

친절한 강화학습 소개

강화학습 알고리즘의 전반적 이해부터 Q-learning 까지

발표자 : 이동진

강화학습 (Reinforcement Learning) 이란?

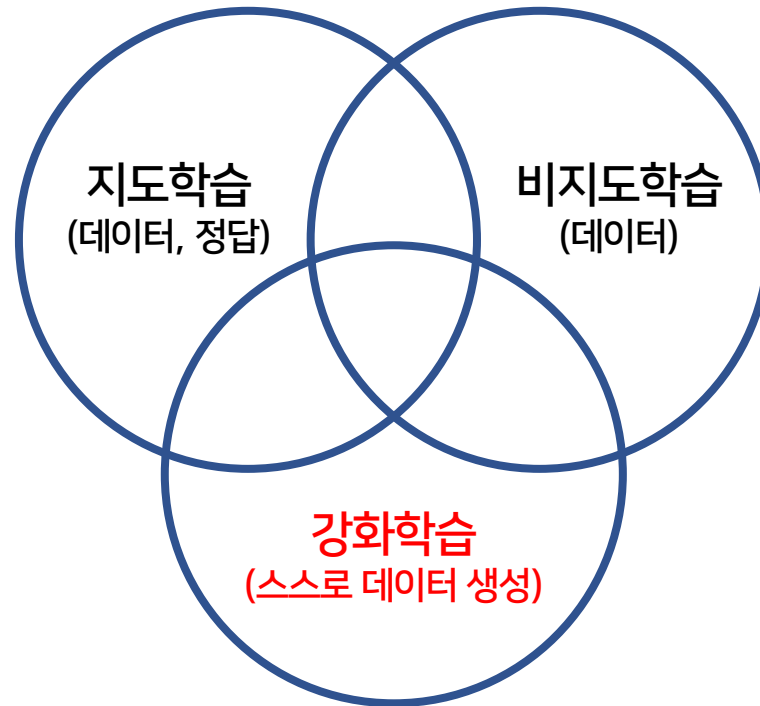


순차적 의사결정 문제 (Sequential decision making problem)

해결에 특화된 머신러닝 알고리즘

강화학습 (Reinforcement Learning) 이란?

머신러닝



순차적 의사결정 문제 (Sequential decision making problem)

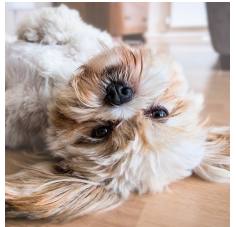
해결에 특화된 머신러닝 알고리즘

강화학습 (Reinforcement Learning) 이란?

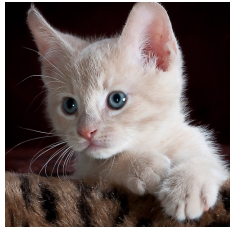


(, 개)

개와 고양이의
특징을 찾습니다 ..



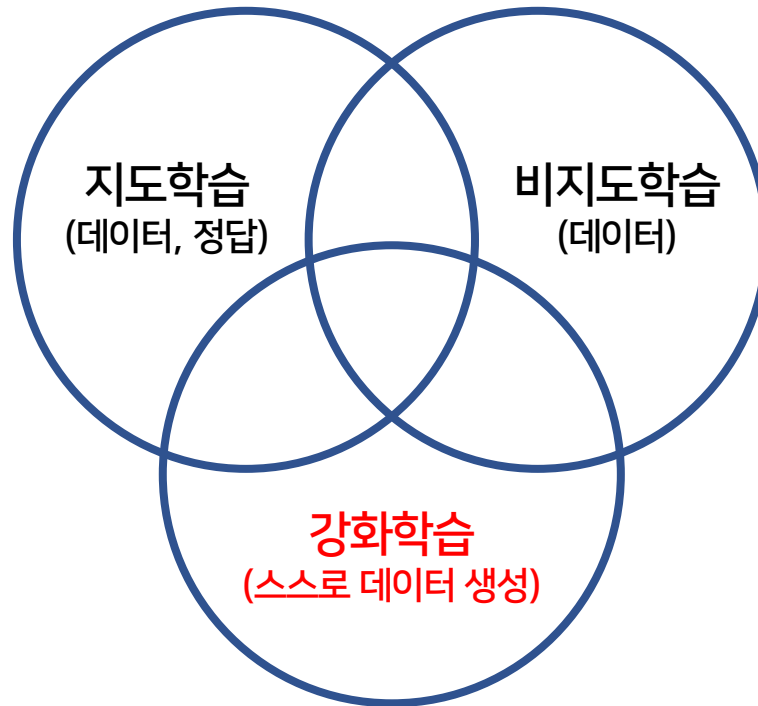
(, 개)



(, 고양이)



머신러닝

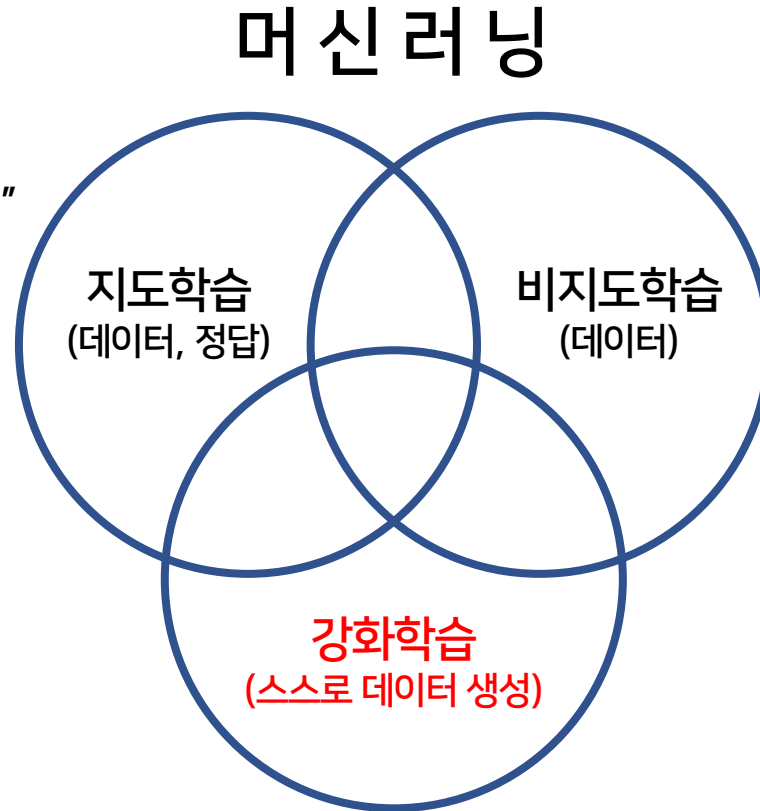


순차적 의사결정 문제 (Sequential decision making problem)

해결에 특화된 머신러닝 알고리즘

https://pixabay.com/users/ty_swartz-617282/
<https://pixabay.com/users/picsbyfran-6087762/>

강화학습 (Reinforcement Learning) 이란?



