An introduction to Reinforcement Learning

26th of July 2022

Summary

What is reinforcement learning (RL)?

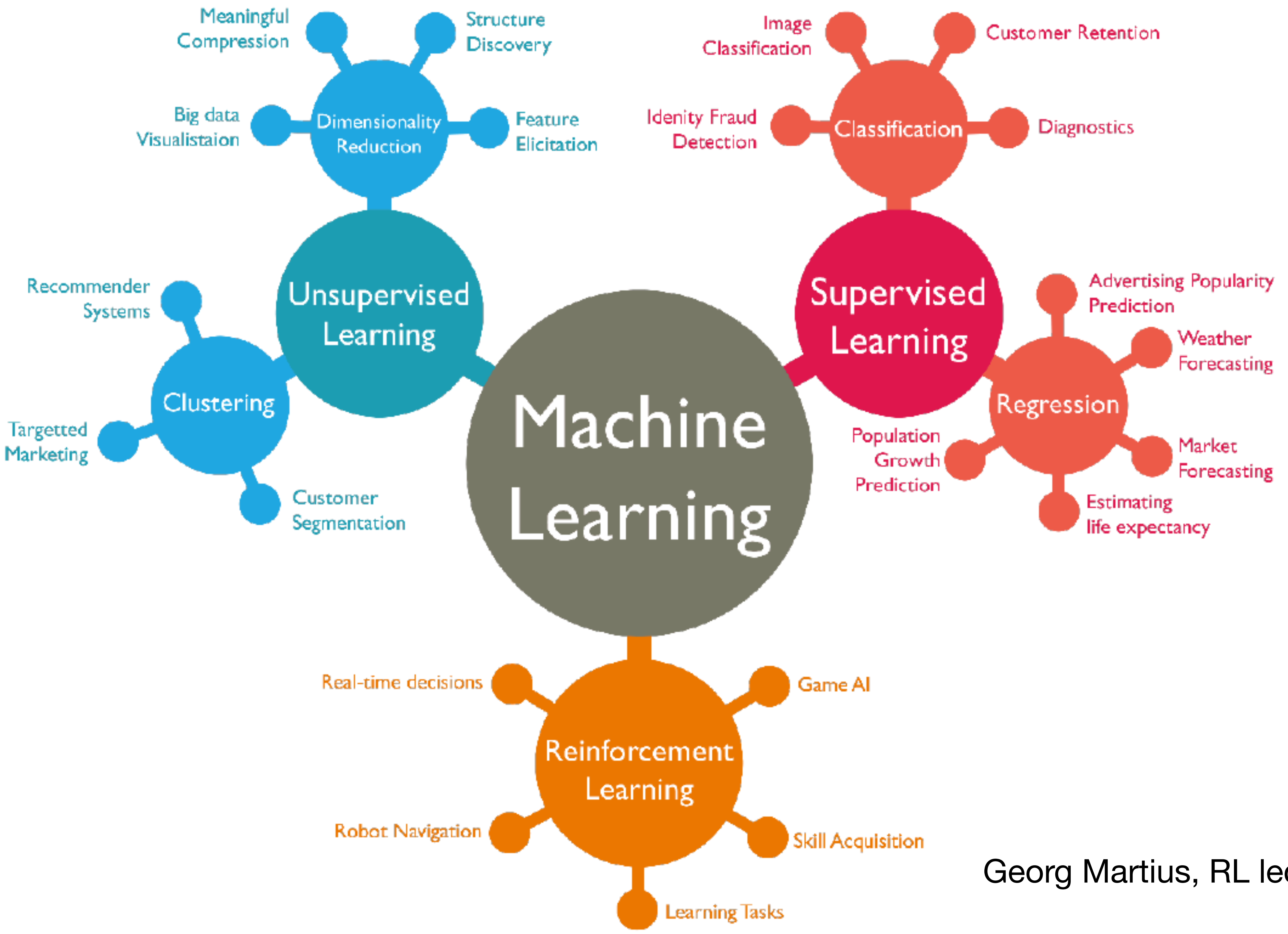
- RL is a **computational approach** to learning from **interactions** with the **environment**
  - Trial-and-error
  - Delayed reward
- Considers whole problem of **goal-directed** agent interacting with an **uncertain** environment
- RL agents
  - Have explicit goals
  - Sense aspects of their environments
  - Choose actions to influence their environments
- Very general

How does RL compare to other types of learning?

