

Plotting Graph In Python

Data Visualizations:

In this practical you will learn how to build the most common graphs used in data analytics Graphs.

I.Scatterplot.

II.Histogram.

III.Density plot.

IV.Bar Chart.

V.Pie Chart.

Using in build mtcars datasets ¶

```
In [1]: 1 import os
        2 print(os.getcwd()) # Hiển thị thư mục hiện tại
        3
```

C:\Users\Vhho-PC

```
In [2]: 1 os.chdir('D:/Test_Python/Day_6') # Thay đổi 'path/to/your/directory' bằng đường dẫn bạn muốn
        2
```

```
In [3]: 1 ## 1. Cài đặt Pandas (nếu chưa cài):
        2 ## pip install pandas
        3
        4 import pandas as pd
        5
```

```
In [4]: 1 # 2.Đọc file mtcars.csv
        2 df = pd.read_csv("mtcars.csv")
        3
        4 # Hiển thị 5 dòng đầu tiên
        5 print(df.head())
        6
```

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
0	21.0	6	160.0	110	3.90	2.620	16.46	0	1	4	4
1	21.0	6	160.0	110	3.90	2.875	17.02	0	1	4	4
2	22.8	4	108.0	93	3.85	2.320	18.61	1	1	4	1
3	21.4	6	258.0	110	3.08	3.215	19.44	1	0	3	1
4	18.7	8	360.0	175	3.15	3.440	17.02	0	0	3	2

```
In [5]: 1 ## 3.Phân tích dữ liệu
2 ## - Mô tả dữ liệu:
3 print(df.describe())
4
```

	mpg	cyl	disp	hp	drat	wt \
count	32.000000	32.000000	32.000000	32.000000	32.000000	32.000000
mean	20.090625	6.187500	230.721875	146.687500	3.596563	3.217250
std	6.026948	1.785922	123.938694	68.562868	0.534679	0.978457
min	10.400000	4.000000	71.100000	52.000000	2.760000	1.513000
25%	15.425000	4.000000	120.825000	96.500000	3.080000	2.581250
50%	19.200000	6.000000	196.300000	123.000000	3.695000	3.325000
75%	22.800000	8.000000	326.000000	180.000000	3.920000	3.610000
max	33.900000	8.000000	472.000000	335.000000	4.930000	5.424000

	qsec	vs	am	gear	carb
count	32.000000	32.000000	32.000000	32.000000	32.000000
mean	17.848750	0.437500	0.406250	3.687500	2.8125
std	1.786943	0.504016	0.498991	0.737804	1.6152
min	14.500000	0.000000	0.000000	3.000000	1.0000
25%	16.892500	0.000000	0.000000	3.000000	2.0000
50%	17.710000	0.000000	0.000000	4.000000	2.0000
75%	18.900000	1.000000	1.000000	4.000000	4.0000
max	22.900000	1.000000	1.000000	5.000000	8.0000

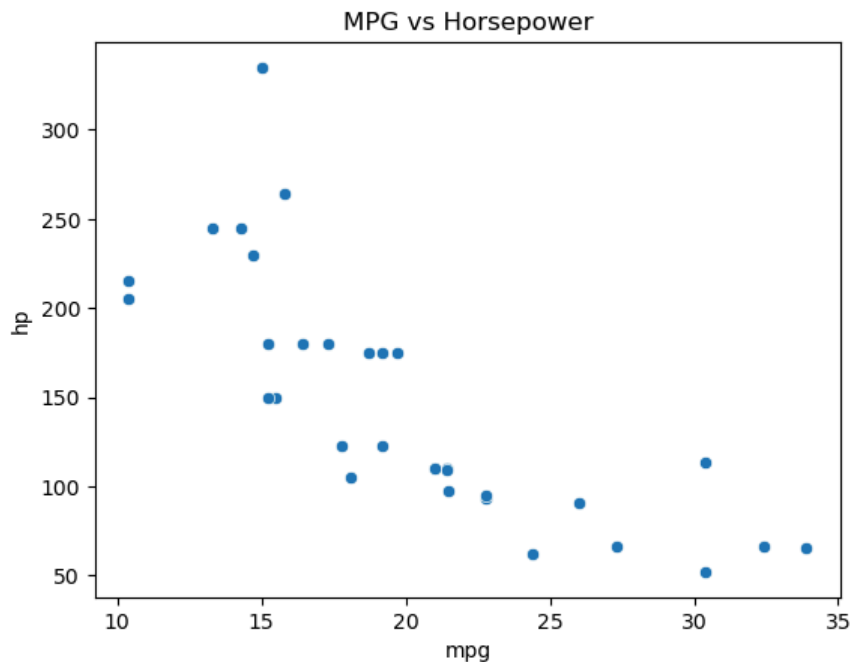
```
In [6]: 1 ## - Lọc dữ liệu:
2 filtered_df = df[df['mpg'] > 20]
3 print(filtered_df)
4
```

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
0	21.0	6	160.0	110	3.90	2.620	16.46	0	1	4	4
1	21.0	6	160.0	110	3.90	2.875	17.02	0	1	4	4
2	22.8	4	108.0	93	3.85	2.320	18.61	1	1	4	1
3	21.4	6	258.0	110	3.08	3.215	19.44	1	0	3	1
7	24.4	4	146.7	62	3.69	3.190	20.00	1	0	4	2
8	22.8	4	140.8	95	3.92	3.150	22.90	1	0	4	2
17	32.4	4	78.7	66	4.08	2.200	19.47	1	1	4	1
18	30.4	4	75.7	52	4.93	1.615	18.52	1	1	4	2
19	33.9	4	71.1	65	4.22	1.835	19.90	1	1	4	1
20	21.5	4	120.1	97	3.70	2.465	20.01	1	0	3	1
25	27.3	4	79.0	66	4.08	1.935	18.90	1	1	4	1
26	26.0	4	120.3	91	4.43	2.140	16.70	0	1	5	2
27	30.4	4	95.1	113	3.77	1.513	16.90	1	1	5	2
31	21.4	4	121.0	109	4.11	2.780	18.60	1	1	4	2

I.Scatterplot

Plot hp against milage

```
In [8]: 1 ## - Vẽ biểu đồ (sử dụng Matplotlib hoặc Seaborn):
2 ## pip install matplotlib
3 ## pip install
4 import matplotlib.pyplot as plt
5 import seaborn as sns
6
7 sns.scatterplot(data=df, x='mpg', y='hp')
8 plt.title("MPG vs Horsepower")
9 plt.show()
```



Is your Graph Salty

S = Scale (e.g., Scale of Axes: 10, 15, ..., 35 and 50, 100, ..., 300)

- Horizontal and vertical scale selected
- Independent variable in the horizontal axis • Dependent variable in the vertical axis

A = Axes (e.g., X Axis and Y Axis: x and y)

- Each axis is shown clearly
- Each axis scale increment evenly
- Each axis scale may or may not start from zero

L = Labels (e.g., Label Axis: x = "mpg", y = "hp")

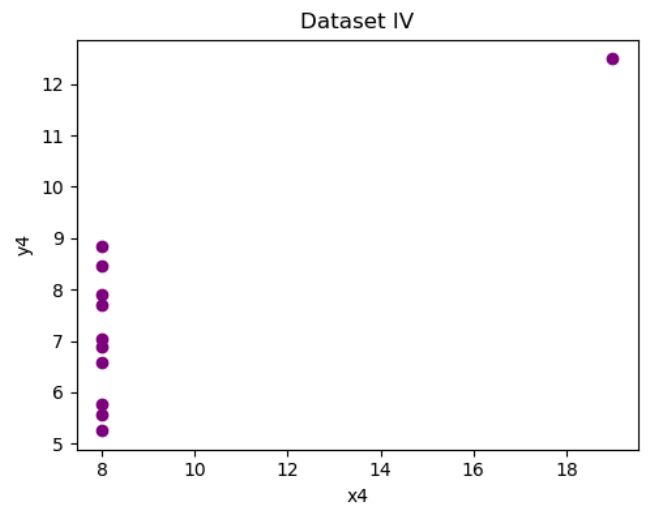
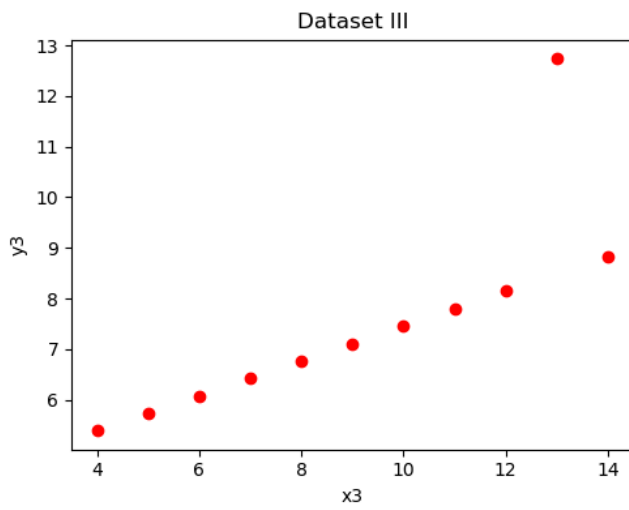
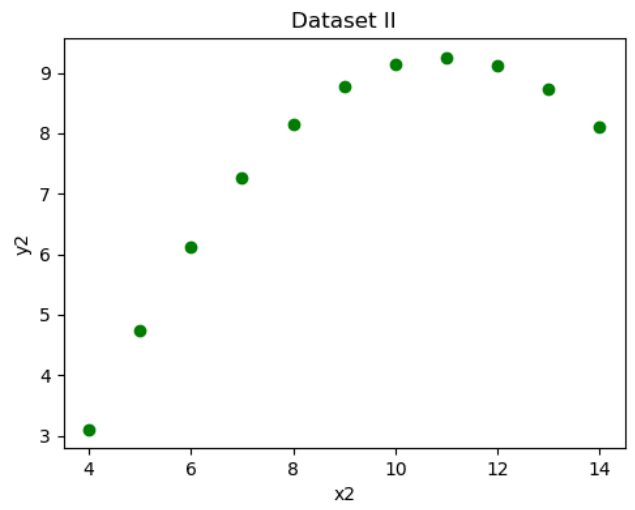
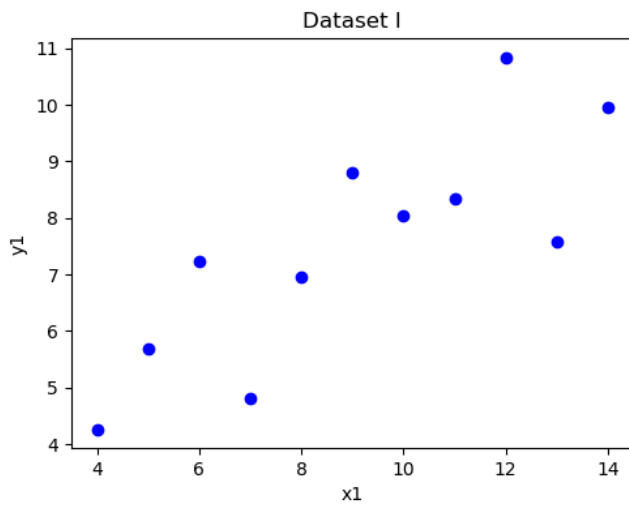
- Each axis identifies the data and units clearly

T = Title (e.g., Title: "MPG vs Horsepower")

- The title should be written clearly usually at the top • The title should identify the purpose of the graph

Create 2x2 Graph

```
In [10]: 1  ## The code to create 2 by 2 graph is
2
3  import matplotlib.pyplot as plt
4  import seaborn as sns
5  import pandas as pd
6  from sklearn.datasets import fetch_openml
7
8  # Tải dataset Anscombe (nếu chưa có)
9  anscombe = sns.load_dataset('anscombe')
10
11 # Lọc dữ liệu theo từng nhóm
12 group1 = anscombe[anscombe['dataset'] == 'I']
13 group2 = anscombe[anscombe['dataset'] == 'II']
14 group3 = anscombe[anscombe['dataset'] == 'III']
15 group4 = anscombe[anscombe['dataset'] == 'IV']
16
17 # Tạo lưới đồ thị 2x2
18 plt.figure(figsize=(10, 8))
19
20 # Đồ thị 1
21 plt.subplot(2, 2, 1)
22 plt.scatter(group1['x'], group1['y'], color='blue')
23 plt.title('Dataset I')
24 plt.xlabel('x1')
25 plt.ylabel('y1')
26
27 # Đồ thị 2
28 plt.subplot(2, 2, 2)
29 plt.scatter(group2['x'], group2['y'], color='green')
30 plt.title('Dataset II')
31 plt.xlabel('x2')
32 plt.ylabel('y2')
33
34 # Đồ thị 3
35 plt.subplot(2, 2, 3)
36 plt.scatter(group3['x'], group3['y'], color='red')
37 plt.title('Dataset III')
38 plt.xlabel('x3')
39 plt.ylabel('y3')
40
41 # Đồ thị 4
42 plt.subplot(2, 2, 4)
43 plt.scatter(group4['x'], group4['y'], color='purple')
44 plt.title('Dataset IV')
45 plt.xlabel('x4')
46 plt.ylabel('y4')
47
48 # Hiển thị đồ thị
49 plt.tight_layout()
50 plt.show()
51
```

