HE VD HAUTE ÉCOLE D'INGÉNIERIE ET DE GESTION DU CANTON DE VAUD

ISD - 2023

Introduction à la science des données

Solutions TP1 – Introduction TP2

Résumé et solutions du TP1

- Environnements virtuels
- Utilisation de Python et des librairies spécialisées:
  - Numpy: Gestion de matrices numériques
  - Pandas: Gestion de matrices plus complexes -> Dataframes
  - Matplotlib: Visualisation des données (relativement bas niveau)
  - Seaborn: Visualisation des données (haut niveau)

- N’hésitez pas à ouvrir votre TP1 en même temps que les suivants comme référence

Solutions – Compréhension de liste

1.2.1-3 Calculer le pourcentage de valeurs paires dans une liste d'un million de valeurs aléatoires entre 1 et 10

```
1
2  import random # -> random.fonction() pour utiliser une fonction
3  from random import randrange # -> randrange() pour utiliser la fonction
4  from random import randint   # -> randint() pour utiliser la fonction
5
6  d = [randrange(1, 11) for _ in range(1_000_000)]
7
8  # d = [randint(1,10) for _ in range(1_000_000)] # créer une liste en utilisant la compréhension de liste
9  # d = random.choices(range(1, 11), k=1_000_000) # retourne directement une liste
10 d_even = [x for x in d if x % 2 == 0]
11
12 # Les f-strings: solution simple de formattage !
13 print(f'{len(d_even) / 10_000:.2f}%')
```

```
49.98%
```

Solutions – Compréhension de liste

1.2.4 Description du code

Toutes les solutions appliquent le carré en se servant de la compréhension de liste (applique la fonction "carré" sur tous les éléments d'une liste, et retourne une liste).

Solution 1: `[x**2 for x in (list(range(1, 11, 2)) + list(range(12, 21, 2)))]`

- On créé d'abord deux listes à l'aide du constructeur list() que l'on concatène, avant de prendre les carrés des éléments. La fonction *range()* fait ici le travail de générer les bons nombre paires ou impaires.

Solution 2: `[x**2 for x in range(1, 21) if (((x <= 10) and (x % 2 != 0)) or ((x > 10) and (x % 2 == 0)))]`

- On créé la liste en une fois directement avec un range de 1 à 20, mais pour chaque élément des conditions vérifient s'il doit être ajouté ou non.

Solution 3: `[x**2 for x in range(1, 11, 2)] + [x**2 for x in range(12, 21, 2)]`

- On utilise deux compréhension de liste pour créer la liste finale en deux partie séparée. Comme pour la solution 1 on se sert du fait qu'on peut concaténer des listes, mais cette fois-ci l'opération est faite à la toute fin.

Solutions – Compréhension de liste

1.2.5 Liste de strings:

- *A partir de la liste "objets" donnée, créez une liste contenant uniquement les mots de la première liste qui **contiennent** la lettre "**z**" ou "**Z**".*
- C'est un exemple de la documentation: https://www.w3schools.com/python/python_lists_comprehension.asp
- Résultat attendu: ['Zoo', 'Zebre', 'Jazz']

```
newlist = [x for x in objets if ("z" in x) or ("Z" in x)]

newlist = [x for x in objets if 'z' in x.lower()]

newlist = [x for x in objets if (x.__contains__("z")) or (x.__contains__("Z"))]
```

Solutions - Numpy

- 1) Pourquoi utiliser NumPy ?
  - Pour utiliser des tableaux, Python fournit le type "List". Cependant les calculs sur les listes sont lents.
- 2) Comment s'appelle (n.b. "de quel type est") l'objet "array" dans NumPy ?
  - ndarray
- 3) Pourquoi utiliser NumPy est-il plus rapide qu'utiliser les listes ?
  - Stockage continu en mémoire. Optimisé pour travailler avec les dernières architectures CPU.
- 4) A quelle question le code "# Question 4" ci-dessous répond-il ?
  - La question 2.