- Association Learning
- Representation Learning
- Supervised Learning
- Unsupervised Learning
- Imitation Learning
- …

- Reinforcement Learning

The Machine Learning view:



Georg Martius, RL lecture 2020

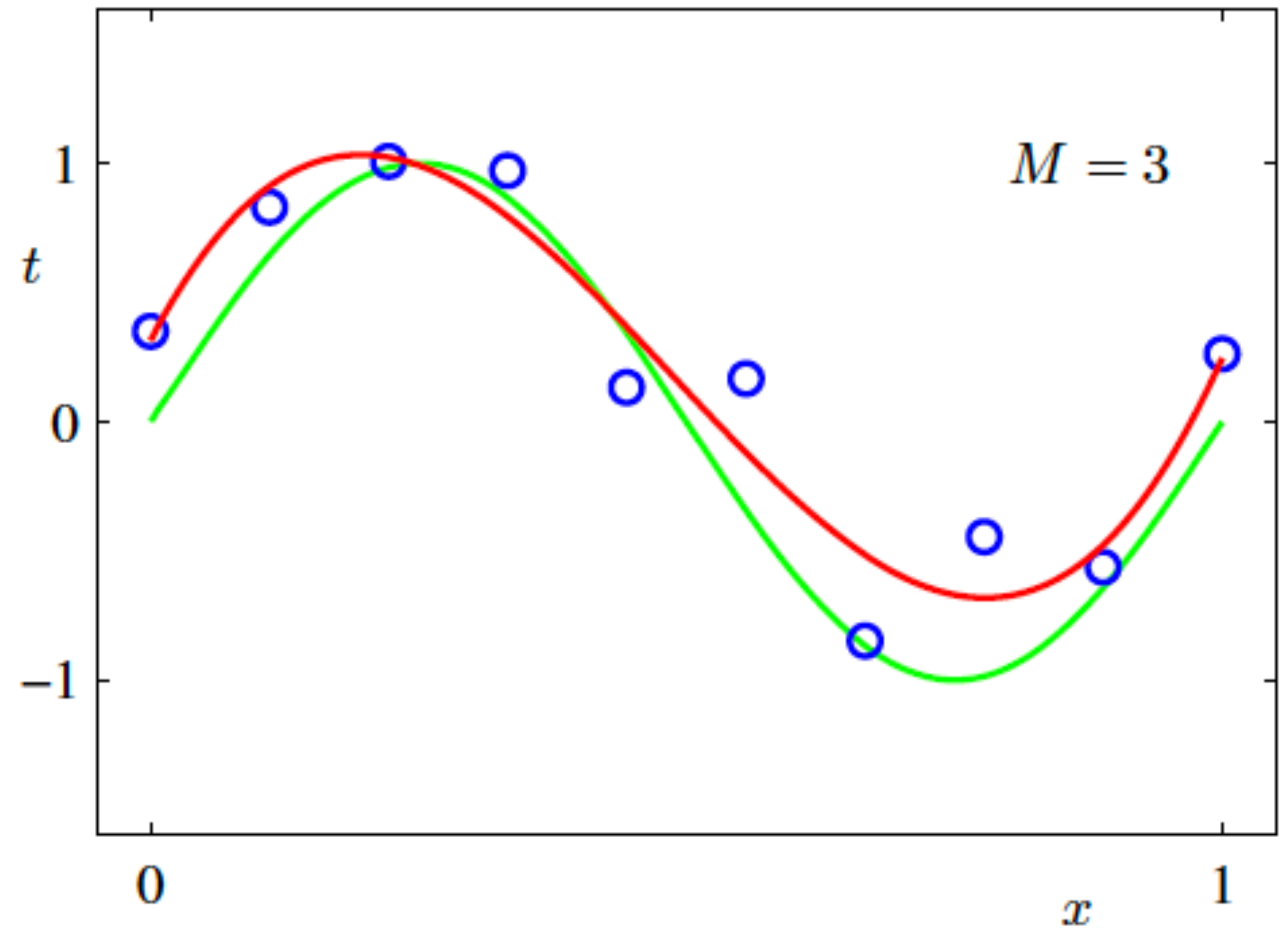
Types of (machine) learning: supervised learning

- Find correct labelling/prediction of data:

Classification

MNIST

Regression



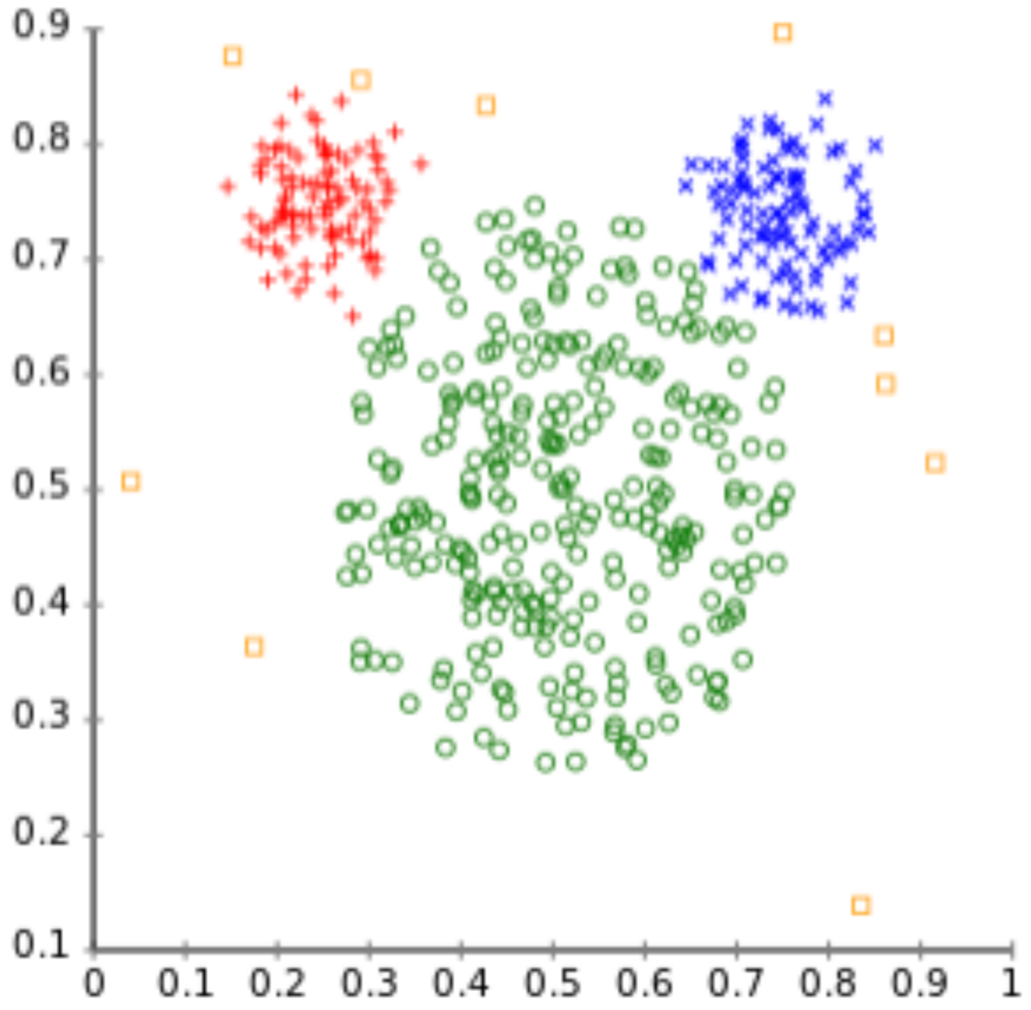
Bishop 2006

- That's not what we want though:
    - Want to learn from *own experience* by *interacting* with the world