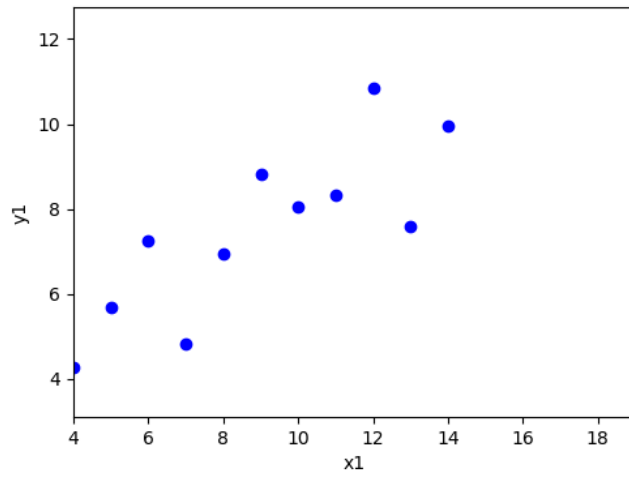


Create 2x2 Graph with xmin, xmax, ymin, ymax

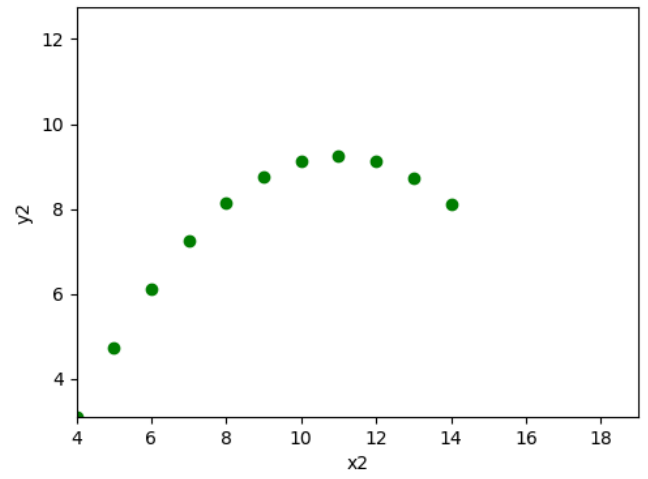
In [11]:

```
1 #Execute this code
2
3 import matplotlib.pyplot as plt
4 import seaborn as sns
5 import pandas as pd
6
7 # Tải dataset Anscombe
8 anscombe = sns.load_dataset('anscombe')
9
10 # Lọc dữ liệu theo từng nhóm
11 group1 = anscombe[anscombe['dataset'] == 'I']
12 group2 = anscombe[anscombe['dataset'] == 'II']
13 group3 = anscombe[anscombe['dataset'] == 'III']
14 group4 = anscombe[anscombe['dataset'] == 'IV']
15
16 # Tính xmin, xmax, ymin, ymax
17 xmin = min(group1['x'].min(), group2['x'].min(), group3['x'].min(), group4['x'].min())
18 xmax = max(group1['x'].max(), group2['x'].max(), group3['x'].max(), group4['x'].max())
19 ymin = min(group1['y'].min(), group2['y'].min(), group3['y'].min(), group4['y'].min())
20 ymax = max(group1['y'].max(), group2['y'].max(), group3['y'].max(), group4['y'].max())
21
22 # Tạo lưới đồ thị 2x2
23 plt.figure(figsize=(10, 8))
24
25 # Đồ thị 1
26 plt.subplot(2, 2, 1)
27 plt.scatter(group1['x'], group1['y'], color='blue')
28 plt.title('Dataset I')
29 plt.xlabel('x1')
30 plt.ylabel('y1')
31 plt.xlim(xmin, xmax)
32 plt.ylim(ymin, ymax)
33
34 # Đồ thị 2
35 plt.subplot(2, 2, 2)
36 plt.scatter(group2['x'], group2['y'], color='green')
37 plt.title('Dataset II')
38 plt.xlabel('x2')
39 plt.ylabel('y2')
40 plt.xlim(xmin, xmax)
41 plt.ylim(ymin, ymax)
42
43 # Đồ thị 3
44 plt.subplot(2, 2, 3)
45 plt.scatter(group3['x'], group3['y'], color='red')
46 plt.title('Dataset III')
47 plt.xlabel('x3')
48 plt.ylabel('y3')
49 plt.xlim(xmin, xmax)
50 plt.ylim(ymin, ymax)
51
52 # Đồ thị 4
53 plt.subplot(2, 2, 4)
54 plt.scatter(group4['x'], group4['y'], color='purple')
55 plt.title('Dataset IV')
56 plt.xlabel('x4')
57 plt.ylabel('y4')
58 plt.xlim(xmin, xmax)
59 plt.ylim(ymin, ymax)
60
61 # Hiển thị đồ thị
62 plt.tight_layout()
63 plt.show()
64
```

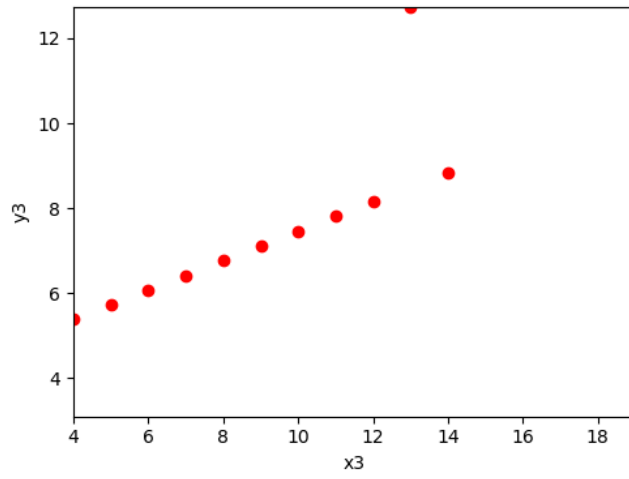
Dataset I



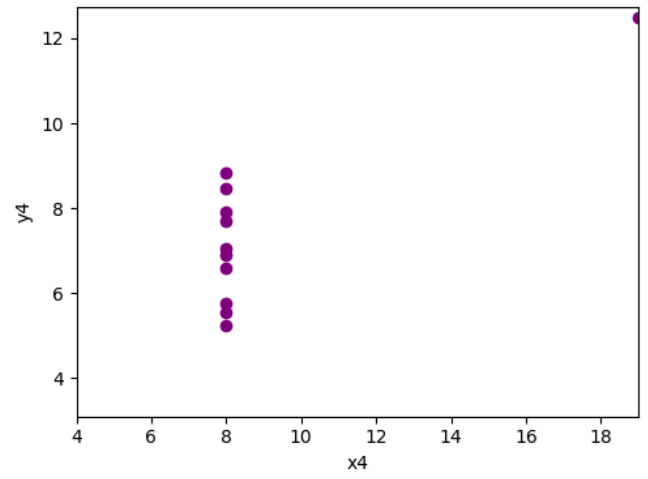
Dataset II



Dataset III



Dataset IV

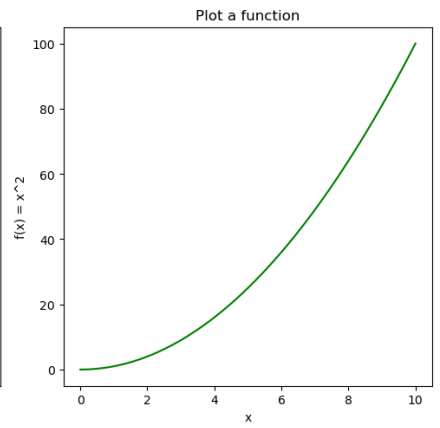
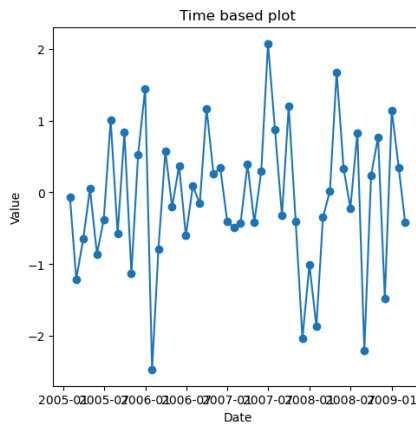
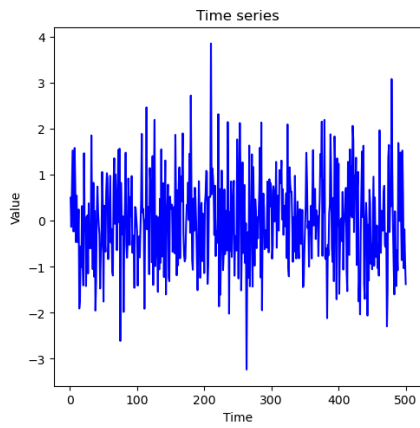
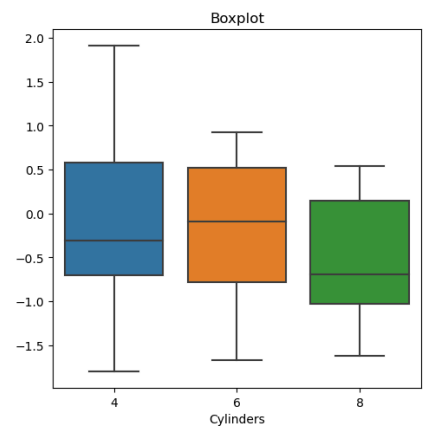
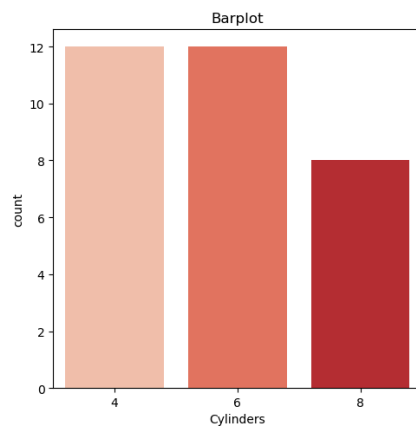
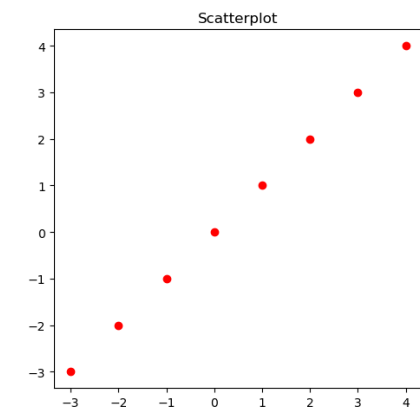


Plot 2x3 Graphs:

1. Scatterplot
2. Barplot
3. Boxplot
4. Time series plot
5. Time-based plot
6. Plot a function

In [12]:

```
1  # Run this code
2  # Examples
3
4  import numpy as np
5  import pandas as pd
6  import matplotlib.pyplot as plt
7  import seaborn as sns
8
9  # Data
10 np.random.seed(42) # Đảm bảo kết quả ngẫu nhiên tái hiện được
11 x = np.array([-3, -2, -1, 0, 1, 2, 3, 4])
12 y = np.array([-3, -2, -1, 0, 1, 2, 3, 4])
13 my_ts = np.random.normal(size=500) # Time series data
14 my_dates = pd.date_range(start="2005-01-01", periods=50, freq="M")
15 my_factor = pd.Series([6, 4, 8, 4, 6, 8, 4, 6] * 4, name="Cylinders") # Tương tự mtcars$cyl
16
17 # Tạo lưới đồ thị 2x3
18 fig, axs = plt.subplots(2, 3, figsize=(15, 10))
19
20 # 1. Scatterplot
21 axs[0, 0].scatter(x, y, color='red')
22 axs[0, 0].set_title("Scatterplot")
23
24 # 2. Barplot
25 sns.countplot(x=my_factor, ax=axs[0, 1], palette="Reds")
26 axs[0, 1].set_title("Barplot")
27
28 # 3. Boxplot
29 sns.boxplot(x=my_factor, y=np.random.normal(size=len(my_factor)), ax=axs[0, 2])
30 axs[0, 2].set_title("Boxplot")
31
32 # 4. Time series plot
33 axs[1, 0].plot(np.arange(1, 501), my_ts, color='blue')
34 axs[1, 0].set_title("Time series")
35 axs[1, 0].set_xlabel("Time")
36 axs[1, 0].set_ylabel("Value")
37
38 # 5. Time-based plot
39 axs[1, 1].plot(my_dates, np.random.normal(size=50), marker='o')
40 axs[1, 1].set_title("Time based plot")
41 axs[1, 1].set_xlabel("Date")
42 axs[1, 1].set_ylabel("Value")
43
44 # 6. Plot a function
45 x_fun = np.linspace(0, 10, 100)
46 y_fun = x_fun**2
47 axs[1, 2].plot(x_fun, y_fun, color='green')
48 axs[1, 2].set_title("Plot a function")
49 axs[1, 2].set_xlabel("x")
50 axs[1, 2].set_ylabel("f(x) = x^2")
51
52 # Hiển thị đồ thị
53 plt.tight_layout()
54 plt.show()
55
```



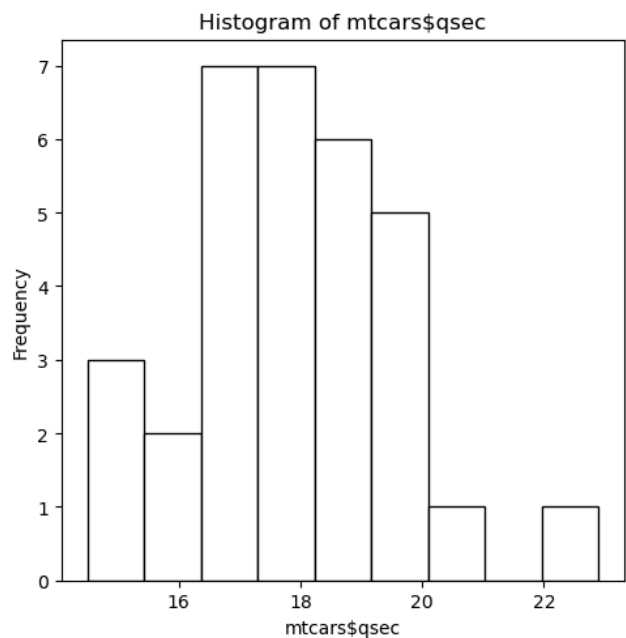
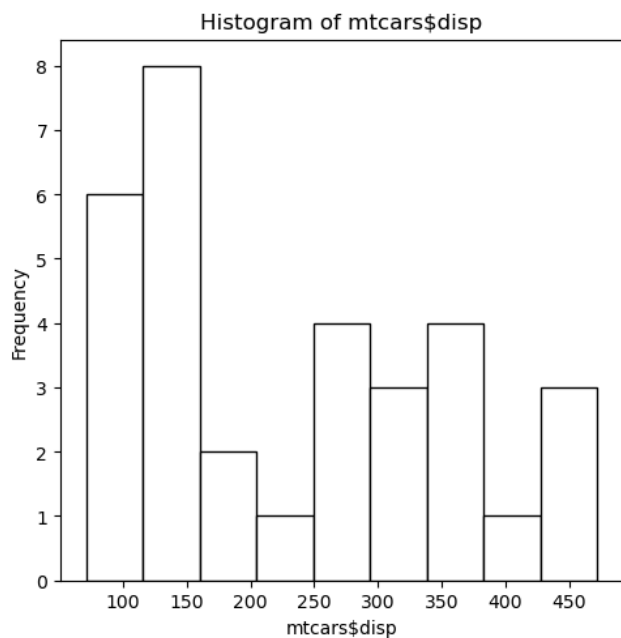
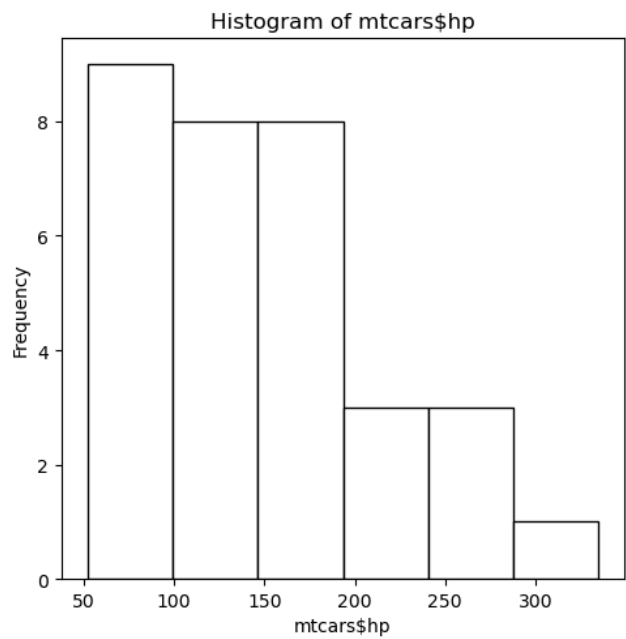
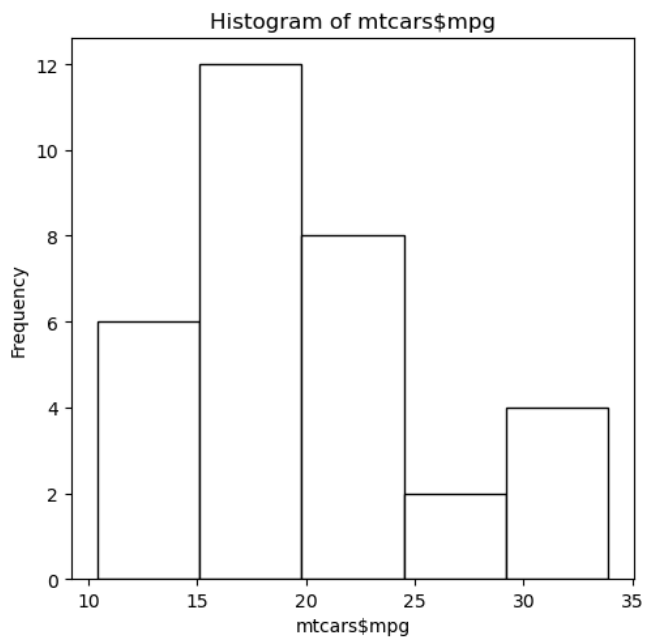
II. Histogram

The Histogram is a **bar plot** used to represent **continuous data**.

It shows the **frequencies** in the **vertical axis** and the **continuous data** on the **horizontal axis**.

Using mtcars datasets

```
In [14]: 1 import pandas as pd
2 import matplotlib.pyplot as plt
3
4 # Đọc dữ liệu mtcars từ file CSV
5 mtcars = pd.read_csv('mtcars.csv')
6
7 # Tạo bố cục 2 hàng, 2 cột
8 fig, axes = plt.subplots(2, 2, figsize=(10, 10))
9
10 # Biểu đồ histogram 1: mpg (5 bins)
11 axes[0, 0].hist(mtcars['mpg'], bins=5, color='white', edgecolor='black')
12 axes[0, 0].set_title('Histogram of mtcars$mpg', fontsize=12)
13 axes[0, 0].set_xlabel('mtcars$mpg', fontsize=10)
14 axes[0, 0].set_ylabel('Frequency', fontsize=10)
15
16 # Biểu đồ histogram 2: hp (6 bins)
17 axes[0, 1].hist(mtcars['hp'], bins=6, color='white', edgecolor='black')
18 axes[0, 1].set_title('Histogram of mtcars$hp', fontsize=12)
19 axes[0, 1].set_xlabel('mtcars$hp', fontsize=10)
20 axes[0, 1].set_ylabel('Frequency', fontsize=10)
21
22 # Biểu đồ histogram 3: disp (9 bins)
23 axes[1, 0].hist(mtcars['disp'], bins=9, color='white', edgecolor='black')
24 axes[1, 0].set_title('Histogram of mtcars$disp', fontsize=12)
25 axes[1, 0].set_xlabel('mtcars$disp', fontsize=10)
26 axes[1, 0].set_ylabel('Frequency', fontsize=10)
27
28 # Biểu đồ histogram 4: qsec (9 bins)
29 axes[1, 1].hist(mtcars['qsec'], bins=9, color='white', edgecolor='black')
30 axes[1, 1].set_title('Histogram of mtcars$qsec', fontsize=12)
31 axes[1, 1].set_xlabel('mtcars$qsec', fontsize=10)
32 axes[1, 1].set_ylabel('Frequency', fontsize=10)
33
34 # Tự động điều chỉnh khoảng cách giữa các biểu đồ
35 plt.tight_layout()
36
37 # Hiển thị biểu đồ
38 plt.show()
39
```

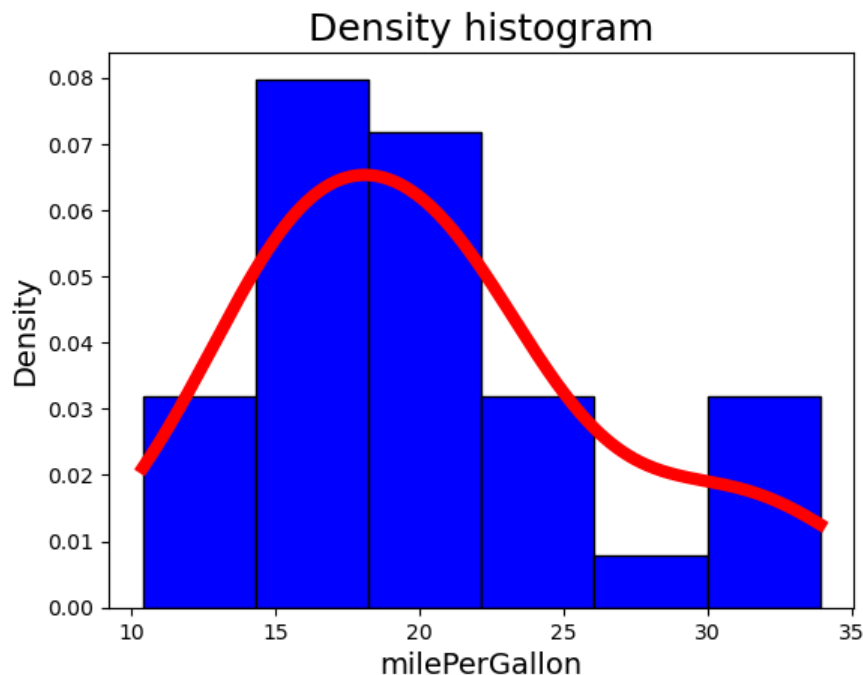


III.Density plot

A Density Plot is a data visualization tool used to estimate the probability density function (PDF) of a continuous variable.

It shows the distribution of data and is smoother than a histogram, making it useful for identifying patterns, trends, and comparing distributions between datasets.

```
In [18]: 1 import pandas as pd
2 import numpy as np
3 import matplotlib.pyplot as plt
4 from scipy.stats import gaussian_kde
5
6 # Đọc dữ liệu mtcars từ file CSV
7 mtcars = pd.read_csv('mtcars.csv')
8
9 # Lấy cột 'mpg' (mile per gallon)
10 mile_per_gallon = mtcars['mpg']
11
12 # Vẽ histogram với xác suất (density=True tương đương prob = TRUE trong R)
13 plt.hist(mile_per_gallon, bins=6, density=True, color='blue', alpha=1.0, edgecolor='black', label='Histogram')
14
15 # Tính density và vẽ đường density
16 density = gaussian_kde(mile_per_gallon)
17 x_vals = np.linspace(mile_per_gallon.min(), mile_per_gallon.max(), 1000)
18 plt.plot(x_vals, density(x_vals), color='red', linewidth=6, label='Density')
19
20 # Thêm tiêu đề và nhãn trục
21 plt.title("Density histogram", fontsize=18)
22 plt.xlabel("milePerGallon", fontsize=14)
23 plt.ylabel("Density", fontsize=14)
24
25 # Hiển thị biểu đồ
26 plt.show()
27
```



Density Plot using mtcars datasets

The probability density function of a variable describes

the probability of the variable taking certain value within the vector space of the variable.

