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 [46]:
```
import os
print(os.getcwd())  # Hiển thị thư mục hiện tại

```

```
D:\Test_Python\Day_6
```

In [47]:
```
os.chdir('D:/Test_Python/Day_6')  # Thay đổi 'path/to/your/directory' bằng

```

In [48]:
```
## 1. Cài đặt Pandas (nếu chưa cài):
## pip install pandas

import pandas as pd

```

In [49]:

```
# 2.Đọc file mtcars.csv
df = pd.read_csv("mtcars.csv")

# Hiển thị 5 dòng đầu tiên
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
```

The mtcars.csv dataset contains the following columns and their meanings:

1. mpg: Miles per gallon (fuel efficiency).

2. cyl: Number of cylinders in the engine.

3. disp: Displacement in cubic inches (engine size).

4. hp: Gross horsepower (engine power).

5. drat: Rear axle ratio (performance measure).

6. wt: Weight of the car (in 1,000 lbs).

7. qsec: 1/4 mile time (acceleration performance in seconds).

8. vs: Engine type (0 = V-shaped, 1 = straight).

9. am: Transmission type (0 = automatic, 1 = manual).

10. gear: Number of forward gears.

11. carb: Number of carburetors.

In [50]:

```
## 3.Phân tích dữ liệu
## - Mô tả dữ liệu:
print(df.describe())

```

```
              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  32.0000
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 [51]:

```
## - Lọc dữ liệu:
filtered_df = df[df['mpg'] > 20]
print(filtered_df)

```

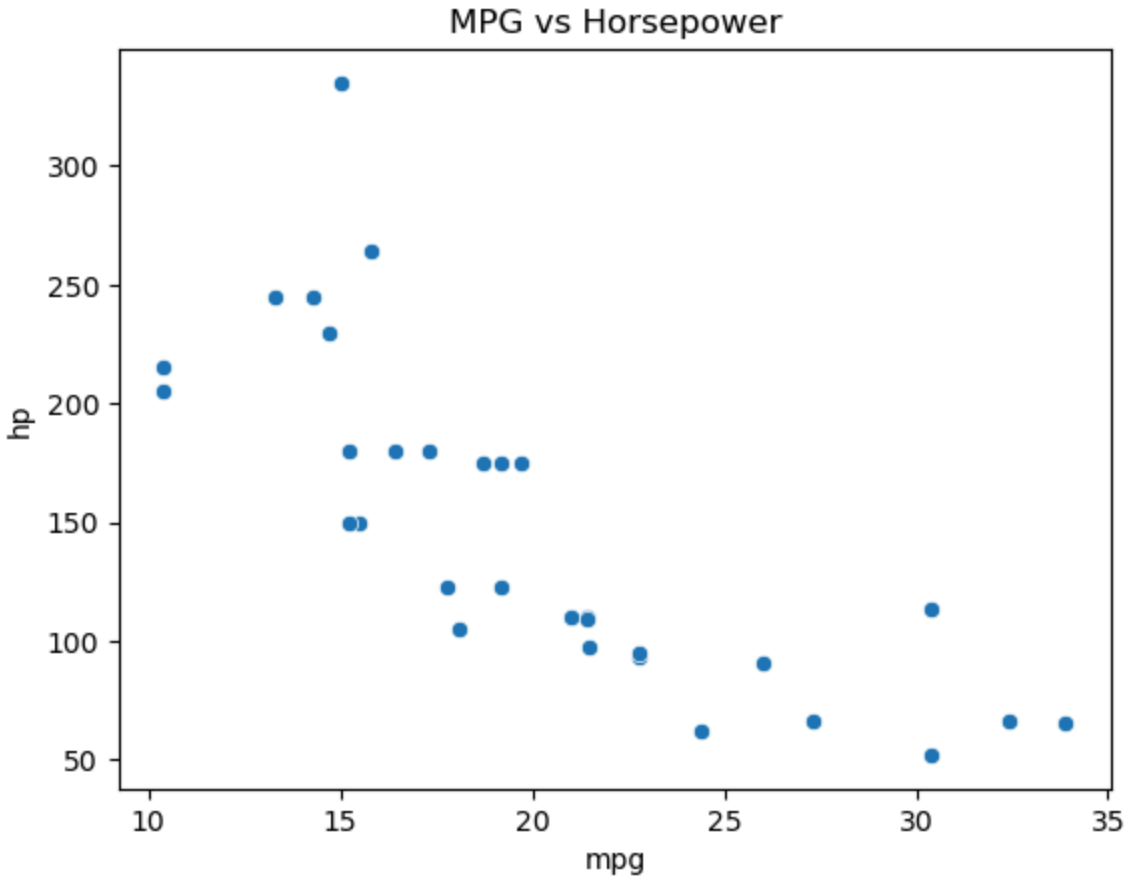
```
     mpg  cyl   disp   hp  drat     wt   qsec  vs  am  gear  carb
0   21.0    6  160.0  110  3.90  2.620  16.46   0   1     4     4
1   21.0    6  160.0  110  3.90  2.875  17.02   0   1     4     4
2   22.8    4  108.0   93  3.85  2.320  18.61   1   1     4     1
3   21.4    6  258.0  110  3.08  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 [52]:

```
## - Vẽ biểu đồ (sử dụng Matplotlib hoặc Seaborn):
## pip install matplotlib
## pip install
import matplotlib.pyplot as plt
import seaborn as sns

sns.scatterplot(data=df, x='mpg', y='hp')
plt.title("MPG vs Horsepower")
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
- Dependent variable in the vertical axis
- Independent variable in the horizontal axis

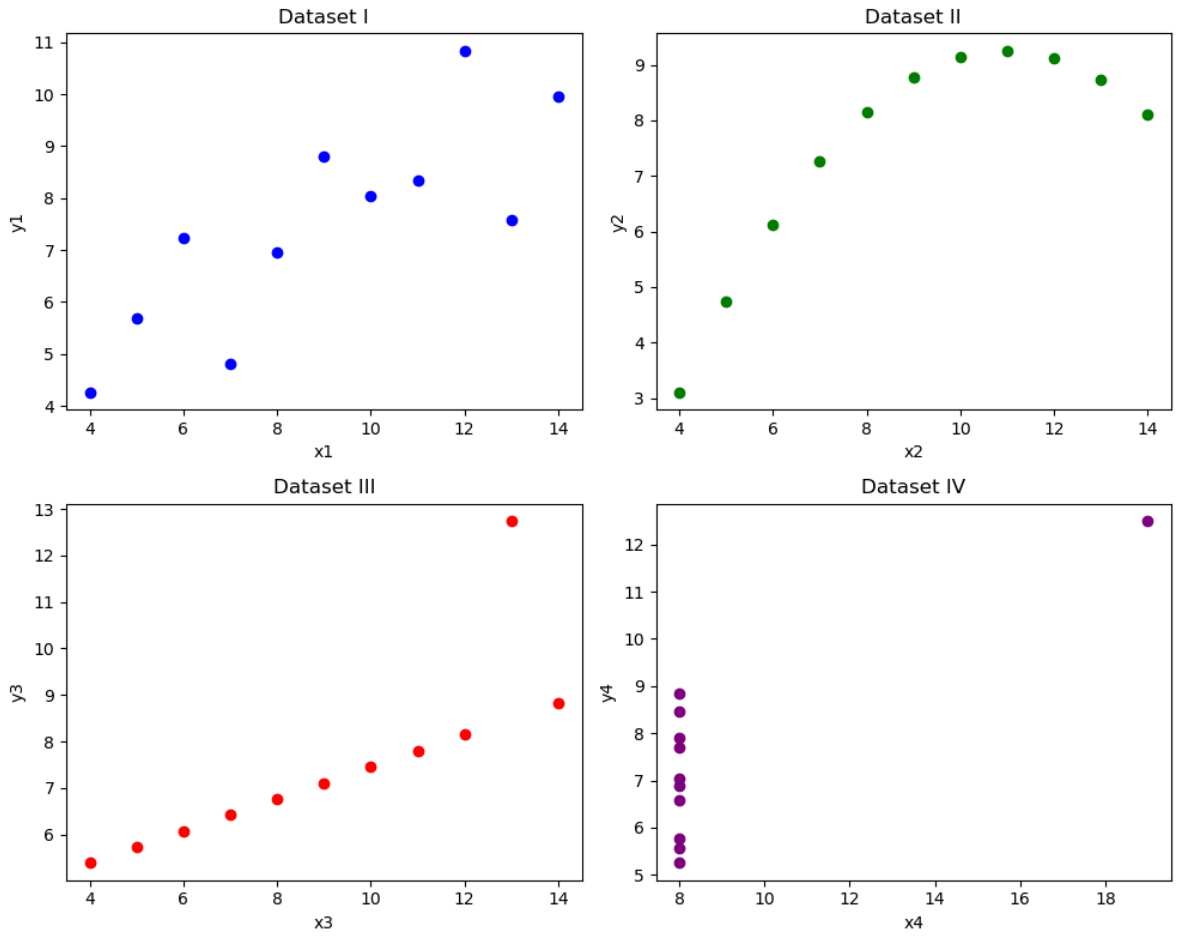
A = Axes (e.g., X Axis and Y Axis: x and y)

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

Create 2x2 Graph

In [53]:

```
## The code to create 2 by 2 graph is

import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd
from sklearn.datasets import fetch_openml

# Tải dataset Anscombe (nếu chưa có)
anscombe = sns.load_dataset('anscombe')

# Lọc dữ liệu theo từng nhóm
group1 = anscombe[anscombe['dataset'] == 'I']
group2 = anscombe[anscombe['dataset'] == 'II']
group3 = anscombe[anscombe['dataset'] == 'III']
group4 = anscombe[anscombe['dataset'] == 'IV']

# Tạo lưới đồ thị 2x2
plt.figure(figsize=(10, 8))

# Đồ thị 1
plt.subplot(2, 2, 1)
plt.scatter(group1['x'], group1['y'], color='blue')
plt.title('Dataset I')
plt.xlabel('x1')
plt.ylabel('y1')

# Đồ thị 2
plt.subplot(2, 2, 2)
plt.scatter(group2['x'], group2['y'], color='green')
plt.title('Dataset II')
plt.xlabel('x2')
plt.ylabel('y2')

# Đồ thị 3
plt.subplot(2, 2, 3)
plt.scatter(group3['x'], group3['y'], color='red')
plt.title('Dataset III')
plt.xlabel('x3')
plt.ylabel('y3')

# Đồ thị 4
plt.subplot(2, 2, 4)
plt.scatter(group4['x'], group4['y'], color='purple')
plt.title('Dataset IV')
plt.xlabel('x4')
plt.ylabel('y4')

# Hiển thị đồ thị
plt.tight_layout()
plt.show()

```

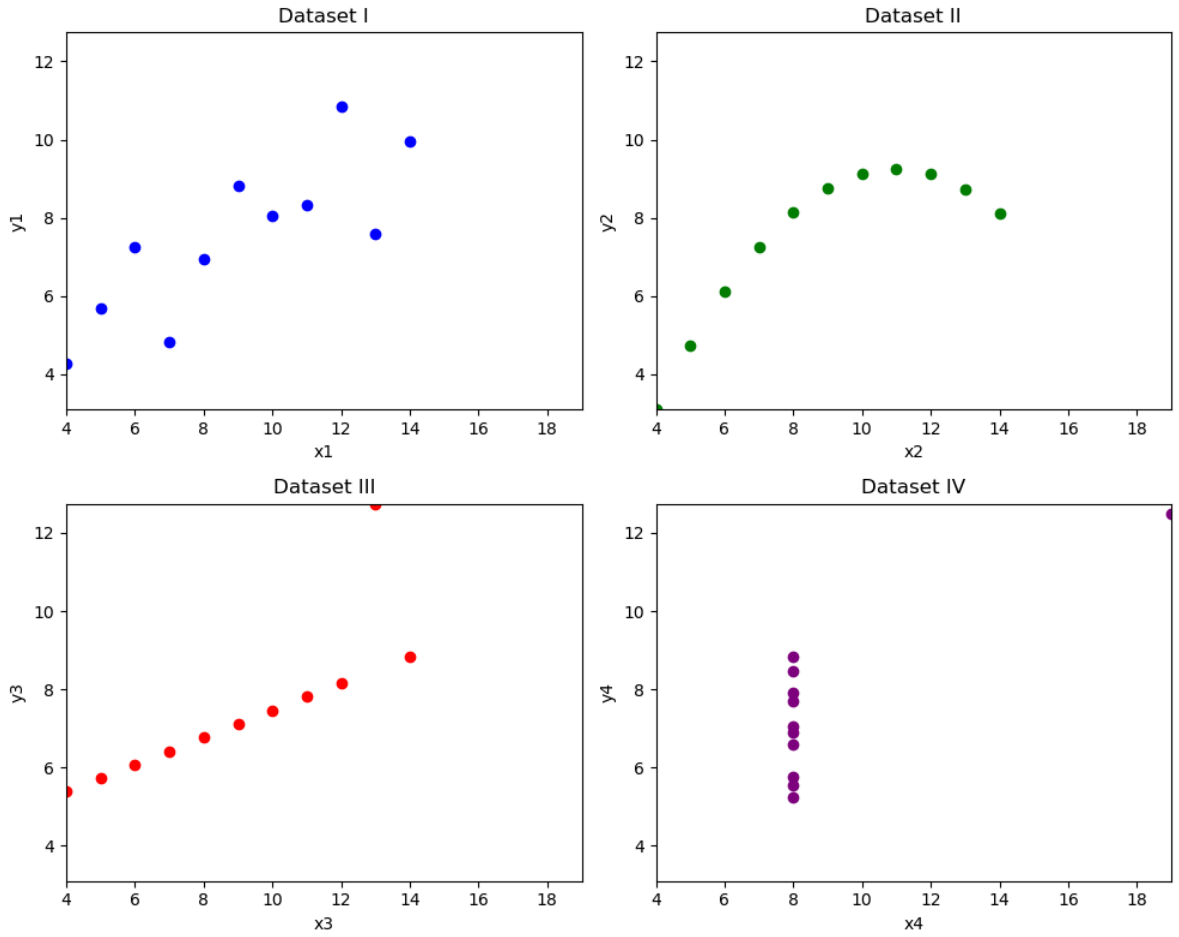


Create 2x2 Graph with xmin, xmax, ymin, ymax

In [54]:

