Thème : Traitement des données

20 Manipulation de fichiers CSV avec une bibliothèque

1. Utilisation du module Pandas

Le module `csv` utilisé précédemment se contente de lire les données structurées. Il ne fait aucun effort particulier pour analyser les données. Nous nous en sommes aperçus lorsqu'il a fallu convertir par `int()` toutes les valeurs numériques, qui étaient interprétées comme des chaînes de caractères.
La bibliothèque `pandas` est par contre spécialement conçue pour l'analyse des données (*data analysis*) : elle est donc naturellement bien plus performante.

On allons utiliser une base de données sur les joueurs NBA.

```
import pandas as pd #import du module pandas, abrégé classiquement par
"pd"
```

```
df = pd.read_csv('data/players.csv', encoding = 'utf-8')
```

La variable est nommée classiquement `df` pour *dataframe* (que l'on peut traduire par *table de données*)

1.1 Premiers renseignements sur les fichiers de données

Que contient la variable `df` ?

```
df
```

Les données sont présentées dans l'ordre originel du fichier.
Il est possible d'avoir uniquement les premières lignes du fichier avec la commande `head()` et les dernières du fichier avec la commande `tail()`. Ces commandes peuvent recevoir en paramètre un nombre entier.

```
df.head()
```

```
df.tail()
```

```
df.head(3)
```

```
df.tail(5)
```

Pour avoir des renseignements globaux sur la structure de notre fichier, on peut utiliser la commande `df.info()`

```
df.info()
```

1.2 Extraction de colonnes, création de graphiques

Pour accéder à une fiche particulière de joueur avec sa position, on peut utiliser la fonction `loc()` :

```
df.loc[45]
```

Pour accéder à un joueur avec son nom :

```
df.loc[df['Name']=='Jaylen Brown']
```

Pour accéder à des données avec une contrainte (Joueurs touchant plus de 40 millions de dollars par an):

```
df.loc[df['Salary']>=40000000]
```

Pour faire afficher une colonne de données :

```
df.loc[:,"Salary"]
```

```
df.loc[:10,"Salary"]  # les 10 premiers dans la liste
```

On peut calculer la moyenne des salaires :

```
df["Salary"].mean()
```

le maximum

```
df["Salary"].max()
```

Le minimum

```
df["Salary"].min()
```

On peut également faire afficher le salaire max et le nom du joueur correspondant :

```
result = df[df["Salary"].max() == df["Salary"]]["Name"].iloc[0]
print(result)
```

Question 1.
Faites de même pour faire apparaitre le nom du joueur ayant le salaire minimum.

```

```

Question 2.
Faire apparaitre le poste de Kemba Walker

```
result = df[df["Name"] == "Kemba Walker"]["Position"].iloc[0]
print(result)
```

Question 3.
Faire apparaitre le plus grand joueur

```
result = df[df["Height"].max() == df["Height"]]["Name"].iloc[0]
print(result)
```

Question 4.

Faire apparaitre le plus petit joueur

```
result = df[df["Height"].min() == df["Height"]]["Name"].iloc[0]
print(result)
```

```
df.loc[df['Name']=='Jared Harper']
```

Question 5.

Donner le nom du meilleur marqueur ainsi que sa moyenne de points

```
pointsmax=df["Points"].max()
result = df[df["Points"].max() == df["Points"]]["Name"].iloc[0]
print(result,pointsmax)
```

```
result = df[df["Points"].max() == df["Points"]]
[["Name","Points"]].iloc[0]
print(result)
```

1.3 Création de graphique

```
%matplotlib inline
import matplotlib.pyplot as plt
X = df['Weight']
Y = df['Height_i']

plt.plot(X,Y,'ro') # r pour red, o pour un cercle. voir
https://matplotlib.org/api/markers_api.html
plt.show()
```

L'interprétation numérique permet à `pandas` d'analyser automatiquement les données, avec notamment la fonction `describe()`.

```
df['Height_i'].describe()
```

```
df.boxplot('Height_i')
```

Donner les joueurs des Brooklyn Nets

```
test=df[['Name','Points']][df['Team']=='Brooklyn Nets']
print(test)
```

```
df[['Name','Team']]
```

1.4 Rajout d'une colonne

Afin de pouvoir trier les joueurs suivant de nouveaux critères, nous allons rajouter un champ pour chaque joueur. Prenons un exemple stupide : fabriquons un nouveau champ `'Taille en cm'` qui contiendra la taille des joueurs aprés conversion en cm. Ceci se fera simplement par :

```
df['Taille'] = round(df['Height_i'] * 0.3048,2)
```

```
df
```

Faite de même pour convertir le poid en kg.

```
df['Poids'] = round(df['Weight'] * 0.45359 ,0)
```

```
df
```

Faire afficher la répartition des tailles (Voir plus haut)

```
df['Taille'].describe()
```

```
df.boxplot('Taille')
```

Trier les joueurs par ordre décroissant de poids

```
p=df.sort_values(by=['Poids'], ascending = False)
p
```

Faire de même pour la taille, les scoreurs et rebondeurs

```
scoreur=df.sort_values(by=['Points'], ascending = False)
scoreur
```

1. Créer une colonne contenant l'IMC de chaque joueur
2. Créer une nouvelle dataframe contenant tous les joueurs du top14 classés par ordre d'IMC croissant.

```
```

Créer une dataframe contenant les joueurs en provenance de Duke

```
dfduke=df[df['College']=='Duke']
```

```
dfduke
```

```
```