F4_Exercises

Kwon do yun

2021-02-23

차 례

Write Pol_eval_Q()

```
def pol_eval_Q (sample_step, q_s_a, alpha):
    q_s_a_copy= q_s_a.copy()
    s = sample_step[0]
    a = sample_step[1]
    r = sample_step[2]
    s_next = sample_step[3]

    q_s_a_copy.loc[s,a] +=alpha*(r+max(q_s_a_copy.loc[s_next, ])-q_s_a_copy.loc[s,a])
    return q_s_a_copy
```

Q-learning

```
num_ep = 10**4
beg_time =time.time()
q_s_a = q_s_a_init
pi = pi_50
exploration_rate = 1
for epi_i in range(1,num_ep) :
    s_now = "0"
    while s_now != "70":
        sample_step = simul_step(pi,s_now, P_normal, P_speed, R_s_a)
        q_s_a = pol_eval_TD(sample_step, q_s_a, alpha = 0.01)
        if(epi_i % 100 == 0):
            pi = pol_imp(pi, q_s_a, epsilon=  exploration_rate)
        s_now = sample_step[3]
        exploration_rate *=max(exploration_rate*0.995, 0.001)
end_time =time.time()
result_q = pd.DataFrame(q_s_a, columns =['n','s'], index= states)
result_pi = pd.DataFrame(pi, columns =['n','s'], index= states)
print("Time difference of {} sec".format(end_time- beg_time))
```

```
## Time difference of 22.16135597229004 sec
```

```
print(result_pi.T)
```

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

```
print(result_q.T)
```

```
##           0        10        20        30        40        50        60   70
## n -5.380989 -4.500130 -3.804086 -2.685968 -1.698566 -1.846947 -1.000000  0.0
## s -5.103440 -4.517545 -3.429833 -3.152010 -1.750993 -1.680272 -1.166735  0.0
```

Write Pol_eval_dbl_Q()

```
def pol_eval_dbl_Q(sample_step, q_s_a_1,q_s_a_2, alpha):
    q_s_a_1_copy= q_s_a_1.copy()
    q_s_a_2_copy= q_s_a_2.copy()
    s = sample_step[0]
    a = sample_step[1]
    r = sample_step[2]
    s_next = sample_step[3]

    if np.random.uniform() < 0.5 : # update q_s_a_1
        q_s_a_1_copy.loc[s,a] += alpha*(r+q_s_a_2_copy.loc[s_next,q_s_a_1_copy.loc[s_next,].idxmax(axis="colu
    else :
        q_s_a_2_copy.loc[s,a] += alpha*(r+q_s_a_1_copy.loc[s_next,q_s_a_2_copy.loc[s_next,].idxmax(axis="colu
    return q_s_a_1_copy, q_s_a_2_copy
```

Double Q-learning

```
num_ep = 10**4
beg_time =time.time()
q_s_a_1 = q_s_a_init
q_s_a_2 = q_s_a_init
pi = pi_50
exploration_rate = 1
for epi_i in range(1,num_ep) :
    s_now = "0"
    while s_now != "70":
        sample_step = simul_step(pi,s_now, P_normal, P_speed, R_s_a)
        q_s_a_1, q_s_a_2 = pol_eval_dbl_Q(sample_step, q_s_a_1, q_s_a_2, alpha = 0.01)
        if(epi_i % 100 == 0):
            pi = pol_imp(pi, q_s_a_1+q_s_a_2, epsilon= exploration_rate)
        s_now = sample_step[3]
        exploration_rate *=max(exploration_rate*0.995, 0.001)
end_time =time.time()
result_q = pd.DataFrame(q_s_a, columns =['n','s'], index= states)
result_pi = pd.DataFrame(pi, columns =['n','s'], index= states)
print("Time difference of {} sec".format(end_time- beg_time))
```

```
## Time difference of 33.715481758117676 sec
```

```
print(result_pi.T)
```

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

```
print(result_q.T)
```

```
##           0        10        20        30        40        50        60   70
## n -5.380989 -4.500130 -3.804086 -2.685968 -1.698566 -1.846947 -1.000000  0.0
## s -5.103440 -4.517545 -3.429833 -3.152010 -1.750993 -1.680272 -1.166735  0.0
```

```
"F4_Exercises"
```