Solutions – Pandas

Les solutions et les exercices proviennent du tutoriel de w3schools:
https://www.w3schools.com/python/pandas/default.asp

- 1) Pourquoi utiliser Pandas ?
  - **Pandas permet d'analyser des Big Data et d'en extraire des conclusions basées sur les statistiques. Grâce à Pandas, on peut nettoyer des données pour les rendre pertinantes.**
- 2) Que peut faire pandas (d'après le tuto w3schools) ?
  - **Donner des informations sur les données telles que la moyenne, la valeur max() ou min() de colonnes. Enlever des colonnes innutiles ou non pertinantes. En résumé, nettoyer les données et en extraire des informations.**
- 3) Que fait l'exemple "Question 3" ci-dessous ?
  - **Créé un DataFrame à partir d'un dictionnaire, puis l'affiche**.

Solutions – Pandas

```
# Question 4
## Code à compléter ##

# Afficher la ligne qui concerne les Volvo.
print(myvar.loc[1])
print(type(myvar.loc[1]))

print("----------------------")
print(myvar.loc[[1]])
print(type(myvar.loc[[1]]))

print("----------------------")
# Afficher les deux dernières lignes du dataframe
print(myvar.loc[[1,2]])
print("----------------------")
print(myvar.iloc[-1])
```

```
cars        Volvo
passings        7
Name: 1, dtype: object
<class 'pandas.core.series.Series'>
----------------------
    cars  passings
1  Volvo         7
<class 'pandas.core.frame.DataFrame'>
----------------------
    cars  passings
1  Volvo         7
2   Ford         2
----------------------
cars        Ford
passings       2
Name: 2, dtype: object
```

4) Compléter la cellule "Question 4" ci-dessous pour afficher les lignes demandées. Utiliser l'attribut *loc* comme décrit dans le tutoriel. Pour afficher les dernières colonnes, vous pouvez utiliser des indexes fixes qui ne fonctionnent qu'avec un dataframe de cette taille, ou essayer la fonction *iloc()* avec des indexes négatifs.* **Que remarquez vous concernant l'utilisation de simples crochets** ([...]) ou **doubles crochets** ([[x]]) pour extraire **une** ligne du dataframe ? En utilisant la fonction type(), donnez le type de données retournées avec les simples crochets ([...]) ou doubles crochets ([[x,y]]).

On remarque que le format affiché n'est pas le même selon l'utilisation de df.loc[1] ou df.loc[[1]]. Avec les simples crochets, on obtient une série, avec les doubles, un dataframe. Pour utiliser des indexes négatifs, il faut utiliser iloc()

Solutions – Pandas

5) Complétez le code comme demandé dans la cellule "Question 5 - exercice". Extrait du tutoriel Pandas de w3school.

```
df_clean = df.dropna().copy()
# df_clean = df_clean.reset_index(drop = True) # Optionnel
```

```
#df_clean['Date'] = pd.to_datetime(df_clean['Date'])
date = pd.to_datetime(df_clean.loc[:, "Date"].copy())

#df_clean["Date"] = date -> returns a warning
df_clean = df_clean.assign(Date = date)
```

```
df.loc[7, 'Duration'] = 45
```

Remplacer les données extrêmes par une donnée arbitraire n'est jamais une bonne idée.

```
# Autre solution:
#for x in df.index:
#    if df.loc[x, "Duration"] > 120:
#        df.drop(x, inplace = True)
#        df.loc[x, "Duration"] = 120 # Solution nul mais dans le tuto !!
```

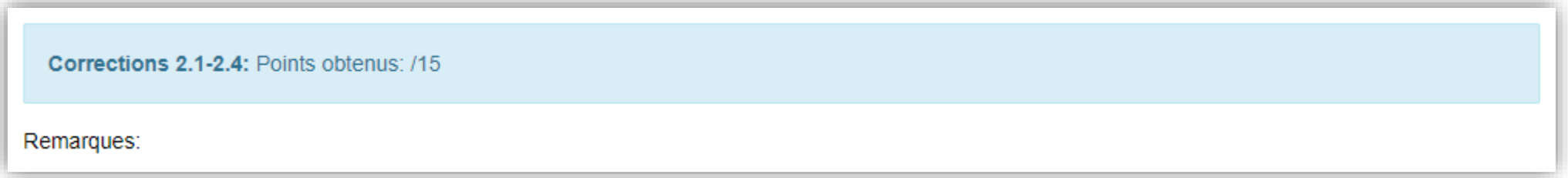
TP2 - Introduction

Manipulation et visualisation des données

TP2 – Manipulation et visualisation

- Utilisation de la librairie Pandas
- Python est non typé, ça change de vos cours de PRG1
  - Nommez vos variables correctement (e.g. préfix *df_...* pour DataFrame)
  - Vérifiez vos types avec la fonction *type(object)* si vous n'êtes pas sûr

- Ne modifiez pas le notebook inutilement; ne modifiez pas les cellules réservées pour la correction



Visualisation et interprétation des données

- Des connaissances de bases de statistiques vous sont demandées.
  - Comment interpréter un graphique ? Que représente un histogramme ? Un boxplot ?
  - Quelle est la différence entre une moyenne et une médiane ?