In [16]:

```
1 # 2.Đọc file mtcars.csv
2 df = pd.read_csv("mtcars.csv")
3
4 # Hiển thị 5 dòng đầu tiên
5 print(df.head())
```

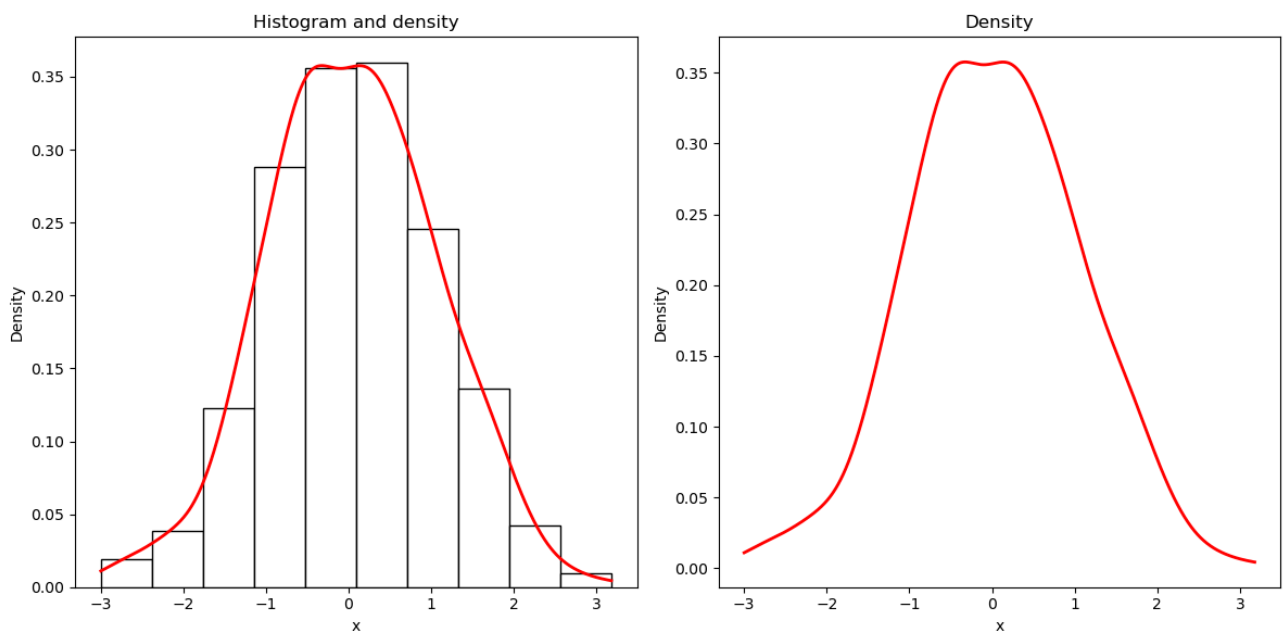
	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
0	21.0	6	160.0	110	3.90	2.620	16.46	0	1	4	4
1	21.0	6	160.0	110	3.90	2.875	17.02	0	1	4	4
2	22.8	4	108.0	93	3.85	2.320	18.61	1	1	4	1
3	21.4	6	258.0	110	3.08	3.215	19.44	1	0	3	1
4	18.7	8	360.0	175	3.15	3.440	17.02	0	0	3	2

Multiple density curves in one plot

```

In [20]: 1 import numpy as np
2 import matplotlib.pyplot as plt
3 from scipy.stats import gaussian_kde
4
5 # Sinh dữ Liệu ngẫu nhiên (500 giá trị) giống như rnorm trong R
6 x = np.random.normal(size=500)
7
8 # Tạo Layout 1 hàng, 2 cột
9 fig, axes = plt.subplots(1, 2, figsize=(12, 6))
10
11 # Biểu đồ 1: Histogram và đường mật độ
12 axes[0].hist(x, bins=10, density=True, color='white', edgecolor='black') # Số cột = 10 để giống R
13 density = gaussian_kde(x)
14 x_vals = np.linspace(min(x), max(x), 1000)
15 axes[0].plot(x_vals, density(x_vals), color='red', linewidth=2)
16 axes[0].set_title("Histogram and density")
17 axes[0].set_xlabel("x")
18 axes[0].set_ylabel("Density")
19
20 # Biểu đồ 2: Chỉ vẽ đường mật độ
21 axes[1].plot(x_vals, density(x_vals), color='red', linewidth=2)
22 axes[1].set_title("Density")
23 axes[1].set_xlabel("x")
24 axes[1].set_ylabel("Density")
25
26 # Hiển thị biểu đồ
27 plt.tight_layout()
28 plt.show()
29

```

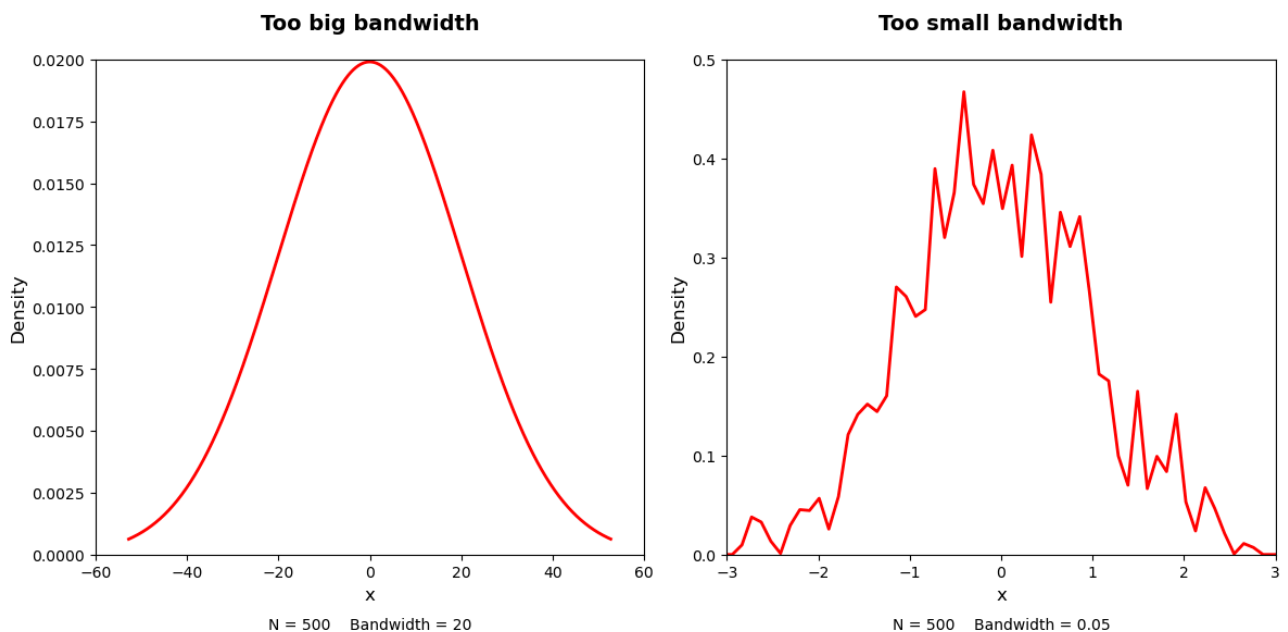


Vẽ density plot với hai băng thông (bandwidth) khác nhau sử dụng Matplotlib và Seaborn.

```

In [28]: 1 import pandas as pd
2 import numpy as np
3 import matplotlib.pyplot as plt
4 from scipy.stats import gaussian_kde
5
6 # Đọc dữ liệu mtcars từ file
7 mtcars = pd.read_csv('mtcars.csv')
8
9 # Tạo dữ liệu ngẫu nhiên giống R
10 np.random.seed(0)
11 x = np.random.normal(size=500)
12
13 # Tạo Layout 1 hàng, 2 cột
14 fig, axes = plt.subplots(1, 2, figsize=(12, 6))
15
16 # Đồ thị 1: Bandwidth Lớn
17 density_big_bw = gaussian_kde(x, bw_method=20 / np.std(x)) # bw = 20
18 x_vals = np.linspace(min(x) - 50, max(x) + 50, 1000)
19 axes[0].plot(x_vals, density_big_bw(x_vals), color='red', linewidth=2)
20 axes[0].set_title("Too big bandwidth", fontsize=14, fontweight='bold', pad=20)
21 axes[0].set_xlabel("x", fontsize=12)
22 axes[0].set_ylabel("Density", fontsize=12)
23 axes[0].set_xlim(-60, 60)
24 axes[0].set_ylim(0, 0.02)
25 axes[0].text(0.5, -0.15, "N = 500    Bandwidth = 20", ha='center', fontsize=10, transform=axes[0].transAxes)
26
27 # Đồ thị 2: Bandwidth nhỏ
28 density_small_bw = gaussian_kde(x, bw_method=0.05 / np.std(x)) # bw = 0.05
29 axes[1].plot(x_vals, density_small_bw(x_vals), color='red', linewidth=2)
30 axes[1].set_title("Too small bandwidth", fontsize=14, fontweight='bold', pad=20)
31 axes[1].set_xlabel("x", fontsize=12)
32 axes[1].set_ylabel("Density", fontsize=12)
33 axes[1].set_xlim(-3, 3)
34 axes[1].set_ylim(0, 0.5)
35 axes[1].text(0.5, -0.15, "N = 500    Bandwidth = 0.05", ha='center', fontsize=10, transform=axes[1].transAxes)
36
37 # Hiển thị đồ thị
38 plt.tight_layout()
39 plt.show()
40

```



In []:

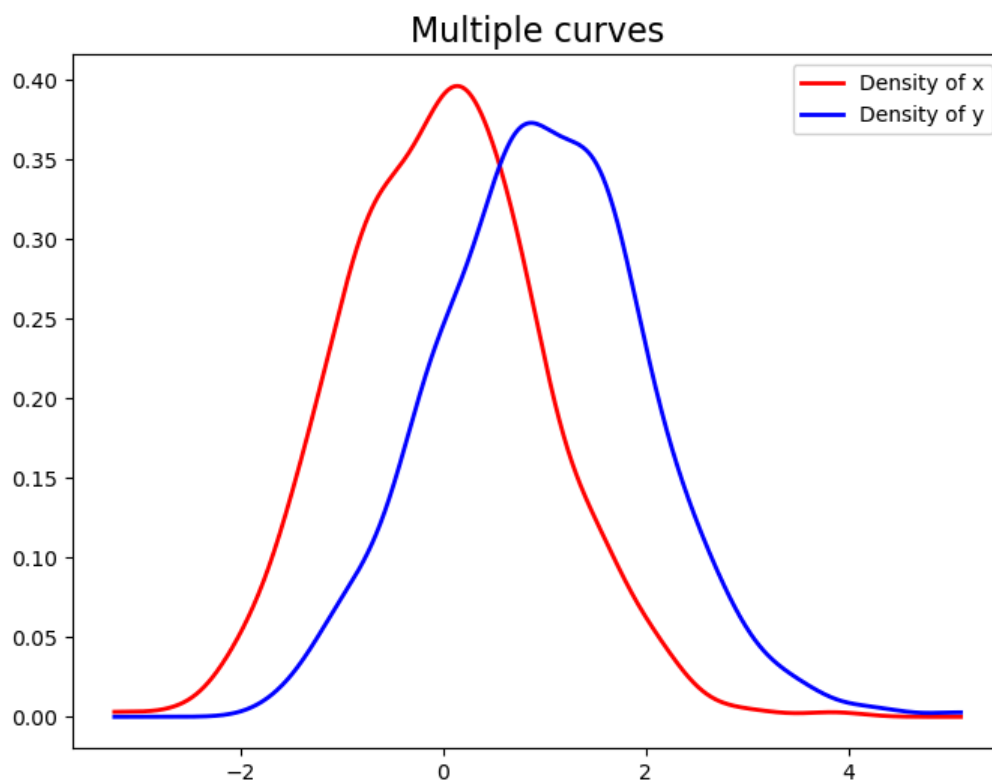
1

Vẽ nhiều đường mật độ (density curves) sử dụng Matplotlib và Scipy.


```

In [19]: 1 import numpy as np
2 import matplotlib.pyplot as plt
3 from scipy.stats import gaussian_kde
4
5 # Tạo dữ liệu x
6 np.random.seed(42)
7 x = np.random.normal(size=500)
8
9 # Tạo dữ liệu y
10 np.random.seed(2)
11 y = np.random.normal(size=500) + 1
12
13 # Tính mật độ cho x
14 density_x = gaussian_kde(x)
15
16 # Tính mật độ cho y
17 density_y = gaussian_kde(y)
18
19 # Tạo lưới giá trị x để vẽ mật độ
20 x_grid = np.linspace(min(min(x), min(y)), max(max(x), max(y)), 1000)
21
22 # Vẽ đồ thị
23 plt.figure(figsize=(8, 6))
24 plt.plot(x_grid, density_x(x_grid), color='red', linewidth=2, label='Density of x')
25 plt.plot(x_grid, density_y(x_grid), color='blue', linewidth=2, label='Density of y')
26 plt.title("Multiple curves", fontsize=16)
27 plt.xlabel("", fontsize=12)
28 plt.legend()
29 plt.show()
30

