E2_Exercises

Kwon do yun

2021-01-25

차 례

policy_eval() (P.4)

```
import numpy as np
import pandas as pd

gamma=1.0
states=np.arange(0,80,10)
P_normal=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]])

P_normal=pd.DataFrame(P_normal,columns=states)

P_speed=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]])

P_speed=pd.DataFrame(P_speed,columns=states)
```

```
def transition(given_pi,states,P_normal,P_speed):
    P_out=pd.DataFrame(np.zeros([8,8]),index=states, columns=states)
    action_dist = given_pi
    P=action_dist["normal"]*P_normal+action_dist["speed"]*P_speed
    P_out=P

    return(P_out)
```

```
R_s_a=np.matrix([[-1,-1,-1,-1,0,-1,-1,0],
                 [-1.5,-1.5,-1.5,-1.5,-0.5,-1.5,-1.5,0]]).T
R_s_a=pd.DataFrame(R_s_a,columns=["normal","speed"],index=states)
```

```
def reward_fn(given_pi, R_s_a):
    R_pi = np.sum(given_pi*R_s_a,axis=1)

    return R_pi.values.reshape([8,1])
```

```
def policy_eval(given_pi,R_s_a,states,P_normal,P_speed):
  R=reward_fn(given_pi,R_s_a)
  P=transition(given_pi,states,P_normal,P_speed)
  gamma = 1.0
  epsilon = 10**(-8)
  v_old=np.zeros([8,1])
  v_new = R + np.dot(gamma*P,v_old)
  while(np.max(np.abs(v_new-v_old)) > epsilon) :
      v_old = v_new
      v_new = R + np.dot(gamma*P,v_old)

  return v_new

pi_speed=np.matrix([[0,0,0,0,0,0,0,0],[1,1,1,1,1,1,1,1]]).T
pi_speed=pd.DataFrame(pi_speed,columns=["normal","speed"],index=states)
```

```
pd.DataFrame(policy_eval(pi_speed,R_s_a,states,P_normal,P_speed).T,columns=states)
```

```
##          0         10        20        30       40        50        60   70
## 0 -5.805929 -5.208781 -4.139262 -3.475765 -2.35376 -1.735376 -1.673538  0.0
```

```
pi_50=pd.DataFrame(np.c_[np.repeat(0.5,len(states)),
np.repeat(0.5,len(states))],index=states, columns=['normal','speed'])
pd.DataFrame(policy_eval(pi_50,R_s_a,states,P_normal,P_speed).T,columns=states)
```

```
##          0         10        20        30       40        50        60   70
## 0 -5.969238 -5.133592 -4.119955 -3.389228 -2.04147 -2.027768 -1.351388  0.0
```

Implementation (P. 12)

```
V_old = policy_eval(pi_speed,R_s_a,states,P_normal,P_speed)
pi_old = pi_speed
q_s_a = R_s_a + np.c_[np.dot(gamma*P_normal,V_old),np.dot(gamma*P_speed,V_old)]
q_s_a
```

```
##       normal     speed
## 0  -6.208781 -5.805929
## 10 -5.139262 -5.208781
## 20 -4.475765 -4.139262
## 30 -3.353760 -3.475765
## 40 -1.735376 -2.353760
## 50 -2.673538 -1.735376
## 60 -1.000000 -1.673538
## 70  0.000000  0.000000
```

```
pi_new=pd.DataFrame(np.zeros(pi_old.shape), index=pi_old.index, columns=pi_old.columns)
idx = q_s_a.idxmax(axis=1).values
count = 0
for i in states:
    pi_new.loc[i][idx[count]] = 1
    count +=1
pi_new
```

```
##     normal  speed
## 0      0.0    1.0
## 10     1.0    0.0
## 20     0.0    1.0
## 30     1.0    0.0
## 40     1.0    0.0
## 50     0.0    1.0
## 60     1.0    0.0
## 70     1.0    0.0
```

```
def policy_imporve(V_old,pi_old,R_s_a,gamma,P_normal,P_speed):
    q_s_a = R_s_a + np.c_[np.dot(gamma*P_normal,V_old),np.dot(gamma*P_speed,V_old)]
    pi_new=pd.DataFrame(np.zeros(pi_old.shape), index=pi_old.index, columns=pi_old.columns)
    idxmax = q_s_a.idxmax(axis=1).values
    count = 0
    for i in states:
        pi_new.loc[i][idxmax[count]] = 1
```

```
        count +=1
    return pi_new
```

```
pi_old = pi_speed
V_old = policy_eval(pi_old,R_s_a,states,P_normal,P_speed)
pi_new = policy_imporve(V_old,pi_old,R_s_a,gamma,P_normal,P_speed)
```

```
pi_old
```

```
##     normal  speed
## 0        0      1
## 10       0      1
## 20       0      1
## 30       0      1
## 40       0      1
## 50       0      1
## 60       0      1
## 70       0      1
```

```
pi_new
```

```
##     normal  speed
## 0      0.0    1.0
## 10     1.0    0.0
## 20     0.0    1.0
## 30     1.0    0.0
## 40     1.0    0.0
## 50     0.0    1.0
## 60     1.0    0.0
## 70     1.0    0.0
```

