IUT d'Orléans - Département d'Informatique
Statistiques Descriptives avec Python

TP 1 : Statistiques univariées

Si $\mathcal{X} = (\mathcal{X}_1, \cdots, \mathcal{X}_N)$ est une statistique de population (i.e. une suite de données), on note $x = (x_1, x_2, \cdots, x_m)$, $m \leq N$ les valeurs (ou modalités dépouillées) présentes (distinctes) dans la suite $\mathcal{X}$ ordonnées par ordre croissant et n_i désigne le nombre de fois où la valeur x_i apparait dans la série $\mathcal{X}$.
Pour ce TP nous utiliserons python et les bibliothèques suivantes :

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

Exercice 1 *Loi uniforme*
1. Pour générer un jeu de $N = 10$ données $\mathcal{X} = (\mathcal{X}_1, \cdots, \mathcal{X}_N)$, comprises entre 0 et 9, en python on peut utiliser :

```
X=np.random.randint(0, 9, 10)
```

Afficher la valeur N de la longueur de `X`.

> *N*=

Afficher la moyenne de $\mathcal{X}$ (notée $\overline{\mathcal{X}}$), en utilisant la méthode `np.mean` *de numpy.*
De même calculer la variance de $\mathcal{X}$ (notée $var(\mathcal{X})$) en utilisant : `np.var`.
2. On va maintenant chercher dans X toutes les valeurs distinctes qui sont apparues et le nombre de fois où elles apparaissent

```
# Extraction des valeurs possibles et de leur effectif

x, counts = np.unique(X, return_counts=True)
# Les valeurs presentent dans la sequence
print(x)
# Le nombre d'apparition de chacune des valeurs dans la sequence
print(counts)

# Frequences pour chaque modalite
f = counts / N
```

Calculer (à la main) les fréquences d'apparition de chacune des valeurs et vérifier en affichant les valeurs de `f` *:*

> `f`=

Générer maintenant un jeu de $N = 100$ *données* $\mathcal{X} = (\mathcal{X}_1, \cdots, \mathcal{X}_N)$*, puis remplir le tableau ci dessous :*

$x_i =$	*0*	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*
$n_i =$										
$f_i =$										

3. Tracer de deux diagrammes en bâtons (horizontal et vertical) de $\mathcal{X}$

```
# Creation de la figure et des sous-graphiques
fig, ax = plt.subplots(1, 2, figsize=(12, 5))

# Histogramme vertical
ax[0].bar(x, counts, color='blue')
ax[0].set_title("Histogramme vertical")
ax[0].set_xlabel("Valeurs")
ax[0].set_ylabel("Effectifs")

# Histogramme horizontal
ax[1].barh(x, counts, color='green')
ax[1].set_title("Histogramme horizontal")
ax[1].set_xlabel("Effectifs")
ax[1].set_ylabel("Valeurs")

# Ajustement des espacements
plt.tight_layout()
plt.show()


# Creation d'un histogramme en barres verticales
plt.bar(x, counts)

# Creation d'un histogramme en barres horizontales
plt.barh(x, counts)
```

Calculer (à la main en vous aidant de la question précédente) le tableau de fréquences cumulées F *(fonction de répartition empirique) :*

x_i	< 0	*0*	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	∞
$F = \sum_{j=0}^{i} f_j$	*0*											*1*

Retrouver ces résultats avec "cumsum" et tracer la fonction de répartition (c'est à dire F *en fonction de* x*) :*

```
# Frequences cumulees
```

```
F = np.cumsum(f)

# Creation du graphique en escalier
plt.step(x, F, where='mid', color='blue', linewidth=2, label="Graphe en escalier")

# Ajout des labels et du titre
plt.xlabel("Valeurs")
plt.ylabel("Frequence cumulee")
plt.title("Graphe en escalier de la frequence cumulee")
plt.legend()
plt.grid()
```

