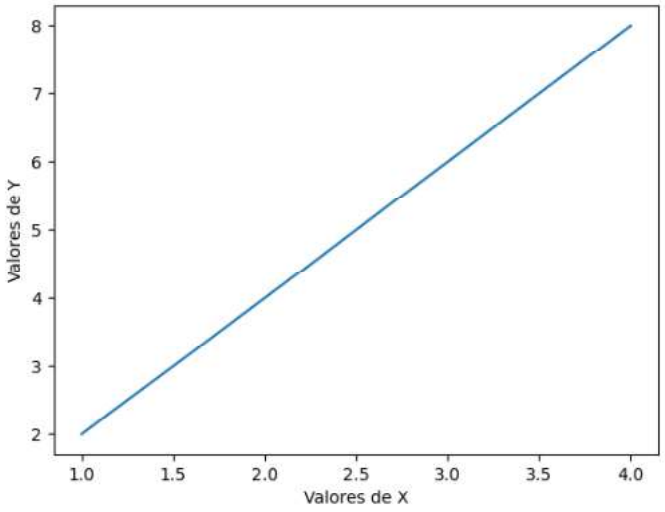
6.2. Plotando Gráficos com plot

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.

Para começarmos a compreender o seu princípio de funcionamento, vamos fazer um exemplo simples:

Ex:

```
# 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)
# Mostrando o plot
plt.show()
```



6.3. Formatando o estilo dos plots

Existe um terceiro argumento customizado que podemos passar a 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

character	description
'.'	point marker
','	pixel marker
'o'	circle marker
'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

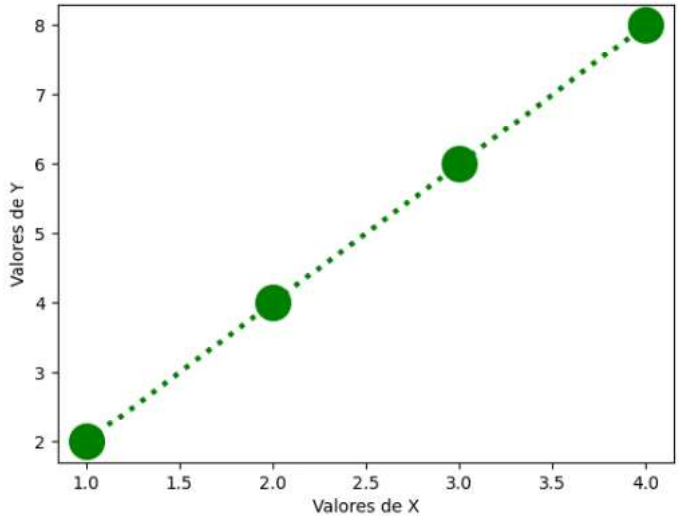
Vamos ver como este procedimento funciona?

6.3. Formatando o estilo dos plots

Com a String customizada podemos facilmente alterar o estilo dos nossos plots.

Ex:

```
# 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)
# Mostrando o plot
plt.show()
```



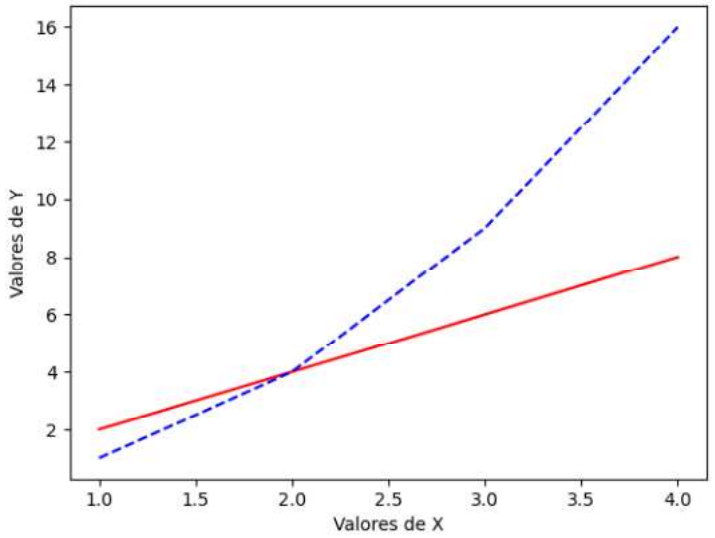
6.4. Plots múltiplos de forma conjunta

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 strings de customização no seguinte formato:

Ex:

```
x = np.array([1, 2, 3, 4])
y = x*2
# Novo valor de y
y2 = x*x
plt.xlabel('Valores de X')
plt.ylabel('Valores de Y')
# Plotando dois gráficos no mesmo plano
plt.plot(x, y, 'r-', x, y2, 'b--')
plt.show()
```



No exemplo temos uma função linear e outra exponencial no mesmo gráfico.

