AI-ML PROJECT

## Q-LEARNING CABS

PROFESSOR: DR. RAVI KUMAR JATOTH

TEAM NAME: THE DUO

TEAM MEMBERS:

SAI TEJA PAITHARI-21ECB0B50

N.SRI DARSHAN-21ECB0B38

Introduction:

Q-Learning cabs are basically the self-driving cabs. The major goal is to demonstrate, in a simplified environment, how you can use Reinforcement learning (RL) techniques to develop an efficient and safe approach for tackling this problem.

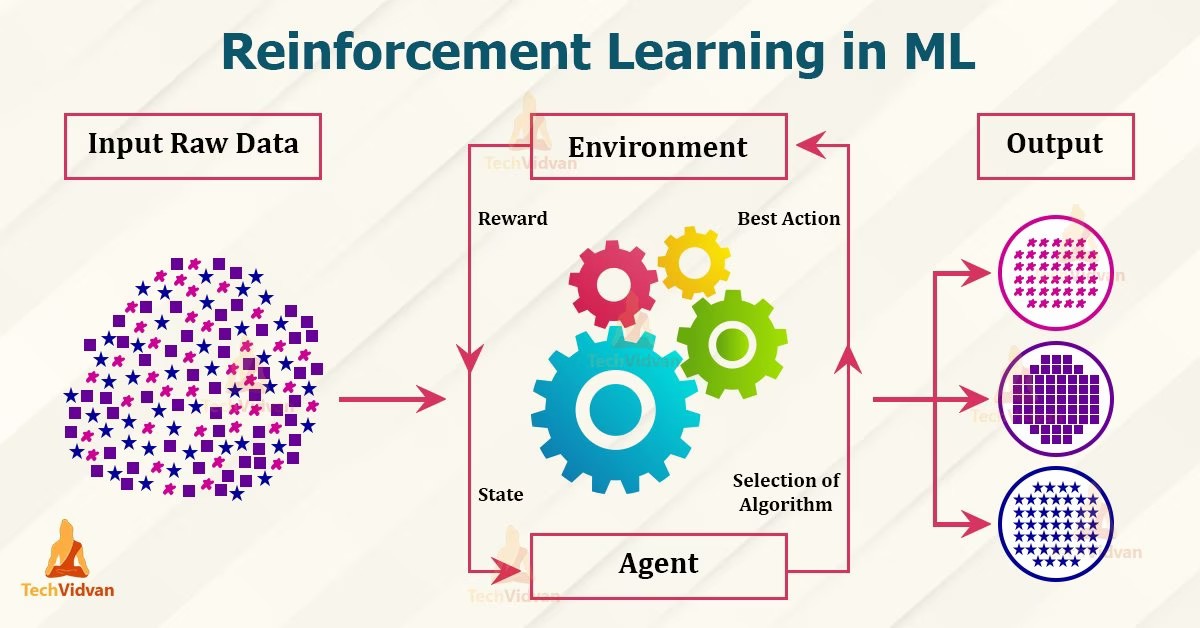
The Q learning cab’s job is to pick up the passenger at one location and drop them off in another. Here are a few things that we'd love our cab to take care of:

* Drop off the passenger to the right location.
* Save passenger's time by taking minimum time possible to drop off.
* Take care of passenger's safety and traffic rules.