Policy iteration (P. 16)

```
#step 0
pi_old = pi_speed
pi_old
```

```
##     normal  speed
## 0        0      1
## 10       0      1
```

```
## 20       0      1
## 30       0      1
## 40       0      1
## 50       0      1
## 60       0      1
## 70       0      1
```

```
#step 1
pi_old = pi_speed
V_old = policy_eval(pi_old,R_s_a,states,P_normal,P_speed)
pi_new = policy_imporve(V_old,pi_old,R_s_a,gamma,P_normal,P_speed)
pi_old=pi_new
pi_old
```

```
##     normal  speed
## 0      0.0    1.0
## 10     1.0    0.0
## 20     0.0    1.0
## 30     1.0    0.0
## 40     1.0    0.0
## 50     0.0    1.0
## 60     1.0    0.0
## 70     1.0    0.0
```

```
#step 2
pi_old = pi_speed
V_old = policy_eval(pi_old,R_s_a,states,P_normal,P_speed)
pi_new = policy_imporve(V_old,pi_old,R_s_a,gamma,P_normal,P_speed)
pi_old=pi_new
pi_old
```

```
##     normal  speed
## 0      0.0    1.0
## 10     1.0    0.0
## 20     0.0    1.0
## 30     1.0    0.0
## 40     1.0    0.0
## 50     0.0    1.0
## 60     1.0    0.0
## 70     1.0    0.0
```

```
#step 3
pi_old = pi_speed
V_old = policy_eval(pi_old,R_s_a,states,P_normal,P_speed)
pi_new = policy_imporve(V_old,pi_old,R_s_a,gamma,P_normal,P_speed)
pi_old=pi_new
pi_old
```

```
##     normal  speed
## 0      0.0    1.0
## 10     1.0    0.0
## 20     0.0    1.0
## 30     1.0    0.0
## 40     1.0    0.0
## 50     0.0    1.0
## 60     1.0    0.0
## 70     1.0    0.0
```

```
pi_old = pi_speed
cnt = 0
while(1) :
    print(cnt,"-th iteration")
    print(pi_old.T)

    V_old = policy_eval(pi_old,R_s_a,states,P_normal,P_speed)
    pi_new = policy_imporve(V_old,pi_old,R_s_a,gamma,P_normal,P_speed)
    if(pi_new.eq(pi_old).all().all() == True):
        break
    pi_old=pi_new
    cnt=cnt+1
```

```
## 0 -th iteration
##         0   10  20  30  40  50  60  70
## normal   0   0   0   0   0   0   0   0
## speed    1   1   1   1   1   1   1   1
## 1 -th iteration
##          0    10   20   30   40   50   60   70
## normal  0.0  1.0  0.0  1.0  1.0  0.0  1.0  1.0
## speed   1.0  0.0  1.0  0.0  0.0  1.0  0.0  0.0
## 2 -th iteration
##          0    10   20   30   40   50   60   70
```

```
## normal  1.0  0.0  0.0  1.0  0.0  0.0  1.0  1.0
## speed   0.0  1.0  1.0  0.0  1.0  1.0  0.0  0.0
```

```
print(policy_eval(pi_old,R_s_a,states,P_normal,P_speed))
```

```
## [[-6.65      ]
##  [-1.5       ]
##  [-6.27777778]
##  [-4.17777778]
##  [-0.5       ]
##  [-2.83333333]
##  [-1.28333333]
##  [ 0.        ]]
```

```
pi_old = pi_50
cnt = 0
while(1) :
    print(cnt,"-th iteration")
    print(pi_old.T)

    V_old = policy_eval(pi_old,R_s_a,states,P_normal,P_speed)
    pi_new = policy_imporve(V_old,pi_old,R_s_a,gamma,P_normal,P_speed)
    if(pi_new.eq(pi_old).all().all() == True):
        break
    pi_old=pi_new
    cnt=cnt+1
```

```
## 0 -th iteration
##          0    10   20   30   40   50   60   70
## normal  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5
## speed   0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5
## 1 -th iteration
##          0    10   20   30   40   50   60   70
## normal  0.0  1.0  0.0  1.0  1.0  0.0  1.0  1.0
## speed   1.0  0.0  1.0  0.0  0.0  1.0  0.0  0.0
## 2 -th iteration
##          0    10   20   30   40   50   60   70
## normal  1.0  0.0  0.0  1.0  0.0  0.0  1.0  1.0
## speed   0.0  1.0  1.0  0.0  1.0  1.0  0.0  0.0
```

```
print(policy_eval(pi_old,R_s_a,states,P_normal,P_speed))
```

```
## [[-6.65      ]
##  [-1.5       ]
##  [-6.27777778]
##  [-4.17777778]
##  [-0.5       ]
##  [-2.83333333]
##  [-1.28333333]
##  [ 0.        ]]
```

```
"E2_Exercises"
```