



42 ARTIFICIAL INTELLIGENCE

Python Module

Pandas


Summary: Aujourd'hui, vous allez apprendre à utiliser une bibliothèque Python qui vous permettra pour manipuler des dataframes.

Contents

I	Exercise 00	2
II	Exercise 01	4
III	Exercise 02	6
IV	Exercise 03	8

Chapter I

Exercise 00

	Exercise :
	FileLoader
Turn-in directory : <i>ex/</i>	
Files to turn in :	
Forbidden functions : None	

Objective

Le but de cet exercice est de créer deux fonctions. Une fonction qui charge un fichier csv et une, qui affiche un Dataframe

Instructions

Implémenter les fonctions suivantes:

- `load(self, path)`: prend en argument le nom du fichier du jeu de données à charger, affiche un message précisant les dimensions de l'ensemble de données (par exemple 340 x 500) et renvoie l'ensemble de données chargé en tant que `pandas.DataFrame`.
- `display(self, df, n)`: prend un `pandas.DataFrame` et un entier comme arguments, affiche les `n` premières lignes de l'ensemble de données si `n` est positif, ou les `n` dernières lignes si `n` est négatif (bonus).

Examples

```
url = "https://raw.githubusercontent.com/A-Mahla/serda-formation-python/main/datasets/athlete_events.csv"
data = load(url)
# Output
Loading dataset of dimensions 271116 x 15

display(data, 12)
# Output
```

ID	Name ...	Event	Medal
0	1 A Dijiang ...	Basketball Men's Basketball	NaN
1	2 A Lamusi ...	Judo Men's Extra-Lightweight	NaN
2	3 Gunnar Nielsen Aaby ...	Football Men's Football	NaN
3	4 Edgar Lindenau Aabye ...	Tug-Of-War Men's Tug-Of-War	Gold
4	5 Christine Jacoba Aaftink ...	Speed Skating Women's 500 metres	NaN
5	5 Christine Jacoba Aaftink ...	Speed Skating Women's 1,000 metres	NaN
6	5 Christine Jacoba Aaftink ...	Speed Skating Women's 500 metres	NaN
7	5 Christine Jacoba Aaftink ...	Speed Skating Women's 1,000 metres	NaN
8	5 Christine Jacoba Aaftink ...	Speed Skating Women's 500 metres	NaN
9	5 Christine Jacoba Aaftink ...	Speed Skating Women's 1,000 metres	NaN
10	6 Per Knut Aaland ...	Cross Country Skiing Men's 10 kilometres	NaN
11	6 Per Knut Aaland ...	Cross Country Skiing Men's 50 kilometres	NaN

```
[12 rows x 15 columns]
```



NB: Votre terminal peut afficher plus de colonnes si la fenêtre est plus large.

Chapter II

Exercise 01

	Exercise :
YoungestFellah	
Turn-in directory : <i>ex/</i>	
Files to turn in :	
Forbidden functions : None	

Objective

Le but de cet exercice est de créer une fonction qui renverra un dictionnaire contenant l'âge de la plus jeune femme et de la plus jeune homme qui a participé aux Jeux olympiques une année donnée.

Instructions

Écrivez une fonction `youngest_athlete` qui prend deux arguments:

- un `pandas.DataFrame` qui contient l'ensemble de données.
- une année olympique.

La fonction renvoie un dictionnaire contenant l'âge du plus jeune femme et homme qui ont participé aux Jeux olympiques cette année-là.


Examples

```
url = "https://raw.githubusercontent.com/A-Mahla/serda-formation-python/main/datasets/athlete_events.csv"
data = load(url)
# Output
Loading dataset of dimensions 271116 x 15

youngest_athlete(data, 2004)
# Output
{'f': 13.0, 'm': 14.0}
```

Chapter III

Exercise 02

	Exercise :
HowManyMedals	
Turn-in directory : <i>ex/</i>	
Files to turn in :	
Forbidden functions : None	

Objective

Le but de cet exercice est d'implémenter une fonction qui renverra un dictionnaire de dictionnaires donnant le nombre et le type de médailles pour chaque année au cours de laquelle le participant a remporté des médailles.

Instructions

Écrivez une fonction `combien_many_medals` qui prend deux arguments:

- un `pandas.DataFrame` qui contient l'ensemble de données,
- un nom de participant.

La fonction renvoie un dictionnaire de dictionnaires donnant le nombre et le type de médailles pour chaque année au cours de laquelle le participant a remporté des médailles. Les clés du dictionnaire principal sont les années des Jeux Olympiques. Dans le dictionnaire de chaque année, les clés sont 'G', 'S', 'B' correspondant au type de médailles remportées (gold, silver, bronze).


Examples

```
url = "https://raw.githubusercontent.com/A-Mahla/serda-formation-python/main/datasets/athlete_events.csv"
data = load(url)
# Output
Loading dataset of dimensions 271116 x 15

how_many_medals(data, 'Kjetil Andr Aamodt')
# Output
{1992: {'G': 1, 'S': 0, 'B': 1},
 1994: {'G': 0, 'S': 2, 'B': 1},
 1998: {'G': 0, 'S': 0, 'B': 0},
 2002: {'G': 2, 'S': 0, 'B': 0},
 2006: {'G': 1, 'S': 0, 'B': 0}}
```


Chapter IV

Exercise 03

	Exercise :
SpatioTemporalData	
Turn-in directory : <i>ex/</i>	
Files to turn in :	
Forbidden functions : None	

Objective

Le but de cet exercice est de créer deux fonctions de recherche par ville ou date dans le dataset.

Instructions

Implémenter les fonctions suivantes:

- **when(location)**: prend un `pd.DataFrame` et un emplacement comme arguments et renvoie une liste contenant les années où les Jeux olympiques ont eu lieu à l'endroit donné,
- **where(date)**: prend un `pd.DataFrame` et un emplacement comme arguments et renvoie le lieu où les Jeux olympiques ont eu lieu au cours de l'année donnée.

Examples

```
url = "https://raw.githubusercontent.com/A-Mahla/serda-formation-python/main/datasets/athlete_events.csv"
data = load(url)
# Output
Loading dataset of dimensions 271116 x 15

where(data, 1896)
# Output
['Athina']

where(data, 2016)
# Output
['Rio de Janeiro']

sp.when(data, 'Athina')
# Output
[2004, 1906, 1896]

sp.when(data, 'Paris')
# Output
[1900, 1924]
```