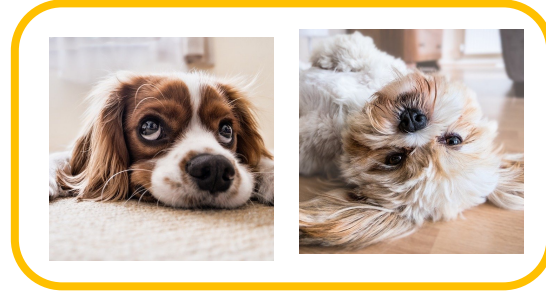
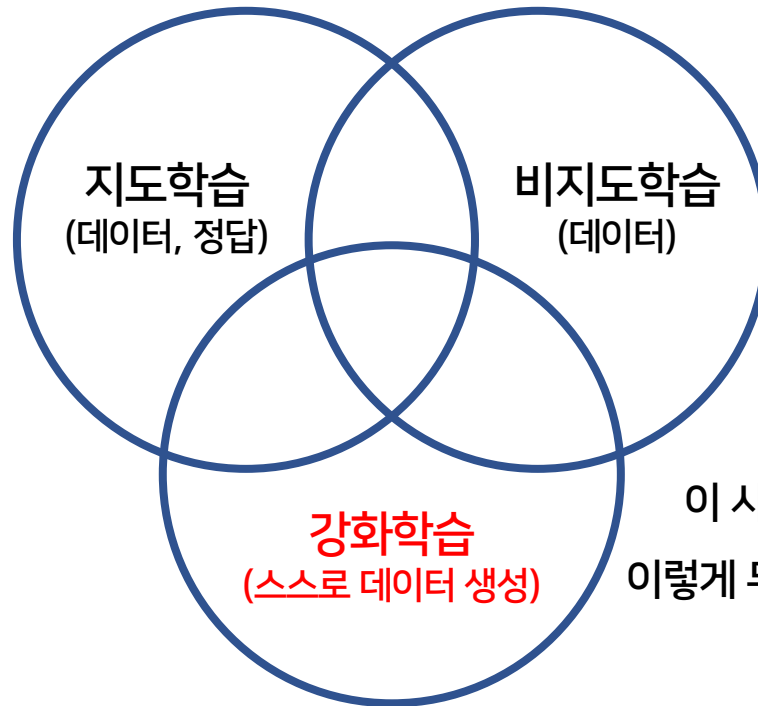
순차적 의사결정 문제 (Sequential decision making problem)

해결에 특화된 머신러닝 알고리즘

https://pixabay.com/users/ty_swartz-617282/
<https://pixabay.com/users/picsbyfran-6087762/>
<https://pixabay.com/photos/cat-pet-licking-animal-tabby-cat-323262/>

강화학습 (Reinforcement Learning) 이란?

머신러닝



이 사진들이 무엇인지 모르겠지만
이렇게 두 집단으로 나눌 수 있을 것 같아

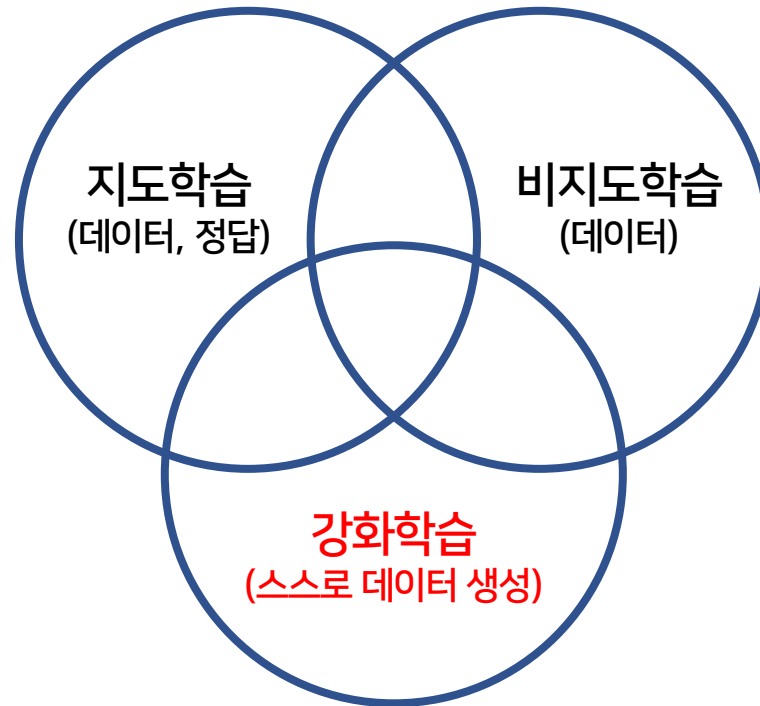
순차적 의사결정 문제 (Sequential decision making problem)

해결에 특화된 머신러닝 알고리즘

https://pixabay.com/users/ty_swartz-617282/
<https://pixabay.com/users/picsbyfran-6087762/>

강화학습 (Reinforcement Learning) 이란?