4. a. Déterminer : le mode de $\mathcal{X}$ c'est à dire sa plus grande valeur : $\max_{i\leq N}(\mathcal{X}_i)$*, la médiane de x (c.f. ci-dessous pour la définition).*
b. Evaluer : `np.mean(X)`, `sum(X)/N` *et* `np.sum(x * f)`*, que remarquez vous ?*
5 a. Nous définissons maintenant la notion de quartile
Premier quartile : *La plus grande valeur dans x telle que la fréquence cumulée F est inférieure ou égale à* 0.25*.*

$$Q_1 = \max\{x \in values \mid F(x) \leq 0.25\}.$$

Deuxième quartile (médiane) : *La plus grande valeur dans x telle que la fréquence cumulée F est inférieure ou égale à* 0.50*.*

$$Q_2 = \max\{x \in values \mid F(x) \leq 0.50\}.$$

Interquartile (IQR) : *La différence entre le troisième quartile et le premier quartile.*

$$IQR = Q_3 - Q_1.$$

Déterminer les quartiles de $\mathcal{X}$ en utilisant la question 3, ainsi que l'inter-quartile.
b. Calculer la variance de $\mathcal{X}$ de deux manières différentes (formule de Koening : voir le TD).
c. Comparer l'écart type à l'interquartile.

Exercice 2 *Loi binomiale*
1. Générer un jeu de n données $\mathcal{X} = (\mathcal{X}_1, \mathcal{X}_2, \cdots, \mathcal{X}_n)$, avec la commande suivante

```
X=np.random.binomial(9, 0.6, 100)
```

2. Calculer le tableau de fréquences avec la commande "tabul" :

$x_i =$	0	1	2	3	4	5	6	7	8	9
$n_i =$										
$f_i =$										

3. Tracer les diagrammes en bâtons de x. Calculer (à la main en vous aidant de la question précédente) le tableau de fréquences cumulées F (fonction de répartition empirique) :

x_i	< 0	0	1	2	3	4	5	6	7	8	9	∞
$F = \sum_{j=1}^{i} f_j$	0											1

Retrouver ces résultats avec "cumsum" et tracer la fonction de répartition.
4. a. Déterminer : le mode de $\mathcal{X}$, la médiane de $\mathcal{X}$.
b. Vérifier que : `np.mean(X)`, `sum(X)/N` *et* `np.sum(x * f)`, *renvoint des valeurs identiques.*
5 a. Déterminer les quartiles ainsi que l'inter-quartile
b. Calculer la variance de $\mathcal{X}$ de deux manières différentes (formule de Koening).
c. Comparer l'écart type à l'interquartile.

Exercice 3 *Loi normale discrétisée*
Faites la même analyse que précédemment en utilisant le jeu de données généré par

```
X=np.floor(np.random.normal(5, 1, 250)).
```

Exercice 4 *Dans cet exercice, nous apprenons à charger des données depuis un fichier csv, les convertir en des tableaux numpy et à faire leur analyse. Pour le faire de manière plus propre, nous définissons une classe qui charge les données et contient quelques méthodes permettant d'effectuer les calculs de base. Nous appliquons cela à des données Forex (ici le taux de change euro-dollar américain).*

```
import pandas as pd
import numpy as np

class CSVLoader:
    def __init__(self, file_path, nrows=None):
        """Charge un fichier CSV et garde uniquement les colonnes numeriques."""
        df = pd.read_csv(file_path, nrows=nrows, sep='\t')
        self.data = df.select_dtypes(include=[np.number]).to_numpy()

    def mean(self):
        """Calcule la moyenne des colonnes."""
        return np.mean(self.data, axis=0)

    def variance(self):
        """Calcule la variance des colonnes."""
        return np.var(self.data, axis=0, ddof=1)

    def quartiles(self):
        """Calcule les quartiles Q1, Q2 (mediane), Q3."""
        return np.percentile(self.data, [25, 50, 75], axis=0)

    # La methode ci-dessous calcul des frequences pour le tracer de l'histogramme.
```

```
    def frequence(self, column_index):
        # Recuperation des donnees :
        data_column = self.data[:, column_index]
        # Nombre d'intervalles :
        num_bins = 10
        #Divise les valeurs de data_column en intervalles (bins) et compte combien
        #de valeurs tombent dans chaque intervalle :
        frequences, bin_edges = np.histogram(data_column, bins=num_bins)

        return frequences, bin_edges
```

Voici comment charger les données puis calculer la moyenne avec cette classe.

```
# Charger les donnees depuis un fichier CSV
csv_loader = CSVLoader("Forex.csv")

# Calculer la moyenne des colonnes
moyenne = csv_loader.mean()
print("Moyenne :", moyenne)
```

Utiliser de la même façon cette classe pour :
1) Calculer la variance, la médiane et l'interquartile,
2) Tracer un histogramme,
3) Intégrer dans la classe le calcul du mode.