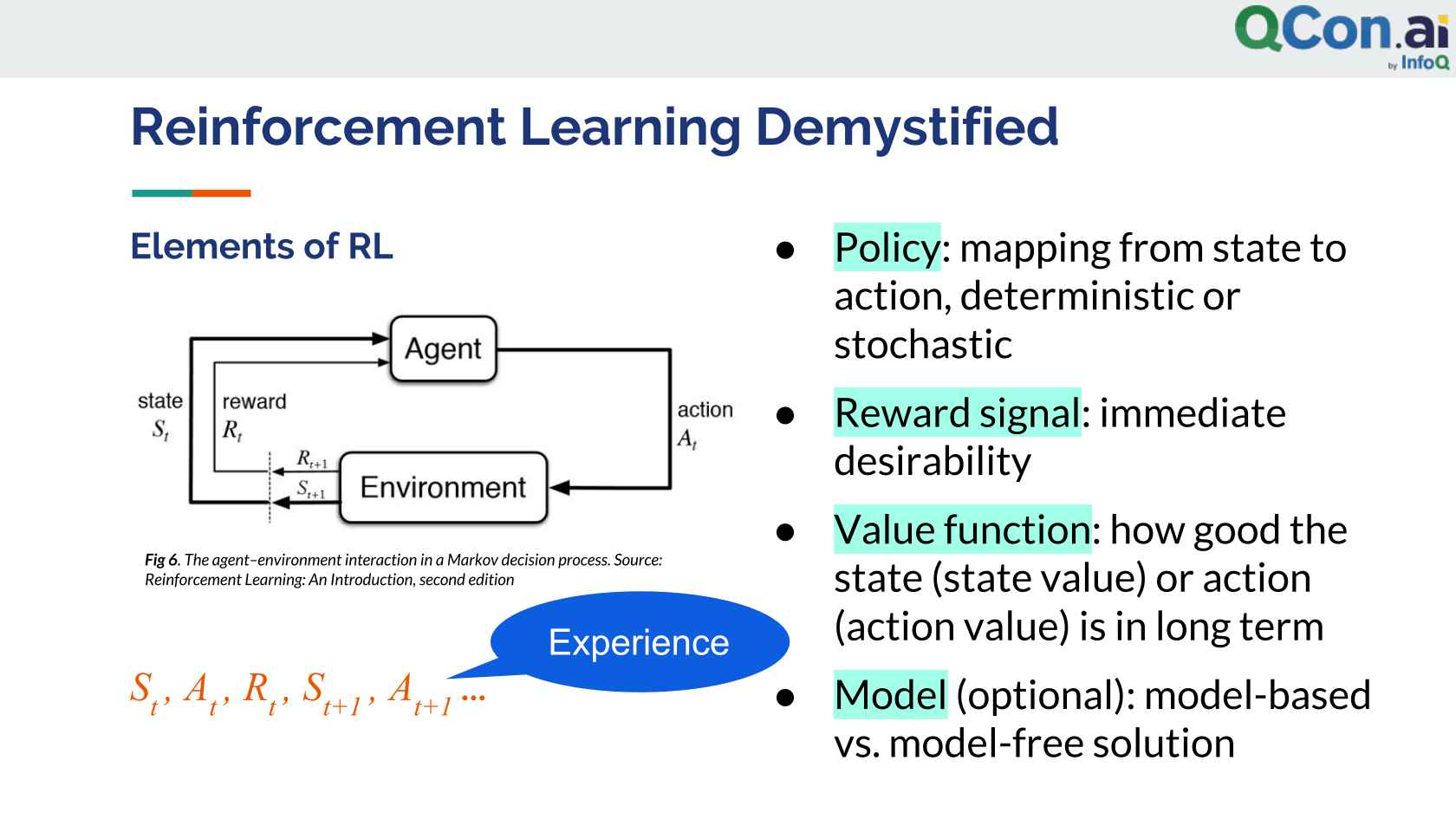
Background: (-concepts and algorithms)

This whole thing is based on the reinforcement type of machine learning specifically the q learning algorithm. Reinforcement Learning differs from a typical “input x, output y” supervised learning problems as**it involves an agent interacting with its surrounding environment to determine what is the best action to take.** The **environment could be uncertain, complex, and the agent’s behaviour can also be probabilistic, not deterministic.**

Q-Learning is a basic form of Reinforcement Learning which uses Q-values (also called action values) to iteratively improve the behaviour of the learning agent.