머신러닝



순차적 의사결정 문제 (Sequential decision making problem)

해결에 특화된 머신러닝 알고리즘

순차적 의사결정 문제 (Sequential decision making problem)

매 timestep 마다 / 현재 주어진 상태를 보고 / 적절한 행동을 취해서 / 보상을 누적

순차적 의사결정 문제 (Sequential decision making problem)



매 timestep 마다 / 현재 주어진 상태를 보고 / 적절한 행동을 취해서 / 보상을 누적

순차적 의사결정 문제 (Sequential decision making problem)



매 timestep 마다 / 현재 주어진 상태를 보고 / 적절한 행동을 취해서 / 보상을 누적

100



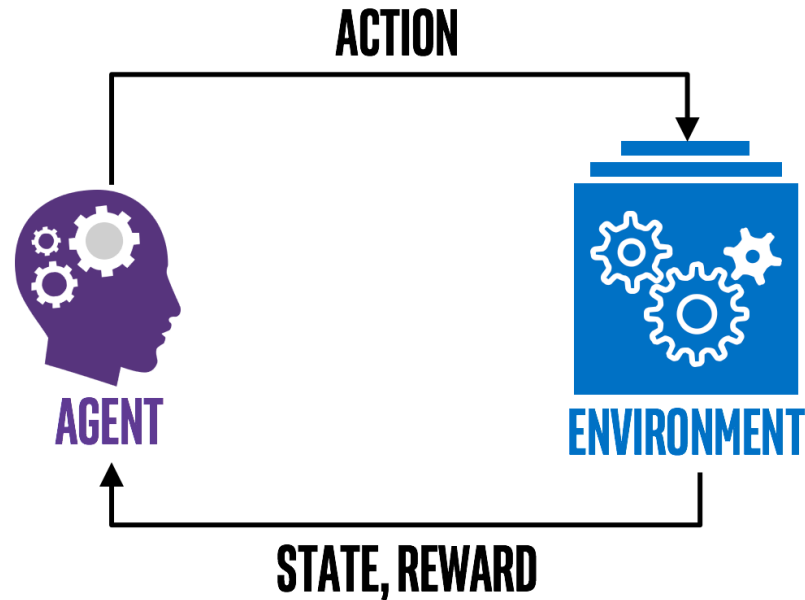
↑
환경의

↑
에이전트가

11/49

에이전트 (Agent) 와 환경 (Environment)

- 에이전트 (Agent) : 의사결정자 (decision maker)
 - 예) 슈퍼마리오 게임을 하는 사람, 인공지능 에어컨 내부에 있는 프로그램, 강화학습 프로그램
- 환경 (Environment) : 에이전트가 상호작용하는 대상
 - 에이전트의 행동에 반응하여 상태를 변경하고 보상을 제공
 - 예) 슈퍼마리오 게임, 에어컨 주변 환경, 시뮬레이션 환경



https://intellabs.github.io/coach/_images/design.png

상태 (State) 와 행동 (Action)

➤ 상태 (State) : 환경의 현재 모습을 수치적으로 표현한 것

▪ \mathcal{S} : 가능한 모든 상태 집합 | s_t : t 시점에서 환경의 상태

예시	상태공간 \mathcal{S}
슈퍼마리오	$\mathcal{S} = \mathbb{R}^4 = [x \text{ 좌표}, y \text{ 좌표}, \text{적과의 거리}, \text{장애물과의 거리}], \mathcal{S} = \text{게임 프레임}$
AI 에어컨	$\mathcal{S} = \mathbb{R}^3 = [\text{온도}, \text{풍속}, \text{풍향}]$

상태 (State) 와 행동 (Action)

➤ 상태 (State) : 환경의 현재 모습을 수치적으로 표현한 것

- \mathcal{S} : 가능한 모든 상태 집합 | s_t : t 시점에서 환경의 상태

예시	상태공간 \mathcal{S}
슈퍼마리오	$\mathcal{S} = \mathbb{R}^4 = [x \text{ 좌표}, y \text{ 좌표}, \text{적과의 거리}, \text{장애물과의 거리}], \mathcal{S} = \text{게임 프레임}$
AI 에어컨	$\mathcal{S} = \mathbb{R}^3 = [\text{온도}, \text{풍속}, \text{풍향}]$

➤ 행동 (Action)

- \mathcal{A} : 가능한 모든 행동 집합 | a_t : t 시점에서 취할 행동

예시	행동공간 \mathcal{A}	행동 $\{a_t\}$
슈퍼마리오	$\mathcal{A} = \{\rightarrow, \leftarrow, \text{점프}, \text{숙이기}, \dots\}$	$a_0 = \text{'}\rightarrow\text{'}, a_1 = \text{'}\rightarrow\text{'}, a_2 = \text{'점프'}$
AI 에어컨	$\mathcal{A} = [-1^\circ\text{C}, 1^\circ\text{C}]$	$a_0 = +0^\circ\text{C}, a_1 = -0.5^\circ\text{C}, a_2 = +0.2^\circ\text{C}$

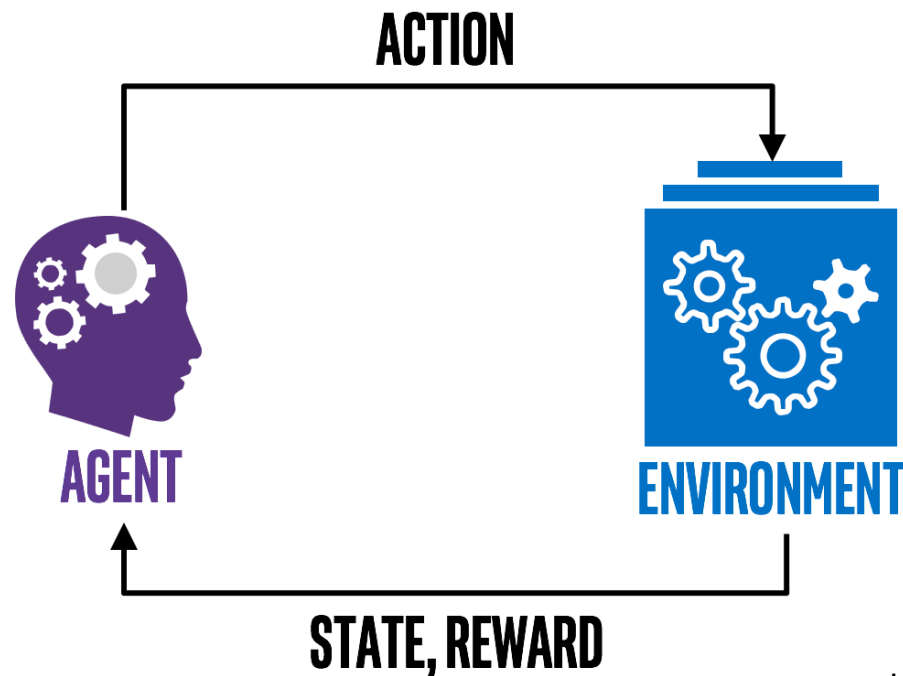
보상 (Reward)