```python
#Execute this code

import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd

# Tải dataset Anscombe
anscombe = sns.load_dataset('anscombe')

# Lọc dữ liệu theo từng nhóm
group1 = anscombe[anscombe['dataset'] == 'I']
group2 = anscombe[anscombe['dataset'] == 'II']
group3 = anscombe[anscombe['dataset'] == 'III']
group4 = anscombe[anscombe['dataset'] == 'IV']

# Tính xmin, xmax, ymin, ymax
xmin = min(group1['x'].min(), group2['x'].min(), group3['x'].min(), group4
xmax = max(group1['x'].max(), group2['x'].max(), group3['x'].max(), group4
ymin = min(group1['y'].min(), group2['y'].min(), group3['y'].min(), group4
ymax = max(group1['y'].max(), group2['y'].max(), group3['y'].max(), group4

# Tạo lưới đồ thị 2x2
plt.figure(figsize=(10, 8))

# Đồ thị 1
plt.subplot(2, 2, 1)
plt.scatter(group1['x'], group1['y'], color='blue')
plt.title('Dataset I')
plt.xlabel('x1')
plt.ylabel('y1')
plt.xlim(xmin, xmax)
plt.ylim(ymin, ymax)

# Đồ thị 2
plt.subplot(2, 2, 2)
plt.scatter(group2['x'], group2['y'], color='green')
plt.title('Dataset II')
plt.xlabel('x2')
plt.ylabel('y2')
plt.xlim(xmin, xmax)
plt.ylim(ymin, ymax)

# Đồ thị 3
plt.subplot(2, 2, 3)
plt.scatter(group3['x'], group3['y'], color='red')
plt.title('Dataset III')
plt.xlabel('x3')
plt.ylabel('y3')
plt.xlim(xmin, xmax)
plt.ylim(ymin, ymax)

# Đồ thị 4
plt.subplot(2, 2, 4)
plt.scatter(group4['x'], group4['y'], color='purple')
plt.title('Dataset IV')
plt.xlabel('x4')
plt.ylabel('y4')
```

```
plt.xlim(xmin, xmax)
plt.ylim(ymin, ymax)

# Hiển thị đồ thị
plt.tight_layout()
plt.show()

```



Plot 2x3 Graphs:

1. Scatterplot

2. Barplot

3. Boxplot

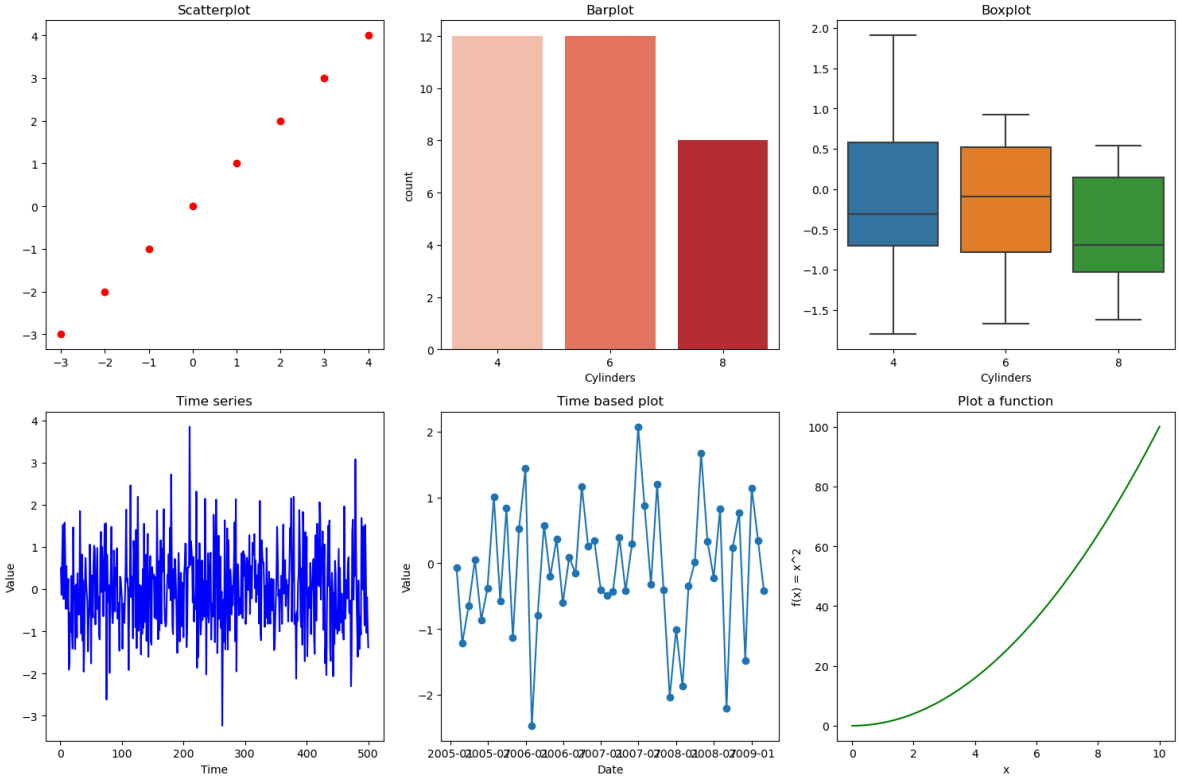
4. Time series plot

5. Time-based plot

6. Plot a function

In [55]:

```python
# Run this code
# Examples

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Data
np.random.seed(42)  # Đảm bảo kết quả ngẫu nhiên tái hiện được
x = np.array([-3, -2, -1, 0, 1, 2, 3, 4])
y = np.array([-3, -2, -1, 0, 1, 2, 3, 4])
my_ts = np.random.normal(size=500)  # Time series data
my_dates = pd.date_range(start="2005-01-01", periods=50, freq="M")
my_factor = pd.Series([6, 4, 8, 4, 6, 8, 4, 6] * 4, name="Cylinders")  #

# Tạo lưới đồ thị 2x3
fig, axs = plt.subplots(2, 3, figsize=(15, 10))

# 1. Scatterplot
axs[0, 0].scatter(x, y, color='red')
axs[0, 0].set_title("Scatterplot")

# 2. Barplot
sns.countplot(x=my_factor, ax=axs[0, 1], palette="Reds")
axs[0, 1].set_title("Barplot")

# 3. Boxplot
sns.boxplot(x=my_factor, y=np.random.normal(size=len(my_factor)), ax=axs[(
axs[0, 2].set_title("Boxplot")

# 4. Time series plot
axs[1, 0].plot(np.arange(1, 501), my_ts, color='blue')
axs[1, 0].set_title("Time series")
axs[1, 0].set_xlabel("Time")
axs[1, 0].set_ylabel("Value")

# 5. Time-based plot
axs[1, 1].plot(my_dates, np.random.normal(size=50), marker='o')
axs[1, 1].set_title("Time based plot")
axs[1, 1].set_xlabel("Date")
axs[1, 1].set_ylabel("Value")

# 6. Plot a function
x_fun = np.linspace(0, 10, 100)
y_fun = x_fun**2
axs[1, 2].plot(x_fun, y_fun, color='green')
axs[1, 2].set_title("Plot a function")
axs[1, 2].set_xlabel("x")
axs[1, 2].set_ylabel("f(x) = x^2")

# Hiển thị đồ thị
plt.tight_layout()
plt.show()
```


Scatterplot
Barplot
count
Cylinders
Boxplot
Cylinders
Time series
Value
Time
Time based plot
Value
Date
Plot a function
f(x) = x^2
x

II.Histogram

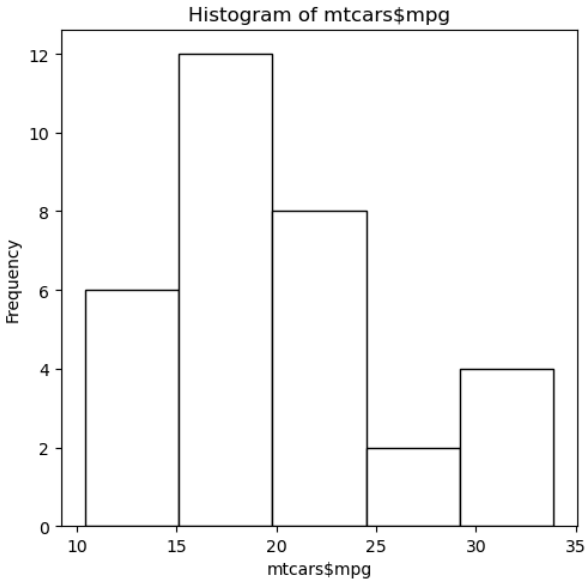
The Histogram is a bar plot used to represent continuous data.

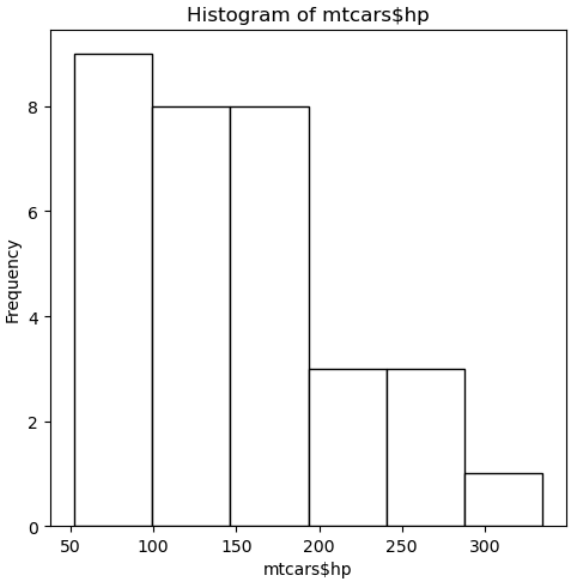
It shows the frequencies in the vertical axis and the continuous data on the horizontal axis.

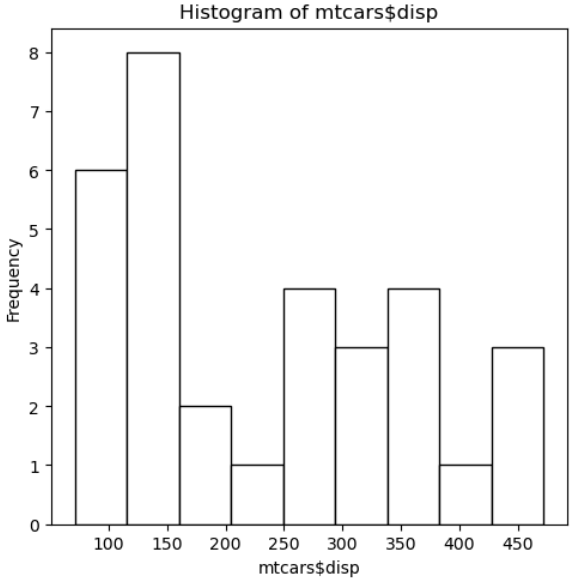
Using mtcars datasets

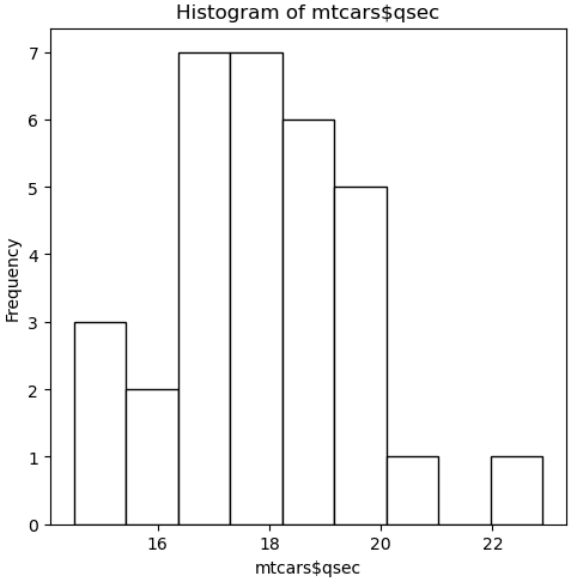
In [56]:

```
import pandas as pd
import matplotlib.pyplot as plt

# Đọc dữ liệu mtcars từ file CSV
mtcars = pd.read_csv('mtcars.csv')

# Tạo bố cục 2 hàng, 2 cột
fig, axes = plt.subplots(2, 2, figsize=(10, 10))

# Biểu đồ histogram 1: mpg (5 bins)
axes[0, 0].hist(mtcars['mpg'], bins=5, color='white', edgecolor='black')
axes[0, 0].set_title('Histogram of mtcars$mpg', fontsize=12)
axes[0, 0].set_xlabel('mtcars$mpg', fontsize=10)
axes[0, 0].set_ylabel('Frequency', fontsize=10)

# Biểu đồ histogram 2: hp (6 bins)
axes[0, 1].hist(mtcars['hp'], bins=6, color='white', edgecolor='black')
axes[0, 1].set_title('Histogram of mtcars$hp', fontsize=12)
axes[0, 1].set_xlabel('mtcars$hp', fontsize=10)
axes[0, 1].set_ylabel('Frequency', fontsize=10)

# Biểu đồ histogram 3: disp (9 bins)
axes[1, 0].hist(mtcars['disp'], bins=9, color='white', edgecolor='black')
axes[1, 0].set_title('Histogram of mtcars$disp', fontsize=12)
axes[1, 0].set_xlabel('mtcars$disp', fontsize=10)
axes[1, 0].set_ylabel('Frequency', fontsize=10)

# Biểu đồ histogram 4: qsec (9 bins)
axes[1, 1].hist(mtcars['qsec'], bins=9, color='white', edgecolor='black')
axes[1, 1].set_title('Histogram of mtcars$qsec', fontsize=12)
axes[1, 1].set_xlabel('mtcars$qsec', fontsize=10)
axes[1, 1].set_ylabel('Frequency', fontsize=10)

# Tự động điều chỉnh khoảng cách giữa các biểu đồ
plt.tight_layout()

# Hiển thị biểu đồ
plt.show()

```


Histogram of mtcars$mpg
Frequency
mtcars$mpg
0
2
4
6
8
10
12
10
15
20
25
30
35


Histogram of mtcars$hp
Frequency
mtcars$hp
0
2
4
6
8
50
100
150
200
250
300


Histogram of mtcars$disp
Frequency
mtcars$disp
0
1
2
3
4
5
6
7
8
100
150
200
250
300
350
400
450


Histogram of mtcars$qsec
Frequency
mtcars$qsec
0
1
2
3
4
5
6
7
16
18
20
22

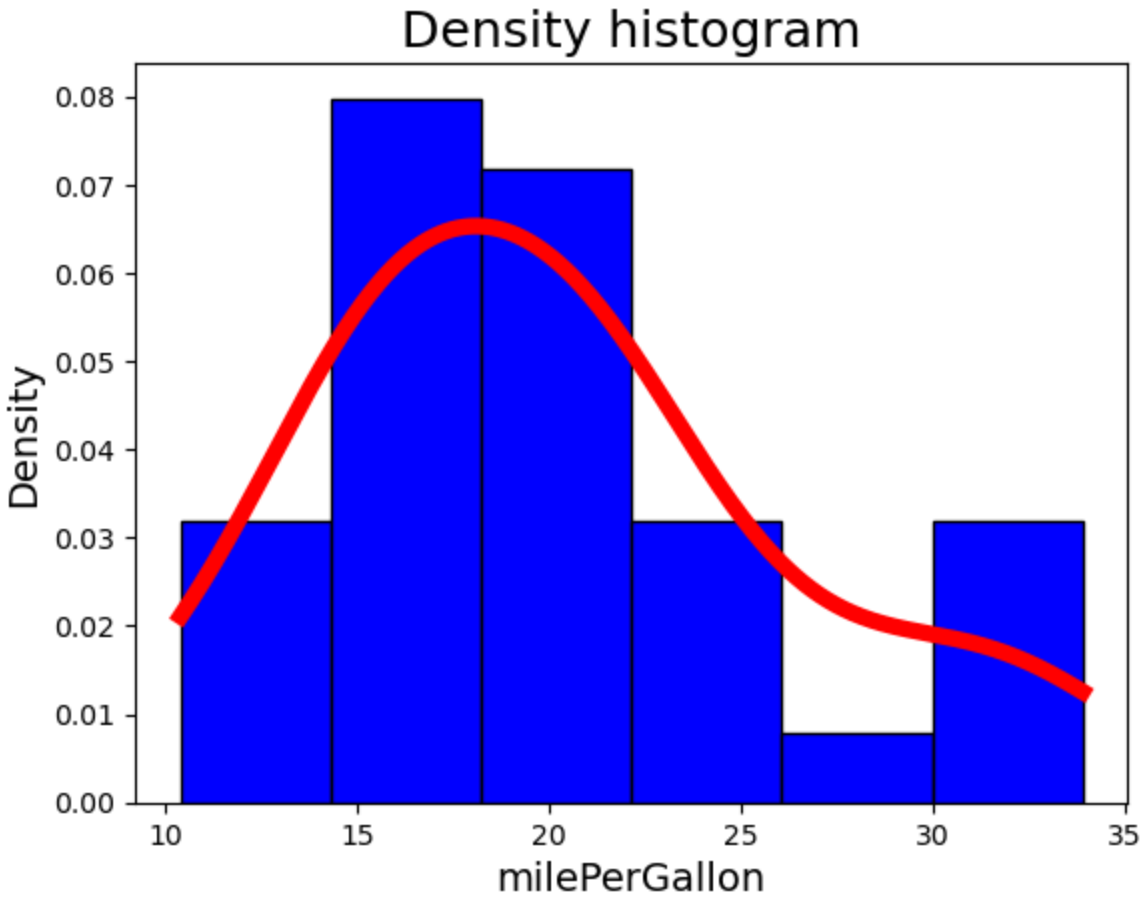
III.Density plot

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 [57]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import gaussian_kde

# Đọc dữ liệu mtcars từ file CSV
mtcars = pd.read_csv('mtcars.csv')

# Lấy cột 'mpg' (mile per gallon)
mile_per_gallon = mtcars['mpg']

# Vẽ histogram với xác suất (density=True tương đương prob = TRUE trong R)
plt.hist(mile_per_gallon, bins=6, density=True, color='blue', alpha=1.0, (

# Tính density và vẽ đường density
density = gaussian_kde(mile_per_gallon)
x_vals = np.linspace(mile_per_gallon.min(), mile_per_gallon.max(), 1000)
plt.plot(x_vals, density(x_vals), color='red', linewidth=6, label='Density

# Thêm tiêu đề và nhãn trục
plt.title("Density histogram", fontsize=18)
plt.xlabel("milePerGallon", fontsize=14)
plt.ylabel("Density", fontsize=14)

# Hiển thị biểu đồ
plt.show()

```



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.

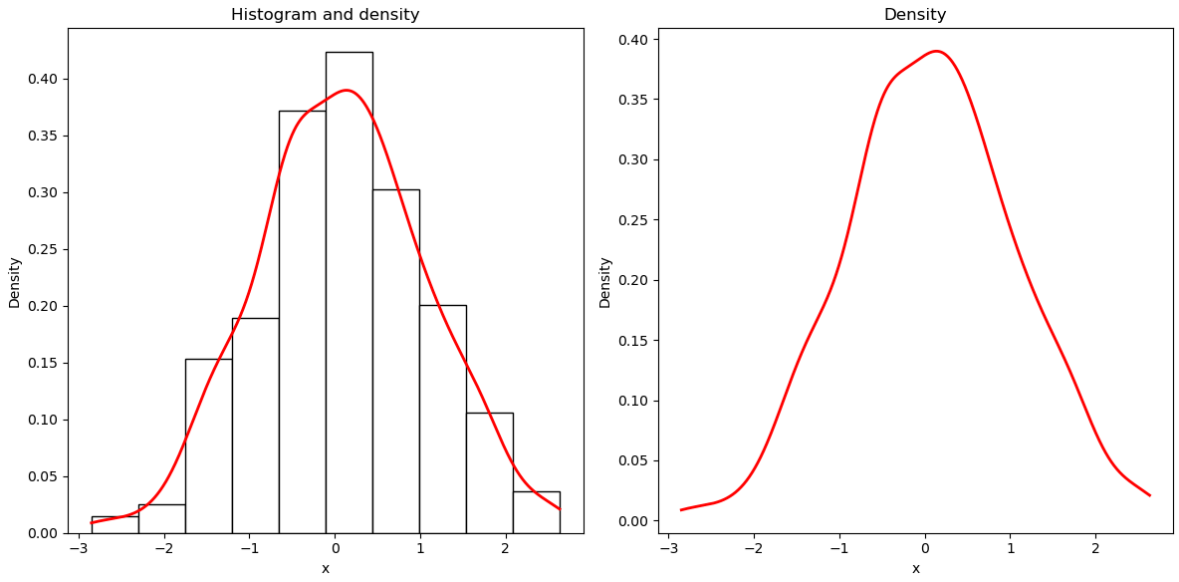
In [58]:
```
# 2.Đọc file mtcars.csv
df = pd.read_csv("mtcars.csv")

# Hiển thị 5 dòng đầu tiên
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 [59]:

```python
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import gaussian_kde

# Sinh dữ liệu ngẫu nhiên (500 giá trị) giống như rnorm trong R
x = np.random.normal(size=500)

# Tạo layout 1 hàng, 2 cột
fig, axes = plt.subplots(1, 2, figsize=(12, 6))

# Biểu đồ 1: Histogram và đường mật độ
axes[0].hist(x, bins=10, density=True, color='white', edgecolor='black')
density = gaussian_kde(x)
x_vals = np.linspace(min(x), max(x), 1000)
axes[0].plot(x_vals, density(x_vals), color='red', linewidth=2)
axes[0].set_title("Histogram and density")
axes[0].set_xlabel("x")
axes[0].set_ylabel("Density")

# Biểu đồ 2: Chỉ vẽ đường mật độ
axes[1].plot(x_vals, density(x_vals), color='red', linewidth=2)
axes[1].set_title("Density")
axes[1].set_xlabel("x")
axes[1].set_ylabel("Density")

# Hiển thị biểu đồ
plt.tight_layout()
plt.show()

```



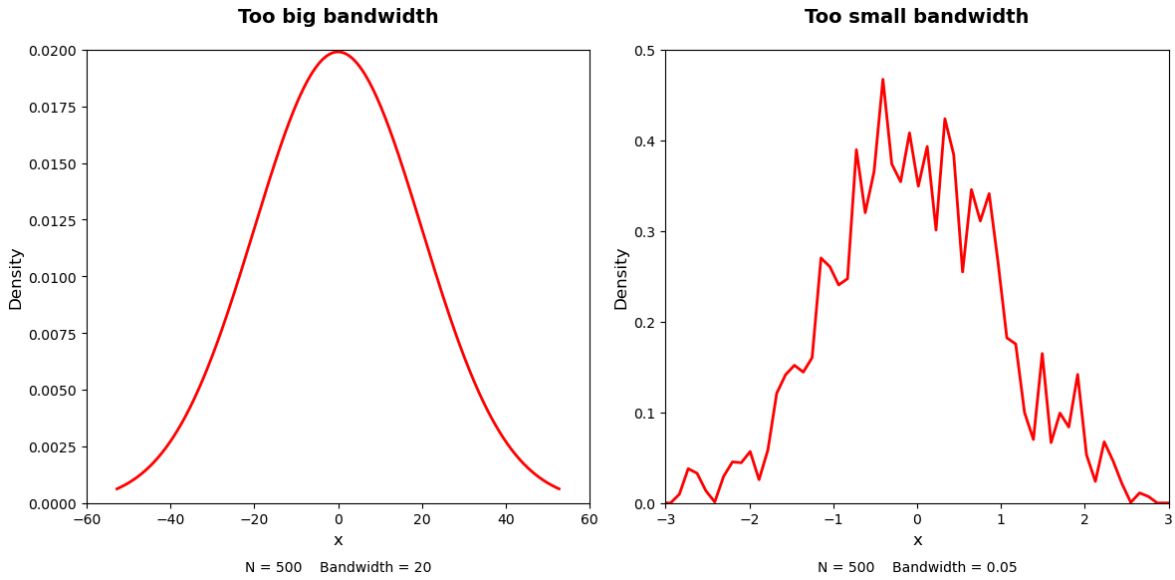
Vẽ density plot với hai băng thông (bandwidth) khác nhau sử dụng Matplotlib và Seaborn.

Mục đích:

Density plot với các băng thông khác nhau giúp dễ dàng khám phá các xu hướng tiềm ẩn và bất thường trong dữ liệu, hỗ trợ trong phân tích thống kê và xây dựng mô hình.

In [60]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import gaussian_kde

# Đọc dữ liệu mtcars từ file
mtcars = pd.read_csv('mtcars.csv')

# Tạo dữ liệu ngẫu nhiên giống R
np.random.seed(0)
x = np.random.normal(size=500)

# Tạo layout 1 hàng, 2 cột
fig, axes = plt.subplots(1, 2, figsize=(12, 6))

# Đồ thị 1: Bandwidth lớn
density_big_bw = gaussian_kde(x, bw_method=20 / np.std(x))  # bw = 20
x_vals = np.linspace(min(x) - 50, max(x) + 50, 1000)
axes[0].plot(x_vals, density_big_bw(x_vals), color='red', linewidth=2)
axes[0].set_title("Too big bandwidth", fontsize=14, fontweight='bold', pa
axes[0].set_xlabel("x", fontsize=12)
axes[0].set_ylabel("Density", fontsize=12)
axes[0].set_xlim(-60, 60)
axes[0].set_ylim(0, 0.02)
axes[0].text(0.5, -0.15, "N = 500    Bandwidth = 20", ha='center', fontsi

# Đồ thị 2: Bandwidth nhỏ
density_small_bw = gaussian_kde(x, bw_method=0.05 / np.std(x))  # bw = 0.
axes[1].plot(x_vals, density_small_bw(x_vals), color='red', linewidth=2)
axes[1].set_title("Too small bandwidth", fontsize=14, fontweight='bold',
axes[1].set_xlabel("x", fontsize=12)
axes[1].set_ylabel("Density", fontsize=12)
axes[1].set_xlim(-3, 3)
axes[1].set_ylim(0, 0.5)
axes[1].text(0.5, -0.15, "N = 500    Bandwidth = 0.05", ha='center', font

# Hiển thị đồ thị
plt.tight_layout()
plt.show()