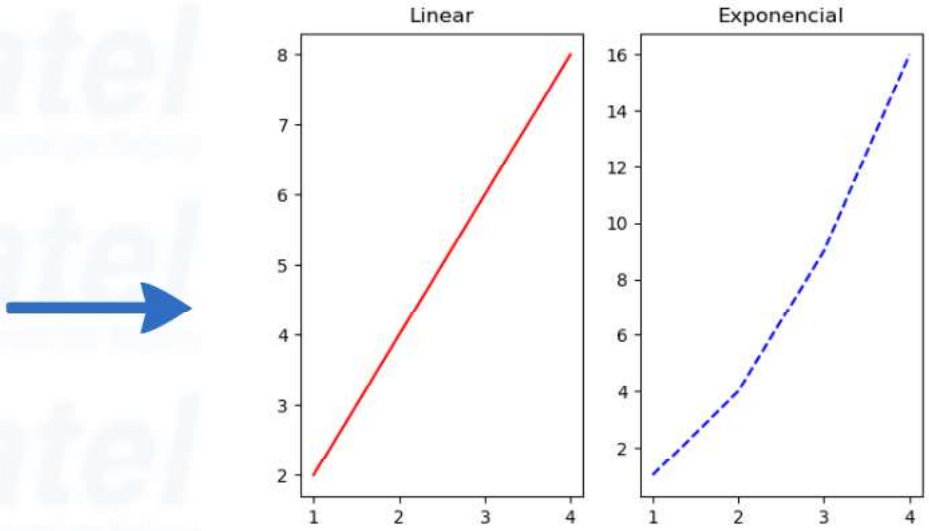
6.5. Plots múltiplos de forma separada

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():

Ex:

```
x = np.array([1, 2, 3, 4])
y = x*2
y2 = x*x
# Plotando dois gráficos separados
plt.subplot(1, 2, 1)
plt.title('Linear')
plt.plot(x, y, 'r-')
plt.subplot(1, 2, 2)
plt.title('Exponencial')
plt.plot(x, y2, 'b--')
plt.show()
```



- No exemplo, o primeiro argumento significa o número de linhas e o segundo o número de colunas que queremos usar para gerar subplots;
- O terceiro argumento é utilizado para indicarmos qual subplot vai receber uma configuração específica que virá logo abaixo.

6.6. Tipos de Plots populares

São inúmeros os tipos de gráficos e configurações disponíveis no Matplotlib para traçarmos nossos gráficos.

Vamos dar uma olhada rápida por meio de exemplos usando 3 formatos populares:

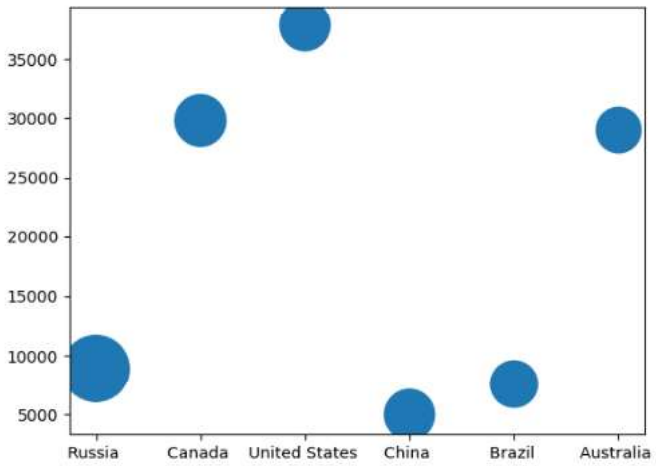
- *SCATTER PLOT (Gráfico de Dispersão)*

Ex:

```
# 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 destes países
plt.scatter(dfPaises2['Country'],dfPaises2['GDP ($ per capita)'],
            s=dfPaises2['Area (sq. mi.)']/10000)
plt.show()
```

O Exemplo ilustra a renda per capita dos 6 maiores países do mundo segundo este dataset.

O Parâmetro s define o tamanho dos pontos baseado em um critério numérico.