- 보상 (Reward) : 환경의 상태에 에이전트의 행동이 얼마나 적절했는지 환경이 에이전트에게 지급하는 것
- 에이전트가 자신의 행동의 좋고 나쁨을 가늠할 수 있는 신호
- 보상을 어떻게 설정해주느냐에 따라서 학습된 에이전트의 행동 양상이 달라짐

예시	보상
슈퍼마리오	목적지에 도착하면 +100, 죽으면 -100, 매 시간 -0.1, 코인 먹으면 +1
AI 에어컨	사용고객의 칭찬 / 불평

순차적 의사결정 문제 (Sequential decision making problem)

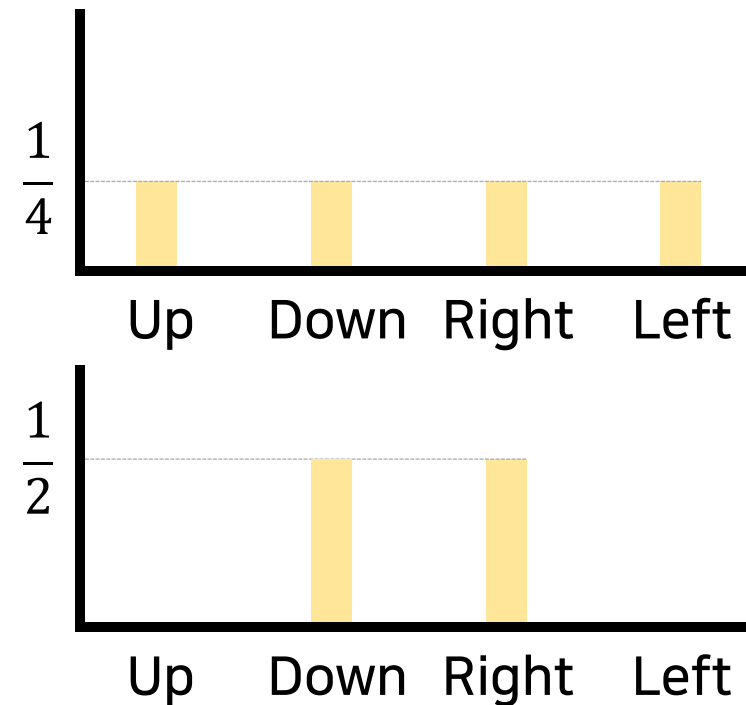
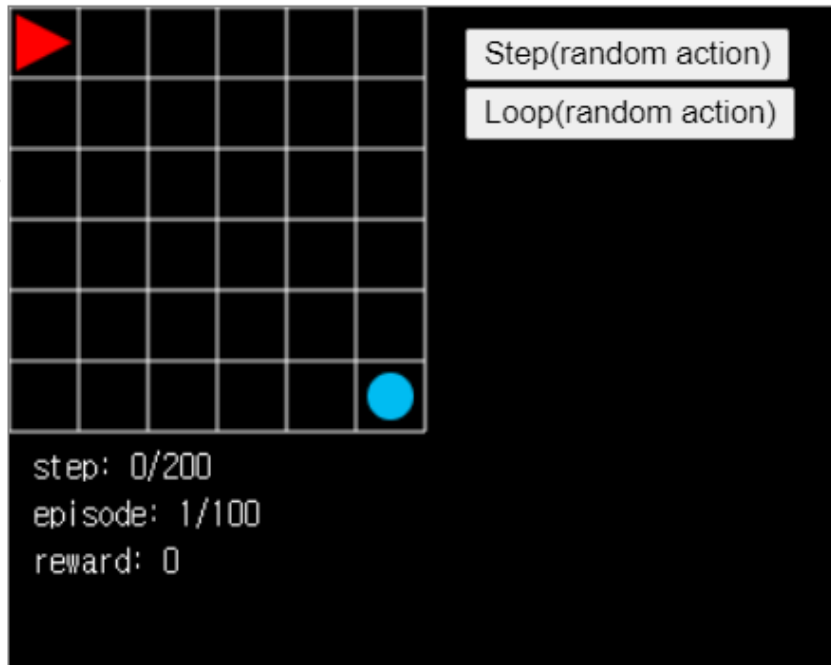
- 환경의 초기 상태 s_0 에서 시작하여, 행동하고, 보상받고, 관찰하는 것을 반복하는 문제
- $s_0, a_0, r_1, s_1, a_1, r_2, s_2, a_2, r_3, \dots, s_{T-1}, a_{T-1}, r_T$
- 에이전트의 목적 : 더 큰 $r_1, r_2, r_3, \dots, r_T$ 을 받을 수 있도록 행동 전략을 수립하는 것



https://intellabs.github.io/coach/_images/design.png

정책 (Policy)