```


Too big bandwidth
Density
x
N = 500 Bandwidth = 20
Too small bandwidth
Density
x
N = 500 Bandwidth = 0.05

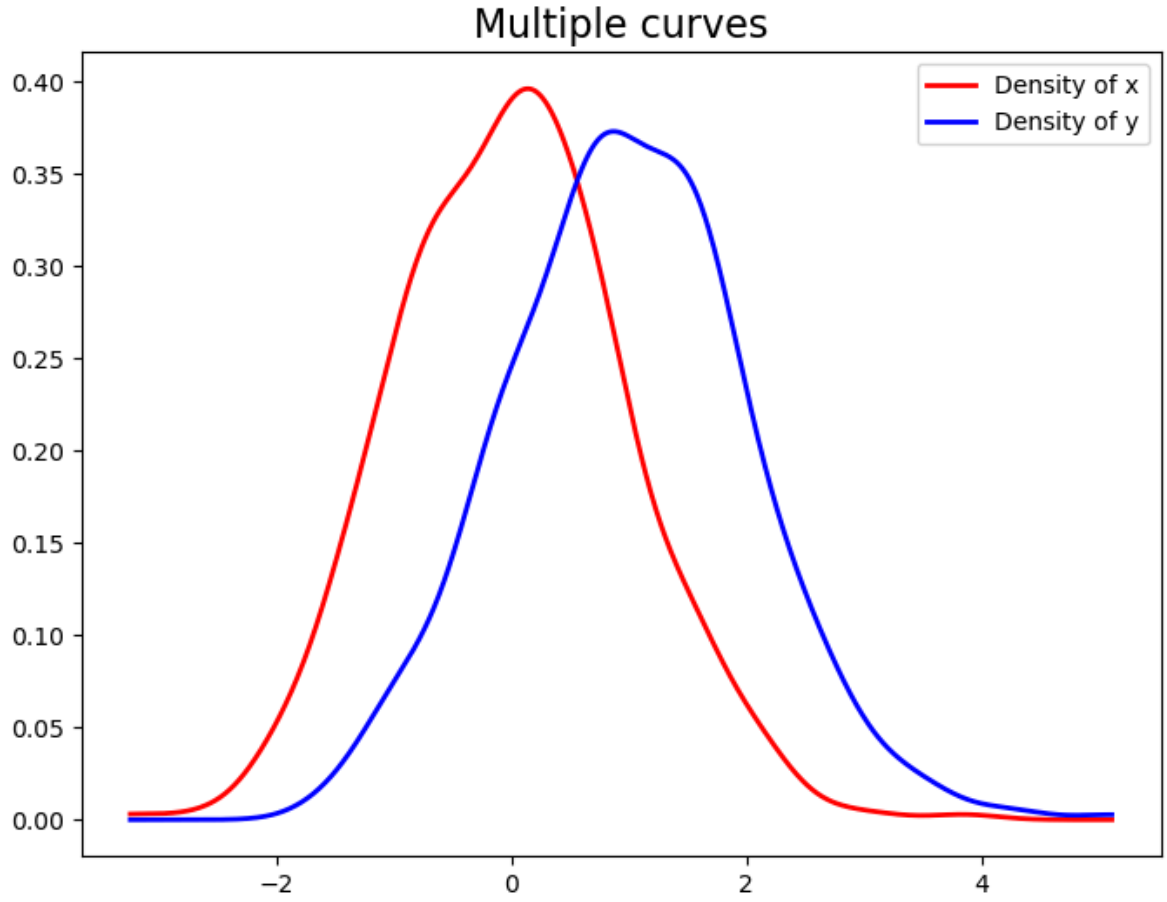
Vẽ nhiều đường mật độ (density curves) sử dụng Matplotlib và Scipy.

Mục đích:

Việc vẽ nhiều đường mật độ là một cách mạnh mẽ để phân tích, so sánh và diễn giải dữ liệu.

In [61]:

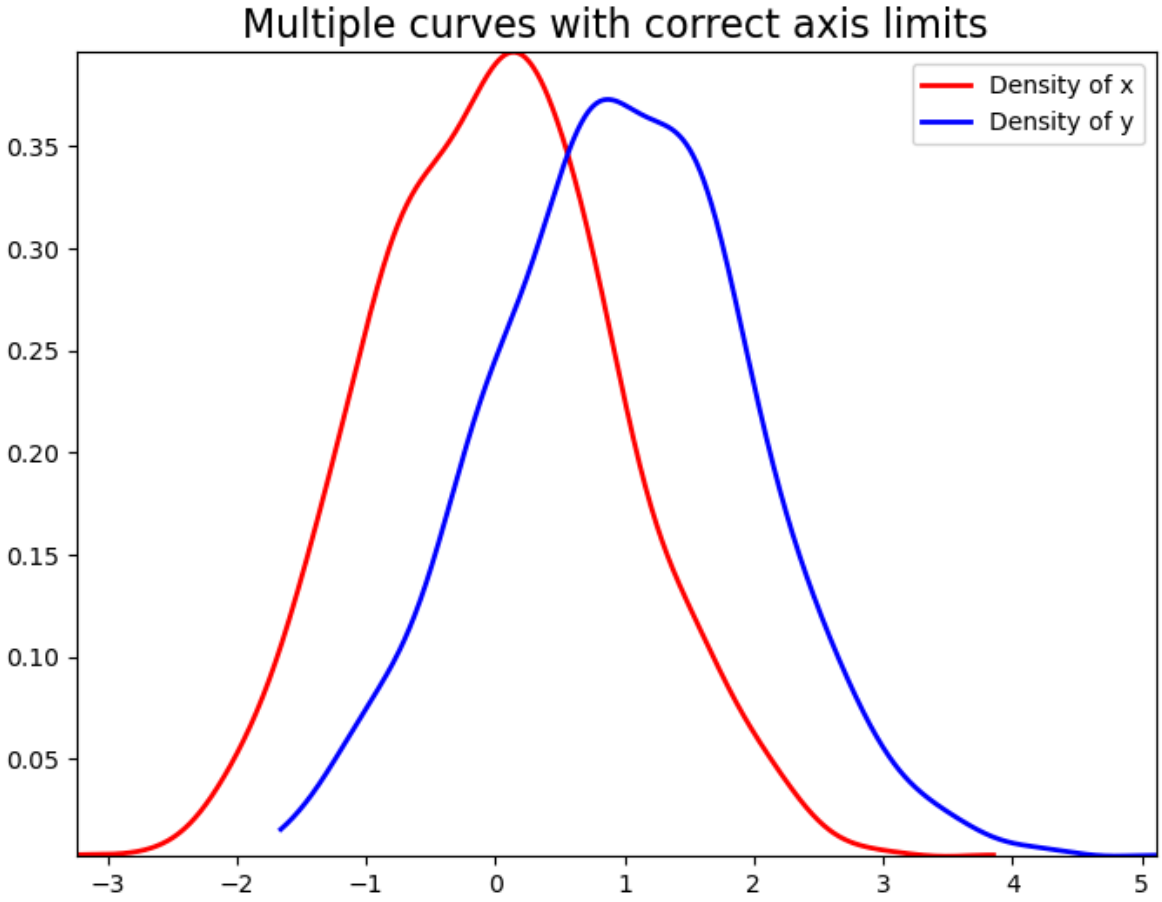
```
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import gaussian_kde

# Tạo dữ liệu x
np.random.seed(42)
x = np.random.normal(size=500)

# Tạo dữ liệu y
np.random.seed(2)
y = np.random.normal(size=500) + 1

# Tính mật độ cho x
density_x = gaussian_kde(x)

# Tính mật độ cho y
density_y = gaussian_kde(y)

# Tạo lưới giá trị x để vẽ mật độ
x_grid = np.linspace(min(min(x), min(y)), max(max(x), max(y)), 1000)

# Vẽ đồ thị
plt.figure(figsize=(8, 6))
plt.plot(x_grid, density_x(x_grid), color='red', linewidth=2, label='Dens
plt.plot(x_grid, density_y(x_grid), color='blue', linewidth=2, label='Den
plt.title("Multiple curves", fontsize=16)
plt.xlabel("", fontsize=12)
plt.legend()
plt.show()

```



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

In [62]:

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import gaussian_kde

# Tạo dữ liệu
np.random.seed(42)
x = np.random.normal(size=500)

np.random.seed(2)
y = np.random.normal(size=500) + 1

# Tính mật độ
density_x = gaussian_kde(x)
density_y = gaussian_kde(y)

# Lưới giá trị cho trục x
x_grid_x = np.linspace(min(x), max(x), 1000)
x_grid_y = np.linspace(min(y), max(y), 1000)

# Xác định giới hạn trục x và y
x_min = min(x.min(), y.min())
x_max = max(x.max(), y.max())
y_min = min(density_x(x_grid_x).min(), density_y(x_grid_y).min())
y_max = max(density_x(x_grid_x).max(), density_y(x_grid_y).max())

# Vẽ đồ thị
plt.figure(figsize=(8, 6))
plt.plot(x_grid_x, density_x(x_grid_x), color='red', linewidth=2, label='
plt.plot(x_grid_y, density_y(x_grid_y), color='blue', linewidth=2, label=
plt.title("Multiple curves with correct axis limits", fontsize=16)
plt.xlabel("", fontsize=12)
plt.xlim(x_min, x_max)  # Giới hạn trục x
plt.ylim(y_min, y_max)  # Giới hạn trục y
plt.legend()
plt.show()

```


Multiple curves with correct axis limits
Density of x
Density of y
0.35
0.30
0.25
0.20
0.15
0.10
0.05
−3
−2
−1
0
1
2
3
4
5

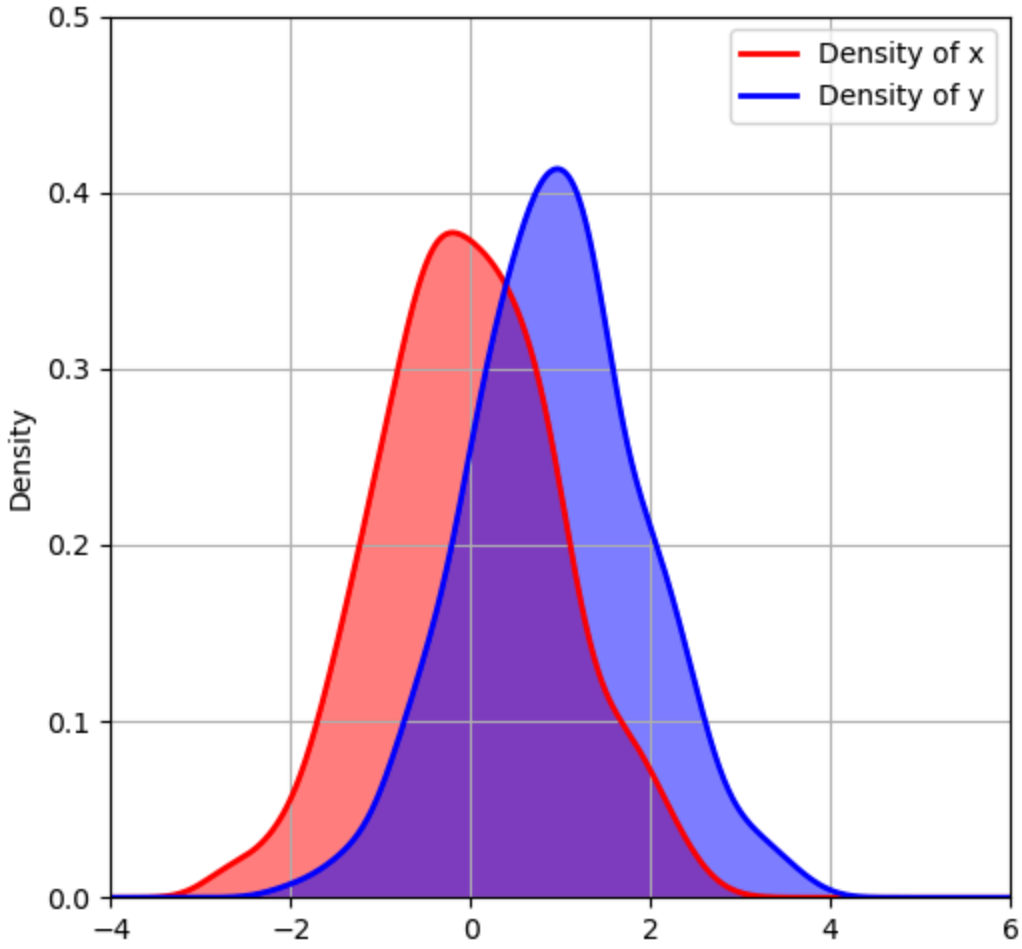
Shade area under curve with transparency

Mục đích:

Giúp tăng cường độ rõ ràng, dễ đọc và khả năng diễn giải của biểu đồ, làm cho chúng hiệu quả hơn trong việc truyền tải thông tin và phân tích.

In [63]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import gaussian_kde

# Tạo dữ liệu x và y
np.random.seed(0)
x = np.random.normal(size=500)
y = np.random.normal(size=500) + 1

# Tính mật độ (density) cho x và y
density_x = gaussian_kde(x)
density_y = gaussian_kde(y)
x_vals = np.linspace(-4, 6, 1000)
y_vals = np.linspace(-4, 6, 1000)

# Vẽ biểu đồ
plt.figure(figsize=(10, 5))

# Biểu đồ đầu tiên: density của x và y với vùng tô màu
plt.subplot(1, 2, 1)
plt.plot(x_vals, density_x(x_vals), color='red', linewidth=2, label='Dens
plt.fill_between(x_vals, density_x(x_vals), color='red', alpha=0.5)
plt.plot(y_vals, density_y(y_vals), color='blue', linewidth=2, label='Den
plt.fill_between(y_vals, density_y(y_vals), color='blue', alpha=0.5)
plt.xlim(-4, 6)
plt.ylim(0, 0.5)
plt.xlabel("")
plt.ylabel("Density")
plt.title("", fontsize=14)
plt.legend(loc='upper right')
plt.grid(True)

# Hiển thị đồ thị
plt.tight_layout()
plt.show()

```



IV.Bar Chart.