- En cas de doutes ou de questions n’hésitez pas à demander
  - à l’assistant et au professeur
  - à Youtube
  - à chatGPT

Visualisation et interprétation des données

Exemple d’histogramme:

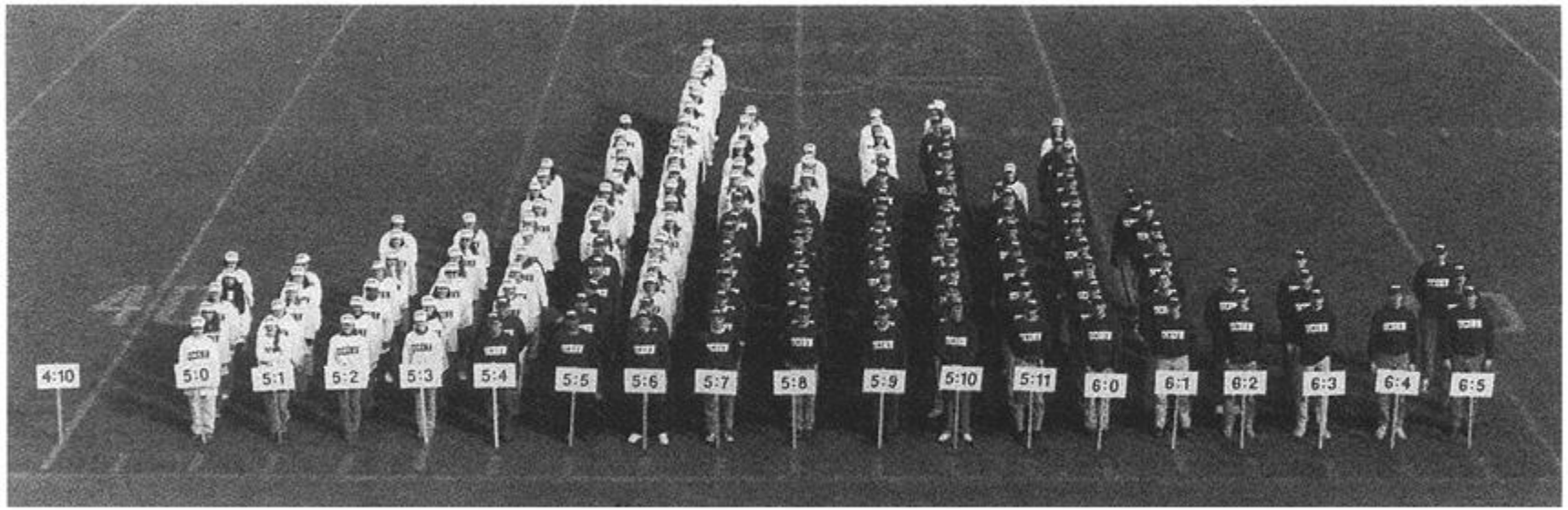
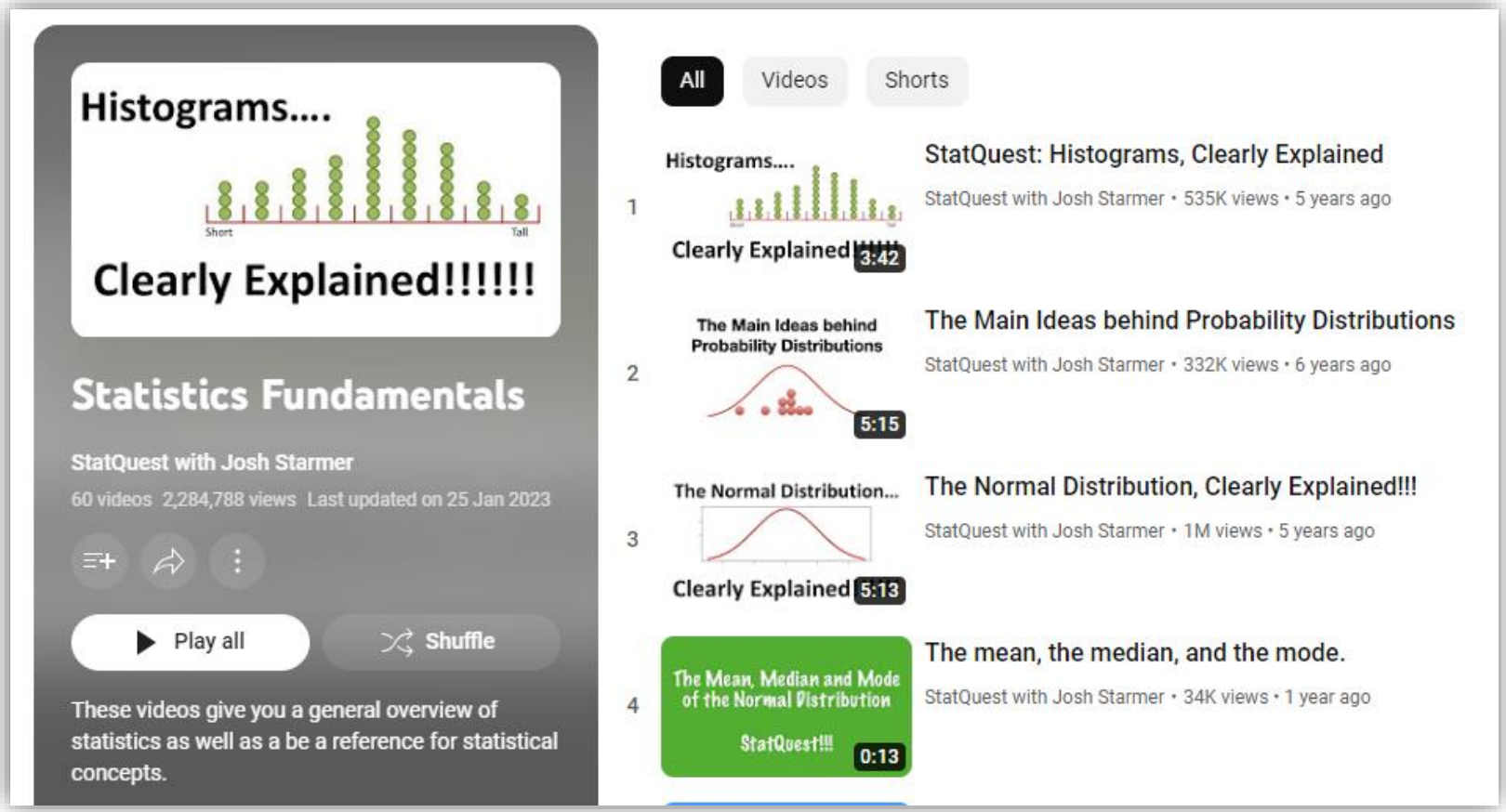