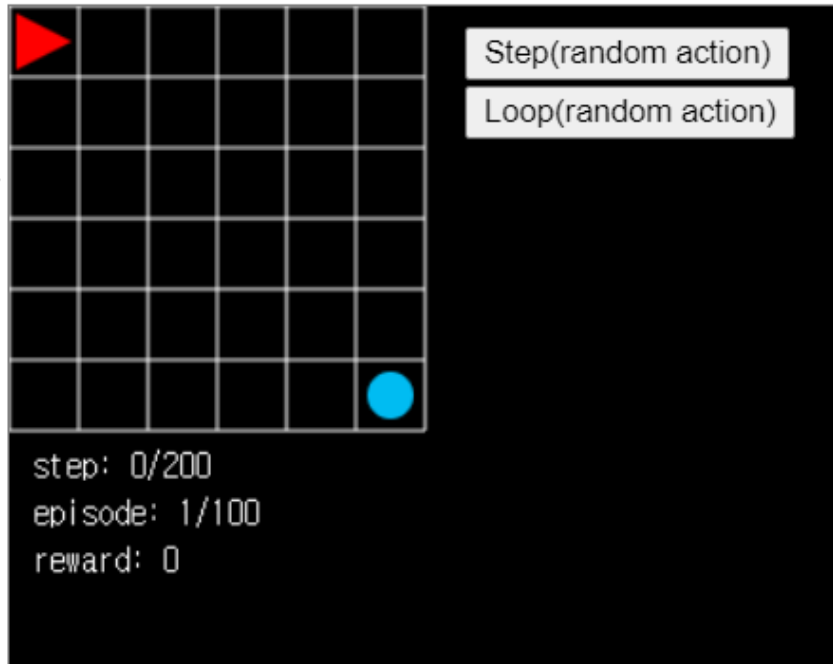
- 주어진 상태 (state)에서 어떻게 행동 (action)해야 할지 알려주는 지침서. 조금 더 명확하게는,
- $\pi : \mathcal{S} \times \mathcal{A} \rightarrow [0,1]$, state $s \in \mathcal{S}$ 에서 각 action $a \in \mathcal{A}$ 을 선택할 조건부 확률 $P(A_t = a | S_t = s)$
- 간결하게 $\pi(a|s)$ 로 적어준다.



<https://greentec.github.io/reinforcement-learning-first/>

정책 (Policy)

- 주어진 상태 (state)에서 어떻게 행동 (action)해야 할지 알려주는 지침서. 조금 더 명확하게는,
- $\pi : \mathcal{S} \times \mathcal{A} \rightarrow [0,1]$, state $s \in \mathcal{S}$ 에서 각 action $a \in \mathcal{A}$ 을 선택할 조건부 확률 $P(A_t = a | S_t = s)$
- 간결하게 $\pi(a|s)$ 로 적어준다.



한 가지 최적의 정책

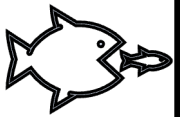
$p(a = \rightarrow s = 1) = 1$	$p(a = \downarrow s = 6) = 1$
$p(a = \rightarrow s = 2) = 1$	$p(a = \downarrow s = 12) = 1$
$p(a = \rightarrow s = 3) = 1$	$p(a = \downarrow s = 18) = 1$
$p(a = \rightarrow s = 4) = 1$	$p(a = \downarrow s = 24) = 1$
$p(a = \rightarrow s = 5) = 1$	$p(a = \downarrow s = 30) = 1$

<https://greentec.github.io/reinforcement-learning-first/>

어떤 정책 (Policy) 을 찾을 것인가?

- 강화학습에 대한 아주 흔한 오해 : 보상을 최대화하도록 행동한다 ?
 - π_1 : 당장 받을 수 있는 가장 큰 보상을 받도록 행동하는 policy

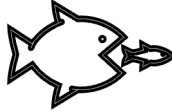
$G = 0$ (누적보상)

	+100	+1	+101
+0			
+100			
+200			

어떤 정책 (Policy) 을 찾을 것인가?

- 강화학습에 대한 아주 흔한 오해 : 보상을 최대화하도록 행동한다 ?
 - π_1 : 당장 받을 수 있는 가장 큰 보상을 받도록 행동하는 policy

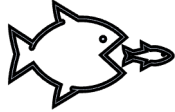
$G = 100$ (누적보상)

+0		+1	+101
+0			
+100			
+200			

어떤 정책 (Policy) 을 찾을 것인가?

- 강화학습에 대한 아주 흔한 오해 : 보상을 최대화하도록 행동한다 ?
 - π_1 : 당장 받을 수 있는 가장 큰 보상을 받도록 행동하는 policy

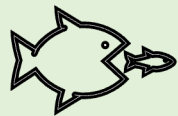
$G = 101$ (누적보상)

+0	+100		+101
+0			
+100			
+200			

어떤 정책 (Policy) 을 찾을 것인가?

- 강화학습에 대한 아주 흔한 오해 : 보상을 최대화하도록 행동한다 ?
 - π_1 : 당장 받을 수 있는 가장 큰 보상을 받도록 행동하는 policy

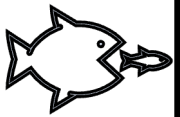
$G = 202$ (누적보상)

+0	+100	+1	
+0			
+100			
+200			

어떤 정책 (Policy) 을 찾을 것인가?

- 강화학습에 대한 아주 흔한 오해 : 보상을 최대화하도록 행동한다 ?
 - π_2 : (어떻게 알았는지는 모르겠지만) 미래 받을 보상까지 고려하여 행동하는 policy

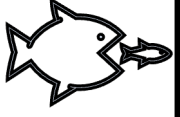
$G = 0$ (누적보상)

	+100	+1	+101
+0			
+100			
+200			

어떤 정책 (Policy) 을 찾을 것인가?

- 강화학습에 대한 아주 흔한 오해 : 보상을 최대화하도록 행동한다 ?
 - π_2 : (어떻게 알았는지는 모르겠지만) 미래 받을 보상까지 고려하여 행동하는 policy

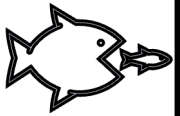
$G = 0$ (누적보상)

+0	+100	+1	+101
			
+100			
+200			

어떤 정책 (Policy) 을 찾을 것인가?

- 강화학습에 대한 아주 흔한 오해 : 보상을 최대화하도록 행동한다 ?
 - π_2 : (어떻게 알았는지는 모르겠지만) 미래 받을 보상까지 고려하여 행동하는 policy


$G = 100$ (누적보상)

+0	+100	+1	+101
+0			
			
+200			

어떤 정책 (Policy) 을 찾을 것인가?

- 강화학습에 대한 아주 흔한 오해 : 보상을 최대화하도록 행동한다 ?
 - π_2 : (어떻게 알았는지는 모르겠지만) 미래 받을 보상까지 고려하여 행동하는 policy

$G = 300$ (누적보상)

+0	+100	+1	+101
+0			
+100			
			

강화학습의 전부인 가치 함수 (1) : 상태-가치 함수

➤ 누적보상 (return, G_t)

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots + \gamma^{T-t-1} R_T$$

강화학습의 전부인 가치 함수 (1) : 상태-가치 함수

- 누적보상 (return, G_t)

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots + \gamma^{T-t-1} R_T$$

- 상태 가치 함수 (state-value function, $v_\pi: \mathcal{S} \rightarrow \mathbb{R}$)

$$v_\pi(s) = \mathbb{E}_\pi[G_t | S_t = s]$$

- t 시점의 상태가 s 일 때, policy π 를 따라서 에피소드를 진행했을 때 얻게 되는 기대 누적보상

강화학습의 전부인 가치 함수 (1) : 상태-가치 함수

- 누적보상 (return, G_t)

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \dots + \gamma^{T-t-1} R_T$$

- 상태 가치 함수 (state-value function, $v_\pi: \mathcal{S} \rightarrow \mathbb{R}$)

$$v_\pi(s) = \mathbb{E}_\pi[G_t | S_t = s]$$

- t 시점의 상태가 s 일 때, policy π 를 따라서 에피소드를 진행했을 때 얻게 되는 기대 누적보상

- 최적의 policy $\pi^* = \underset{\pi}{\operatorname{argmax}} v_\pi(s) \quad \forall s$

강화학습의 전부인 가치 함수 (2) : 행동-가치 함수

- 누적보상 (return, G_t)

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \dots + \gamma^{T-t-1} R_T$$

- 행동 가치 함수 (state-value function, $q_\pi: \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$)

$$q_\pi(s, a) = \mathbb{E}_\pi[G_t | S_t = s, A_t = a]$$

- t 시점의 상태가 s 일 때, 행동 a 를 취하고 policy π 를 따라서 에피소드를 진행했을 때 얻게 되는 기대 누적보상

강화학습의 전부인 가치 함수 (2) : 행동-가치 함수

- 누적보상 (return, G_t)

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \dots + \gamma^{T-t-1} R_T$$

- 행동 가치 함수 (state-value function, $q_\pi: \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$)

$$q_\pi(s, a) = \mathbb{E}_\pi[G_t | S_t = s, A_t = a]$$

- t 시점의 상태가 s 일 때, 행동 a 를 취하고 policy π 를 따라서 에피소드를 진행했을 때 얻게 되는 기대 누적보상

- 상태 s 에서 $v_\pi(s) \leq q_\pi(s, a)$ 인 행동 a 가 있다는 것의 의미?

강화학습의 전부인 가치 함수 (2) : 행동-가치 함수

- 누적보상 (return, G_t)

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \dots + \gamma^{T-t-1} R_T$$

- 행동 가치 함수 (state-value function, $q_\pi: \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$)

$$q_\pi(s, a) = \mathbb{E}_\pi[G_t | S_t = s, A_t = a]$$

- t 시점의 상태가 s 일 때, 행동 a 를 취하고 policy π 를 따라서 에피소드를 진행했을 때 얻게 되는 기대 누적보상
- 상태 s 에서 $v_\pi(s) \leq q_\pi(s, a)$ 인 행동 a 가 있다는 것의 의미?
 - 상태 s 에서는 policy π 에 따라 행동하는 것보다 다른 행동 a 를 취했을 때 더 기대보상이 높다.

강화학습의 전부인 가치 함수 (2) : 행동-가치 함수

- 누적보상 (return, G_t)

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \dots + \gamma^{T-t-1} R_T$$

- 행동 가치 함수 (state-value function, $q_\pi: \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$)

$$q_\pi(s, a) = \mathbb{E}_\pi[G_t | S_t = s, A_t = a]$$

- t 시점의 상태가 s 일 때, 행동 a 를 취하고 policy π 를 따라서 에피소드를 진행했을 때 얻게 되는 기대 누적보상
- 상태 s 에서 $v_\pi(s) \leq q_\pi(s, a)$ 인 행동 a 가 있다는 것의 의미?
 - 상태 s 에서는 policy π 에 따라 행동하는 것보다 다른 행동 a 를 취했을 때 더 기대보상이 높다.
 - 지금 policy π 는 안 좋으니까 개선하자! 어떻게?

강화학습의 전부인 가치 함수 (2) : 행동-가치 함수

- 누적보상 (return, G_t)

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \dots + \gamma^{T-t-1} R_T$$

- 행동 가치 함수 (state-value function, $q_\pi: \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$)

$$q_\pi(s, a) = \mathbb{E}_\pi[G_t | S_t = s, A_t = a]$$

- t 시점의 상태가 s 일 때, 행동 a 를 취하고 policy π 를 따라서 에피소드를 진행했을 때 얻게 되는 기대 누적보상

- 상태 s 에서 $v_\pi(s) \leq q_\pi(s, a)$ 인 행동 a 가 있다는 것의 의미?

- 상태 s 에서는 policy π 에 따라 행동하는 것보다 다른 행동 a 를 취했을 때 더 기대보상이 높다.

- 지금 policy π 는 안 좋으니까 개선하자! 어떻게?
$$\begin{aligned} \pi'(a|s) &= 1 && \text{if } a = \operatorname{argmax}_{a \in \mathcal{A}} q_\pi(s, a) \\ &= 0 && \text{otherwise} \end{aligned}$$

정책 평가와 개선을 반복하면 최적의 policy를 찾을 수 있다.

- Policy evaluation (정책 평가) : 주어진 policy π 의 $v_\pi(s)$ 와 $q_\pi(s, a)$ 을 계산
- Policy improvement (정책 개선) :
$$\pi'(a|s) = \begin{cases} 1 & \text{if } a = \operatorname{argmax}_{a \in \mathcal{A}} q_\pi(s, a) \\ 0 & \text{otherwise} \end{cases}$$
- 그래서 어떻게 $v_\pi(s)$ 와 $q_\pi(s, a)$ 을 구하죠?

$$v_\pi(s) = \mathbb{E}_\pi[G_t | S_t = s]$$

$$q_\pi(s, a) = \mathbb{E}_\pi[G_t | S_t = s, A_t = a]$$

Stochastic Approximation

- 기댓값을 점차 근사시키는 방법
- 표본평균을 이용

$$\begin{aligned}\mathbb{E}[X] \approx S_N &= \frac{1}{N} \sum_{i=1}^N X_i \\ &= \frac{1}{N} \left(\sum_{i=1}^{N-1} X_i + X_N \right) \\ &= \frac{1}{N} ((N-1)S_{N-1} + X_N) \\ &= S_{N-1} + \frac{1}{N} (X_N - S_{N-1})\end{aligned}$$

- Stochastic approximation

$$S_N \leftarrow S_{N-1} + \alpha(X - S_{N-1})$$

MonteCarlo를 이용한 policy evaluation

Stochastic approximation $S_N \approx \mathbb{E}[X]$

$$S_N \leftarrow S_{N-1} + \alpha(X - S_{N-1})$$

➤ 목적 : 주어진 policy π 의 행동-가치 함수 $q_\pi(s, a)$ 찾기

- 행동-가치 함수 $q_\pi(s_t, a_t) = \mathbb{E}_\pi[G_t | S_t = s_t, A_t = a_t]$
- 업데이트식 $Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha(G_t - Q(s_t, a_t))$

MonteCarloPrediction($\pi, \gamma, \alpha, n_episodes$):

1. $Q \leftarrow |S| \times |A|$ 크기의 영행렬
2. for i in range(n_episodes):
3. $s_0, a_0, r_1, s_1, a_1, r_2, \dots, s_{T-1}, a_{T-1}, r_T \leftarrow$ policy π 를 따라서 에피소드 진행
4. for (s_t, a_t) 순서쌍 in the trajectory:
5. $G_t \leftarrow r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \dots + \gamma^{T-t-1} r_T$
6. $Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha(G_t - Q(s_t, a_t))$
7. Return Q

MonteCarlo를 이용한 최적의 Policy 찾기

➤ 목적 : 주어진 환경에서 최적의 policy 찾기

- 행동-가치 함수 $q_{\pi}(s_t, a_t) = \mathbb{E}_{\pi}[G_t | S_t = s_t, A_t = a_t]$
- 업데이트식 $Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha(G_t - Q(s_t, a_t))$
- Policy improvement (정책 개선) 자동화 : ϵ - greedy policy

MonteCarloControl($\gamma, \alpha, n_episodes$):

1. $Q \leftarrow |\mathcal{S}| \times |\mathcal{A}|$ 크기의 영행렬
2. $\pi \leftarrow \epsilon$ - greedy policy
3. for i in range($n_episodes$):
4. $s_0, a_0, r_1, s_1, a_1, r_2, \dots, s_{T-1}, a_{T-1}, r_T \leftarrow$ policy π 를 따라서 에피소드 진행
5. for (s_t, a_t) 순서쌍 in the trajectory:
6. $G_t \leftarrow r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \dots + \gamma^{T-t-1} r_T$
7. $Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha(G_t - Q(s_t, a_t))$
8. Return Q

Stochastic approximation $S_N \approx \mathbb{E}[X]$

$$S_N \leftarrow S_{N-1} + \alpha(X - S_{N-1})$$

Policy improvement (정책 개선)

$$\begin{aligned} \pi'(a|s) &= 1 && \text{if } a = \underset{a \in \mathcal{A}}{\operatorname{argmax}} q_{\pi}(s, a) \\ &= 0 && \text{otherwise} \end{aligned}$$

ϵ - greedy policy

ϵ 의 확률로 랜덤 행동을 취하고,
 $1 - \epsilon$ 의 확률로 $a = \underset{a \in \mathcal{A}}{\operatorname{argmax}} Q(s, a)$ 를 취하는 policy

TD를 이용한 policy evaluation

Stochastic approximation $S_N \approx \mathbb{E}[X]$

$$S_N \leftarrow S_{N-1} + \alpha(X - S_{N-1})$$

- 목적 : 주어진 policy π 의 행동-가치 함수 $q_\pi(s, a)$ 찾기
 - 행동-가치 함수

$$q_\pi(s_t, a_t) = \mathbb{E}_\pi[G_t | S_t = s_t, A_t = a_t]$$

TD를 이용한 policy evaluation

Stochastic approximation $S_N \approx \mathbb{E}[X]$

$$S_N \leftarrow S_{N-1} + \alpha(X - S_{N-1})$$

➤ 목적 : 주어진 policy π 의 행동-가치 함수 $q_\pi(s, a)$ 찾기

▪ 행동-가치 함수

$$\begin{aligned} q_\pi(s_t, a_t) &= \mathbb{E}_\pi[G_t | S_t = s_t, A_t = a_t] \\ &= \mathbb{E}_\pi[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \dots | S_t = s_t, A_t = a_t] \end{aligned}$$

TD를 이용한 policy evaluation

Stochastic approximation $S_N \approx \mathbb{E}[X]$

$$S_N \leftarrow S_{N-1} + \alpha(X - S_{N-1})$$

➤ 목적 : 주어진 policy π 의 행동-가치 함수 $q_\pi(s, a)$ 찾기

▪ 행동-가치 함수

$$\begin{aligned} q_\pi(s_t, a_t) &= \mathbb{E}_\pi[G_t | S_t = s_t, A_t = a_t] \\ &= \mathbb{E}_\pi[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots | S_t = s_t, A_t = a_t] \\ &= \mathbb{E}_\pi[R_{t+1} + \gamma G_{t+1} | S_t = s_t, A_t = a_t] \end{aligned}$$

TD를 이용한 policy evaluation

Stochastic approximation $S_N \approx \mathbb{E}[X]$

$$S_N \leftarrow S_{N-1} + \alpha(X - S_{N-1})$$

➤ 목적 : 주어진 policy π 의 행동-가치 함수 $q_\pi(s, a)$ 찾기

▪ 행동-가치 함수

$$\begin{aligned} q_\pi(s_t, a_t) &= \mathbb{E}_\pi[G_t | S_t = s_t, A_t = a_t] \\ &= \mathbb{E}_\pi[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \dots | S_t = s_t, A_t = a_t] \\ &= \mathbb{E}_\pi[R_{t+1} + \gamma G_{t+1} | S_t = s_t, A_t = a_t] \\ &= \mathbb{E}_\pi[R_{t+1} + \gamma q_\pi(S_{t+1}, A_{t+1}) | S_t = s_t, A_t = a_t] \end{aligned}$$

TD를 이용한 policy evaluation

Stochastic approximation $S_N \approx \mathbb{E}[X]$

$$S_N \leftarrow S_{N-1} + \alpha(X - S_{N-1})$$

➤ 목적 : 주어진 policy π 의 행동-가치 함수 $q_\pi(s, a)$ 찾기

- 행동-가치 함수 $q_\pi(s_t, a_t) = \mathbb{E}_\pi[R_{t+1} + \gamma q_\pi(S_{t+1}, A_{t+1}) | S_t = s_t, A_t = a_t]$
- 업데이트식 $Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha(r_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t))$

TemporalDifferenceLearning($\pi, \gamma, \alpha, n_episodes$):

1. $Q \leftarrow |S| \times |A|$ 크기의 영행렬
2. for i in range($n_episodes$):
3. $s_0 \leftarrow$ 초기 상태
4. $a_0 \leftarrow$ policy π 를 따라서 선택
5. for t in range(T):
6. $r_{t+1}, s_{t+1} \leftarrow$ 환경에 a_t 를 실시하고 관찰한 보상과 다음 상태
7. $a_{t+1} \leftarrow$ policy π 를 따라서 선택
6. $Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha(r_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t))$
7. $s_{t+1} \leftarrow s_t, a_{t+1} \leftarrow a_t$
8. Return Q

TD를 이용한 최적의 policy 찾기

➤ 목적 : 주어진 policy π 의 행동-가치 함수 $q_\pi(s, a)$ 찾기

- 행동-가치 함수 $q_\pi(s_t, a_t) = \mathbb{E}_\pi[R_{t+1} + \gamma q_\pi(S_{t+1}, A_{t+1}) | S_t = s_t, A_t = a_t]$
- 업데이트식 $Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha(r_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t))$
- Policy improvement (정책 개선) 자동화 : $\epsilon - greedy$ policy

SARSA($\gamma, \alpha, n_episodes$):

1. $Q \leftarrow |\mathcal{S}| \times |\mathcal{A}|$ 크기의 영행렬
2. $\pi \leftarrow \epsilon - greedy$ policy
3. for i in range(n_episodes):
4. $s_0 \leftarrow$ 초기 상태
5. $a_0 \leftarrow$ policy π 를 따라서 선택
6. for t in range(T):
7. $r_{t+1}, s_{t+1} \leftarrow$ 환경에 a_t 를 실시하고 관찰한 보상과 다음 상태
8. $a_{t+1} \leftarrow$ policy π 를 따라서 선택
9. $Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha(r_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t))$
10. $s_{t+1} \leftarrow s_t, a_{t+1} \leftarrow a_t$
11. Return Q

Stochastic approximation $S_N \approx \mathbb{E}[X]$

$$S_N \leftarrow S_{N-1} + \alpha(X - S_{N-1})$$

Policy improvement (정책 개선)

$$\begin{aligned} \pi'(a|s) &= 1 && \text{if } a = \underset{a \in \mathcal{A}}{\operatorname{argmax}} q_\pi(s, a) \\ &= 0 && \text{otherwise} \end{aligned}$$

$\epsilon - greedy$ policy

ϵ 의 확률로 랜덤 행동을 취하고,
 $1 - \epsilon$ 의 확률로 $a = \underset{a \in \mathcal{A}}{\operatorname{argmax}} Q(s, a)$ 를 취하는 policy

Q-learning을 이용한 최적의 policy 찾기

Stochastic approximation $S_N \approx \mathbb{E}[X]$

$$S_N \leftarrow S_{N-1} + \alpha(X - S_{N-1})$$

Policy improvement (정책 개선)

$$\begin{aligned}\pi'(a|s) &= 1 && \text{if } a = \operatorname{argmax}_{a \in \mathcal{A}} q_\pi(s, a) \\ &= 0 && \text{otherwise}\end{aligned}$$

ϵ - greedy policy

ϵ 의 확률로 랜덤 행동을 취하고,
 $1 - \epsilon$ 의 확률로 $a = \operatorname{argmax}_{a \in \mathcal{A}} Q(s, a)$ 를 취하는 policy

➤ 목적 : 주어진 policy π 의 행동-가치 함수 $q_\pi(s, a)$ 찾기

- 행동-가치 함수 $q_\pi(s_t, a_t) = \mathbb{E}_\pi[R_{t+1} + \gamma q_\pi(S_{t+1}, A_{t+1}) | S_t = s_t, A_t = a_t]$
- 업데이트식 $Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha(r_{t+1} + \gamma \max_{a'} Q(s_{t+1}, a') - Q(s_t, a_t))$
- Policy improvement (정책 개선) 자동화 : ϵ - greedy policy

Q-learning($\gamma, \alpha, n_episodes$):

1. $Q \leftarrow |\mathcal{S}| \times |\mathcal{A}|$ 크기의 영행렬
2. $\pi \leftarrow \epsilon$ - greedy policy
3. for i in range($n_episodes$):
4. $s_0 \leftarrow$ 초기 상태
5. $a_0 \leftarrow$ policy π 를 따라서 선택
6. for t in range(T):
7. $r_{t+1}, s_{t+1} \leftarrow$ 환경에 a_t 를 실시하고 관찰한 보상과 다음 상태
8. $a_{t+1} \leftarrow$ policy π 를 따라서 선택
9. $Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha(r_{t+1} + \gamma \max_{a'} Q(s_{t+1}, a') - Q(s_t, a_t))$
10. $s_{t+1} \leftarrow s_t, a_{t+1} \leftarrow a_t$
11. Return Q

강화학습 누가 물어보면?

- 강화학습은 순차적 의사결정 문제 (sequential decision-making problem) 을 해결하는 머신러닝 알고리즘
- 강화학습은 보상을 최대화하는 것이 아닌, (할인률이 적용된) 보상의 기댓값을 최대화하도록 행동전략 수립하는 방법
- 주어진 policy의 가치함수를 추정하는 policy evaluation과 가치함수를 사용하여 policy를 개선하는 policy improvement를 반복하여 최적의 policy 근사
- 가장 기초적인 강화학습 알고리즘 : MonteCarlo Control , SARSA, Q-learning

들어주셔서 감사합니다 휴먼