Bar Chart using mtcars dataset

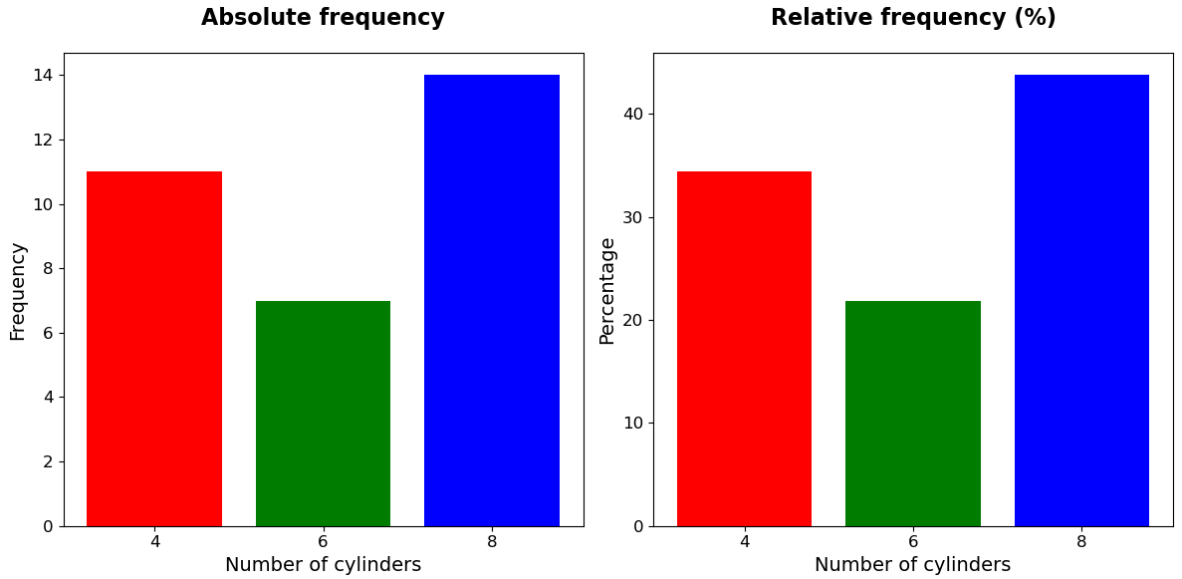
This is for plotting discrete and/or categorical values;

values that take a finite number or categorize items

Making a Frequency table of Column

In [64]:

```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np

# Đọc dữ liệu mtcars từ file
mtcars = pd.read_csv('mtcars.csv')

# Tạo bảng tần suất tuyệt đối cho số xi-lanh (cột 'cyl')
my_table = mtcars['cyl'].value_counts().sort_index()

# Bố cục 1 hàng, 2 cột
plt.figure(figsize=(12, 6))

# Biểu đồ tần suất tuyệt đối
plt.subplot(1, 2, 1)
plt.bar(my_table.index.astype(str), my_table.values, color=['red', 'green
plt.title("Absolute frequency", fontsize=16, fontweight='bold', pad=20)
plt.xlabel("Number of cylinders", fontsize=14)
plt.ylabel("Frequency", fontsize=14)
plt.xticks(fontsize=12)
plt.yticks(fontsize=12)

# Biểu đồ tần suất tương đối
plt.subplot(1, 2, 2)
relative_freq = (my_table / my_table.sum()) * 100
plt.bar(relative_freq.index.astype(str), relative_freq.values, color=['re
plt.title("Relative frequency (%)", fontsize=16, fontweight='bold', pad=2
plt.xlabel("Number of cylinders", fontsize=14)
plt.ylabel("Percentage", fontsize=14)
plt.xticks(fontsize=12)
plt.yticks(fontsize=12)

# Hiển thị đồ thị
plt.tight_layout()
plt.show()

```



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.

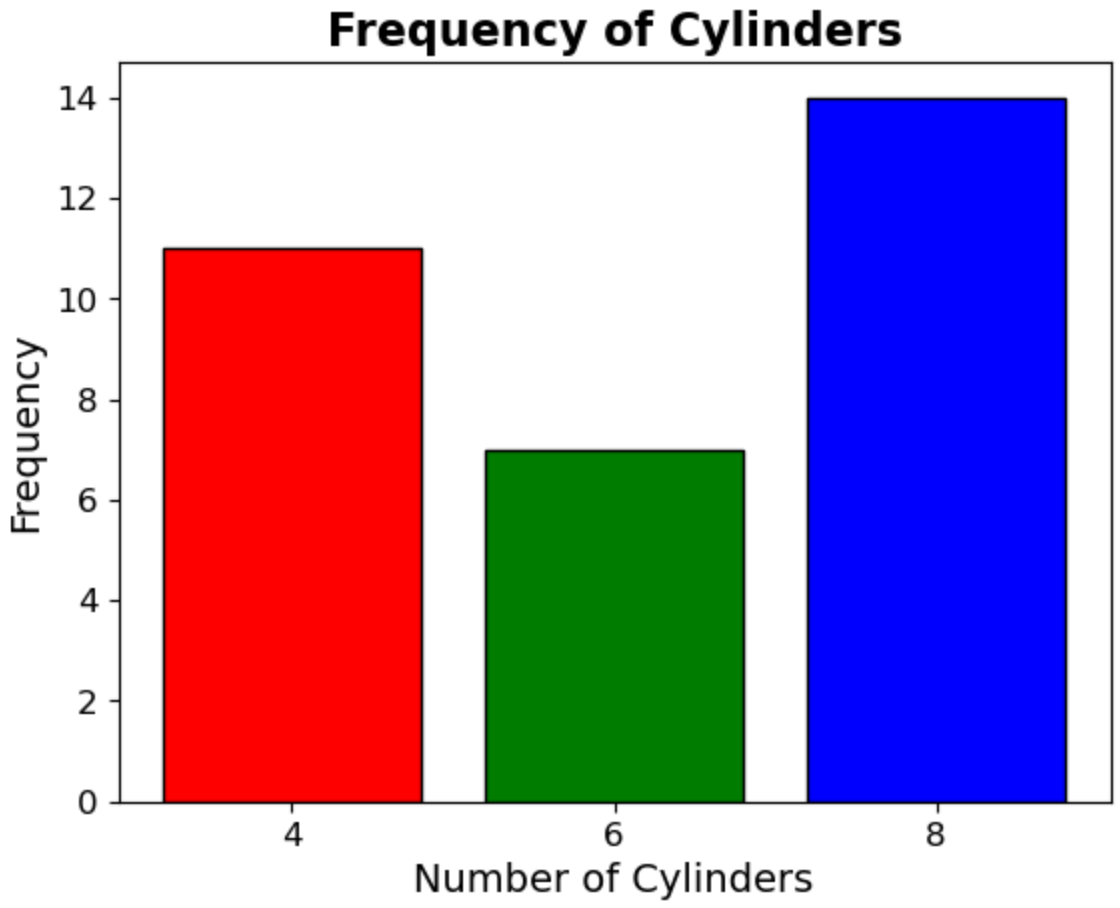
Biểu đồ Tần suất Tương đối:

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 [65]:

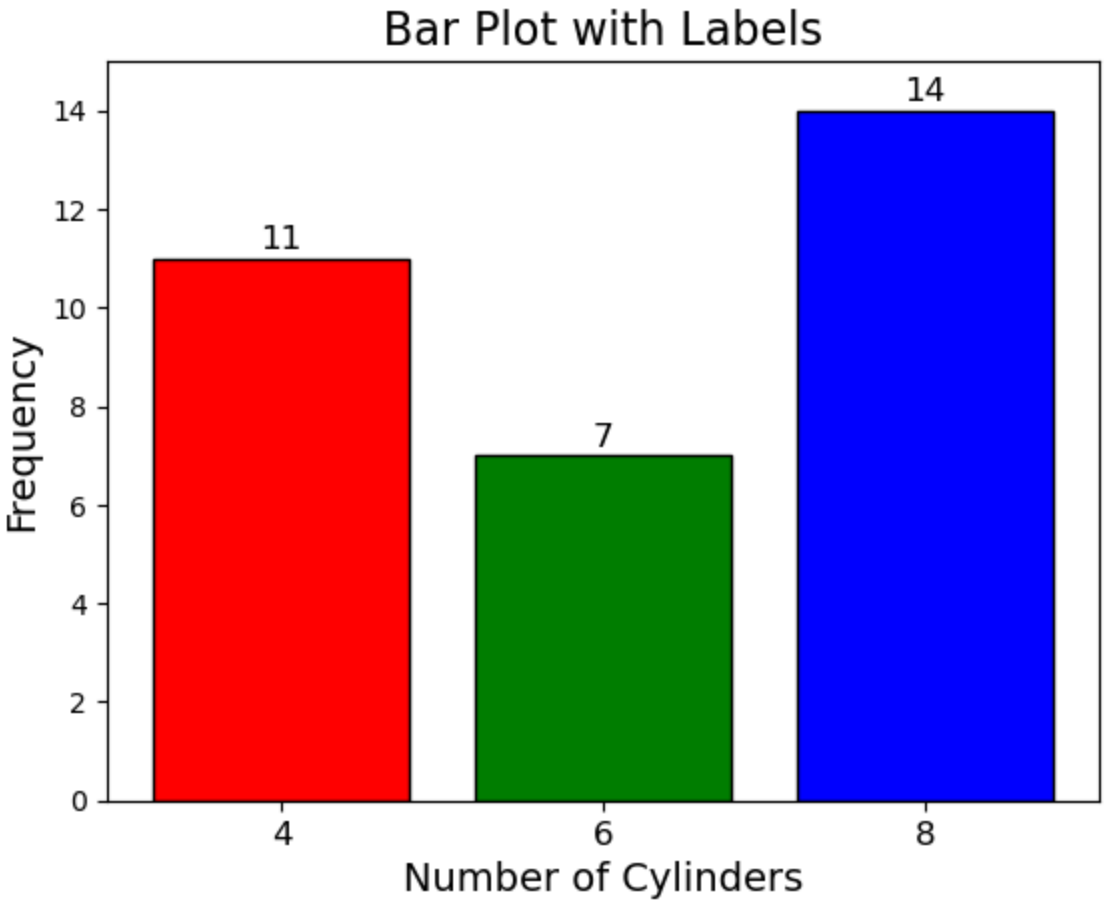
```python
import pandas as pd
import matplotlib.pyplot as plt

# Đọc dữ liệu mtcars từ file
mtcars = pd.read_csv('mtcars.csv')

# Tạo bảng tần suất của số xi-lanh (cyl)
freq_table = mtcars['cyl'].value_counts().sort_index()

# Vẽ biểu đồ với màu đỏ, xanh lá, xanh dương
colors = ['red', 'green', 'blue']
plt.bar(freq_table.index.astype(str), freq_table.values, color=colors, ed

# Thêm tiêu đề và nhãn trục
plt.title("Frequency of Cylinders", fontsize=16, fontweight='bold')
plt.xlabel("Number of Cylinders", fontsize=14)
plt.ylabel("Frequency", fontsize=14)

# Điều chỉnh vị trí nhãn trục x
plt.xticks(fontsize=12)
plt.yticks(fontsize=12)

# Hiển thị biểu đồ
plt.show()

```



Text can be written on the bar plot as below

In [66]:

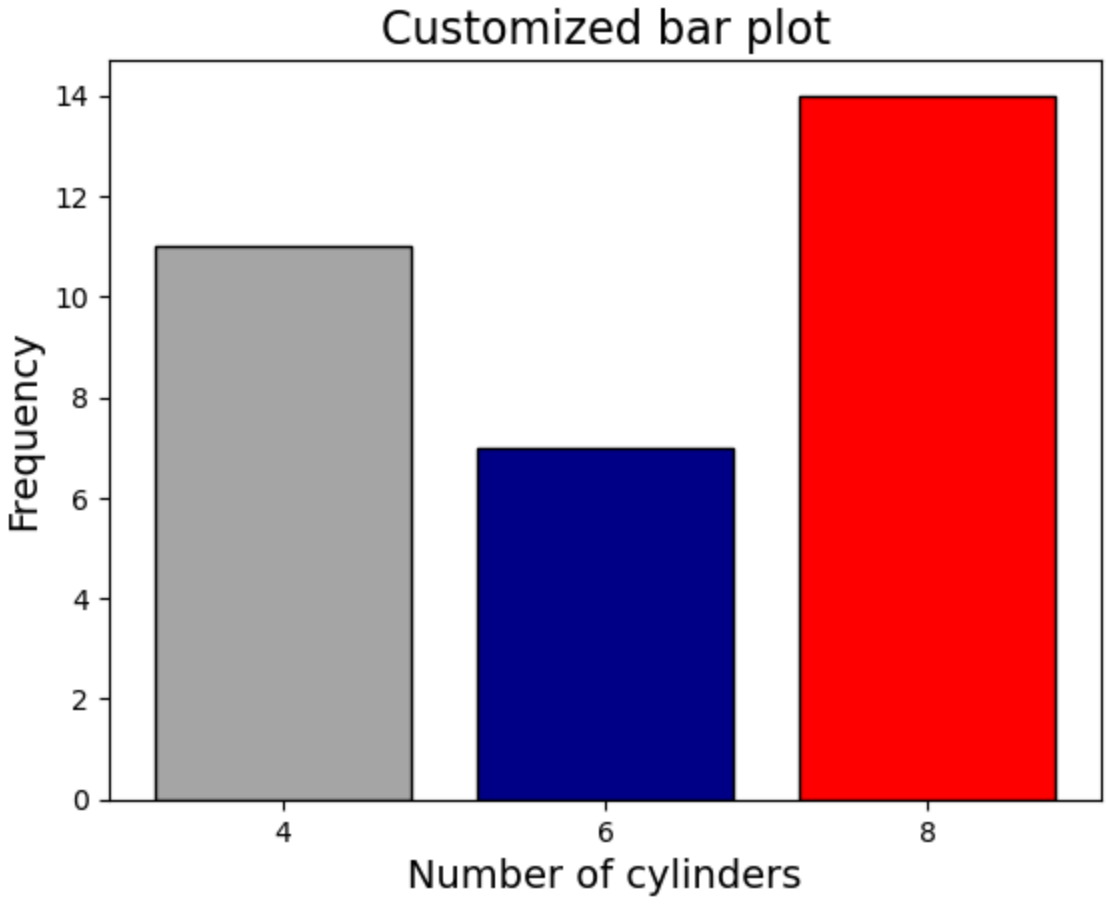
```python
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np

# Đọc dữ liệu mtcars từ file
mtcars = pd.read_csv('mtcars.csv')

# Tạo bảng tần suất của số xi-lanh (cyl)
my_table = mtcars['cyl'].value_counts().sort_index()

# Vẽ biểu đồ cột
colors = ['red', 'green', 'blue']  # Màu sắc theo thứ tự giống R
bar_positions = np.arange(len(my_table))  # Vị trí các cột
plt.bar(bar_positions, my_table.values, color=colors, edgecolor='black')

# Thêm nhãn vào các cột
for i, value in enumerate(my_table.values):
    plt.text(bar_positions[i], value + 0.2, str(value), ha='center', font

# Cài đặt trục y
plt.ylim(0, 15)

# Thêm tiêu đề và nhãn trục
plt.title("Bar Plot with Labels", fontsize=16)
plt.xlabel("Number of Cylinders", fontsize=14)
plt.ylabel("Frequency", fontsize=14)

# Đặt nhãn trục x
plt.xticks(bar_positions, my_table.index.astype(str), fontsize=12)

# Hiển thị biểu đồ
plt.show()

```



“SALT” the Bar Plot

In [67]:

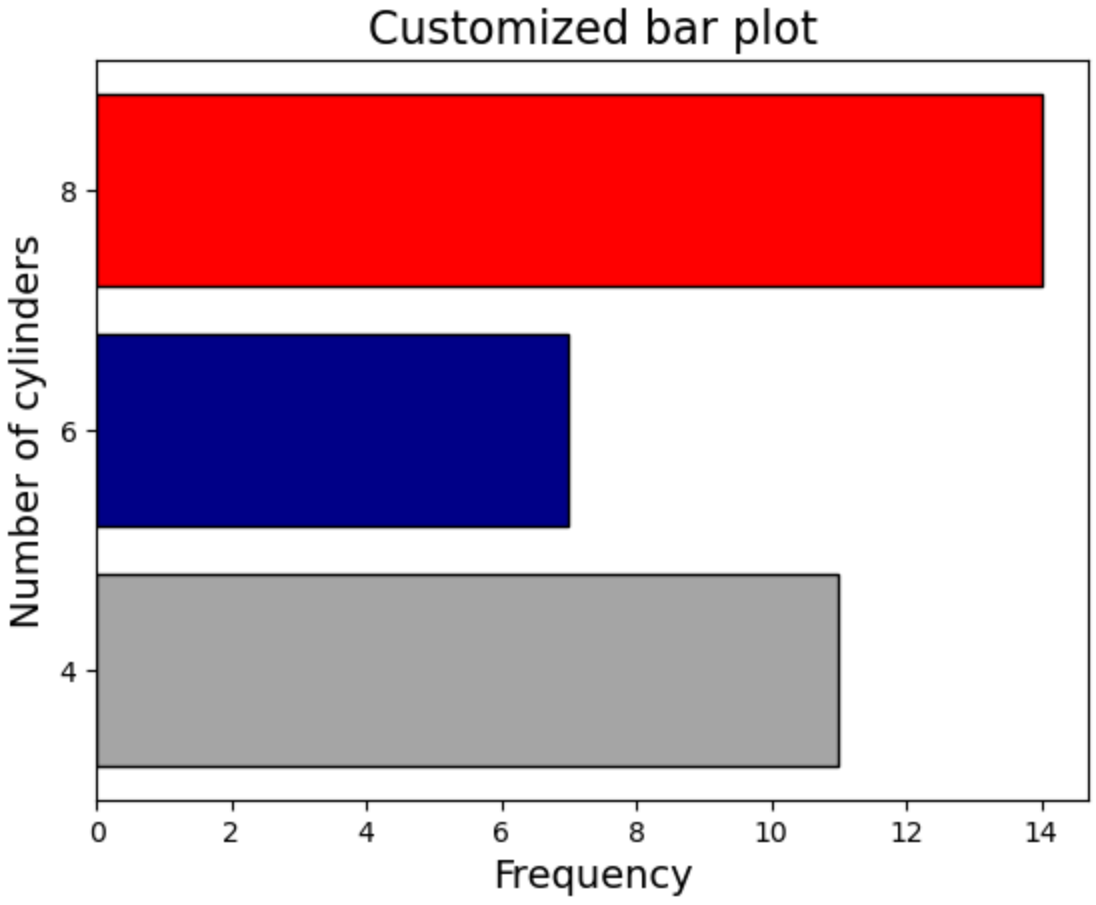
```
import pandas as pd
import matplotlib.pyplot as plt

# Đọc dữ liệu mtcars từ file
mtcars = pd.read_csv('mtcars.csv')

# Tạo bảng tần suất của số xi-lanh (cyl)
my_table = mtcars['cyl'].value_counts().sort_index()

# Màu sắc cho các cột
colors = ['darkgrey', 'darkblue', 'red']

# Vẽ biểu đồ cột
plt.bar(my_table.index.astype(str), my_table.values, color=colors, edgeco

# Thêm tiêu đề và nhãn trục
plt.title("Customized bar plot", fontsize=16)
plt.xlabel("Number of cylinders", fontsize=14)
plt.ylabel("Frequency", fontsize=14)

# Hiển thị biểu đồ
plt.show()

```



Horizontal Barplot

In [68]:

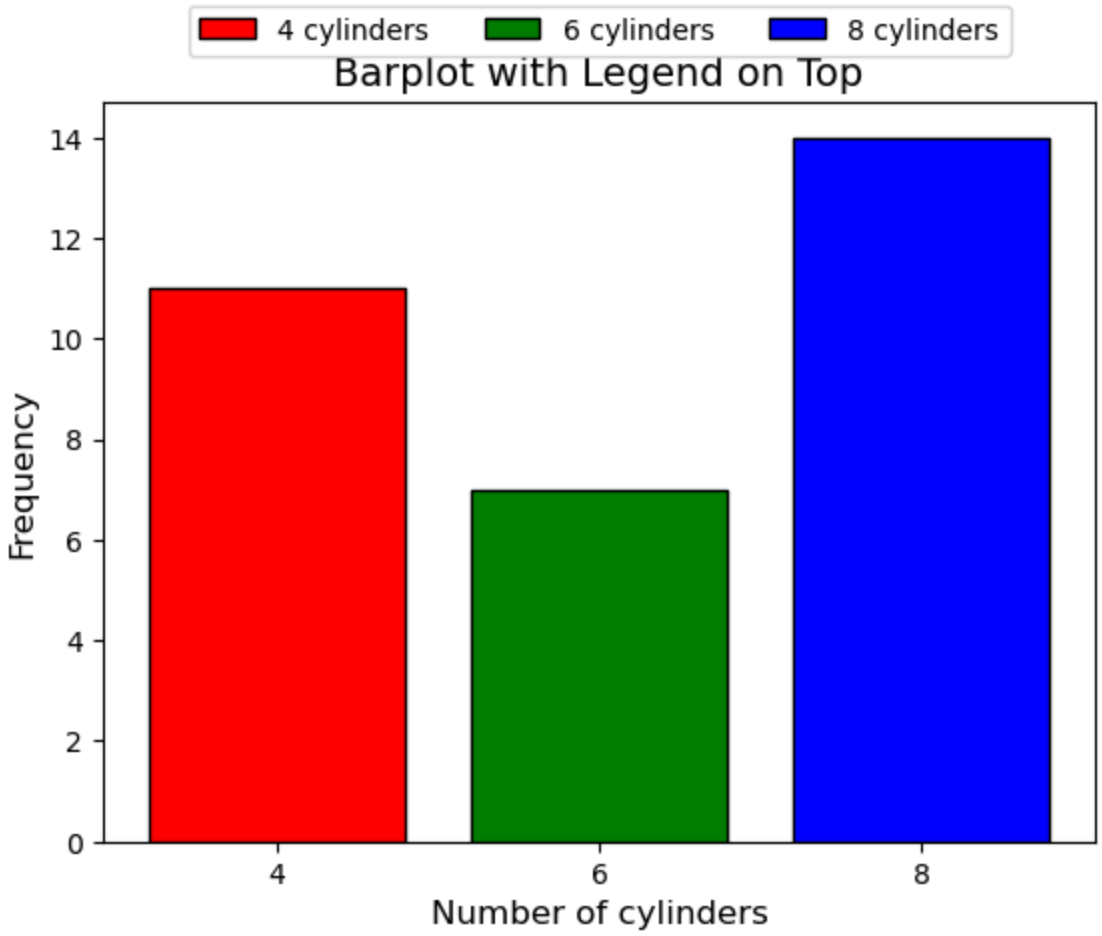
```
import pandas as pd
import matplotlib.pyplot as plt

# Đọc dữ liệu mtcars từ file
mtcars = pd.read_csv('mtcars.csv')

# Tạo bảng tần suất của số xi-lanh (cyl)
my_table = mtcars['cyl'].value_counts().sort_index()

# Màu sắc cho các cột
colors = ['darkgrey', 'darkblue', 'red']

# Vẽ biểu đồ cột ngang
plt.barh(my_table.index.astype(str), my_table.values, color=colors, edgec

# Thêm tiêu đề và nhãn trục
plt.title("Customized bar plot", fontsize=16)
plt.xlabel("Frequency", fontsize=14)
plt.ylabel("Number of cylinders", fontsize=14)

# Hiển thị biểu đồ
plt.show()