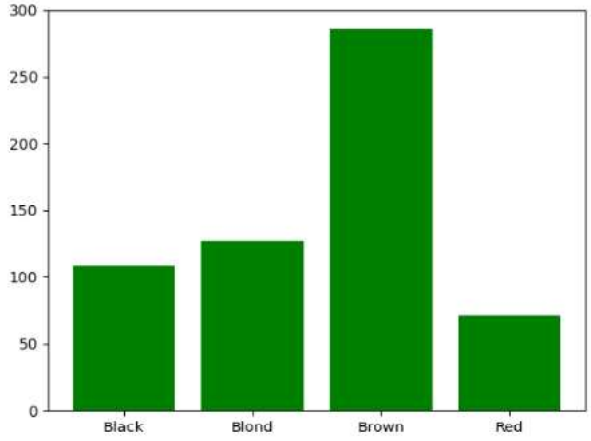
6.6. Tipos de Plots populares

- *BAR PLOT (Gráfico em Barras)*

Ex:

```
import pydataset as pds
# Carregando o dataset HairEyeColor
df = pd.DataFrame(pds.data('HairEyeColor'))
# Agrupando os dados pela cor do cabelo e somando
grupoCorCabelo = df.groupby('Hair').sum()
# Traçando o gráfico em barras
# Nomes das cores no eixo x
# Quantidades no eixo y
plt.bar(grupoCorCabelo.index.values, grupoCorCabelo['Freq'],
        color='green')
plt.show()
```

O Exemplo ilustra a quantidade de alunos que possuem cada cor de cabelo dentro deste dataset.



6.6. Tipos de Plots populares

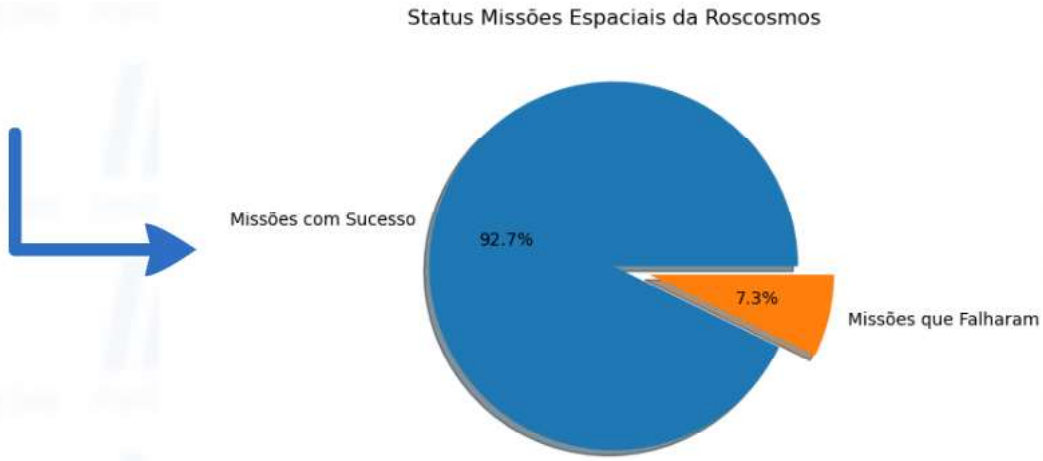
- *PIE PLOT (Gráfico em Torta)*

Ex:

```
# Lendo o dataset space.csv
dfSpace = pd.read_csv('space.csv',delimiter=';')
# Extraindo somente dados da Agência Roscosmos
dfRosc = dfSpace[dfSpace['Company Name'].str.contains('Roscosmos')]
# Contando quantas missões desta agência deram errado
falha = len(dfRosc[dfRosc['Status Mission'].str.contains('Failure')])
# Contando quantas missões desta agência deram certo
sucesso = len(dfRosc[dfRosc['Status Mission'].str.contains('Success')])
# Plotando um gráfico em torta ilustrando o status das missões
plt.pie(x=[sucesso, falha], labels=['Missões com Sucesso', 'Missões que Falharam'],
        shadow=True, explode=[0, 0.2], autopct='%1.1f%%')
plt.title('Status Missões Espaciais da Roscosmos')
plt.show()
```

O Exemplo ilustra a quantidade de missões da Roscosmos que deram certo e que deram errado.

O Parâmetro x recebe os dados, shadow coloca sombra no gráfico, o explode permite destacar partes do mesmo e autopct calcula a porcentagem de valores no gráfico automaticamente.



6.6. Tipos de Plots populares

Exercícios Propostos

Baseado nos fundamentos de Matplotlib e outros conceitos importantes que vimos até o momento, crie scripts em Python que resolvam os seguintes problemas:

1. Por meio do Dataset space.csv, trace um gráfico em barras mostrando quantas empresas Espaciais os EUA e a CHINA possuem;

Dica: após realizar os devidos condicionais no dataset para cada um dos países, utilize do método unique() para retirar resultados repetidos.

2. 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 America do Norte (NORTHERN AMERICA);

Dica: após realizar o condicional no dataset para pegar os países da América do Norte, você já poderá chamar o método plot, passando como argumento para ele slicings sobre o dataset resultante do condicional.

FIM
DO
CAPÍTULO 6