F1_Jeong,wonryeol

Jeong, wonryeol

1/25/2021

Contents

Preparation

```
import numpy as np
import pandas as pd
gamma = 1
states = np.arange(0,80,10).astype('str')
P_normal=pd.DataFrame(np.matrix([[0,1,0,0,0,0,0,0],
                                 [0,0,1,0,0,0,0,0],
                                 [0,0,0,1,0,0,0,0],
                                 [0,0,0,0,1,0,0,0],
                                 [0,0,0,0,0,1,0,0],
                                 [0,0,0,0,0,0,1,0],
                                 [0,0,0,0,0,0,0,1],
                                 [0,0,0,0,0,0,0,1]]), index=states,columns=states)
P_speed=pd.DataFrame(np.matrix([[.1,0,.9,0,0,0,0,0],
                                [.1,0,0,.9,0,0,0,0],
                                [0,.1,0,0,.9,0,0,0],
                                [0,0,.1,0,0,.9,0,0],
                                [0,0,0,.1,0,0,.9,0],
                                [0,0,0,0,.1,0,0,.9],
                                [0,0,0,0,0,.1,0,.9],
                                [0,0,0,0,0,0,0,1]]), index=states, columns=states)



def transition(given_pi, states, P_normal, P_speed):
    P_out=pd.DataFrame(np.zeros((len(states),len(states))),index=states, columns=states)

    for s in states:
        action_dist=given_pi.loc[s]
        P=action_dist['normal']*P_normal+action_dist['speed']*P_speed
        P_out.loc[s]=P.loc[s]

    return P_out
```

```
# reword
R_s_a=pd.DataFrame(np.array([-1,-1,-1,-1,0.0,-1,-1,0,
                             -1.5,-1.5,-1.5,-1.5,-0.5,-1.5,-1.5,0]).reshape(len(states),2,order='F'),c
R_s_a.T
```

```
##      0   10   20   30   40   50   60   70
## n -1.0 -1.0 -1.0 -1.0  0.0 -1.0 -1.0  0.0
## s -1.5 -1.5 -1.5 -1.5 -0.5 -1.5 -1.5  0.0
```

```
# pi_speed
pi_speed=pd.DataFrame(np.c_[np.repeat(0,len(states)), np.repeat(1,len(states))],index=states, columns=[
pi_speed.T
```

```
##    0  10  20  30  40  50  60  70
## n  0   0   0   0   0   0   0   0
## s  1   1   1   1   1   1   1   1
```

```
pi_50=pd.DataFrame(np.repeat(0.5,len(states)*2).reshape(8,2),index=states, columns=['n','s'])
pi_50
```

```
##       n    s
## 0   0.5  0.5
## 10  0.5  0.5
## 20  0.5  0.5
## 30  0.5  0.5
## 40  0.5  0.5
## 50  0.5  0.5
## 60  0.5  0.5
## 70  0.5  0.5
```