```



Barplot with Legend

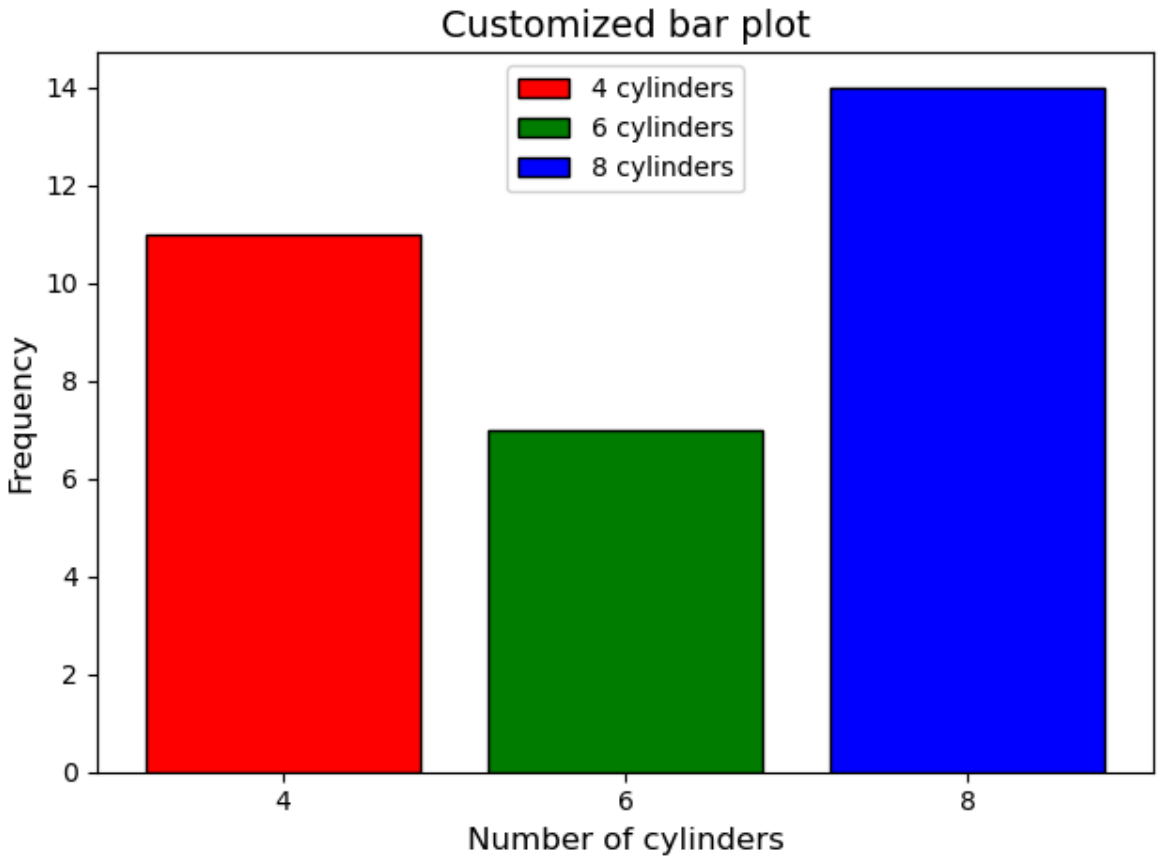
In [69]:

```
import pandas as pd
import matplotlib.pyplot as plt

# Đọc dữ liệu mtcars từ file
mtcars = pd.read_csv('mtcars.csv')

# Tạo bảng tần suất của số xi-lanh (cyl)
my_table = mtcars['cyl'].value_counts().sort_index()

# Màu sắc cho các cột (tương tự rainbow(3) trong R)
colors = ['red', 'green', 'blue']

# Vẽ biểu đồ cột
bars = plt.bar(my_table.index.astype(str), my_table.values, color=colors,

# Thêm chú thích (legend) phía trên
plt.legend(bars, [f"{cyl} cylinders" for cyl in my_table.index], loc='upp

# Thêm nhãn trục và tiêu đề
plt.xlabel("Number of cylinders", fontsize=12)
plt.ylabel("Frequency", fontsize=12)
plt.title("Barplot with Legend on Top", fontsize=14)

# Hiển thị biểu đồ
plt.show()

```



In [70]:

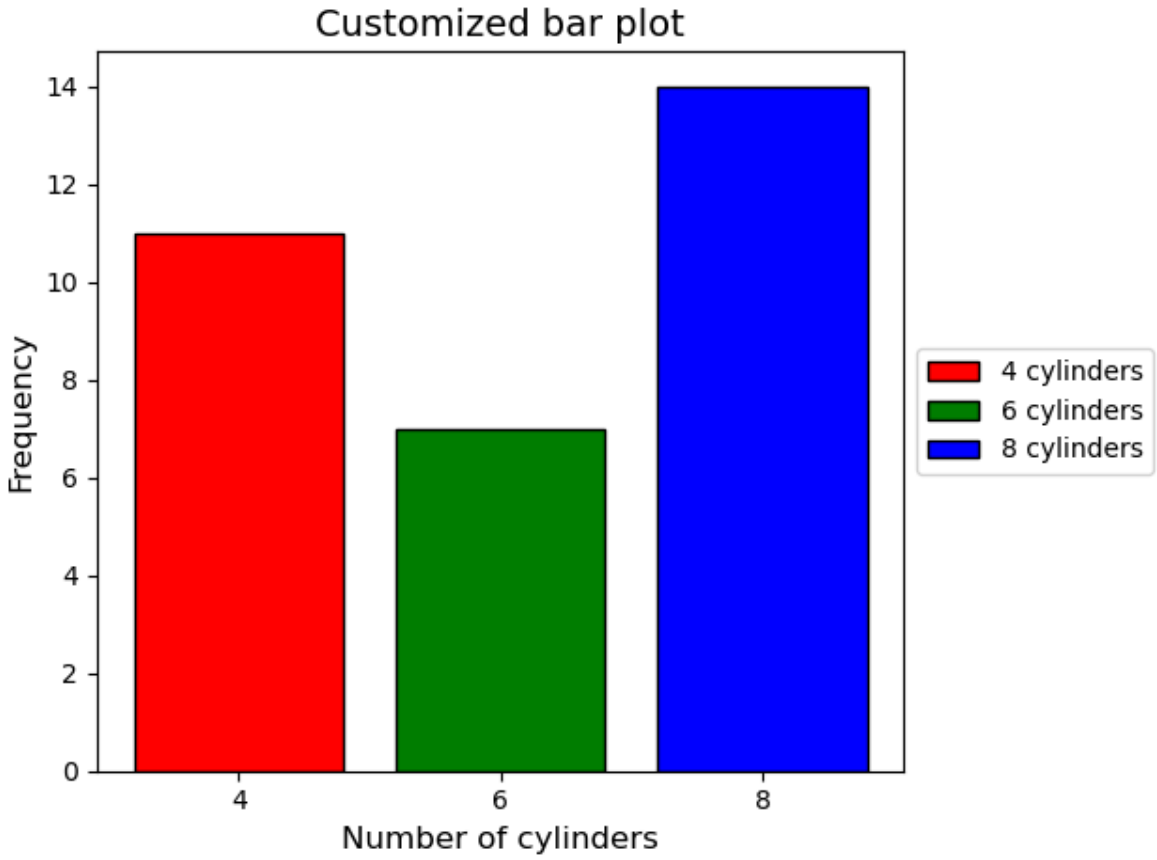
```python
import pandas as pd
import matplotlib.pyplot as plt

# Đọc dữ liệu mtcars từ file
mtcars = pd.read_csv('mtcars.csv')

# Tạo bảng tần suất của số xi-lanh (cyl)
my_table = mtcars['cyl'].value_counts().sort_index()

# Màu sắc cho các cột (tương tự rainbow(3) trong R)
colors = ['red', 'green', 'blue']

# Vẽ biểu đồ cột
bars = plt.bar(my_table.index.astype(str), my_table.values, color=colors,

# Thêm chú thích (legend) với một cột, đặt bên dưới tiêu đề
plt.legend(bars, [f"{cyl} cylinders" for cyl in my_table.index], loc='upp

# Thêm nhãn trục và tiêu đề
plt.xlabel("Number of cylinders", fontsize=12)
plt.ylabel("Frequency", fontsize=12)
plt.title("Customized bar plot", fontsize=14)

# Hiển thị biểu đồ
plt.tight_layout()  # Đảm bảo không bị cắt nội dung
plt.show()

```



Another legend on the right side

In [71]:

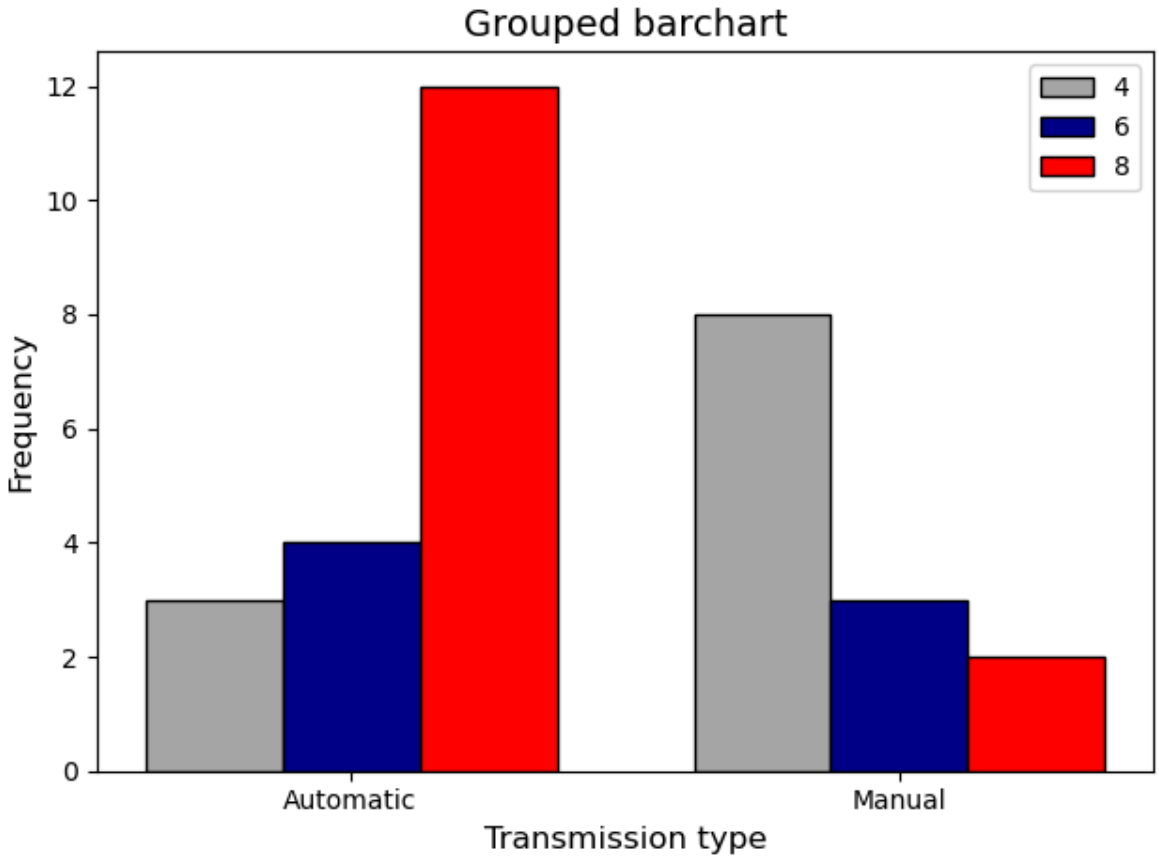
```python
import pandas as pd
import matplotlib.pyplot as plt

# Đọc dữ liệu mtcars từ file
mtcars = pd.read_csv('mtcars.csv')

# Tạo bảng tần suất của số xi-lanh (cyl)
my_table = mtcars['cyl'].value_counts().sort_index()

# Màu sắc cho các cột (tương tự rainbow(3) trong R)
colors = ['red', 'green', 'blue']

# Vẽ biểu đồ cột
bars = plt.bar(my_table.index.astype(str), my_table.values, color=colors,

# Thêm chú thích (legend) nằm bên phải của biểu đồ
plt.legend(bars, [f"{cyl} cylinders" for cyl in my_table.index], loc='cen

# Thêm nhãn trục và tiêu đề
plt.xlabel("Number of cylinders", fontsize=12)
plt.ylabel("Frequency", fontsize=12)
plt.title("Customized bar plot", fontsize=14)

# Hiển thị biểu đồ
plt.tight_layout()  # Đảm bảo không bị cắt nội dung
plt.show()

```



Grouping Bar Plot

In [72]:

```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np

# Tải dataset mtcars.csv từ file bên ngoài
mtcars = pd.read_csv('mtcars.csv')

# Chuyển đổi cột 'am' thành dạng phân loại
mtcars['am'] = mtcars['am'].map({0: "Automatic", 1: "Manual"})

# Tạo bảng tần suất giữa 'cyl' và 'am'
other_table = pd.crosstab(mtcars['cyl'], mtcars['am'])

# Màu sắc tùy chỉnh
colors = ["darkgrey", "darkblue", "red"]

# Vẽ barplot nhóm
bar_width = 0.25  # Chiều rộng của cột
x = np.arange(len(other_table.columns))  # Vị trí các nhóm (Automatic, Ma

fig, ax = plt.subplots()

# Lặp qua từng số xi-lanh và vẽ các nhóm
for i, (cyl, values) in enumerate(other_table.iterrows()):
    ax.bar(x + i * bar_width, values, bar_width,
           label=f"{cyl}", color=colors[i], edgecolor="black")

# Thêm tiêu đề và nhãn trục
ax.set_title("Grouped barchart", fontsize=14)
ax.set_xlabel("Transmission type", fontsize=12)
ax.set_ylabel("Frequency", fontsize=12)
ax.set_xticks(x + bar_width)  # Điều chỉnh nhãn trục x để nằm giữa
ax.set_xticklabels(other_table.columns, fontsize=10)

# Thêm legend
ax.legend(fontsize=10, title_fontsize=11, loc="upper right")

# Hiển thị biểu đồ
plt.tight_layout()
plt.show()

```


Grouped barchart
Frequency
Transmission type
Automatic
Manual
0
2
4
6
8
10
12
4
6
8

V.Pie Chart.

