Markov Decision Process (MDP)

MDP is a mathematical framework to describe an environment in reinforcement learning.

A state is a status the agent (decision-maker) can hold. In the dice game, the agent can either be in the game or out of the game.

An action is a movement the agent can choose. These move the agent between states, with certain penalties or rewards.

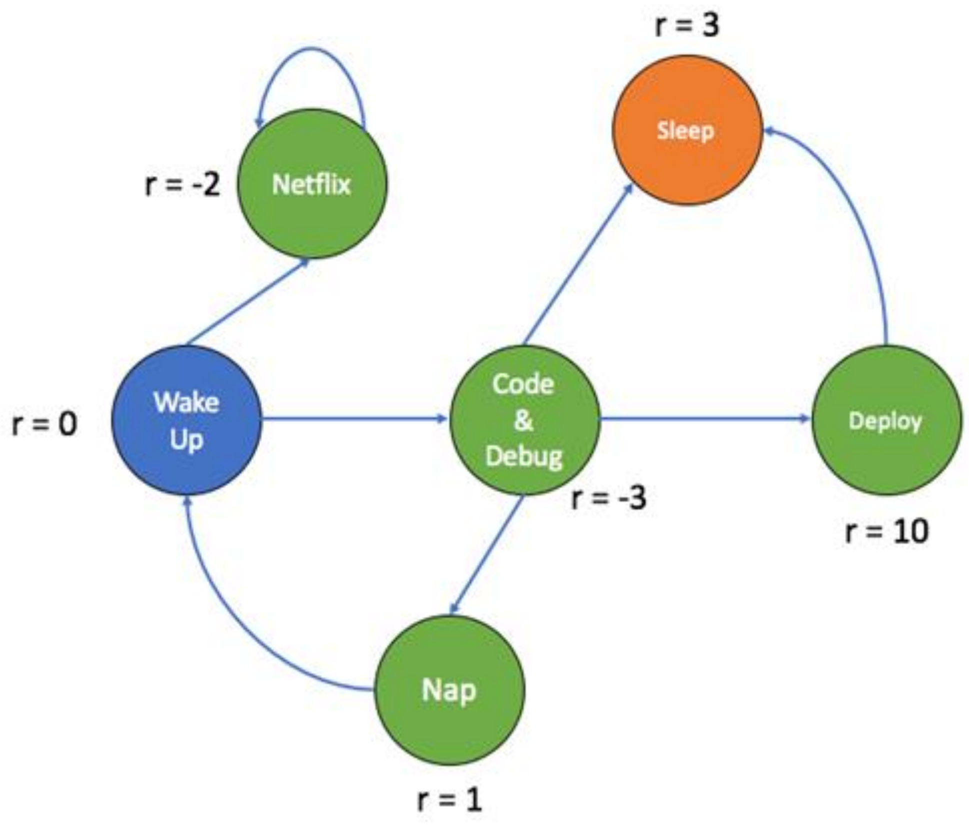
Transition probabilities describe the probability of ending up in a state s’ (s prime) given an action a. These will be often denoted as a function P(s, a, s’) that outputs the probability of ending up in s’ given current state s and action a.

For example, P(s=playing the game, a=choose to continue playing, s’=not playing the game) is ⅓, since there is a two-sixths (one-third) chance of losing the dice roll.

Rewards are given depending on the action. The reward for continuing the game is 3, whereas the reward for quitting is $5. The ‘overall’ reward is to be optimized.

We can formally describe a Markov Decision Process as m = (S, A, P, R, gamma), where:

- S represents the set of all states.
- A represents the set of possible actions.
- P represents the transition probabilities.
- R represents the rewards.
- Gamma is known as the discount factor. More on this later.



The goal of the MDP m is to find a policy, often denoted pi, that yields the optimal long-term reward. Policies are simply a mapping of each state s to a distribution of actions a. For each state s, the agent should take action a with a certain probability. Alternatively, policies can also be deterministic (i.e. the agent will take action a in state s).

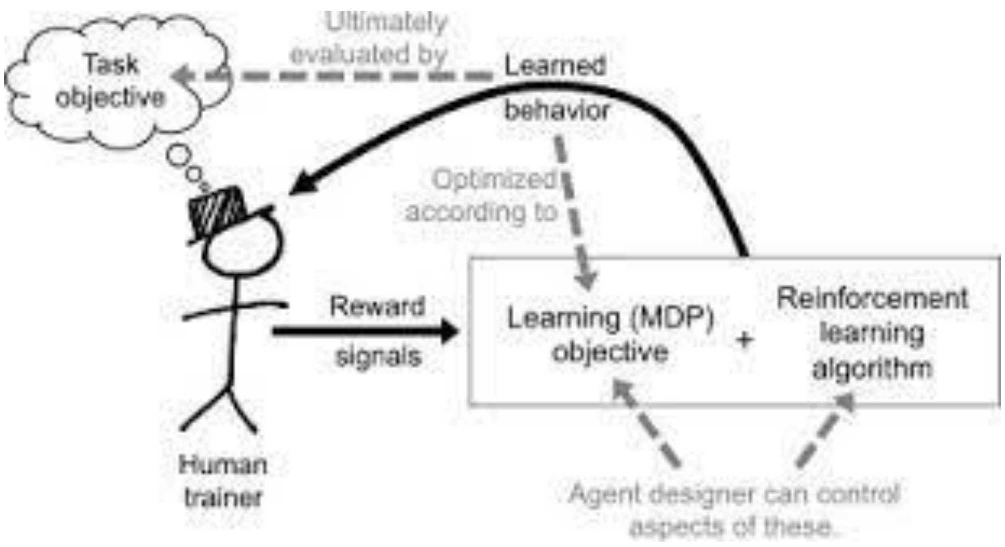
What is q-learning?

Q-learning is an off policy reinforcement learning algorithm that seeks to find the best action to take given the current state

It's considered off-policy because the q-learning function learns from actions that are outside the current policy, like taking random actions, and therefore a policy isn't needed. More specifically, q-learning seeks to learn a policy that maximizes the total reward.

The 'q' in q-learning stands for quality. Quality in this case represents how useful a given action is in gaining some future reward. So then here we will see q learing reinforcement learning with reward function in these code we will import numpy library as ql

- ql is an nxp matrix from A for QL decomposition
- A is list of two matrices Q and L so that A = QL
- Q - nxp matrix with orthonormal columns with the same span as A
- L - is a lower triangular pxp matrix with nonnegative diagonal entries
- For MDP we will use two variable R as reward and Q as learininf Matrix
- R - is reward matrix for each state
- Q - is the Learning Matrix in which rewards will be learned/stored


Task objective
Ultimately evaluated by
Learned behavior
Optimized according to
Reward signals
Learning (MDP) objective
+
Reinforcement learning algorithm
Human trainer
Agent designer can control aspects of these