Figure 7. Living histogram of 143 student heights at University of Connecticut.

Statistics Fundamentals by Josh Starmer

- https://www.youtube.com/playlist?list=PLblh5JKOoLUK0FLuzwntyYI10UQFUhsY9



Des bonnes bases en
15 minutes.

Termes utiles:
- Moyenne
- Médiane (quantilles)
- Min/Max
- Écart-type (Standard deviation)

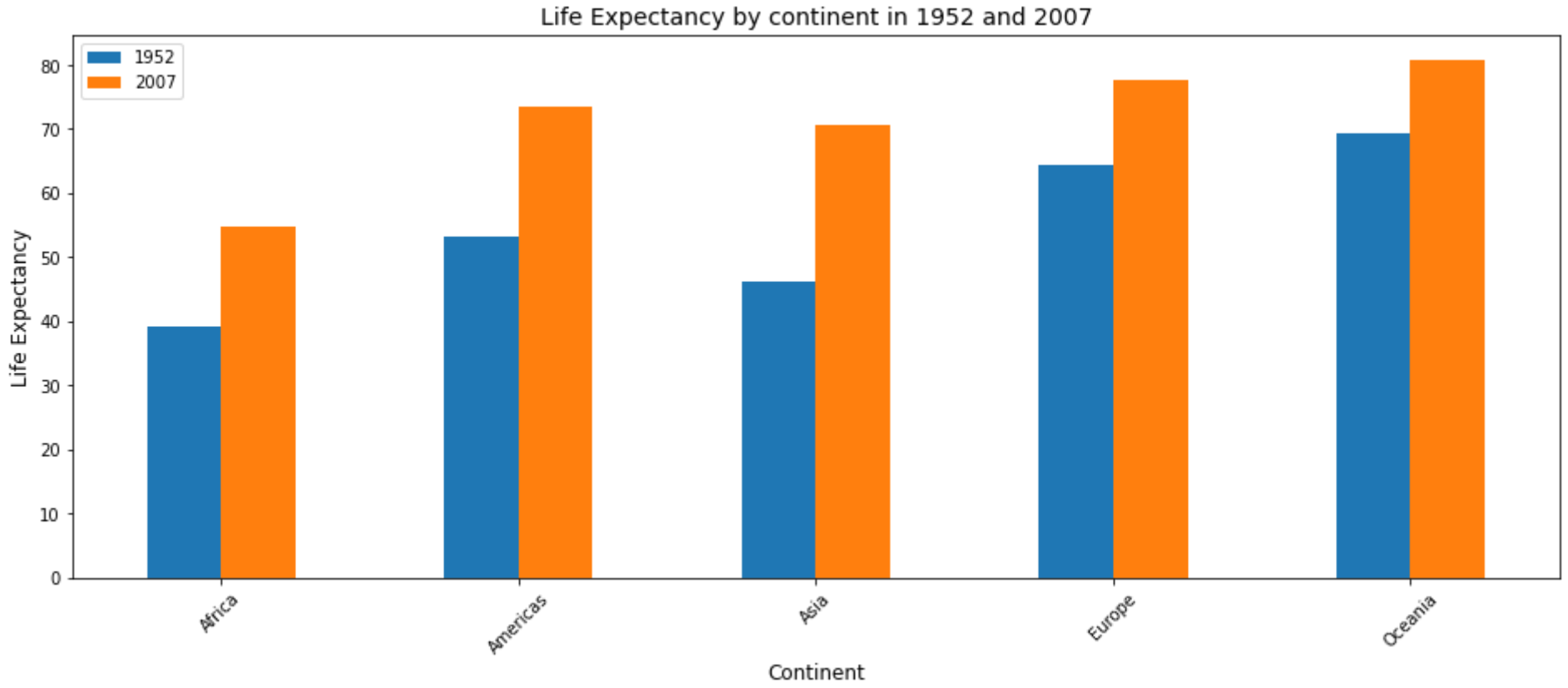
Côté programmation:

- Beaucoup d'information et beaucoup de ressources
- Une fois que vous connaissez les bases (après le TP1 par exemple) vous pouvez rapidement trouver des informations grâce à une série de **cheat sheets**:
  - https://web.itu.edu.tr/iguzel/files/Python_Cheat_Sheets.pdf
  - Dispo en pdf sur Cyberlearn

- Donnent une vue globale des possibilités de ces librairies

TP2 – Exercice 4.2

Résultat attendu:



Rendus

- L’exercice le plus court du TP
  - **Lire la consigne de rendu**
  - Écrire vos noms (Noms dans le nom de fichier et Prénoms et noms directement dans le notebook)
  - Rendre sur la plateforme spécifiée
  - Par groupe de 2 ou 3
    - Utilisez Teams ou parlez-vous pour trouver un groupe
    - Pas de rendu solo sauf cas spécifiques (e.g. abandon des membres du groupe en cours de TP)
- Plus de 120 étudiant.e.s
  - Régulièrement des prénoms ou noms identiques
  - Chaque erreur coûte du temps pour comprendre qui a rendu quoi
  - **Points négatifs en cas d’erreur**

Rendus

- Kernel -> *Restart and run all*
  - Supprime toutes les variables et les imports avant de réexécuter le notebook complet
  - Vérifiez que votre code fonctionne pour une nouvelle exécution (par exemple sur l'ordinateur de l'assistant qui corrigera vos TPs)
  - Permet de vérifier l'exécution dans l'ordre, il arrive vite d'avoir une variable:
    - Utilisée avant d'être déclarée
    - Déclarée dans un autre notebook utilisant le même kernel (même environnement)

Au boulot !

Et hésitez pas si vous avez des questions !