Python Libraries

This document explains, with examples, how to perform descriptive statistics using Python. You can copy or rewrite the code at any time to test it on the compiler available on the course website or in the appendix section.

Using Python
Calculating Univariate Statistical Parameters

1. Python libraries for statistics

A Python library is a collection of modules that contain predefined functions, classes, and variables. These modules are Python files (.py) containing reusable code that simplifies the execution of specific tasks without the need to code them from scratch. Libraries are reusable and organized.

In this first session of the Second Block (focused on univariate descriptive statistics), we use these libraries:

Pandas

Pandas provides powerful and easy-to-use data structures and data manipulation tools for Python. It simplifies data manipulation and analysis, making complex data operations more intuitive and faster. Pandas allows the creation of data structures called DataFrames and Series, which facilitate the calculation of statistical parameters.

```
import pandas as pd
data = [10, 20, 30, 40, 50]
series = pd.Series(data)
```

NumPy

NumPy is a fundamental library for scientific computing in Python, providing support for large-dimensional arrays and mathematical functions. It is highly efficient for mathematical operations on large amounts of data and is used to perform effective statistical calculations.

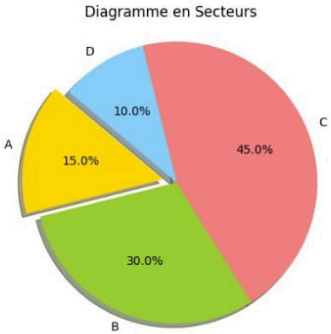
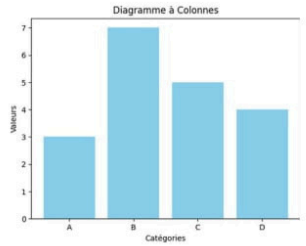
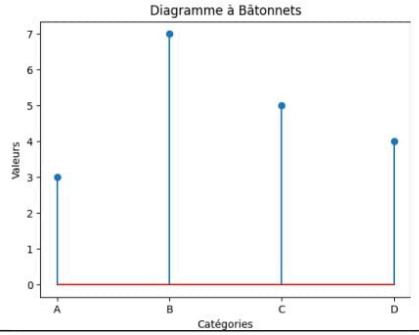
```
import numpy as np
data = np.array([10, 20, 30, 40, 50])
mean = np.mean(data)
```

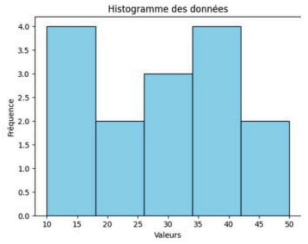
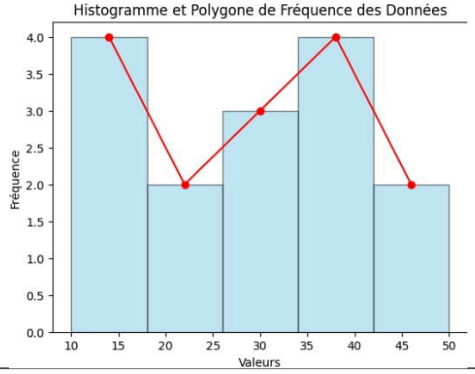
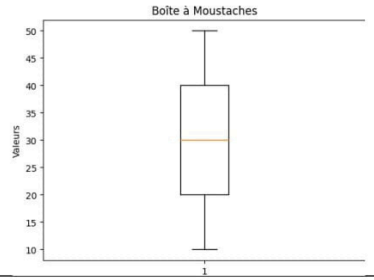
Matplotlib

Matplotlib is a plotting library for Python that enables the creation of static, animated, and interactive graphs. It is highly flexible and widely used for data visualization. Matplotlib is used to create graphs of statistical parameters.

```
import matplotlib.pyplot as plt
plt.hist(data, color='skyblue', edgecolor='black')
plt.xlabel('Valeurs')
plt.ylabel('Fréquence')
plt.title('Histogramme des données')
plt.show()
```

2. Graphical Representation

Diagram	Python Code	Result
Pie Chart *A pie chart representing the proportions of different categories.*	import matplotlib.pyplot as plt labels = ['A', 'B', 'C', 'D'] sizes = [15, 30, 45, 10] colors = ['gold', 'yellowgreen', 'lightcoral', 'lightskyblue'] explode = (0.1, 0, 0, 0) plt.pie(sizes, explode=explode, labels=labels, colors=colors, autopct='%1.1f%%', shadow=True, startangle=140) plt.axis('equal') plt.title(Pie Chart) plt.show()	
Column Chart *A chart representing values with vertical columns.*	import matplotlib.pyplot as plt categories = ['A', 'B', 'C', 'D'] values = [3, 7, 5, 4] plt.bar(categories, values, color='skyblue') plt.xlabel('Categories') plt.ylabel('Valeurs') plt.title('Column Chart') plt.show()	
Bar Chart *Un graphique représentant les valeurs par des bâtonnets.*	import matplotlib.pyplot as plt categories = ['A', 'B', 'C', 'D'] values = [3, 7, 5, 4] plt.stem(categories, values, use_line_collection=True) plt.xlabel('Categories') plt.ylabel('Valeurs') plt.title('Bar Chart') plt.show()	

