Lecture F1. MDP without Model 1

Baek, Jong min

2021-02-15

차 례

Preparation

```
states = np.arange(0,80,10).astype(str)
p_normal = pd.DataFrame(np.array([
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
]).reshape(8,8),index=states, columns=states)
p_speed = pd.DataFrame(np.array([
.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,
]).reshape(8,8),index=states, columns=states)
p_normal
```

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

```
p_speed
```

```
##       0   10   20   30   40   50   60   70
## 0   0.1  0.0  0.9  0.0  0.0  0.0  0.0  0.0
## 10  0.1  0.0  0.0  0.9  0.0  0.0  0.0  0.0
## 20  0.0  0.1  0.0  0.0  0.9  0.0  0.0  0.0
## 30  0.0  0.0  0.1  0.0  0.0  0.9  0.0  0.0
```

```
## 40  0.0  0.0  0.0  0.1  0.0  0.0  0.9  0.0
## 50  0.0  0.0  0.0  0.0  0.1  0.0  0.0  0.9
## 60  0.0  0.0  0.0  0.0  0.0  0.1  0.0  0.9
## 70  0.0  0.0  0.0  0.0  0.0  0.0  0.0  1.0
```

```
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(8,2,orc
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 = pd.DataFrame(np.c_[np.zeros(len(states)),np.repeat(1,len(states))],columns=['n','s'],index=states)
pi_50 = pd.DataFrame(np.c_[np.repeat(0.5,len(states)),np.repeat(0.5,len(states))],columns=['n','s'],index=sta
print(pi_speed.T)
```

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

```
print(pi_50.T)
```

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

Simulation - pi_speed

```
pi = pi_speed
np.random.seed(1234)
history = []
MC_N = 10000

for MC_i in range(MC_N):
  s_now = '0'
  history_i = list(s_now)

  while s_now !='70' :
    if np.random.uniform(0,1) < pi.loc[s_now,'n'] :
      a_now = 'n'
      P = p_normal
    else:
      a_now = 's'
      P = p_speed

    r_now = str(R_s_a.loc[s_now,a_now])
    s_next = states[np.argmin(P.loc[s_now].cumsum() < 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
```

```
pd.Series(map(lambda x : ','.join(x), history_speed[:20]))
```

```
## 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
## dtype: object
```

Siumlator pi_50

```
pi = pi_50
np.random.seed(1234)
history = []
MC_N = 10000

for MC_i in range(MC_N):
  s_now = '0'
  history_i = list(s_now)

  while s_now !='70' :
    if np.random.uniform(0,1) < pi.loc[s_now,'n'] :
      a_now = 'n'
      P = p_normal
    else:
      a_now = 's'
      P = p_speed

    r_now = str(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
```

```
pd.Series(map(lambda x : ','.join(x), history_50[:20]))
```

```
## 0     0,n,-1.0,10,n,-1.0,20,s,-1.5,40,n,0.0,50,s,-1....
## 1     0,n,-1.0,10,s,-1.5,30,n,-1.0,40,s,-0.5,30,s,-1...
## 2     0,s,-1.5,20,n,-1.0,30,n,-1.0,40,s,-0.5,60,s,-1...
## 3     0,s,-1.5,20,n,-1.0,30,n,-1.0,40,n,0.0,50,n,-1....
## 4     0,n,-1.0,10,n,-1.0,20,n,-1.0,30,s,-1.5,20,s,-1...
## 5     0,n,-1.0,10,n,-1.0,20,n,-1.0,30,n,-1.0,40,n,0....
## 6     0,n,-1.0,10,n,-1.0,20,n,-1.0,30,n,-1.0,40,n,0....
## 7             0,s,-1.5,20,s,-1.5,40,s,-0.5,60,n,-1.0,70
## 8     0,s,-1.5,20,n,-1.0,30,s,-1.5,50,n,-1.0,60,s,-1...
## 9     0,s,-1.5,20,s,-1.5,40,n,0.0,50,n,-1.0,60,n,-1....
## 10    0,n,-1.0,10,s,-1.5,30,n,-1.0,40,s,-0.5,60,s,-1...
## 11    0,s,-1.5,20,n,-1.0,30,n,-1.0,40,n,0.0,50,n,-1....
## 12    0,n,-1.0,10,s,-1.5,30,n,-1.0,40,n,0.0,50,s,-1....
```

```
## 13            0,n,-1.0,10,s,-1.5,30,s,-1.5,50,s,-1.5,70
## 14            0,n,-1.0,10,s,-1.5,30,s,-1.5,50,s,-1.5,70
## 15            0,n,-1.0,10,s,-1.5,30,s,-1.5,50,s,-1.5,70
## 16    0,s,-1.5,20,s,-1.5,40,n,0.0,50,n,-1.0,60,n,-1....
## 17            0,s,-1.5,20,s,-1.5,40,s,-0.5,60,n,-1.0,70
## 18    0,n,-1.0,10,n,-1.0,20,s,-1.5,40,n,0.0,50,n,-1....
## 19            0,s,-1.5,20,s,-1.5,40,s,-0.5,60,s,-1.5,70
## dtype: object
```

