Inote2_Python_code

Jeong, wonryeol

1/13/2021

Contents

- Prob matrix

$$P = \begin{pmatrix} 0.6 & 0.4 & 0 & 0 & 0 & 0 & 0 \\ 0.4 & 0.2 & 0.4 & 0 & 0 & 0 & 0 \\ 0 & 0.4 & 0.2 & 0.4 & 0 & 0 & 0 \\ 0 & 0 & 0.4 & 0.2 & 0.4 & 0 & 0 \\ 0 & 0 & 0 & 0.4 & 0.2 & 0.4 & 0 \\ 0 & 0 & 0 & 0 & 0.4 & 0.2 & 0.4 \\ 0 & 0 & 0 & 0 & 0 & 0.4 & 0.6 \end{pmatrix}$$

Trainsition Diagram :

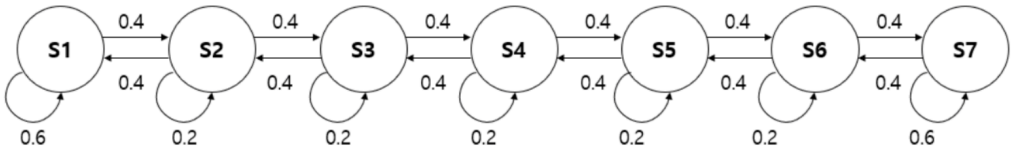

Figure 1: Mars Rover MarkovProcess

- State i is recurrent if, sum of getting prob is 1
- State i is trainsient if, sum of getting prob 1

Therefore, all state is recurrent state

```
import numpy as np

Mc_N = 10000

def mars_simul(state):
  prob = np.random.rand()
  if state == 1 and prob <= 0.4:
    state = 2
  elif state ==7 and prob <= 0.4:
    state = 6
  else:
```

```
    if prob < 0.4:
      state = state+1
    elif prob>0.6:
      state = state-1

  return state
def reward(state):
  if state == 7:
    return 10
  elif state ==1:
    return 1
  else:
    return 0


episode_i = 0

start_state = 4
H = 100
episode_reward = []

while episode_i < Mc_N:
    st = start_state
    result = []

    for i in range(H):
        st = mars_simul(st)
        result.append(st)
        result = list( map (lambda x : reward(x),result))
    re = np.sum(np.array(result))

    episode_reward.append(re)
    episode_i = episode_i +1

episode_reward = np.array(episode_reward)

print(np.mean(episode_reward))
```

```
## 7.9944
```

```
P = np.matrix([[0.6,0.4,0,0,0,0,0],[0.4,0.2,0.4,0,0,0,0],[0,0.4,0.2,0.4,0,0,0],[0,0,0.4,0.2,0.4,0,0],[0
R = np.matrix([1,0,0,0,0,0,10]).reshape(7,1)
H=10
V_t1 = np.matrix([0,0,0,0,0,0,0]).reshape(7,1)
t=H-1
while(t>=0):
    V_t = R + P*V_t1
    t-=1
    V_t1 = V_t
```

```
V_t
```

```
## matrix([[ 5.06485555],
##         [ 3.99416218],
##         [ 4.74779648],
##         [ 8.11882086],
##         [15.21931059],
##         [27.31909325],
##         [45.53596109]])
```