```

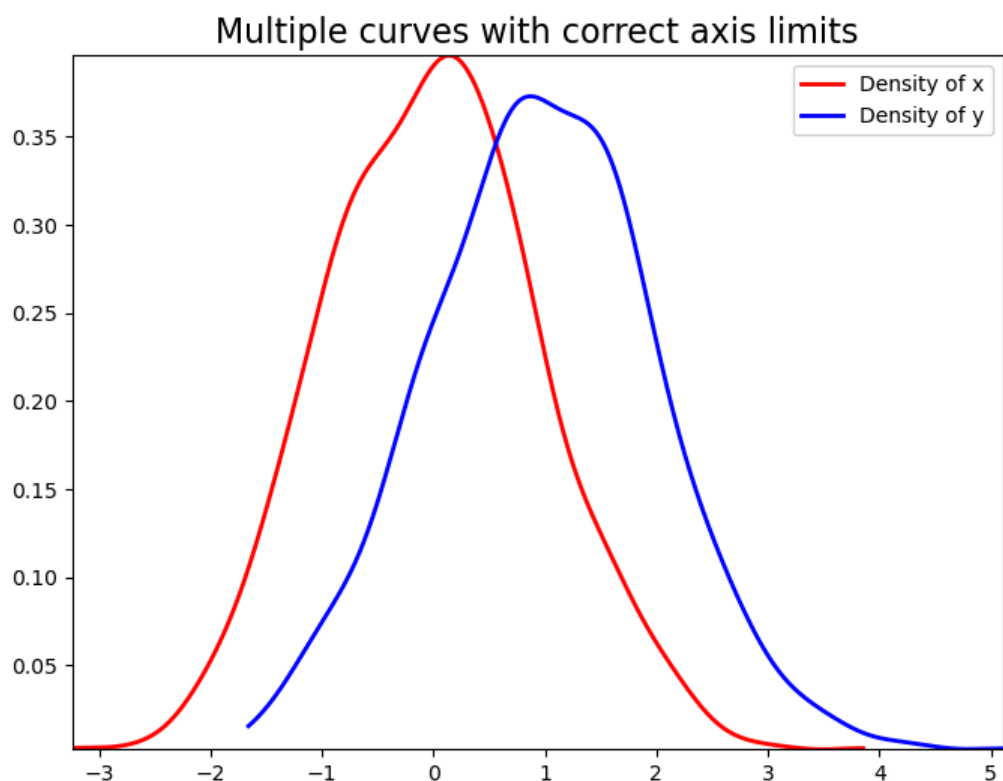


Vẽ nhiều đường mật độ (density curves) với giới hạn trục x và y được điều chỉnh chính xác

```

In [20]: 1 import numpy as np
2 import matplotlib.pyplot as plt
3 from scipy.stats import gaussian_kde
4
5 # Tạo dữ liệu
6 np.random.seed(42)
7 x = np.random.normal(size=500)
8
9 np.random.seed(2)
10 y = np.random.normal(size=500) + 1
11
12 # Tính mật độ
13 density_x = gaussian_kde(x)
14 density_y = gaussian_kde(y)
15
16 # Lưới giá trị cho trục x
17 x_grid_x = np.linspace(min(x), max(x), 1000)
18 x_grid_y = np.linspace(min(y), max(y), 1000)
19
20 # Xác định giới hạn trục x và y
21 x_min = min(x.min(), y.min())
22 x_max = max(x.max(), y.max())
23 y_min = min(density_x(x_grid_x).min(), density_y(x_grid_y).min())
24 y_max = max(density_x(x_grid_x).max(), density_y(x_grid_y).max())
25
26 # Vẽ đồ thị
27 plt.figure(figsize=(8, 6))
28 plt.plot(x_grid_x, density_x(x_grid_x), color='red', linewidth=2, label='Density of x')
29 plt.plot(x_grid_y, density_y(x_grid_y), color='blue', linewidth=2, label='Density of y')
30 plt.title("Multiple curves with correct axis limits", fontsize=16)
31 plt.xlabel("", fontsize=12)
32 plt.xlim(x_min, x_max) # Giới hạn trục x
33 plt.ylim(y_min, y_max) # Giới hạn trục y
34 plt.legend()
35 plt.show()
36

```

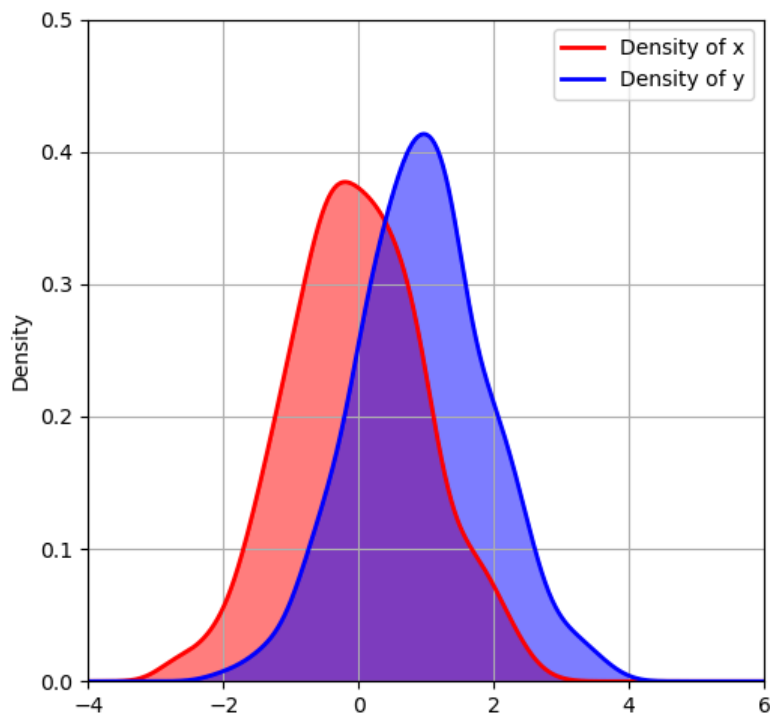


Shade area under curve with transparency

```

In [31]: 1 import pandas as pd
2 import numpy as np
3 import matplotlib.pyplot as plt
4 from scipy.stats import gaussian_kde
5
6 # Tạo dữ liệu x và y
7 np.random.seed(0)
8 x = np.random.normal(size=500)
9 y = np.random.normal(size=500) + 1
10
11 # Tính mật độ (density) cho x và y
12 density_x = gaussian_kde(x)
13 density_y = gaussian_kde(y)
14 x_vals = np.linspace(-4, 6, 1000)
15 y_vals = np.linspace(-4, 6, 1000)
16
17 # Vẽ biểu đồ
18 plt.figure(figsize=(10, 5))
19
20 # Biểu đồ đầu tiên: density của x và y với vùng tô màu
21 plt.subplot(1, 2, 1)
22 plt.plot(x_vals, density_x(x_vals), color='red', linewidth=2, label='Density of x')
23 plt.fill_between(x_vals, density_x(x_vals), color='red', alpha=0.5)
24 plt.plot(y_vals, density_y(y_vals), color='blue', linewidth=2, label='Density of y')
25 plt.fill_between(y_vals, density_y(y_vals), color='blue', alpha=0.5)
26 plt.xlim(-4, 6)
27 plt.ylim(0, 0.5)
28 plt.xlabel("")
29 plt.ylabel("Density")
30 plt.title("", fontsize=14)
31 plt.legend(loc='upper right')
32 plt.grid(True)
33
34 # Hiển thị đồ thị
35 plt.tight_layout()
36 plt.show()
37

```

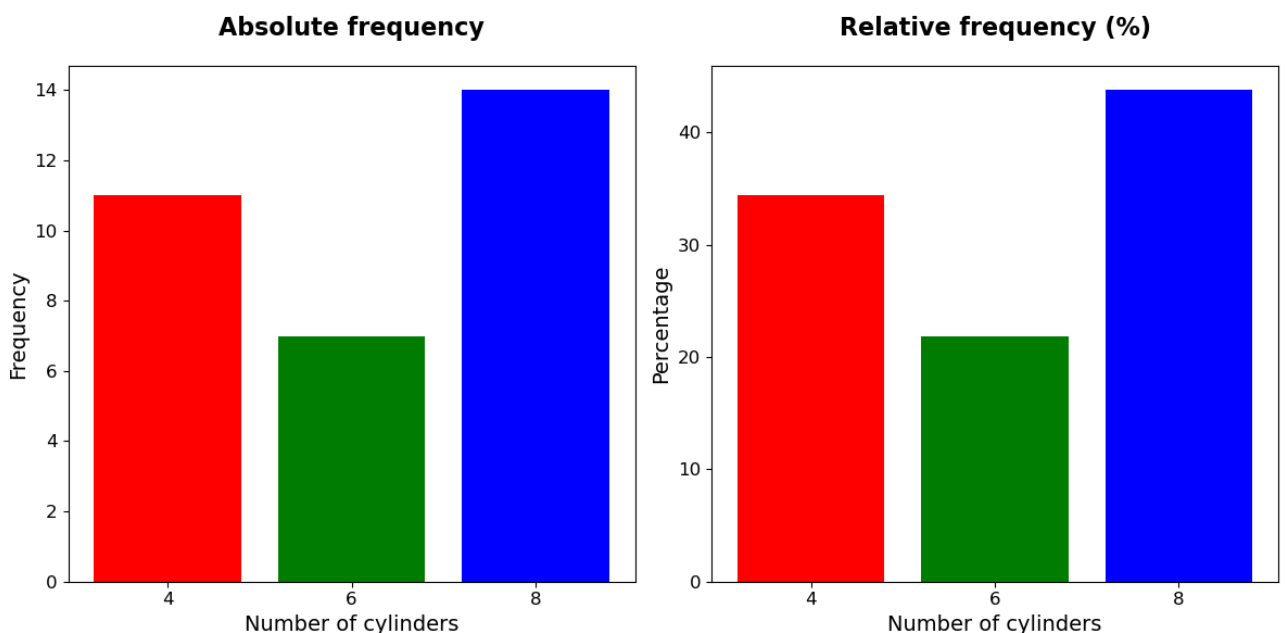


IV.Bar Chart.

