NumPy (Numerical Python) is a powerful library in Python for numerical computing. It provides high-performance multidimensional arrays and tools for working with them—making it a cornerstone of data science, machine learning, and scientific computing.

pip install numpy

import numpy as np

arr = np.array([1, 2, 3])

print(arr) # [1 2 3]

print(type(arr)) # <class 'numpy.ndarray'>

np.zeros((2, 3)) # 2x3 array of zeros

np.ones((3, 2)) # 3x2 array of ones

np.arange(0, 10, 2) # [0 2 4 6 8]

np.linspace(0, 1, 5) # 5 numbers between 0 and 1 =🡺

np.random.rand(2, 2) # 2x2 random array (0 to 1)

a = np.array([1, 2, 3])

b = np.array([4, 5, 6])

print(a + b) # [5 7 9]

print(a \* b) # [4 10 18]

print(a \*\* 2) # [1 4 9]

print(np.sqrt(a)) # [1. 1.41 1.73]

A = np.array([[1, 2], [3, 4]])

B = np.array([[2, 0], [1, 3]])

print(np.dot(A, B)) # Matrix multiplication

print(np.transpose(A)) # Transpose

print(np.linalg.inv(A)) # Inverse (if A is invertible)

arr = np.array([[10, 20, 30], [40, 50, 60]])

print(arr[0, 1]) # 20

print(arr[:, 1]) # [20 50] ==🡺

print(arr[1]) # [40 50 60]

arr = np.array([1, 2, 3, 4])

print(arr.mean()) # 2.5

print(arr.sum()) # 10

print(arr.std()) # Standard deviation

print(arr.max()) # 4

print(arr.min()) # 1

Pandas

pandas is one of the most powerful and popular Python libraries for **data manipulation and analysis**. It's widely used in data science, finance, analytics, and machine learning.

1. **Series** – A one-dimensional labeled array

import pandas as pd

s = pd.Series([10, 20, 30, 40], index=['a', 'b', 'c', 'd'])

print(s)

output:

a 10

b 20

c 30

d 40

dtype: int64

1. **DataFrame** – A 2D labeled table (like a spreadsheet or SQL table)

data = {

'Name': ['Alice', 'Bob', 'Charlie'],

'Age': [25, 30, 35],

'City': ['New York', 'Paris', 'London']

}

df = pd.DataFrame(data)

print(df)

Name Age City

0 Alice 25 New York

1 Bob 30 Paris

2 Charlie 35 London

Key operations

df.head() # First 5 rows

df.tail(2) # Last 2 rows

df.info() # Summary info

df.describe() # Stats summary

df.columns # Column names

df.shape # (rows, columns)

selection and filtering

df['Age'] # Select a column

df[['Name', 'City']] # Select multiple columns

df[df['Age'] > 28] # Filter rows

df.iloc[1] # Row by index (position)

df.loc[1] # Row by label/index

df['Age'] += 1 # Modify a column

df['Country'] = 'USA' # Add new column

df.rename(columns={'Age': 'Years'}, inplace=True)

df.drop('City', axis=1) # Drop column

Reading/writing files

pd.read\_csv('data.csv') # Read CSV

df.to\_csv('output.csv', index=False) # Write CSV

df.groupby('City').mean() # Group by city, get average

df['Age'].sum() # Sum of a column

pd.concat([df1, df2]) # Combine rows

pd.merge(df1, df2, on='id') # SQL-style join

Matplotlib

Matplotlib is a powerful Python library for **data visualization**. It's widely used for creating static, animated, and interactive plots. Think of it as Python's version of Excel charts—but much more customizable and code-driven.

pip install matplotlib

import matplotlib.pyplot as plt



x = [1, 2, 3, 4]

y = [10, 20, 25, 30]



plt.plot(x, y)



plt.title("Simple Line Plot")

plt.xlabel("X-axis")

plt.ylabel("Y-axis")



plt.show()



Common Plot Types

1. **Line Plot**

plt.plot([1, 2, 3], [4, 5, 6])

plt.show()

2.Bar chart

x = ['A', 'B', 'C']

y = [5, 7, 3]

plt.bar(x, y, color='skyblue')

plt.title("Bar Chart Example")

plt.show()

1. **Histogram**

import numpy as np

data = np.random.randn(1000)

plt.hist(data, bins=30, color='purple')

plt.title("Histogram")

plt.show()

4.Scatter plot

x = [1, 2, 3, 4, 5]

y = [5, 20, 14, 32, 22]

plt.scatter(x, y, color='red')

plt.title("Scatter Plot")

plt.show()

5. Pie chart

sizes = [40, 30, 20, 10]

labels = ['A', 'B', 'C', 'D']

plt.pie(sizes, labels=labels, autopct='%1.1f%%', startangle=140)

plt.title("Pie Chart")

plt.axis('equal') # Makes it a circle

plt.show()