lec4_step2_BarStack_Aligned_Stage1

November 30, 2022

[ ]:
```
## Python basics for novice data scientists, supported by Wagatsuma Lab@Kyutech
#
# The MIT License (MIT): Copyright (c) 2020 Hiroaki Wagatsuma and Wagatsuma␣
 ↪Lab@Kyutech
#
# Permission is hereby granted, free of charge, to any person obtaining a copy␣
 ↪of this software and associated documentation files (the "Software"), to␣
 ↪deal in the Software without restriction, including without limitation the␣
 ↪rights to use, copy, modify, merge, publish, distribute, sublicense, and/or␣
 ↪sell copies of the Software, and to permit persons to whom the Software is␣
 ↪furnished to do so, subject to the following conditions:
# The above copyright notice and this permission notice shall be included in␣
 ↪all copies or substantial portions of the Software.
# THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR␣
 ↪IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,␣
 ↪FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE␣
 ↪AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER␣
 ↪LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING␣
 ↪FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS␣
 ↪IN THE SOFTWARE. */
#
# # @Time    : 2020-11-30
# # @Author  : Hiroaki Wagatsuma
# # @Site    : https://github.com/hirowgit/2A1_python_intermediate_course
# # @IDE     : Python 3.9.14 (main, Sep  6 2022, 23:29:09) [Clang 13.1.6␣
 ↪(clang-1316.0.21.2.5)] on darwin
# # @File    : lec4_step2_BarStack_Aligned_Stage1.py
```

[1]:
```
import numpy as np
#prFill=[90    60    50    50    50    90    40    30    80    40  20 ]/100;
prFill=np.array([90, 60, 50, 50, 50, 90, 40, 30, 80, 40, 20])
prFill=prFill/100
fillLine=np.full(len(prFill),True)
LineT=[]
k=1
for i in range(len(prFill)):
    print(i)
```

```
    print(prFill[i])
```

```
0
0.9
1
0.6
2
0.5
3
0.5
4
0.5
5
0.9
6
0.4
7
0.3
8
0.8
9
0.4
10
0.2
```

[ ]:

[1]:
```
1
```

[1]: 1

[9]:
```
import numpy as np
#prFill=[90    60    50    50    50    90    40    30    80    40  20 ]/100;
prFill=np.array([90, 60, 50, 50, 50, 90, 40, 30, 80, 40, 20])
prFill=prFill/100
fillLine=np.full(len(prFill),True)
LineT=[]
k=0
for i in range(len(prFill)):
    if fillLine[i]:
        remF=1-prFill[i]
        IDrem=np.where((prFill[i+1:-1]<=remF) & fillLine[i+1:-1])
        tmp=i
        fID=i
        j=0
        while IDrem[0].size > 0:
```

```
            fID=IDrem[0][0]+fID
            tmp=np.append(tmp,fID)
            remF=remF-prFill[IDrem[j][0]+i]
            IDrem=np.where((prFill[fID+1:-1]<=remF) & fillLine[fID+1:-1])
        LineT.append(tmp)
        fillLine[tmp]=False
        print(k)
        k=k+1
```

```
0
1
2
3
4
5
6
7
8
```

[ ]:

[ ]:

[ ]:

[118]:
```
prFill=np.array([90, 60, 50, 50, 50, 90, 40, 30, 80, 40, 20])
prFill=prFill/100
fillLine=np.full(len(prFill),True)
LineT=[]
k=1
```

[119]:
```
i=0
fillLine[i]
```

[119]: True

[20]:
```
remF=1-prFill[i]
print(remF)
IDrem=np.where((prFill[i+1:-1]<=remF) & fillLine[i+1:-1])
print(IDrem)
tmp=i
fID=i
j=0
i
```

```
0.09999999999999998
(array([], dtype=int64),)
```

[20]:
```
0
```

[15]:
```
i=0
print(remF)
print(prFill[1:-1])
np.where(prFill[i+1:-1]<=remF)
IDrem=np.where((prFill[fID+1:-1]<=remF) & fillLine[fID+1:-1])
print(IDrem)
print(len(IDrem))
if IDrem:
    print('IDrem is NOT empty')
else:
    print('IDrem is empty')
np.array(IDrem)
print(IDrem[0])
```

```
0.8
[0.6 0.5 0.5 0.5 0.9 0.4 0.3 0.8 0.4]
(array([], dtype=int64),)
1
IDrem is NOT empty
[]
```

[2]:
```
if IDrem[0]:
    print('IDrem is NOT empty')
else:
    print('IDrem is empty')
print(IDrem[0])
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
/var/folders/mg/w5t8lkhc8xj79f001s7kzpfh0000gp/T/ipykernel_26401/3945485694.py␣
 ↪in <module>
----> 1 if IDrem[0]:
      2     print('IDrem is NOT empty')
      3 else:
      4     print('IDrem is empty')
      5 print(IDrem[0])

NameError: name 'IDrem' is not defined
```

[116]:
```
LineT
```

[116]:
```
[0]
```

[117]:
```
k
```

[117]:
```
1
```

[131]:
```
np.where((prFill[fID+1:-1]<=remF) & fillLine[fID+1:-1])
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-131-769d5665eeb7> in <module>
----> 1 np.where((prFill[fID+1:-1]<=remF) & fillLine[fID+1:-1])

ValueError: operands could not be broadcast together with shapes (4,) (3,)
```

[130]:
```
fID
```

[130]:
```
5
```

[128]:
```
IDrem[0][0]
```

[128]:
```
4
```

[132]:
```
prFill[fID+1:-1]
```

[132]:
```
array([0.4, 0.3, 0.8, 0.4])
```

[133]:
```
fillLine[fID+1:-1]
```

[133]:
```
array([ True,  True,  True,  True])
```

[134]:
```
np.where((prFill[fID+1:-1]<=remF) )
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-134-c7e1a02de47f> in <module>
----> 1 np.where((prFill[fID+1:-1]<=remF) )

ValueError: operands could not be broadcast together with shapes (4,) (3,)
```

[135]:
```
remF
```

[135]:
```
array([-0.5,  0. , -0.4])
```

[136]:
```
IDrem[j]
```

[136]:
```
array([4, 5, 7])
```

[137]:
```
IDrem[j][0]
```

[137]:
```
4
```

[139]:
```
LineT
```

[139]:
```
[0,
 array([1, 5]),
 array([2, 2]),
 array([3, 3]),
 array([4, 5]),
 array([6, 6]),
 array([7, 8]),
 9,
 10]
```

[ ]:

[ ]: