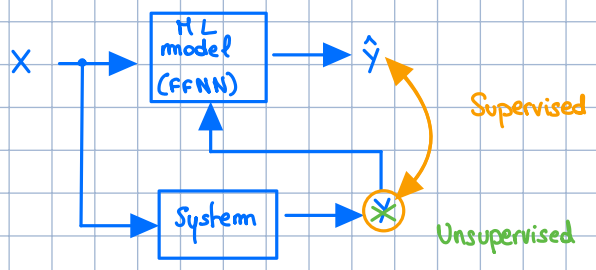
Reinforcement Learning

Summary

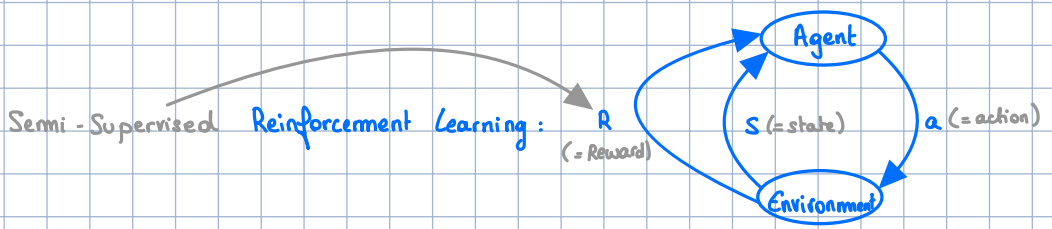
- HDP
- Policy function
- Value function
- Bellman equation
- Optimal value function
- Optimal policy function
- Value interaction
- Policy interaction
- Q - Learning

Overview of Machine Learning

HL { Supervised Learning
Unsupervised Learning
Reinforcement Learning ← Today



	Known	Unknown
Supervised Learning	X, Y	$\theta = (v, w)$ → model parameters
Unsupervised Learning	X	Y, θ
Reinforcement Learning	X, ____	a → actions



HDP

RL models the world (environment + agent) using the Markov Decision Process formalism
HDP

Markov : whenever the future can be predicted knowing only the present

HDP is a 5-tuple $(S, A, \{P_{sa}\}, \gamma, R)$:

- $S \triangleq$ state space : all the values that a state can take
- $A \triangleq$ action space : all the values that an action can take
- $\{P_{sa}\} \triangleq$ state transition distribution : probability of ending up at state s' from state s by taking action a

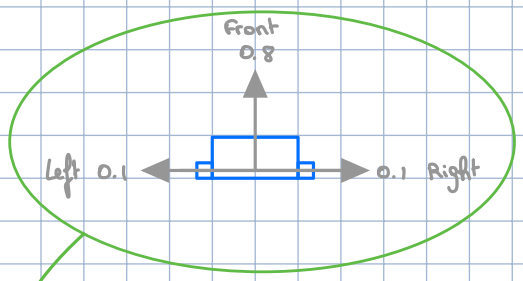
$$s \xrightarrow[R]{a} s', \quad 0 \leq P_{sa}(s') \leq 1$$
$$\sum_{s'} P_{sa}(s') = 1$$

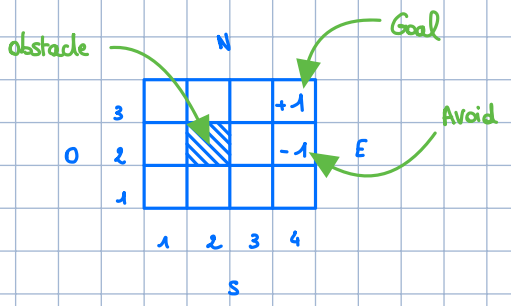
- $\gamma \triangleq$ discount factor : The importance of the future with respect to the present

$$0 \leq \gamma \leq 1$$

- $R : S \longrightarrow \mathbb{R}$ $\triangleq$ Reward function

Example :





$S = \{(1,1), (2,1), (3,1), \ldots, (4,3)\}$

$|S| = 11$

$A = \{N, S, E, W\}$

$|A|$

$\{P_{sa}\}$ $\longrightarrow$ $P_{(3,1),N}((3,2)) = 0.8$ $\quad P_{(3,1),N}((2,1)) = 0.1$ $\quad P_{(3,1),N}((4,1)) = 0.1$

$P_{(3,1),N}((3,3)) = 0$ $\quad P_{(3,1),W}((2,1)) = 0$

$\gamma = 0.99$ (future is very important)

$$R = \begin{cases} +1 & \text{if } s = (4,3) \\ -1 & \text{if } s = (4,2) \\ -0.02 & \text{otherwise} \end{cases}$$

(encourage robot to reach goal as quickly as possible + energy consumption)

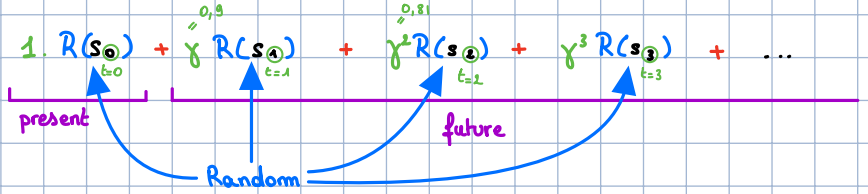
Stop condition: An episode finishes when the robot hits the celle (4,3) or (4,2)

Performance measure : How well the robot did by visiting the sequence of states $s_0, s_1, s_2, \ldots$?

s_0 : time = 0, s_1 : time = 1, s_2 : time = 2

1) Define the reward function
2) Apply it to the sequence of states
3) Compare the discounted cumulative reward or total payoff

Expected total payoff $= \mathbb{E}\left[1 \cdot R(s_0) + \gamma R(s_1) + \gamma^2 R(s_2) + \gamma^3 R(s_3) + \ldots \right]$



Goal of RL

Choose actions overtime $(a_0, a_1, a_2, \ldots)$ that maximize the expected total payoff

How ?

Define :

1) Policy function $\pi : S \longrightarrow A$

It is a function that recommends which action one should take for a given state

2) State-value function $V^\pi : S \longrightarrow \mathbb{R}$

For any given π, $V^\pi(s)$ is the expected total payoff starting from state s and following the policy π.

$$V^\pi(s) = \mathbb{E}\left[\underbrace{R(s_0) + \gamma R(s_1) + \gamma^2 R(s_2) + \dots}_{\text{Random}} \;\middle|\; \underbrace{s_0 = s, \pi}_{\text{Knowns}} \right]$$

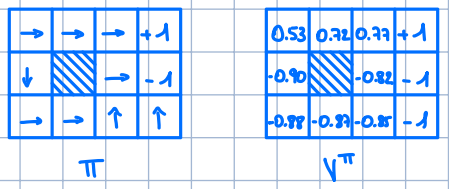
(knowing that)

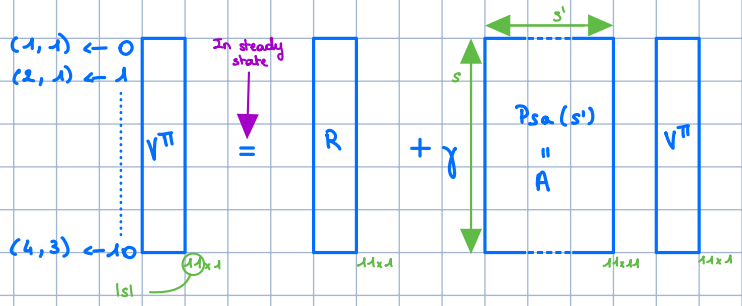
$$= R(s) + \gamma \mathbb{E}\left[R(s_1) + \gamma R(s_2) + \dots \mid \pi \right]$$

$$= R(s) + \gamma \sum_{s'} P_{s\pi(s)}(s') \, \mathbb{E}\left[R(s_1) + \gamma R(s_2) + \dots \mid s_1 = s', \pi \right]$$

$$\boxed{V^\pi(s) = R(s) + \gamma \sum_{s'} P_{s\pi(s)}(s') \, V^\pi(s')} \longrightarrow \text{Bellman equation}$$

Example :





$$V^\pi = I \cdot V^\pi = R + \gamma A V^\pi$$

$$\Leftrightarrow \boxed{V^\pi = (I - \gamma A)^{-1} R} \longrightarrow \text{Bellman equation}$$

$$V^\pi((3,1)) = R(3,1) + \gamma \left[0.8 \, V^\pi((3,2)) + 0.1 \, V^\pi((2,1)) + 0.1 \, V^\pi((4,1)) \right]$$

$$= -0.02 + 0.9 \left[0.8 \, (-0.82) + 0.1 \, (-0.87) + 0.1 \, (-1) \right]$$

$\simeq -0.7$ we improved the previous outcome (-0.85) by taking into account the future

Optimal value function

$$V^*(s) = \max_\pi V^\pi(s)$$

$$V^*(s) = \underbrace{R(s)}_{\text{present}} + \underbrace{\gamma \max_a \sum_{s'} P_{sa}(s') V^*(s')}_{\text{future}} \qquad \text{Bellman form}$$

Optimal policy function

$$\pi^*(s) = \arg\max_a \sum_{s'} P_{sa}(s') V^*(s')$$

Note : argmax is the index of max . $z = \begin{bmatrix} 2 \\ 3 \\ 1 \end{bmatrix}$ (0, 1, 2) $\max z = 3$ $\arg\max z = 1$

→ Goal of Reinforcement Learning !

Value iteration

Initialize $V(s) = 0 \quad \forall s$
Repeat until convergence { → episodes
  for every s {
    $V(s) = R(s) + \gamma \max_a \sum_{s'} P_{sa}(s') V(s')$ → Bellman
  }
}

$\Rightarrow V \longrightarrow V^* \Rightarrow \pi^*(s) = \arg\max_a \sum_{s'} P_{sa}(s') V(s')$

Policy iteration

Initialize π randomly $\forall s$
Repeat until convergence { → episodes
  $V = V^\pi = (I - \gamma A)^{-1} R$
  for every s {
    $\pi(s) = \arg\max_a \sum_{s'} P_{sa}(s') V(s')$ → Bellman
  }
}

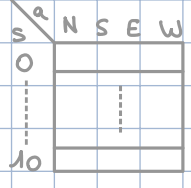
$\Rightarrow V \longrightarrow V^*$, $\pi \longrightarrow \pi^*$

Q-Learning

Q (Quality) ≙ state-action value function

Initialize $Q(s, a) = 0 \quad \forall s, \forall a$
Repeat until convergence { → episodes
  Choose s_0 randomly
  Repeat until stop condition is satisfied { } choose action
    m = rand()
    10% if ($m < \underbrace{\varepsilon}_{=0.1}$):
      choose a_t randomly → Exploration
    90% else :
      $a_t = \arg\max_a Q(s_t, a)$ → Exploitation
    Obtain $(s_{t+1}, R(s_t))$ from (s_t, a_t)
    $Q(s_t, a_t) = (1-\alpha) Q(s_t, a_t) + \alpha \left[R(s_t) + \gamma \max_a Q(s_{t+1}, a) \right]$
    $s_t = s_{t+1}$
  }
}

$\Rightarrow Q \longrightarrow Q^* \Rightarrow \pi^*(s) = \arg\max_a Q^*(s, a)$



En espérant que ce cours vous sera utile. Bon courage pour le prelab et le lab, plus que deux semaine avant la fin des cours !

GK