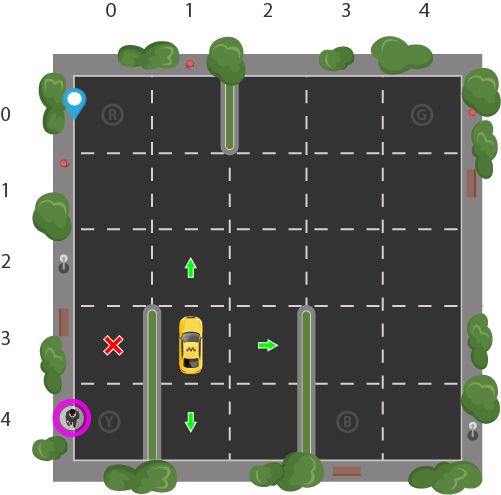


• Reinforcement learning systems have 4 main elements (as shown in the below figure):



**Problem specification:**

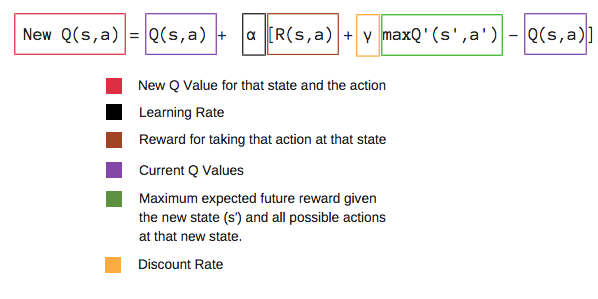
* Starting at a random state, our job is to get the taxi to the passenger’s location, pick up the passenger and drive to the destination, drop the customer, and then the episode ends.
* There are 4 designated locations in the grid indicated by **Red — 0 , Green — 1, Yellow — 2, and Blue — 3**, the blue letter correspond to pick up location and purple letter indicate the drop off location. The solid lines indicate walls that the taxi cannot pass, whereas the filled rectangle is the taxi, when it is yellow it is empty and when it is green it is carrying a passenger.



* State Space: We can see that our state space consists of 500 possible states, with 25 possible taxi positions, 5 possible locations of the passenger (including the case when the passenger is in the taxi), and 4 destination locations
* Action space: There are **6 discrete deterministic actions:** 0 — move south, 1 — move north, 2 — move east, 3 — move west, 4 — pickup passenger, 5 — drop off passenger
* Rewards: Except for delivering the passenger with gets a reward of +20, each extra step has a penalty of R=-1, executing “pickup” and “drop-off” actions illegally results in R=-10

Mathematical Approach:

The **Q-function** uses the Bellman equation and takes two inputs: state (**s**) and action (**a**).