Bar Chart using mtcars dataset

Making a Frequency table of Column

```
In [33]: 1 import pandas as pd
2 import matplotlib.pyplot as plt
3 import numpy as np
4
5 # Đọc dữ liệu mtcars từ file
6 mtcars = pd.read_csv('mtcars.csv')
7
8 # Tạo bảng tần suất tuyệt đối cho số xi-lanh (cột 'cyl')
9 my_table = mtcars['cyl'].value_counts().sort_index()
10
11 # Bố cục 1 hàng, 2 cột
12 plt.figure(figsize=(12, 6))
13
14 # Biểu đồ tần suất tuyệt đối
15 plt.subplot(1, 2, 1)
16 plt.bar(my_table.index.astype(str), my_table.values, color=['red', 'green', 'blue'])
17 plt.title("Absolute frequency", fontsize=16, fontweight='bold', pad=20)
18 plt.xlabel("Number of cylinders", fontsize=14)
19 plt.ylabel("Frequency", fontsize=14)
20 plt.xticks(fontsize=12)
21 plt.yticks(fontsize=12)
22
23 # Biểu đồ tần suất tương đối
24 plt.subplot(1, 2, 2)
25 relative_freq = (my_table / my_table.sum()) * 100
26 plt.bar(relative_freq.index.astype(str), relative_freq.values, color=['red', 'green', 'blue'])
27 plt.title("Relative frequency (%)", fontsize=16, fontweight='bold', pad=20)
28 plt.xlabel("Number of cylinders", fontsize=14)
29 plt.ylabel("Percentage", fontsize=14)
30 plt.xticks(fontsize=12)
31 plt.yticks(fontsize=12)
32
33 # Hiển thị đồ thị
34 plt.tight_layout()
35 plt.show()
36
```



Khi nào sử dụng mỗi loại biểu đồ:

Biểu đồ Tần suất Tuyệt đối:

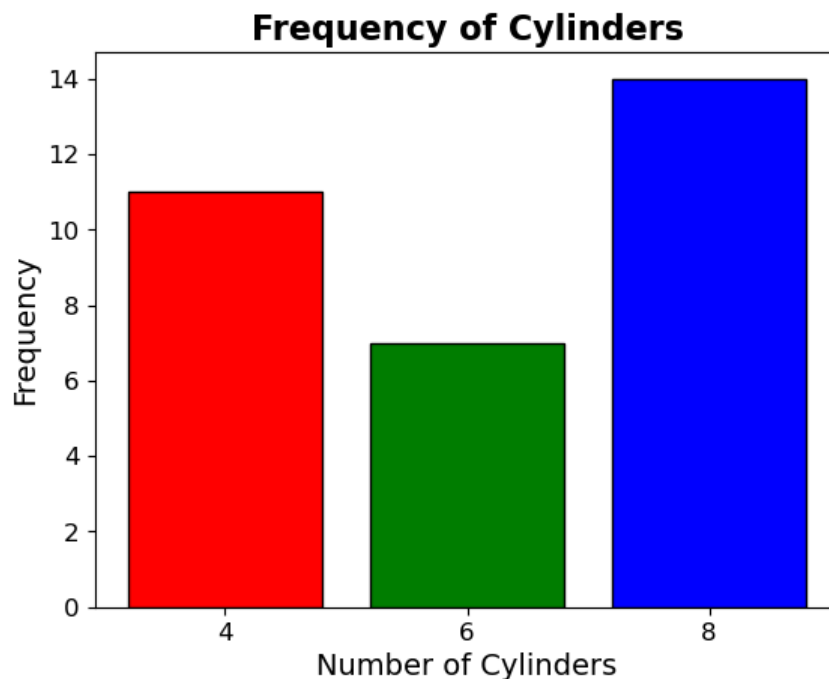
Khi bạn cần biểu diễn số lượng thực tế của các nhóm. Phù hợp để so sánh trực tiếp giữa các nhóm.

Biểu đồ Tần suất Tương đối:

Khi bạn cần biểu diễn tỷ lệ hoặc phần trăm của mỗi nhóm so với tổng thể. Thích hợp để so sánh dữ liệu trong các tập dữ liệu có quy mô khác nhau.

The same barplot can be created with factor data with the plot function see below.

```
In [35]: 1 import pandas as pd
2 import matplotlib.pyplot as plt
3
4 # Đọc dữ liệu mtcars từ file
5 mtcars = pd.read_csv('mtcars.csv')
6
7 # Tạo bảng tần suất của số xi-lanh (cyl)
8 freq_table = mtcars['cyl'].value_counts().sort_index()
9
10 # Vẽ biểu đồ với màu đỏ, xanh lá, xanh dương
11 colors = ['red', 'green', 'blue']
12 plt.bar(freq_table.index.astype(str), freq_table.values, color=colors, edgecolor='black')
13
14 # Thêm tiêu đề và nhãn trục
15 plt.title("Frequency of Cylinders", fontsize=16, fontweight='bold')
16 plt.xlabel("Number of Cylinders", fontsize=14)
17 plt.ylabel("Frequency", fontsize=14)
18
19 # Điều chỉnh vị trí nhãn trục x
20 plt.xticks(fontsize=12)
21 plt.yticks(fontsize=12)
22
23 # Hiển thị biểu đồ
24 plt.show()
25
```

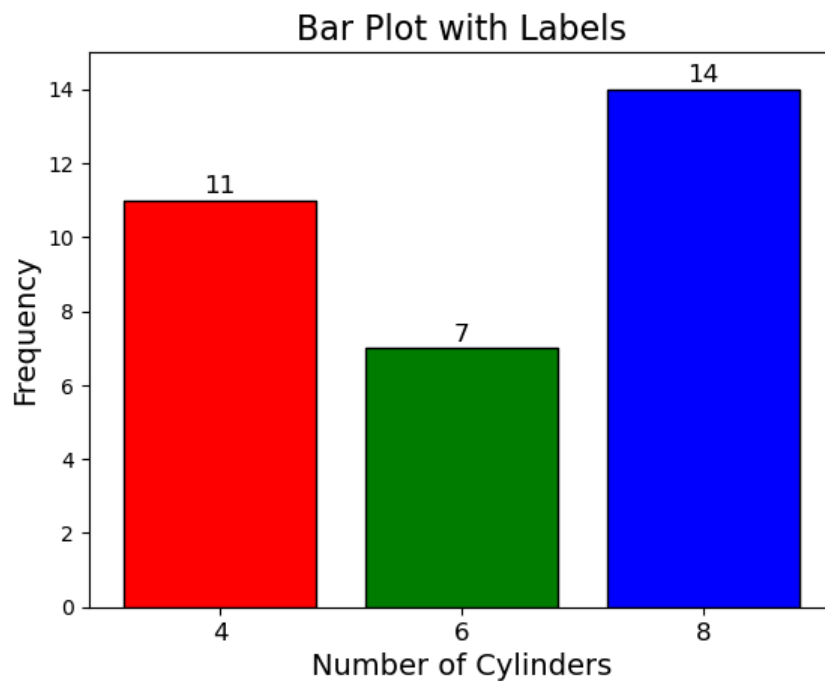


Text can be written on the bar plot as below

```

In [37]: 1 import pandas as pd
2 import matplotlib.pyplot as plt
3 import numpy as np
4
5 # Đọc dữ liệu mtcars từ file
6 mtcars = pd.read_csv('mtcars.csv')
7
8 # Tạo bảng tần suất của số xi-lanh (cyl)
9 my_table = mtcars['cyl'].value_counts().sort_index()
10
11 # Vẽ biểu đồ cột
12 colors = ['red', 'green', 'blue'] # Màu sắc theo thứ tự giống R
13 bar_positions = np.arange(len(my_table)) # Vị trí các cột
14 plt.bar(bar_positions, my_table.values, color=colors, edgecolor='black')
15
16 # Thêm nhãn vào các cột
17 for i, value in enumerate(my_table.values):
18     plt.text(bar_positions[i], value + 0.2, str(value), ha='center', fontsize=12)
19
20 # Cài đặt trục y
21 plt.ylim(0, 15)
22
23 # Thêm tiêu đề và nhãn trục
24 plt.title("Bar Plot with Labels", fontsize=16)
25 plt.xlabel("Number of Cylinders", fontsize=14)
26 plt.ylabel("Frequency", fontsize=14)
27
28 # Đặt nhãn trục x
29 plt.xticks(bar_positions, my_table.index.astype(str), fontsize=12)
30
31 # Hiển thị biểu đồ
32 plt.show()
33

```

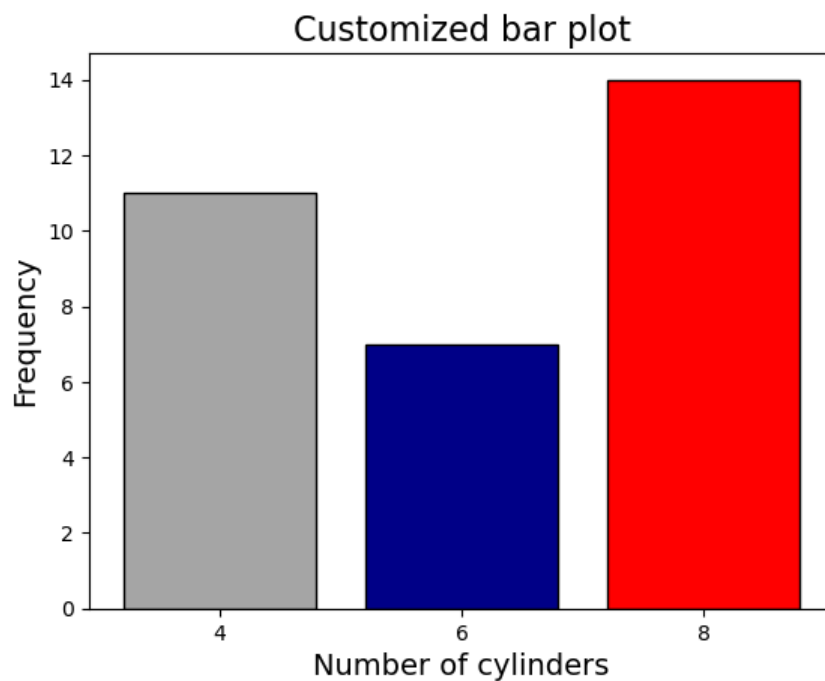


“SALT” the Bar Plot

```

In [38]: 1 import pandas as pd
2 import matplotlib.pyplot as plt
3
4 # Đọc dữ liệu mtcars từ file
5 mtcars = pd.read_csv('mtcars.csv')
6
7 # Tạo bảng tần suất của số xi-lanh (cyl)
8 my_table = mtcars['cyl'].value_counts().sort_index()
9
10 # Màu sắc cho các cột
11 colors = ['darkgrey', 'darkblue', 'red']
12
13 # Vẽ biểu đồ cột
14 plt.bar(my_table.index.astype(str), my_table.values, color=colors, edgecolor='black')
15
16 # Thêm tiêu đề và nhãn trục
17 plt.title("Customized bar plot", fontsize=16)
18 plt.xlabel("Number of cylinders", fontsize=14)
19 plt.ylabel("Frequency", fontsize=14)
20
21 # Hiển thị biểu đồ
22 plt.show()
23

```

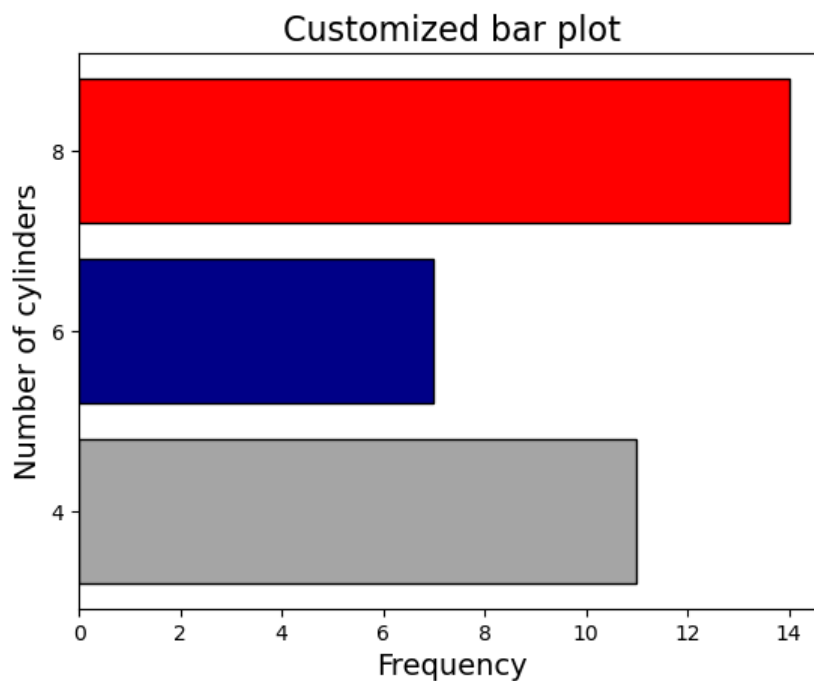


Horizontal Barplot

```

In [39]: 1 import pandas as pd
          2 import matplotlib.pyplot as plt
          3
          4 # Đọc dữ liệu mtcars từ file
          5 mtcars = pd.read_csv('mtcars.csv')
          6
          7 # Tạo bảng tần suất của số xi-lanh (cyl)
          8 my_table = mtcars['cyl'].value_counts().sort_index()
          9
         10 # Màu sắc cho các cột
         11 colors = ['darkgrey', 'darkblue', 'red']
         12
         13 # Vẽ biểu đồ cột ngang
         14 plt.barh(my_table.index.astype(str), my_table.values, color=colors, edgecolor='black')
         15
         16 # Thêm tiêu đề và nhãn trục
         17 plt.title("Customized bar plot", fontsize=16)
         18 plt.xlabel("Frequency", fontsize=14)
         19 plt.ylabel("Number of cylinders", fontsize=14)
         20
         21 # Hiển thị biểu đồ
         22 plt.show()
         23