11_page Simulator -π^{speed}

```
# simulator pi_speed

pi = pi_speed
np.random.seed(1234)
history = list()
MC_N = 10000
for MC_i in range(1,MC_N+1):
    s_now = '0'
    history_i = [s_now]
    while s_now != '70':
        if np.random.uniform(0,1,1) < pi.loc[s_now,"n"] :
            a_now = "n"
            P = P_normal
        else:
            a_now = "s"
            P = P_speed

        r_now = R_s_a.loc[s_now,a_now]
        s_next = states[np.argmin(np.cumsum(P.loc[s_now,])<np.random.uniform(0,1))]
        history_i.extend([a_now,r_now,s_next])
        s_now = s_next


    history.append(history_i)

history_speed = history


def paste0 (x):
    x = np.array(x).astype('str')
    return ','.join(x)

result =list(map(paste0,history_speed[:20]))
pd.DataFrame(result)
```

```
##                                                    0
## 0          0,s,-1.5,20,s,-1.5,40,s,-0.5,60,s,-1.5,70
## 1          0,s,-1.5,20,s,-1.5,40,s,-0.5,60,s,-1.5,70
## 2   0,s,-1.5,0,s,-1.5,20,s,-1.5,40,s,-0.5,60,s,-1....
## 3          0,s,-1.5,20,s,-1.5,40,s,-0.5,60,s,-1.5,70
## 4          0,s,-1.5,20,s,-1.5,40,s,-0.5,60,s,-1.5,70
## 5          0,s,-1.5,20,s,-1.5,40,s,-0.5,60,s,-1.5,70
## 6   0,s,-1.5,20,s,-1.5,10,s,-1.5,30,s,-1.5,50,s,-1...
## 7   0,s,-1.5,20,s,-1.5,40,s,-0.5,30,s,-1.5,50,s,-1...
## 8          0,s,-1.5,20,s,-1.5,40,s,-0.5,60,s,-1.5,70
## 9          0,s,-1.5,20,s,-1.5,40,s,-0.5,60,s,-1.5,70
## 10         0,s,-1.5,20,s,-1.5,40,s,-0.5,60,s,-1.5,70
## 11         0,s,-1.5,20,s,-1.5,40,s,-0.5,60,s,-1.5,70
## 12         0,s,-1.5,20,s,-1.5,40,s,-0.5,60,s,-1.5,70
## 13         0,s,-1.5,20,s,-1.5,40,s,-0.5,60,s,-1.5,70
## 14         0,s,-1.5,20,s,-1.5,40,s,-0.5,60,s,-1.5,70
## 15  0,s,-1.5,20,s,-1.5,10,s,-1.5,30,s,-1.5,50,s,-1...
```

```
## 16          0,s,-1.5,20,s,-1.5,40,s,-0.5,60,s,-1.5,70
## 17          0,s,-1.5,20,s,-1.5,40,s,-0.5,60,s,-1.5,70
## 18          0,s,-1.5,20,s,-1.5,40,s,-0.5,60,s,-1.5,70
## 19          0,s,-1.5,20,s,-1.5,40,s,-0.5,60,s,-1.5,70
```

13_page Simulator -π^{50}

```
# simulator pi_50


pi = pi_50
np.random.seed(1234)
history = list()
MC_N = 10000
for MC_i in range(1,MC_N+1):
    s_now = '0'
    history_i = [s_now]
    while s_now != '70':
        if np.random.uniform(0,1,1) < pi.loc[s_now,"n"] :
            a_now = "n"
            P = P_normal
        else:
            a_now = "s"
            P = P_speed

        r_now = R_s_a.loc[s_now,a_now]
        s_next = states[np.argmin(np.cumsum(P.loc[s_now,])<np.random.uniform(0,1))]
        history_i.extend([a_now,r_now,s_next])
        s_now = s_next


    history.append(history_i)

history_50 = history

def paste0 (x):
    x = np.array(x).astype('str')
    return ','.join(x)

result =list(map(paste0,history_speed[:20]))
pd.DataFrame(result)
```

```
##                                                    0
## 0           0,s,-1.5,20,s,-1.5,40,s,-0.5,60,s,-1.5,70
## 1           0,s,-1.5,20,s,-1.5,40,s,-0.5,60,s,-1.5,70
## 2   0,s,-1.5,0,s,-1.5,20,s,-1.5,40,s,-0.5,60,s,-1....
## 3           0,s,-1.5,20,s,-1.5,40,s,-0.5,60,s,-1.5,70
## 4           0,s,-1.5,20,s,-1.5,40,s,-0.5,60,s,-1.5,70
## 5           0,s,-1.5,20,s,-1.5,40,s,-0.5,60,s,-1.5,70
## 6   0,s,-1.5,20,s,-1.5,10,s,-1.5,30,s,-1.5,50,s,-1...
## 7   0,s,-1.5,20,s,-1.5,40,s,-0.5,30,s,-1.5,50,s,-1...
## 8           0,s,-1.5,20,s,-1.5,40,s,-0.5,60,s,-1.5,70
## 9           0,s,-1.5,20,s,-1.5,40,s,-0.5,60,s,-1.5,70
## 10          0,s,-1.5,20,s,-1.5,40,s,-0.5,60,s,-1.5,70
## 11          0,s,-1.5,20,s,-1.5,40,s,-0.5,60,s,-1.5,70
## 12          0,s,-1.5,20,s,-1.5,40,s,-0.5,60,s,-1.5,70
## 13          0,s,-1.5,20,s,-1.5,40,s,-0.5,60,s,-1.5,70
## 14          0,s,-1.5,20,s,-1.5,40,s,-0.5,60,s,-1.5,70
## 15  0,s,-1.5,20,s,-1.5,10,s,-1.5,30,s,-1.5,50,s,-1...
```

```
## 16          0,s,-1.5,20,s,-1.5,40,s,-0.5,60,s,-1.5,70
## 17          0,s,-1.5,20,s,-1.5,40,s,-0.5,60,s,-1.5,70
## 18          0,s,-1.5,20,s,-1.5,40,s,-0.5,60,s,-1.5,70
## 19          0,s,-1.5,20,s,-1.5,40,s,-0.5,60,s,-1.5,70
```