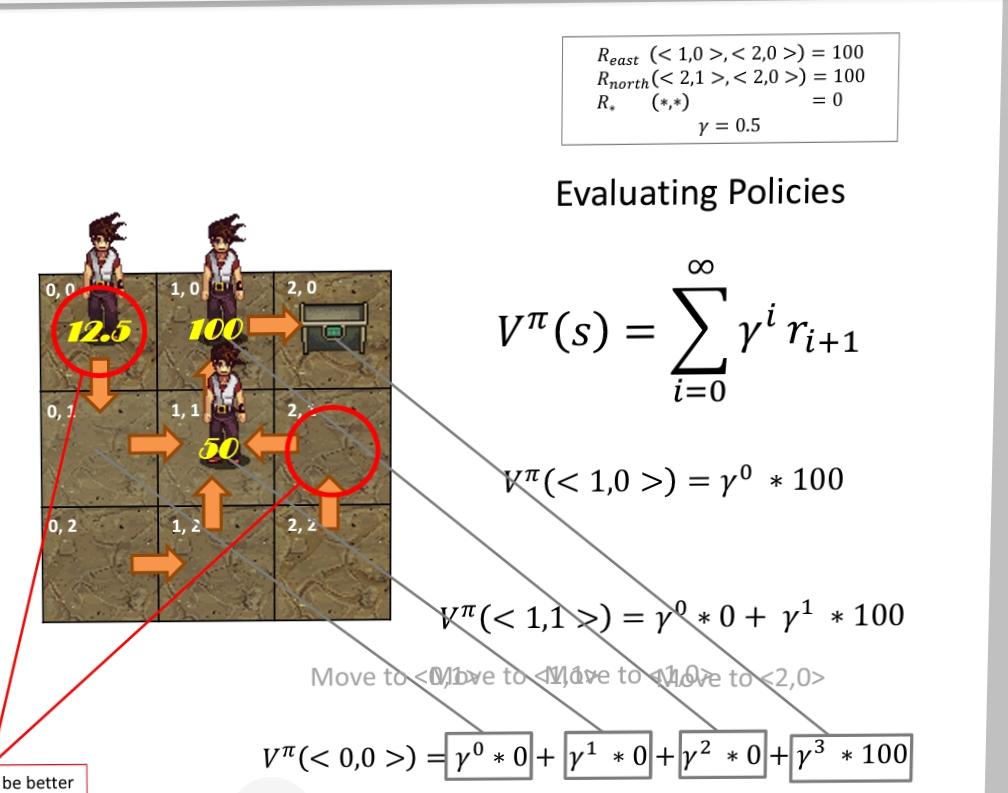
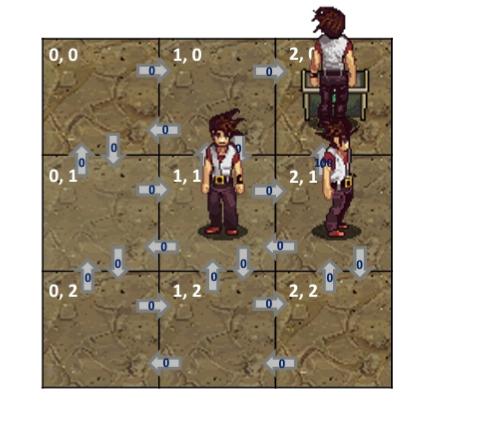
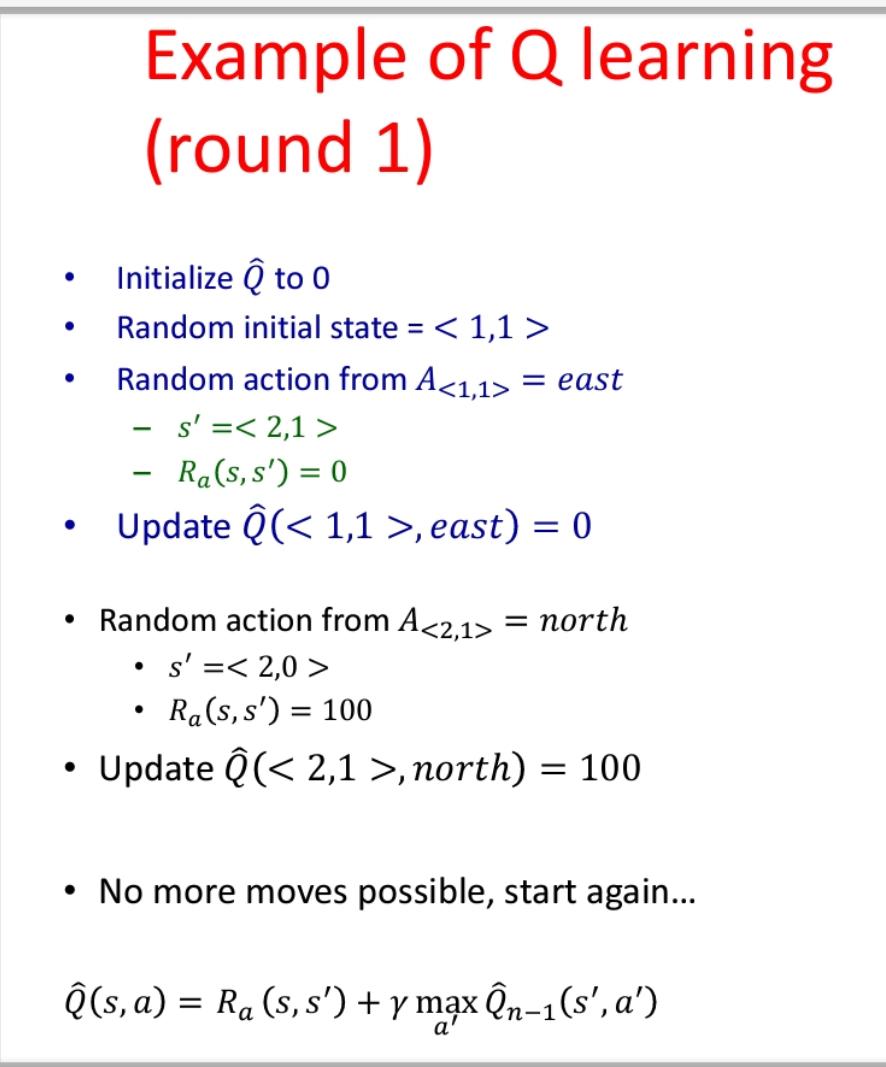
When we start, all the values in the Q-table are zeros. [A Q-table is a matrix of order (no. of states\*no. of actions) with the corresponding Q values filled in their boxes].