```

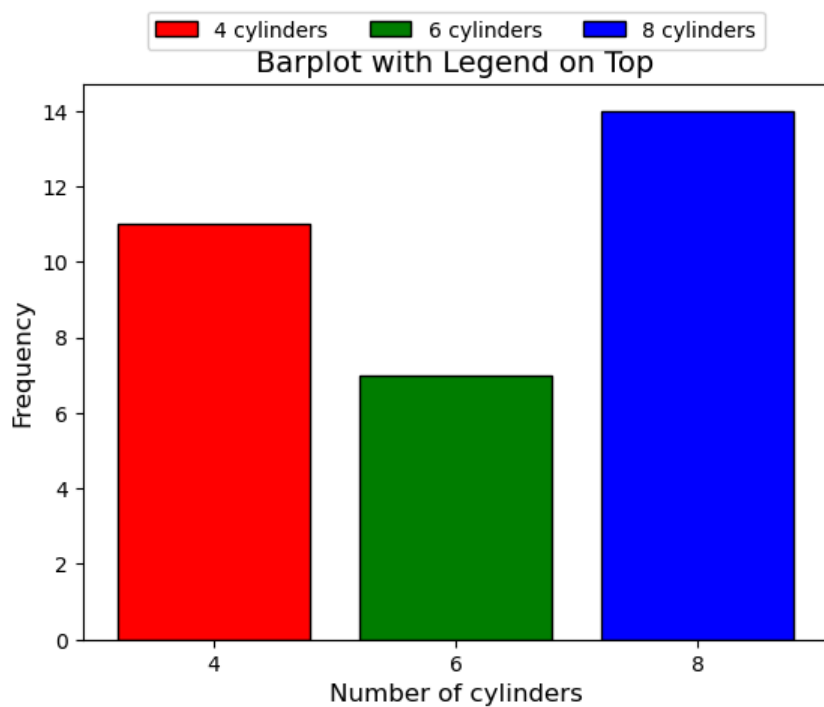


Barplot with Legend


```

In [40]: 1 import pandas as pd
2 import matplotlib.pyplot as plt
3
4 # Đọc dữ liệu mtcars từ file
5 mtcars = pd.read_csv('mtcars.csv')
6
7 # Tạo bảng tần suất của số xi-lanh (cyl)
8 my_table = mtcars['cyl'].value_counts().sort_index()
9
10 # Màu sắc cho các cột (tương tự rainbow(3) trong R)
11 colors = ['red', 'green', 'blue']
12
13 # Vẽ biểu đồ cột
14 bars = plt.bar(my_table.index.astype(str), my_table.values, color=colors, edgecolor='black')
15
16 # Thêm chú thích (Legend) phía trên
17 plt.legend(bars, [f"{cyl} cylinders" for cyl in my_table.index], loc='upper center', bbox_to_anchor=(0.5, 1.
18
19 # Thêm nhãn trục và tiêu đề
20 plt.xlabel("Number of cylinders", fontsize=12)
21 plt.ylabel("Frequency", fontsize=12)
22 plt.title("Barplot with Legend on Top", fontsize=14)
23
24 # Hiển thị biểu đồ
25 plt.show()
26

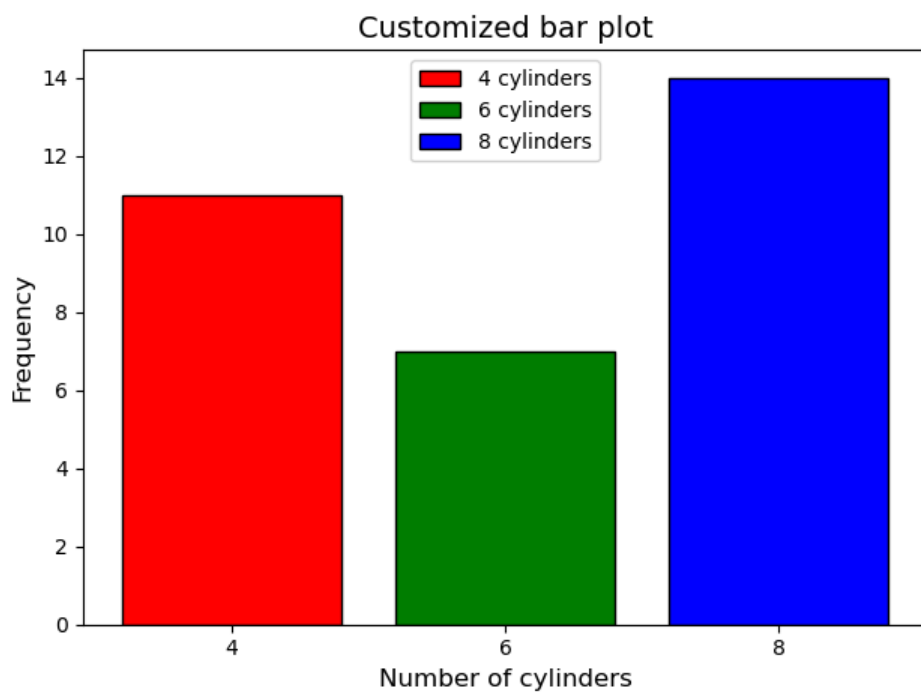
```



```

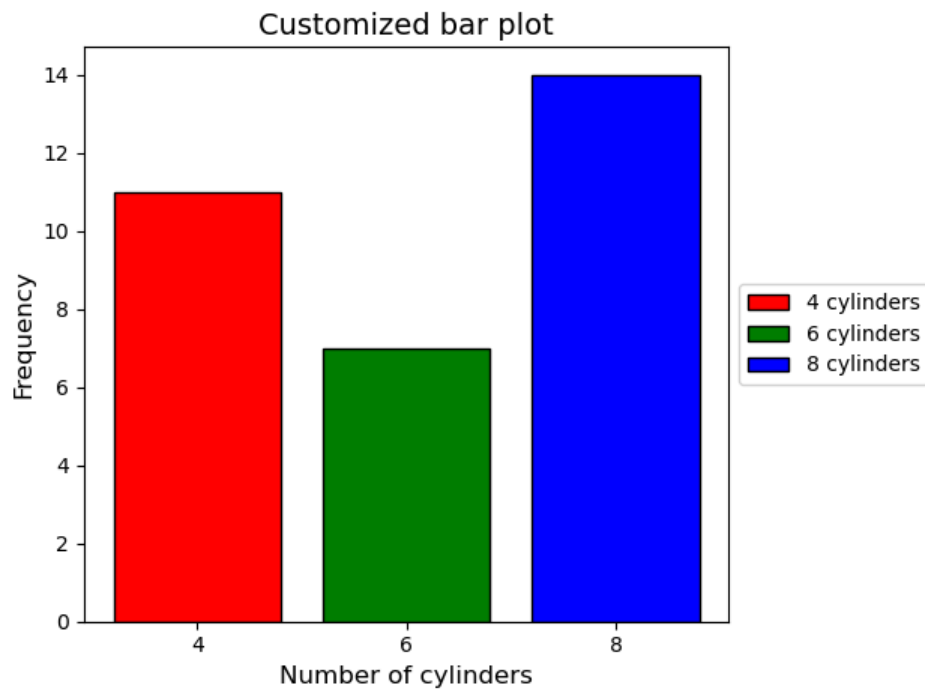
In [44]: 1 import pandas as pd
2 import matplotlib.pyplot as plt
3
4 # Đọc dữ liệu mtcars từ file
5 mtcars = pd.read_csv('mtcars.csv')
6
7 # Tạo bảng tần suất của số xi-lanh (cyl)
8 my_table = mtcars['cyl'].value_counts().sort_index()
9
10 # Màu sắc cho các cột (tương tự rainbow(3) trong R)
11 colors = ['red', 'green', 'blue']
12
13 # Vẽ biểu đồ cột
14 bars = plt.bar(my_table.index.astype(str), my_table.values, color=colors, edgecolor='black')
15
16 # Thêm chú thích (legend) với một cột, đặt bên dưới tiêu đề
17 plt.legend(bars, [f"{cyl} cylinders" for cyl in my_table.index], loc='upper center', bbox_to_anchor=(0.5, 1.
18
19 # Thêm nhãn trục và tiêu đề
20 plt.xlabel("Number of cylinders", fontsize=12)
21 plt.ylabel("Frequency", fontsize=12)
22 plt.title("Customized bar plot", fontsize=14)
23
24 # Hiển thị biểu đồ
25 plt.tight_layout() # Đảm bảo không bị cắt nội dung
26 plt.show()
27

```



Another legend on the right side

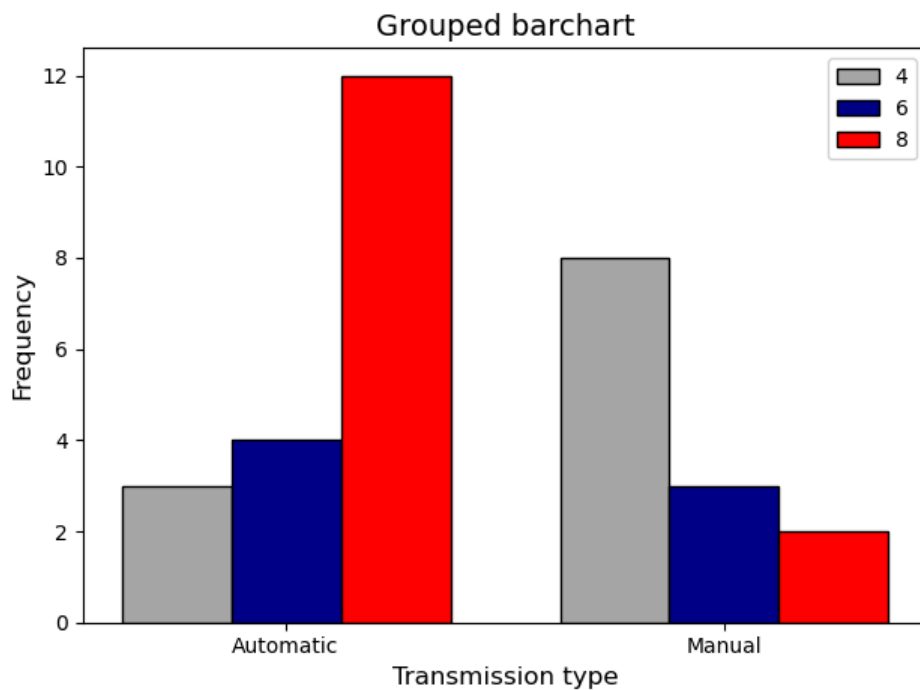
```
In [45]: 1 import pandas as pd
2 import matplotlib.pyplot as plt
3
4 # Đọc dữ liệu mtcars từ file
5 mtcars = pd.read_csv('mtcars.csv')
6
7 # Tạo bảng tần suất của số xi-lanh (cyl)
8 my_table = mtcars['cyl'].value_counts().sort_index()
9
10 # Màu sắc cho các cột (tương tự rainbow(3) trong R)
11 colors = ['red', 'green', 'blue']
12
13 # Vẽ biểu đồ cột
14 bars = plt.bar(my_table.index.astype(str), my_table.values, color=colors, edgecolor='black')
15
16 # Thêm chú thích (Legend) nằm bên phải của biểu đồ
17 plt.legend(bars, [f"{cyl} cylinders" for cyl in my_table.index], loc='center left', bbox_to_anchor=(1, 0.5),
18
19 # Thêm nhãn trục và tiêu đề
20 plt.xlabel("Number of cylinders", fontsize=12)
21 plt.ylabel("Frequency", fontsize=12)
22 plt.title("Customized bar plot", fontsize=14)
23
24 # Hiển thị biểu đồ
25 plt.tight_layout() # Đảm bảo không bị cắt nội dung
26 plt.show()
27
```