Histogram *A chart showing the distribution of data across intervals.*	import matplotlib.pyplot as plt data = [10, 20, 20, 30, 30, 30, 40, 40, 40, 40, 10, 10, 10, 50, 50] plt.hist(data, bins=5, color='skyblue', edgecolor='black') plt.xlabel('Values') plt.ylabel('Frequency') plt.title('Histogram') plt.show()	
Frequency Polygon *A chart showing the frequency of data across its quartiles.*	import matplotlib.pyplot as plt import numpy as np data = [10, 20, 20, 30, 30, 30, 40, 40, 40, 40, 10, 10, 10, 50, 50] counts, bin_edges = np.histogram(data, bins=5) bin_centers = (bin_edges[:-1] + bin_edges[1:]) / 2 plt.hist(data, bins=5, color='skyblue', edgecolor='black', alpha=0.5) plt.plot(bin_centers, counts, marker='o', color='red') plt.xlabel('Values') plt.ylabel('Fréquency') plt.title('Histogram and data Frequency Polygon') plt.show()	
Boxplot *A chart showing the distribution of data across its quartiles.*	import matplotlib.pyplot as plt data = [10, 20, 30, 40, 50] plt.boxplot(data) plt.ylabel('Values') plt.title('Box Plot') plt.show()	

3. Statistical Parameteres

Indicator	Formula	Python Code	Exemple
Mode *The most frequent value in the dataset.*	Find the most frequent value.	import pandas as pd data = [10, 20, 20, 30, 30, 30, 40, 50] series = pd.Series(data) mode = series.mode()[0] print(mode)	Data : [10, 20, 20, 30, 30, 30, 40, 50] Mode: 30
Médian *The value that separates the upper half from the lower half of the data.*	Find the central value after sorting.	median = series.median() print(median)	Data : [10, 20, 20, 30, 30, 30, 40, 50] Médiane: 30
Mean *The sum of the values divided by the number of values.*	\(\bar{x} = \frac{\ sum_{i=1}^{n} x_i}{n} \)	mean = series.mean() print(mean)	Data : [10, 20, 20, 30, 30, 30, 40, 50] Moyenne: 28.75
Range *The difference between the maximum and minimum values.*	\(\text{Range} = \max(x_i) - \ min(x_i) \)	range_ = series.max() - series.min() print(range_)	Data : [10, 20, 20, 30, 30, 30, 40, 50] Étendue: 40
Variance *The measure of dispersion of values relative to the mean.*	\(\sigma^2 = \frac{\ sum_{i=1}^{n} (x_i - \bar{x})^2} {n} \)	variance = series.var() print(variance)	Data : [10, 20, 20, 30, 30, 30, 40, 50] Variance: 164.64285714285714
Standard Deviation *The square root of the variance.*	\(\sigma = \sqrt{\sigma^2} \)	std_dev = series.std() print(std_dev)	Data : [10, 20, 20, 30, 30, 30, 40, 50] Écart-Type: 12.828172463606714
Quartiles *Quartiles divide the data into four equal parts.*	Calculate the 25th, 50th, and 75th percentiles.	quartiles = series.quantile([0.25, 0.5, 0.75]) print(quartiles)	Data : [10, 20, 20, 30, 30, 30, 40, 50] Quartiles: 25.0, 30.0, 35.0
Deciles *Deciles divide the data into ten equal parts.*	Calculate the 10th, 20th, ..., 90th percentiles.	deciles = series.quantile([0.1 * i for i in range(1, 10)]) print(deciles)	Data : [10, 20, 20, 30, 30, 30, 40, 50] Déciles: 13.0, 20.0, 20.0, 30.0, 30.0, 30.0, 33.0, 40.0, 46.0
Percentiles *Percentiles divide the data into one hundred equal parts.*	Calculate the 1st, 2nd, ..., 99th percentiles.	centiles = series.quantile([0.01 * i for i in range(1, 100)]) print(centiles)	Data : [10, 20, 20, 30, 30, 30, 40, 50] Centiles: [11.4, 12.8, 14.2, 15.6, ..., 45.4, 46.8, 48.2]