E2_MDP Python

Jaemin Park

2021-01-22

차 례

p.2 policy_eval()

```
gamma = 1
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_speed = np.matrix([[0.1,0,0.9,0,0,0,0,0],[0.1,0,0,0.9,0,0,0,0],
[0,0.1,0,0,0.9,0,0,0],[0,0,0.1,0,0,0.9,0,0],[0,0,0,0.1,0,0,0.9,0],
[0,0,0,0,0.1,0,0,0.9],[0,0,0,0,0,0.1,0,0.9],[0,0,0,0,0,0,0,1]])

def transition(given_pi, states, P_normal, P_speed):
    P_out = pd.DataFrame(np.zeros((len(states),len(states))),states,states)
    for i,s in enumerate(states):
        action_dist = given_pi.loc[s]
        P = action_dist["normal"]*P_normal + action_dist["speed"]*P_speed
        P_out.loc[s] = P[i,:]

    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,states,["normal","speed"])

def reward_fn(given_pi):
    R_pi = np.matrix(given_pi*R_s_a).sum(axis=1)
    R_pi = pd.DataFrame(R_pi,states)
    return(R_pi)
def policy_eval(given_pi):
    R=reward_fn(given_pi)
    P=transition(given_pi, states, P_normal, P_speed)
    gamma=1.0
    epsilon=10**(-8)
    v_old=np.repeat(0,8).reshape(8,1)
    v_new=R+np.dot(gamma*P,v_old)

    while(np.linalg.norm(v_new-v_old)>epsilon):
        v_old=v_new
        v_new=R+np.dot(gamma*P,v_old)
    return v_new
```

```
pi_speed = np.hstack((np.repeat(0,len(states)).reshape(8,1),np.repeat(1,len(states)).reshape(8,1)))
pi_speed = pd.DataFrame(pi_speed,states,["normal","speed"])
print(policy_eval(pi_speed).T)
```

```
##           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 = np.hstack((np.repeat(0.5,len(states)).reshape(8,1),np.repeat(0.5,len(states)).reshape(8,1)))
pi_50 = pd.DataFrame(pi_50,states,["normal","speed"])
print(policy_eval(pi_50).T)
```

```
##           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
```

p.12 Implementation

```
V_old=policy_eval(pi_speed)
pi_old=pi_speed
q_s_a=R_s_a+np.hstack((np.dot(gamma*P_normal,V_old),np.dot(gamma*P_speed,V_old)))
print(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_vec=q_s_a.idxmax(axis=1)
pi_new=pd.DataFrame(np.zeros(pi_old.shape), index=pi_old.index, columns=pi_old.columns)

for i in range(len(pi_new_vec)):
    pi_new.iloc[i][pi_new_vec.iloc[i]]=1
print(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
```

++ For문 없이 구현해본 코드

```
# Not using For loop. But it takes more computational time - ineffiecient
pi_new_speed=q_s_a["speed"]-q_s_a["normal"]
pi_new_normal=pd.DataFrame(np.repeat(1,len(pi_new_speed)).T,states)

pi_new_speed[pi_new_speed<0]=0
pi_new_speed[pi_new_speed>0]=1

pi_new_normal =pi_new_normal - pi_new_speed

pi_new = pd.concat([pi_new_normal,pi_new_speed],axis=1)
pi_new.columns = ['normal','speed']
```

Policy Improve

```
def policy_improve(V_old, pi_old = pi_old, R_s_a = R_s_a, gamma = gamma,
P_normal = P_normal, P_speed = P_speed):
    q_s_a=R_s_a+np.hstack((np.dot(gamma*P_normal,V_old),np.dot(gamma*P_speed,V_old)))
    pi_new_vec=q_s_a.idxmax(axis=1)
    pi_new=pd.DataFrame(np.zeros(pi_old.shape), index=pi_old.index, columns=pi_old.columns)

    for i in range(len(pi_new_vec)):
        pi_new.iloc[i][pi_new_vec.iloc[i]]=1
    return pi_new
```

One step improvement from π^{speed}

```
pi_old = pi_speed
V_old = policy_eval(pi_old)
pi_new = policy_improve(V_old, pi_old = pi_old, R_s_a = R_s_a, gamma = gamma, P_normal = P_normal, P
print(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
```

```
print(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
```

p.18 Policy iteration process (from π^{speed})

```
pi_old = pi_speed
cnt = 0
while True:
    print(cnt,"-th iteration")
    print(pi_old.T)
    V_old = policy_eval(pi_old)
    pi_new = policy_improve(V_old, pi_old = pi_old, R_s_a = R_s_a, gamma = gamma, P_normal = P_norma
    if pi_new.equals(pi_old):
        break
    pi_old = pi_new
    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  0.0  0.0  0.0  1.0  1.0  0.0  1.0  1.0
## speed   1.0  1.0  1.0  0.0  0.0  1.0  0.0  0.0
```

```
print(policy_eval(pi_new))
```

```
##              0
## 0  -5.107744
## 10 -4.410774
## 20 -3.441077
## 30 -2.666667
## 40 -1.666667
## 50 -1.666667
## 60 -1.000000
## 70  0.000000
```

p.19 Policy iteration process (from π^{50})

```
pi_old = pi_50
cnt = 0
while True:
    print(cnt,"-th iteration")
    print(pi_old.T)
    V_old = policy_eval(pi_old)
    pi_new = policy_improve(V_old, pi_old = pi_old, R_s_a = R_s_a, gamma = gamma, P_normal = P_norma
    if pi_new.equals(pi_old):
        break
    pi_old = pi_new
    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  0.0  0.0  0.0  1.0  1.0  0.0  1.0  1.0
## speed   1.0  1.0  1.0  0.0  0.0  1.0  0.0  0.0
```

```
print(policy_eval(pi_new))
```

```
##             0
## 0  -5.107744
## 10 -4.410774
## 20 -3.441077
## 30 -2.666667
## 40 -1.666667
## 50 -1.666667
## 60 -1.000000
## 70  0.000000
```