Grouping Bar Plot

In [44]:

```
1 import pandas as pd
2 import matplotlib.pyplot as plt
3 import numpy as np
4
5 # Tải dataset mtcars.csv từ file bên ngoài
6 mtcars = pd.read_csv('mtcars.csv')
7
8 # Chuyển đổi cột 'am' thành dạng phân loại
9 mtcars['am'] = mtcars['am'].map({0: "Automatic", 1: "Manual"})
10
11 # Tạo bảng tần suất giữa 'cyl' và 'am'
12 other_table = pd.crosstab(mtcars['cyl'], mtcars['am'])
13
14 # Màu sắc tùy chỉnh
15 colors = ["darkgrey", "darkblue", "red"]
16
17 # Vẽ barplot nhóm
18 bar_width = 0.25 # Chiều rộng của cột
19 x = np.arange(len(other_table.columns)) # Vị trí các nhóm (Automatic, Manual)
20
21 fig, ax = plt.subplots()
22
23 # Lặp qua từng số xi-lanh và vẽ các nhóm
24 for i, (cyl, values) in enumerate(other_table.iterrows()):
25     ax.bar(x + i * bar_width, values, bar_width,
26           label=f"{cyl}", color=colors[i], edgecolor="black")
27
28 # Thêm tiêu đề và nhãn trục
29 ax.set_title("Grouped barchart", fontsize=14)
30 ax.set_xlabel("Transmission type", fontsize=12)
31 ax.set_ylabel("Frequency", fontsize=12)
32 ax.set_xticks(x + bar_width) # Điều chỉnh nhãn trục x để nằm giữa
33 ax.set_xticklabels(other_table.columns, fontsize=10)
34
35 # Thêm Legend
36 ax.legend(fontsize=10, title_fontsize=11, loc="upper right")
37
38 # Hiển thị biểu đồ
39 plt.tight_layout()
40 plt.show()
41
```



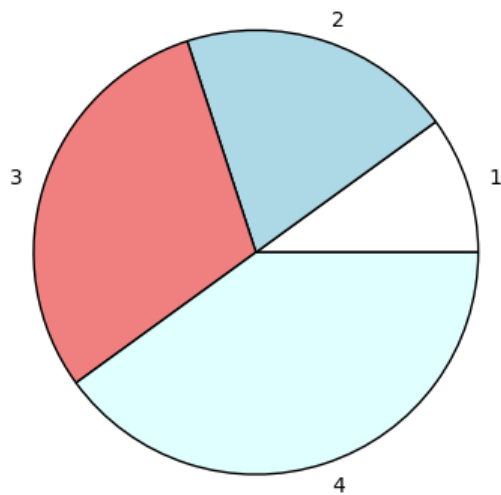
V.Pie Chart.

Pie or circular chart are used to represents discrete or categorical data.

The proportions or percentages in slices shows the quantities of various categories.

```
In [48]: 1 import matplotlib.pyplot as plt
2
3 # Tạo một danh sách chứa số Lượng của mỗi phần trong Pie Chart
4 count = [10, 20, 30, 40]
5
6 # Màu sắc tương tự hình đính kèm
7 colors = ['white', 'lightblue', 'lightcoral', 'lightcyan']
8
9 # Vẽ Pie Chart đơn giản với viền
10 plt.pie(count,
11         labels=[1, 2, 3, 4],
12         colors=colors,
13         wedgeprops={'edgecolor': 'black', 'linewidth': 1})
14
15 # Thêm tiêu đề
16 plt.title("Simple Pie Chart", fontsize=14)
17
18 # Hiển thị biểu đồ
19 plt.show()
20
21
```

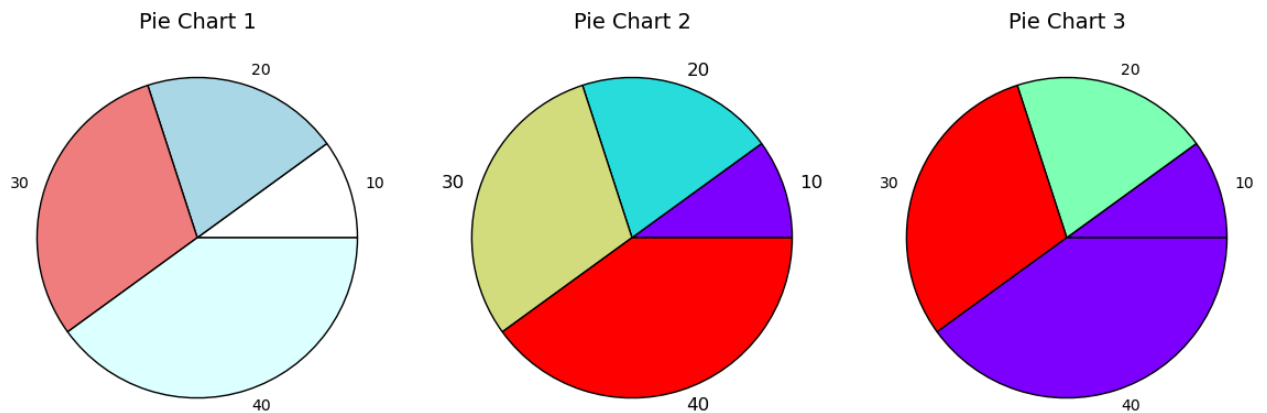
Simple Pie Chart



```

In [49]: 1 import matplotlib.pyplot as plt
2
3 # Dữ liệu cho Pie Chart
4 count = [10, 20, 30, 40]
5
6 # Tạo layout 1 hàng, 3 cột
7 fig, axes = plt.subplots(1, 3, figsize=(12, 4)) # figsize để điều chỉnh kích thước tổng thể
8
9 # Pie Chart 1: Nhân và kích thước chữ lớn hơn
10 axes[0].pie(count, labels=count, colors=['white', 'lightblue', 'lightcoral', 'lightcyan'],
11            wedgeprops={'edgecolor': 'black', 'linewidth': 1}, textprops={'fontsize': 10})
12 axes[0].set_title("Pie Chart 1", fontsize=14)
13
14 # Pie Chart 2: Màu cầu vồng, nhân lớn hơn
15 axes[1].pie(count, labels=count, colors=plt.cm.rainbow(np.linspace(0, 1, 4))),
16            wedgeprops={'edgecolor': 'black', 'linewidth': 1}, textprops={'fontsize': 12})
17 axes[1].set_title("Pie Chart 2", fontsize=14)
18
19 # Pie Chart 3: Màu cầu vồng (ít màu hơn), nhân lớn hơn
20 axes[2].pie(count, labels=count, colors=plt.cm.rainbow(np.linspace(0, 1, 3))),
21            wedgeprops={'edgecolor': 'black', 'linewidth': 1}, textprops={'fontsize': 10})
22 axes[2].set_title("Pie Chart 3", fontsize=14)
23
24 # Hiển thị biểu đồ
25 plt.tight_layout()
26 plt.show()
27
28

```

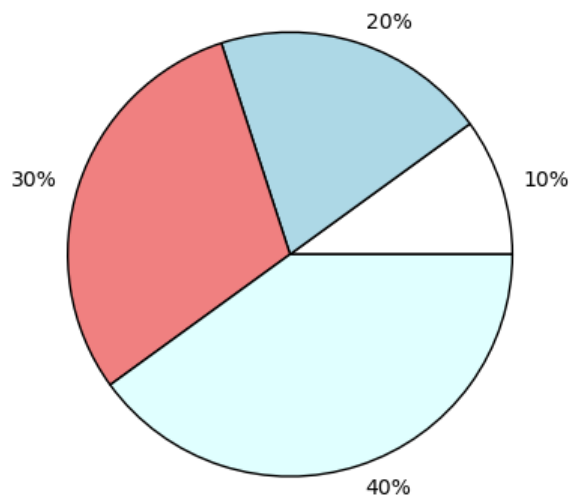


```

In [50]: 1 import matplotlib.pyplot as plt
2
3 # Dữ liệu cho Pie Chart
4 count = [10, 20, 30, 40]
5 labels = [f'{c}%' for c in count] # Tạo nhãn với dấu phần trăm
6
7 # Màu sắc giống hình đính kèm
8 colors = ['white', 'lightblue', 'lightcoral', 'lightcyan']
9
10 # Vẽ Pie Chart với nhãn phần trăm
11 plt.pie(count,
12         labels=labels,
13         colors=colors,
14         autopct=None, # Không hiển thị thêm phần trăm tự động
15         wedgeprops={'edgecolor': 'black', 'linewidth': 1}) # Thêm viền cho Pie Chart
16
17 # Thêm tiêu đề
18 plt.title("Pie Chart with Percentage Labels", fontsize=14)
19
20 # Hiển thị biểu đồ
21 plt.show()
22

```

Pie Chart with Percentage Labels

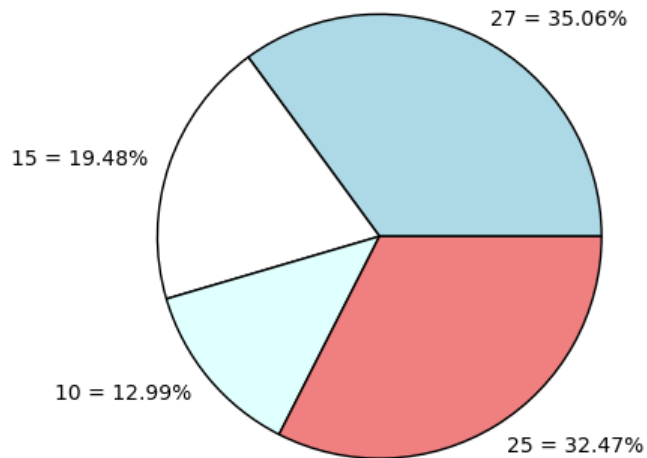


```

In [51]: 1 import matplotlib.pyplot as plt
2
3 # Dữ liệu cho Pie Chart
4 count_2 = [27, 15, 10, 25]
5
6 # Tạo nhãn tùy chỉnh với giá trị và phần trăm
7 total = sum(count_2)
8 pie_labels = [f"{value} = {value / total * 100:.2f}%" for value in count_2]
9
10 # Màu sắc tùy chỉnh
11 colors = ['lightblue', 'white', 'lightcyan', 'lightcoral']
12
13 # Vẽ Pie Chart với nhãn tùy chỉnh
14 plt.pie(count_2,
15         labels=pie_labels,
16         colors=colors,
17         wedgeprops={'edgecolor': 'black', 'linewidth': 1}) # Thêm viền đen cho các phần
18
19 # Thêm tiêu đề
20 plt.title("Pie Chart with Custom Labels", fontsize=14)
21
22 # Hiển thị biểu đồ
23 plt.show()
24

```

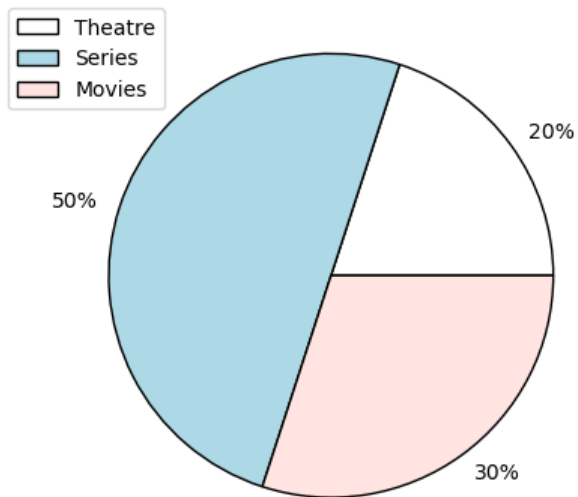
Pie Chart with Custom Labels




```

In [53]: 1 import matplotlib.pyplot as plt
2
3 # Dữ liệu cho Pie Chart
4 count_3 = [20, 50, 30]
5
6 # Tạo nhãn với phần trăm
7 labels = [f"{value}%" for value in count_3]
8
9 # Màu sắc cho Pie Chart
10 colors = ['white', 'lightblue', 'mistyrose']
11
12 # Vẽ Pie Chart
13 plt.pie(count_3,
14         labels=labels,
15         colors=colors,
16         wedgeprops={'edgecolor': 'black', 'linewidth': 1}) # Thêm viền đen cho các phần
17
18 # Thêm Legend, đặt ở ngoài Pie Chart
19 plt.legend(["Theatre", "Series", "Movies"], loc="upper left", bbox_to_anchor=(-0.1, 1), fontsize=10, frameon=0)
20
21 # Hiển thị biểu đồ
22 plt.show()
23

```



In []:

1