17_page Implementation1 - $\pi^{speed}(vectorized)$

```
# 17_page Implementation1 - $\pi^{speed}(vectorized)$

pol_eval = pd.DataFrame(np.zeros(16).reshape(states.shape[0],2),index =states,columns = ['count','sum'])


def paste0 (x):
    x = np.array(x).astype('str')
    return ','.join(x)



for MC_i in range(len(history_speed)):
    history_i = history_speed[MC_i]
    for j in range(0,len(history_i),3):

        pol_eval.loc[history_i[j],'count']+=1
        if j < len(history_i):
            pol_eval.loc[history_i[j],'sum']+=pd.Series(history_i)[list(range(j+2,len(history_i)-1,3))]
        else:
            pol_eval.loc[history_i[j],'sum'] =pol_eval.loc[history_i[j],'sum'] +0


pol_eval.T
```

```
##              0      10       20      30       40      50       60       70
## count  11225.0  1076.0  10291.0  1887.0   9485.0  2563.0   8563.0  10000.0
## sum   -65136.0 -5619.0 -42703.0 -6539.0 -22275.5 -4472.5 -14355.0      0.0
```

```
pol_eval["sum"]/pol_eval["count"]
```

```
## 0    -5.802762
## 10   -5.222119
## 20   -4.149548
## 30   -3.465289
## 40   -2.348498
## 50   -1.745025
## 60   -1.676398
## 70    0.000000
## dtype: float64
```

18_page Implementation2 π^{speed} (running estimate)

```
pol_eval = pd.DataFrame(np.zeros(16).reshape(states.shape[0],2),index =states,columns = ['count','est'])


for MC_i in range(len(history_speed)):
    history_i = history_speed[MC_i]

    for j in range(0,len(history_i),3):
        #update count
        pol_eval.loc[history_i[j],'count']+=1
        current_cnt = pol_eval.loc[history_i[j],'count']
        #return is the new info
        if j < len(history_i):
            new_info =pd.Series(history_i)[list(range(j+2,len(history_i)-1,3))].astype('float').sum()
        else:
            new_info = 0
        alpha = 1/current_cnt
        pol_eval.loc[history_i[j],'est'] = pol_eval.loc[history_i[j],'est']+ alpha * (new_info - pol_eva


pol_eval.T
```

```
##                   0           10  ...           60       70
## count  11225.000000  1076.000000  ...  8563.000000  10000.0
## est       -5.802762    -5.222119  ...    -1.676398      0.0
##
## [2 rows x 8 columns]
```

