F3

Bongseokkim

2021-02-21

차 례

Write Pol_eval Q()

```
def pol_eval_TD(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]
    a_next = sample_step[4]

    q_s_a_copy[np.where(states== int(s)),list(action_dict.values()).index(a)]+=alpha*(r+q_s_a_copy[np.where(s

    return q_s_a_copy



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]
    a_next = sample_step[4]
```

```
    q_s_a_copy[np.where(states== int(s)),list(action_dict.values()).index(a)] +=alpha*(r+max(q_s_a_copy[np.wh

    return q_s_a_copy
```

Q-learning

```
num_ep = 10**5
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_Q(sample_step, q_s_a, alpha = max(1/epi_i, 0.01))
        if(epi_i % 100 ==0 ):
            pi = pol_imp(pi, q_s_a, epsilon= exploration_rate)

        s_now = int(sample_step[3])
        exploration_rate = max(exploration_rate*0.9995, 0.01)

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 28.511730909347534 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 -5.468586 -4.468654 -3.664097 -2.648173 -1.671863 -1.984651 -1.00000  0.0
## s -5.141157 -4.524735 -3.508173 -2.983116 -1.720620 -1.673031 -1.68304  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]
    a_next = sample_step[4] # Not use here

    if np.random.uniform() < 0.5 : # update q_s_a_1
        q_s_a_1_copy[np.where(states== int(s)),list(action_dict.values()).index(a)] +=\
            alpha*(r+max(q_s_a_2_copy[np.where(states== int(s_next))[0][0]]) - q_s_a_1_copy[np.where(states==

    else :
        q_s_a_2_copy[np.where(states== int(s)),list(action_dict.values()).index(a)] +=\
            alpha*(r+max(q_s_a_1_copy[np.where(states== int(s_next))[0][0]]) - q_s_a_2_copy[np.where(states==

    return q_s_a_1_copy, q_s_a_2_copy
```

Double Q-learning

```
num_ep = 10**5
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 = max(1/epi_i, 0.01))

        if(epi_i % 100 ==0 ):

            pi = pol_imp(pi, q_s_a_1+q_s_a_2, epsilon= exploration_rate)

        s_now = int(sample_step[3])
        exploration_rate = max(exploration_rate*0.9995, 0.001)

end_time =time.time()
result_q = pd.DataFrame((q_s_a_1+q_s_a_2)/2, 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 29.562909364700317 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 -5.321469 -4.474759 -3.634614 -2.670574 -1.670637 -1.894187 -1.000000  0.0
## s -5.150704 -4.494792 -3.489028 -2.849408 -1.714412 -1.661479 -1.081319  0.0
```

```
"Done "
```

```
## [1] "Done "
```