D1 python ver

Lee SungHo

2021-01-13

⚜ ⚜

page 10 Recap

```
import numpy as np

def soda_simul(this_state):
  u=np.random.rand()
  if (this_state == "c"):
    if(u<=0.7):
      next_state = "c"
    else:
      next_state = "p"

  else:
    if(u<=0.5):
      next_state = "c"
    else:
      next_state = "p"

  return next_state

def cost_eval(path):
  cost_one_path = path.count('c')*1.5 + path.count('p')*1
  return cost_one_path

MCN = 10000
spending_record = ['0']*MCN



for i in range(MCN):
  path = 'c'
  for t in range(9):
    this_state = path[-1]
    next_state = soda_simul(this_state)
    path = path+next_state

  spending_record[i] = cost_eval(path)
```

page 11 MC simulation for estimating state-value function

```
import numpy as np

#MC evalutaion for state-value function
#with state s, time 0, reward r, time horizon H
num_episode = 100000
episode_i = 0
cum_sum_G_i = 0

while episode_i < num_episode:
  path = 's'

  for t in range(9):
    this_state = path[-1]
    next_state = soda_simul(this_state)
    path = path+next_state

  G_i = cost_eval(path)
  cum_sum_G_i = cum_sum_G_i + G_i

  episode_i +=1

V_t = cum_sum_G_i / num_episode

print(V_t)
```

```
## 11.733525
```

page 20 Iterative solution

```
import numpy as np

P = np.array([[0.7,0.3],[0.5,0.5]])
R = np.array([1.5,1.0])
H = 10 #time horizon
V_t1 = np.array([0,0])

t = H-1
while (t>=0):
  V_t = R + np.dot(P,V_t1)
  t-=1
  V_t1 = V_t

print(V_t1)
```

```
## [13.35937498 12.73437504]
```

```
print("V0(c) is",V_t1[0])
```

```
## V0(c) is 13.359374975999996
```

```
print("V0(p) is",V_t1[1])
```

```
## V0(p) is 12.734375039999998
```