Q-Learining

Enrique Alifio Ditya

What is Reinforcement Learning?

Reinforcement Learning is a branch of machine learning that deals with an agent interacting with an environment. The agent takes actions in the environment, and the environment returns a reward and a new state. The agent's goal is to maximize the total reward it receives over time. It follows a similar pattern like how you would teach a dog to do tricks. You give the dog a treat when it does something good, and you punish it when it does something bad. The dog's goal is to maximize the number of treats it receives over time.

What is Q-Learning and how does it work?

Q-Learning is a model-free Reinforcement Learning policy that will find the best action given a current state of the agent. It's considered model-free because the agent doesn't need to know the environment's transition probabilities. It only needs to know the current state, the action it wants to take, the reward it will receive, and the next state it will end up in. The agent will use this information to update its Q-Table, which is a table of Q-Values for each state-action pair. The Q-Value is the expected future reward the agent will receive if it takes that action in that state. The agent will use the Q-Table to decide which action to take in a given state. The agent will continue to update its Q-Table until it has found the optimal policy for the environment. The Q-Table is updated with the Bellman equation, which is a recursive equation that calculates the Q-Value for a given state-action pair. The Bellman equation is as follows:

$$Q(s,a) = Q(s,a) + \alpha(r + \gamma \max_{a'} Q(s',a') - Q(s,a))$$

where s is the current state, a is the action taken, r is the reward received, s' is the next state, α is the learning rate, and γ is the discount factor. The discount factor is used to weigh the importance of future rewards. The learning rate is used to weigh the importance of new information. Algorithmically, it works as follows:

1. Initialize the Q-Table with random values.
2. Observe the current state.
3. Choose an action to take based on the current state using the Q-Table.
4. Take the action and observe the reward and the next state.
5. Update the Q-Table using the Bellman Equation.
6. Repeat steps 2-5 until the Q-Table converges.

Q-Learning vs SARSA

SARSA is another model-free Reinforcement Learning policy. It's very similar to Q-Learning, but it uses a different update rule. The update rule for SARSA is as follows:

$$Q(s,a) = Q(s,a) + \alpha(r + \gamma Q(s',a') - Q(s,a))$$

where s is the current state, a is the action taken, r is the reward received, s' is the next state, α is the learning rate, and γ is the discount factor. The difference between the two update rules is that SARSA uses the Q-Value of the next state-action pair to update the current state-action pair, whereas Q-Learning uses the maximum Q-Value of the next state-action pair to update the current state-action pair. This subtle difference results in SARSA being

an on-policy algorithm, and Q-Learning being an off-policy algorithm.

On-policy means that the agent learns by following and improving the same policy that it is using to interact with the environment. This means that the agent's experience directly impacts the policy it is trying to optimize. The agent on off-policy algorithm, on the other hand, learns by observing and interacting with the environment using one policy (behavior policy) while simultaneously learning and improving a different policy (target policy). The behavior policy is responsible for selecting actions and exploring the environment, while the target policy is the policy the agent is trying to optimize and improve. This means that SARSA is more conservative than Q-Learning, because it will always choose the action that it thinks is best, whereas Q-Learning will sometimes choose an action that it thinks is worse.

Advantages of Q-Learning

- Q-Learning is an off-policy algorithm, which means it can learn from data generated by any behavior policy. This flexibility allows for more efficient exploration and faster convergence to the optimal policy.
- Under certain conditions, Q-Learning is guaranteed to converge to the optimal action-value function (Q-function), leading to the optimal policy.
- Q-Learning is more aggressive than SARSA, which means that it will explore the environment more than SARSA. This means that Q-Learning will find the optimal policy faster than SARSA.

Advantages of SARSA

- SARSA is an on-policy algorithm, meaning it learns while following the same policy it is trying to improve. This means that SARSA is more conservative than Q-Learning, and will not explore the environment as much as Q-Learning. This can be advantageous in situations where exploration is costly, such as in robotics.
- SARSA is less prone to overestimating Q-values compared to Q-Learning, which can lead to better performance in some cases.
- SARSA is more stable than Q-Learning, because it uses the Q-Value of the next state-action pair to update the current state-action pair. This means that SARSA is less likely to diverge than Q-Learning.