21_page Implementation3 - π^{50}(vectorized)

```
pol_eval = pd.DataFrame(np.zeros(16).reshape(states.shape[0],2),index =states,columns = ['count','sum'])


for MC_i in range(len(history_50)):
    history_i = history_50[MC_i]
    for j in range(0,len(history_i),3):

        pol_eval.loc[history_i[j],'count']+=1
        if j < len(history_i):
            pol_eval.loc[history_i[j],'sum']+=pd.Series(history_i)[list(range(j+2,len(history_i)-1,3))]
        else:
            pol_eval.loc[history_i[j],'sum'] =pol_eval.loc[history_i[j],'sum'] +0


pol_eval.T
```

```
##              0       10       20       30       40       50      60       70
## count  10863.0   5792.0   8140.0   7121.0   7549.0   7363.0  6991.0  10000.0
## sum   -64904.5 -29662.5 -33549.0 -24133.0 -15410.0 -14874.5 -9436.5      0.0
```

```
pol_eval["sum"]/pol_eval["count"]
```

```
## 0    -5.974823
## 10   -5.121288
## 20   -4.121499
## 30   -3.388990
## 40   -2.041330
## 50   -2.020168
## 60   -1.349807
## 70    0.000000
## dtype: float64
```

23_page Implementation4 - π^{50}(Running Estimate)

```
pol_eval = pd.DataFrame(np.zeros(16).reshape(states.shape[0],2),index =states,columns = ['count','est'])


for MC_i in range(len(history_50)):
    history_i = history_50[MC_i]

    for j in range(0,len(history_i),3):
        #update count
        pol_eval.loc[history_i[j],'count']+=1
        current_cnt = pol_eval.loc[history_i[j],'count']
        #return is the new info
        if j < len(history_i):
            new_info =pd.Series(history_i)[list(range(j+2,len(history_i)-1,3))].astype('float').sum()
        else:
            new_info = 0
        alpha = 1/current_cnt
        pol_eval.loc[history_i[j],'est'] = pol_eval.loc[history_i[j],'est']+ alpha * (new_info - pol_eva


pol_eval.T
```

```
##                   0           10  ...           60       70
## count  10863.000000  5792.000000  ...  6991.000000  10000.0
## est       -5.974823    -5.121288  ...    -1.349807      0.0
##
## [2 rows x 8 columns]
```

35_page Implementation5_TD - π^{speed}(Running Estimate)

```
pol_eval = pd.DataFrame(np.zeros(16).reshape(states.shape[0],2),index =states,columns = ['count','est'])


for MC_i in range(len(history_speed)):
    history_i = history_speed[MC_i]

    for j in range(0,len(history_i),3):
        #update count
        pol_eval.loc[history_i[j],'count']+=1
        current_cnt = pol_eval.loc[history_i[j],'count']
        #return is the new info
        if j+3 < len(history_i):
            TD_tgt = history_i[j+2]+pol_eval.loc[history_i[j+3]]['est']


        else:
            TD_tgt = 0
        alpha = 1/current_cnt
        pol_eval.loc[history_i[j],'est'] = pol_eval.loc[history_i[j],'est']+ alpha * (TD_tgt - pol_eval


pol_eval.T
```

```
##                   0           10  ...           60       70
## count  11225.000000  1076.000000  ...  8563.000000  10000.0
## est       -5.738838    -5.186466  ...    -1.675699      0.0
##
## [2 rows x 8 columns]
```

38_page Implementation6_TD - π^{50}(Running Estimate)

```
pol_eval = pd.DataFrame(np.zeros(16).reshape(states.shape[0],2),index =states,columns = ['count','est'])


for MC_i in range(len(history_50)):
    history_i = history_50[MC_i]

    for j in range(0,len(history_i),3):
        #update count
        pol_eval.loc[history_i[j],'count']+=1
        current_cnt = pol_eval.loc[history_i[j],'count']
        #return is the new info
        if j+3 < len(history_i):
            TD_tgt = history_i[j+2]+pol_eval.loc[history_i[j+3]]['est']


        else:
            TD_tgt = 0
        alpha = 1/current_cnt
        pol_eval.loc[history_i[j],'est'] = pol_eval.loc[history_i[j],'est']+ alpha * (TD_tgt - pol_eval


pol_eval.T
```

```
##                  0           10           20  ...           50           60       70
## count  10863.00000  5792.000000  8140.000000  ...  7363.000000  6991.000000  10000.0
## est       -5.84492    -5.052485    -4.079273  ...    -2.026683    -1.351198      0.0
##
## [2 rows x 8 columns]
```