There is an iterative process of updating the values. As we start to explore the environment**,** the Q-function gives us better and better approximations by continuously updating the Q-values in the table.

Explore vs Exploit (Life long learning): Exploitation is defined as a greedy approach in which agents try to get more rewards by using estimated value but not the actual value. So, in this technique, agents make the best decision based on current information. Exploitation is defined as a greedy approach in which agents try to get more rewards by using estimated value but not the actual value. So, in this technique, agents make the best decision based on current information.

Exploration rate is denoted by epsilon.

Numerical and computational work:

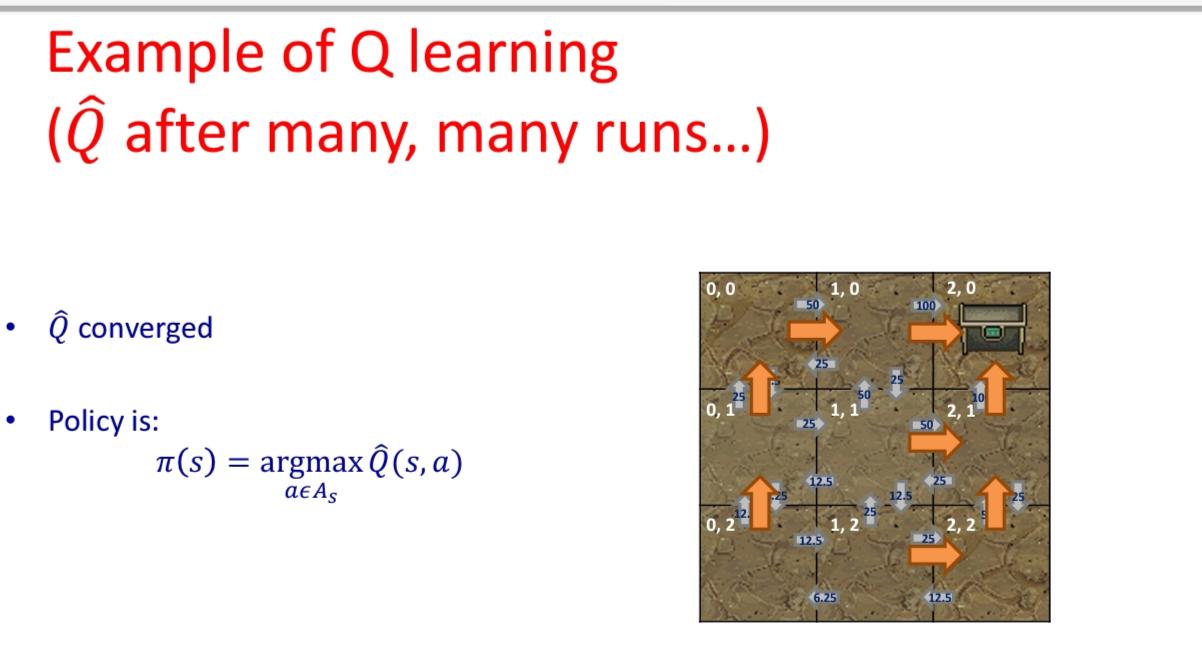
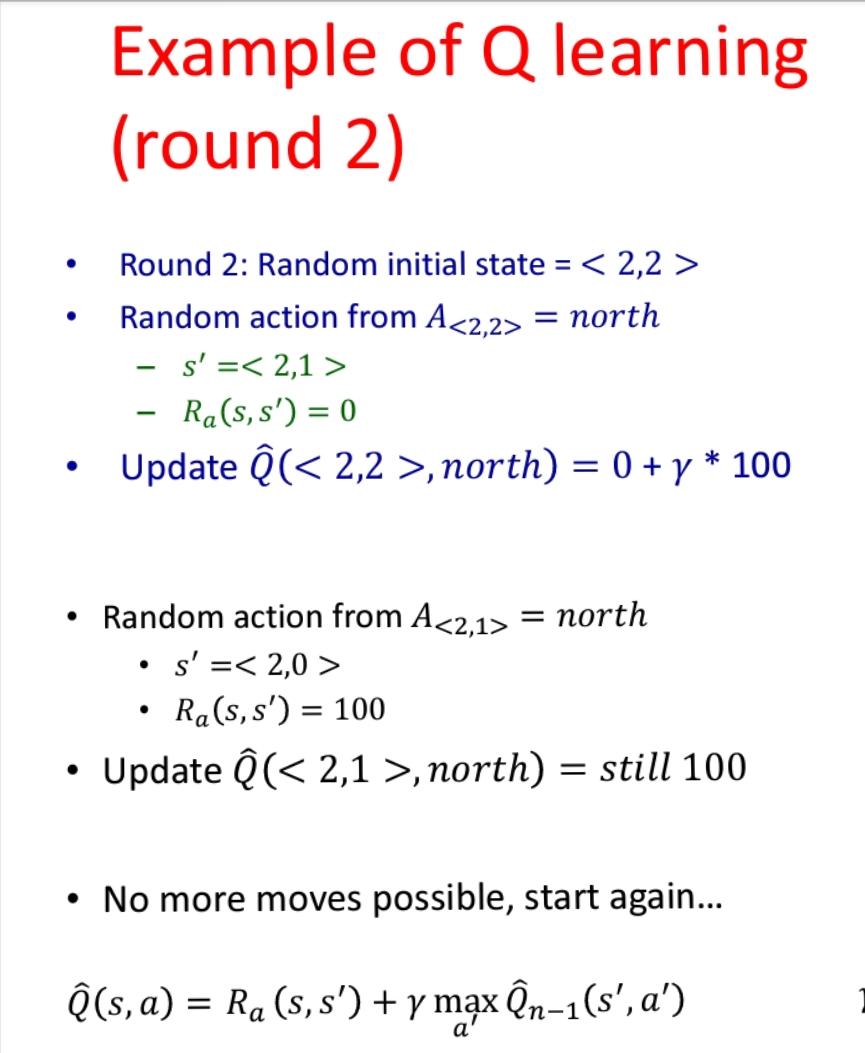
>>>>>>>  

|

|

|

\ /

<<<<<<<<  

Code:

<https://colab.research.google.com/drive/1RG8K7FknjynjoIKsf2a1Jsbiun0Z_Rjp#scrollTo=V1lGZpYCqYXu>

for detailed simulations and results please run the above code because the simulation is dynamic (in video format).

Conclusion:

We’ve completed modelling our first RL agent. It does not take complicated mathematics neither hard algorithms to understand the basics of RL. And with only several lines of code, we were able to train an agent to play the cab game.

References:

<https://drive.google.com/file/d/1rHD9ARBW4iB0iB28uxIQjK3dwIGYIzRR/view?usp=share_link>

<https://nitwarangal.webex.com/nitwarangal/ldr.php?RCID=49eebd315016a4a510205d0d3694b7f7> (Password: MLDSp123)

<https://towardsdatascience.com/reinforcement-learning-teach-a-taxi-cab-to-drive-around-with-q-learning-9913e611028f>