seance5_approche_fonctionnelle_correction

July 2, 2023

1 Données, approches fonctionnelles - correction

Correction de l'approche fonctionnelle. Elle s'appuie principalement sur des itérateurs et le module cytoolz.

[1]:
```
%matplotlib inline
import matplotlib.pyplot as plt
plt.style.use('ggplot')
import pyensae
from jyquickhelper import add_notebook_menu
add_notebook_menu()
```

```
Populating the interactive namespace from numpy and matplotlib
```

[1]:
```
<IPython.core.display.HTML object>
```

Le notebook utilisera des données issues d'une table de mortalité extraite de table de mortalité de 1960 à 2010 qu'on récupère à l'aide de la fonction table_mortalite_euro_stat.

1.1 Exercice 1 : application aux grandes bases de données

Imaginons qu'on a une base de données de 10 milliards de lignes. On doit lui appliquer deux traitements : `f1`, `f2`. On a deux options possibles :

- Appliquer la fonction `f1` sur tous les éléments, puis appliquer `f2` sur tous les éléments transformés par `f1`.
- Application la combinaison des générateurs `f1`, `f2` sur chaque ligne de la base de données.

Que se passe-t-il si on a fait une erreur d'implémentation dans la fonction `f2` ?

[2]:
```

```

1.2 Exercice 2 : cytoolz

La note d'un candidat à un concours de patinage artistique fait la moyenne de trois moyennes parmi cinq, les deux extrêmes n'étant pas prises en compte. Il faut calculer cette somme pour un ensemble de candidats avec cytoolz.

[3]:
```
notes = [dict(nom="A", juge=1, note=8),
         dict(nom="A", juge=2, note=9),
         dict(nom="A", juge=3, note=7),
         dict(nom="A", juge=4, note=4),
         dict(nom="A", juge=5, note=5),
```

```
        dict(nom="B", juge=1, note=7),
        dict(nom="B", juge=2, note=4),
        dict(nom="B", juge=3, note=7),
        dict(nom="B", juge=4, note=9),
        dict(nom="B", juge=1, note=10),
        dict(nom="C", juge=2, note=0),
        dict(nom="C", juge=3, note=10),
        dict(nom="C", juge=4, note=8),
        dict(nom="C", juge=5, note=8),
        dict(nom="C", juge=5, note=8),
        ]

import pandas
pandas.DataFrame(notes)
```

[3]:

	juge	nom	note
0	1	A	8
1	2	A	9
2	3	A	7
3	4	A	4
4	5	A	5
5	1	B	7
6	2	B	4
7	3	B	7
8	4	B	9
9	1	B	10
10	2	C	0
11	3	C	10
12	4	C	8
13	5	C	8
14	5	C	8

[4]:

```
import cytoolz.itertoolz as itz
import cytoolz.dicttoolz as dtz
from functools import reduce
from operator import add
```

[5]:

```
gr = itz.groupby(lambda d: d["nom"], notes)

def select_note(key_value):
    key, value = key_value
    return key, map(lambda d: d["note"], value)

gr_notes = dtz.itemmap(select_note,  gr)

def enleve_extreme(key_value):
    key, value = key_value
    return key, itz.take(3, itz.drop(1,sorted(value)))

def moyenne(key_value):
    key, value = key_value
    return key, reduce(add, value)/3
```

```
no_ext = dtz.itemmap( enleve_extreme, gr_notes)

moy = dtz.itemmap( moyenne, no_ext)
moy
```

[5]: {'A': 6.666666666666667, 'B': 7.666666666666667, 'C': 8.0}

[6]:

```

```