Implementation1-pi_speed(vetorized)(page 17-18)

```
pol_eval = pd.DataFrame(np.zeros(shape=(len(states),2)),columns=['count','sum'],index=states)
print(pol_eval.T)
```

```
##          0   10   20   30   40   50   60   70
## count  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
## sum    0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
```

```
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'] += np.sum(np.asarray(history_i)[range(j+2,len(history_i)-1,3)].astyp
    else :
        pol_eval.loc[history_i[j],'sum'] +=  0
```

```
print(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
```

```
print(pol_eval.loc[:,'sum']/pol_eval.loc[:,'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
```

Implementation2 - pi_speed(running estimate)(page 19-20)

```
pol_eval = pd.DataFrame(np.zeros(shape=(len(states),2)),columns=['count','est'],index=states)
pol_eval.T
```

```
##          0   10   20   30   40   50   60   70
## count  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
## est    0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
```

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

  # updated 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 = np.sum(np.asarray(history_i)[range(j+2,len(history_i)-1,3)].astype('float'))
    else:
      new_info = 0

  # update the last estimate with new info
    alpha = 1/current_cnt
    pol_eval.loc[history_i[j],'est'] += alpha*(new_info - pol_eval.loc[history_i[j],'est'])
```

```
print(round(pol_eval.T,1))
```

```
##              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
## est       -5.8    -5.2     -4.1    -3.5    -2.3    -1.7    -1.7      0.0
```

Implementation2 - pi_50(running estimate)(page 21-22)

```
pol_eval = pd.DataFrame(np.zeros(shape=(len(states),2)),columns=['count','sum'],index=states)
print(pol_eval.T)
```

```
##          0   10   20   30   40   50   60   70
## count  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
## sum    0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
```

```
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'] += np.sum(np.asarray(history_i)[range(j+2,len(history_i)-1,3)].astyp
    else :
        pol_eval.loc[history_i[j],'sum'] +=  0
```

```
print(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
```

```
print(pol_eval.loc[:,'sum']/pol_eval.loc[:,'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
```

Implementation4 - pi_50(running estimate)(page 23-24)

```
pol_eval = pd.DataFrame(np.zeros(shape=(len(states),2)),columns=['count','est'],index=states)
pol_eval.T
```

```
##          0   10   20   30   40   50   60   70
## count  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
## est    0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
```

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

  # updated 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 = np.sum(np.asarray(history_i)[range(j+2,len(history_i)-1,3)].astype('float'))
    else:
      new_info = 0

  # update the last estimate with new info
    alpha = 1/current_cnt
    pol_eval.loc[history_i[j],'est'] += alpha*(new_info - pol_eval.loc[history_i[j],'est'])
```

```
print(round(pol_eval.T),1)
```

```
##              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
## est       -6.0    -5.0    -4.0    -3.0    -2.0    -2.0    -1.0      0.0 1
```

Temporal Difference policy Evaluation

Implementation 5 - pi_speed(page 35-36)

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

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

  for j in range(0,len(history_i),3):
    # updated count
    pol_eval.loc[history_i[j],'count'] += 1
    current_cnt = pol_eval.loc[history_i[j],'count']

    # build TD target
    if j < len(history_i)-3 :
      TD_tgt = np.array(history_i[j+2]).astype('float')+ pol_eval.loc[history_i[j+3]]['est']
    else :
      TD_tgt = 0

  # update the last estimate with new info
    alpha = 1/current_cnt
    pol_eval.loc[history_i[j],'est'] += alpha*(TD_tgt - pol_eval.loc[history_i[j],'est'])

print(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]
```

Implementation 6 - pi_50(page 37-38)

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

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

  for j in range(0,len(history_i),3):
    # updated count
    pol_eval.loc[history_i[j],'count'] += 1
    current_cnt = pol_eval.loc[history_i[j],'count']

    # build TD target
    if j < len(history_i)-3 :
      TD_tgt = np.array(history_i[j+2]).astype('float')+ pol_eval.loc[history_i[j+3]]['est']
    else :
      TD_tgt = 0

  # update the last estimate with new info
    alpha = 1/current_cnt
    pol_eval.loc[history_i[j],'est'] += alpha*(TD_tgt - pol_eval.loc[history_i[j],'est'])

print(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]
```

F1.Rmd

```
"Hello"
```

```
## [1] "Hello"
```