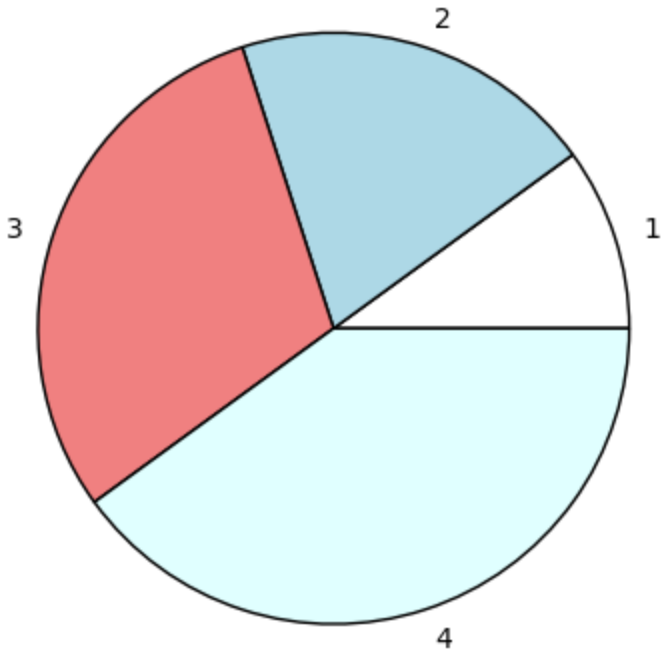
Pie or circular chart are used to represents discrete or categorical data.

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

The pie() function is used to implement it

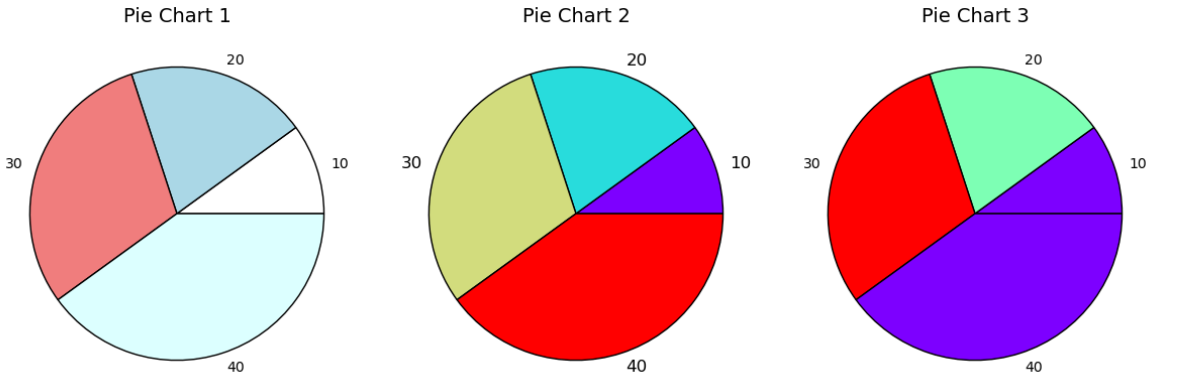
In [73]:

```
import matplotlib.pyplot as plt

# Tạo một danh sách chứa số lượng của mỗi phần trong Pie Chart
count = [10, 20, 30, 40]

# Màu sắc tương tự hình đính kèm
colors = ['white', 'lightblue', 'lightcoral', 'lightcyan']

# Vẽ Pie Chart đơn giản với viền
plt.pie(count,
        labels=[1, 2, 3, 4],
        colors=colors,
        wedgeprops={'edgecolor': 'black', 'linewidth': 1})

# Thêm tiêu đề
plt.title("Simple Pie Chart", fontsize=14)

# Hiển thị biểu đồ
plt.show()


```


Simple Pie Chart
2
1
3
4

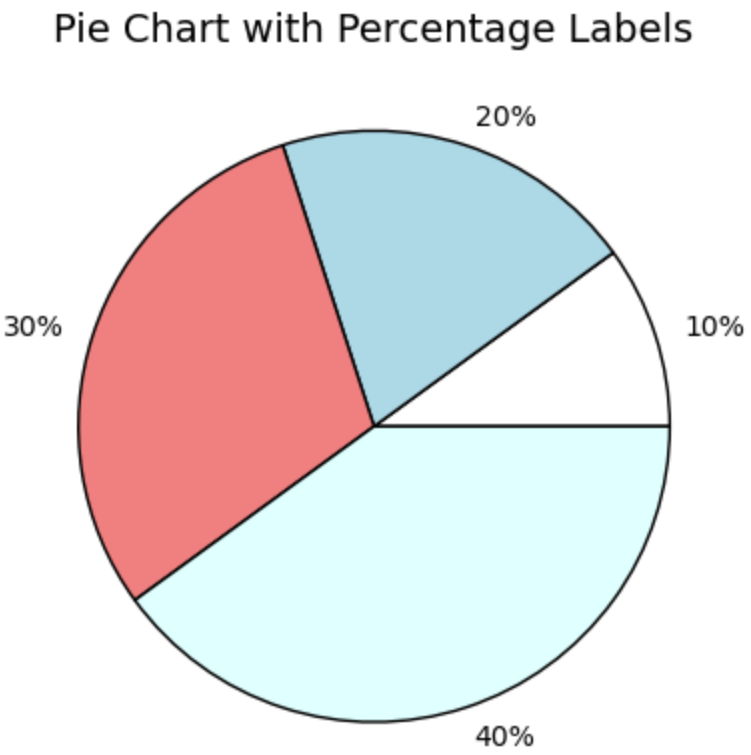
In [74]:

```
import matplotlib.pyplot as plt

# Dữ liệu cho Pie Chart
count = [10, 20, 30, 40]

# Tạo layout 1 hàng, 3 cột
fig, axes = plt.subplots(1, 3, figsize=(12, 4))  # figsize để điều chỉnh k

# Pie Chart 1: Nhãn và kích thước chữ lớn hơn
axes[0].pie(count, labels=count, colors=['white', 'lightblue', 'lightcora
            wedgeprops={'edgecolor': 'black', 'linewidth': 1}, textprops=
axes[0].set_title("Pie Chart 1", fontsize=14)

# Pie Chart 2: Màu cầu vồng, nhãn lớn hơn
axes[1].pie(count, labels=count, colors=plt.cm.rainbow(np.linspace(0, 1, 
            wedgeprops={'edgecolor': 'black', 'linewidth': 1}, textprops=
axes[1].set_title("Pie Chart 2", fontsize=14)

# Pie Chart 3: Màu cầu vồng (ít màu hơn), nhãn lớn hơn
axes[2].pie(count, labels=count, colors=plt.cm.rainbow(np.linspace(0, 1, 
            wedgeprops={'edgecolor': 'black', 'linewidth': 1}, textprops=
axes[2].set_title("Pie Chart 3", fontsize=14)

# Hiển thị biểu đồ
plt.tight_layout()
plt.show()


```



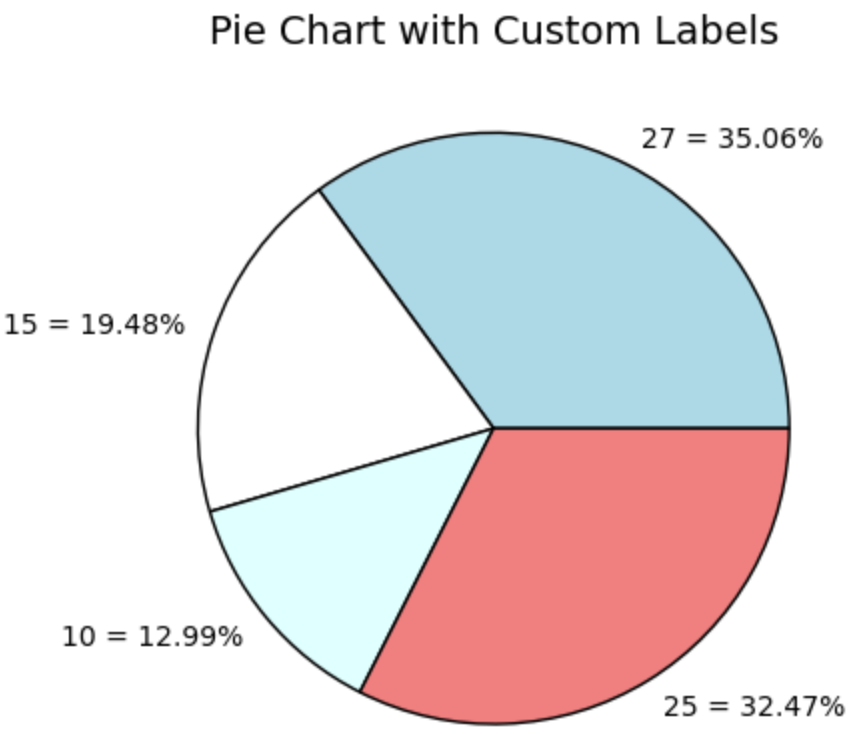
In [75]:

```python
import matplotlib.pyplot as plt

# Dữ liệu cho Pie Chart
count = [10, 20, 30, 40]
labels = [f"{c}%" for c in count]  # Tạo nhãn với dấu phần trăm

# Màu sắc giống hình đính kèm
colors = ['white', 'lightblue', 'lightcoral', 'lightcyan']

# Vẽ Pie Chart với nhãn phần trăm
plt.pie(count,
        labels=labels,
        colors=colors,
        autopct=None,  # Không hiển thị thêm phần trăm tự động
        wedgeprops={'edgecolor': 'black', 'linewidth': 1})  # Thêm viền cl

# Thêm tiêu đề
plt.title("Pie Chart with Percentage Labels", fontsize=14)

# Hiển thị biểu đồ
plt.show()

```



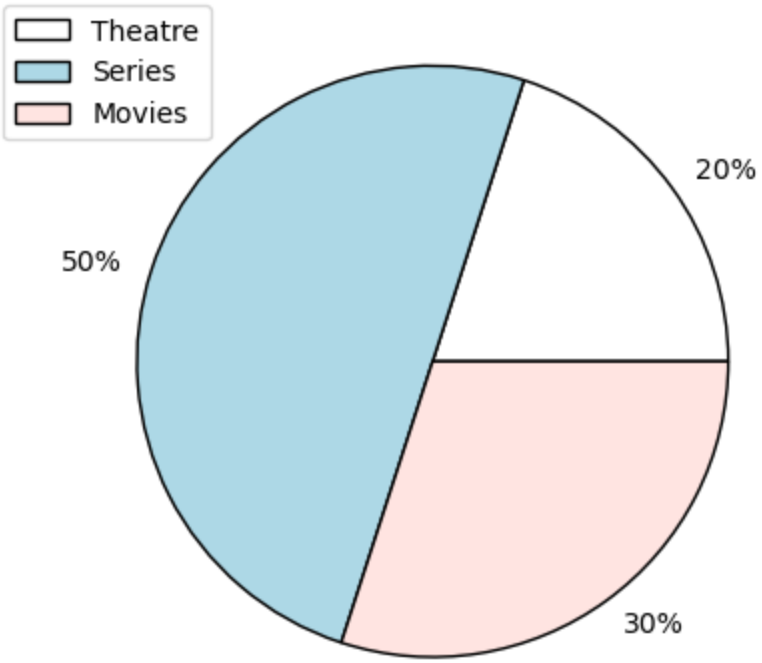
In [76]:

```python
import matplotlib.pyplot as plt

# Dữ liệu cho Pie Chart
count_2 = [27, 15, 10, 25]

# Tạo nhãn tùy chỉnh với giá trị và phần trăm
total = sum(count_2)
pie_labels = [f"{value} = {value / total * 100:.2f}%" for value in count_

# Màu sắc tùy chỉnh
colors = ['lightblue', 'white', 'lightcyan', 'lightcoral']

# Vẽ Pie Chart với nhãn tùy chỉnh
plt.pie(count_2,
        labels=pie_labels,
        colors=colors,
        wedgeprops={'edgecolor': 'black', 'linewidth': 1})  # Thêm viền đ

# Thêm tiêu đề
plt.title("Pie Chart with Custom Labels", fontsize=14)

# Hiển thị biểu đồ
plt.show()

```



In [77]:

```
import matplotlib.pyplot as plt

# Dữ liệu cho Pie Chart
count_3 = [20, 50, 30]

# Tạo nhãn với phần trăm
labels = [f"{value}%" for value in count_3]

# Màu sắc cho Pie Chart
colors = ['white', 'lightblue', 'mistyrose']

# Vẽ Pie Chart
plt.pie(count_3,
        labels=labels,
        colors=colors,
        wedgeprops={'edgecolor': 'black', 'linewidth': 1})  # Thêm viền đ

# Thêm legend, đặt ở ngoài Pie Chart
plt.legend(["Theatre", "Series", "Movies"], loc="upper left", bbox_to_ancl

# Hiển thị biểu đồ
plt.show()

```



In [ ]:

```

```