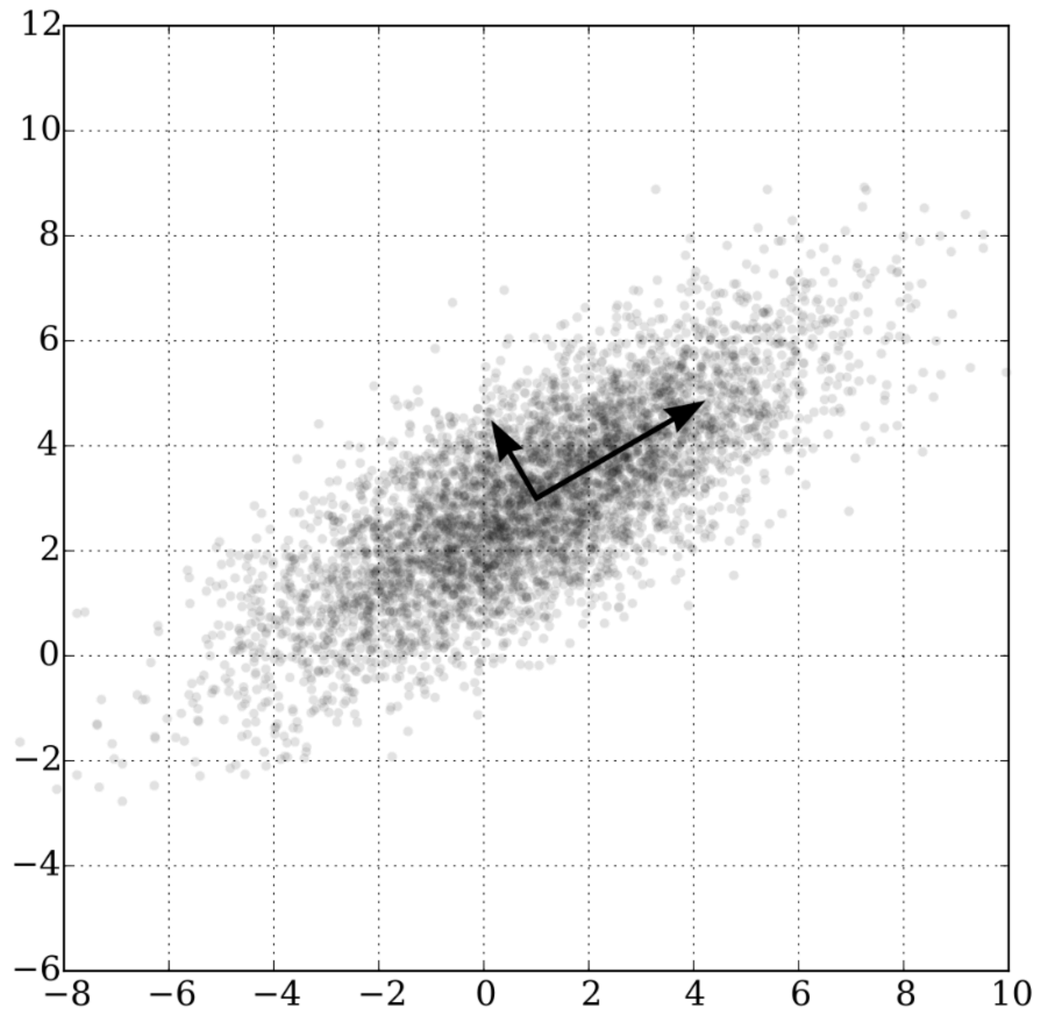
Types of (machine) learning: unsupervised learning

- Find structure in data:

Clustering



Dimensionality Reduction



- That’s also not what we want:
  - Don’t (necessarily) want to learn hidden structure, rather: maximise reward

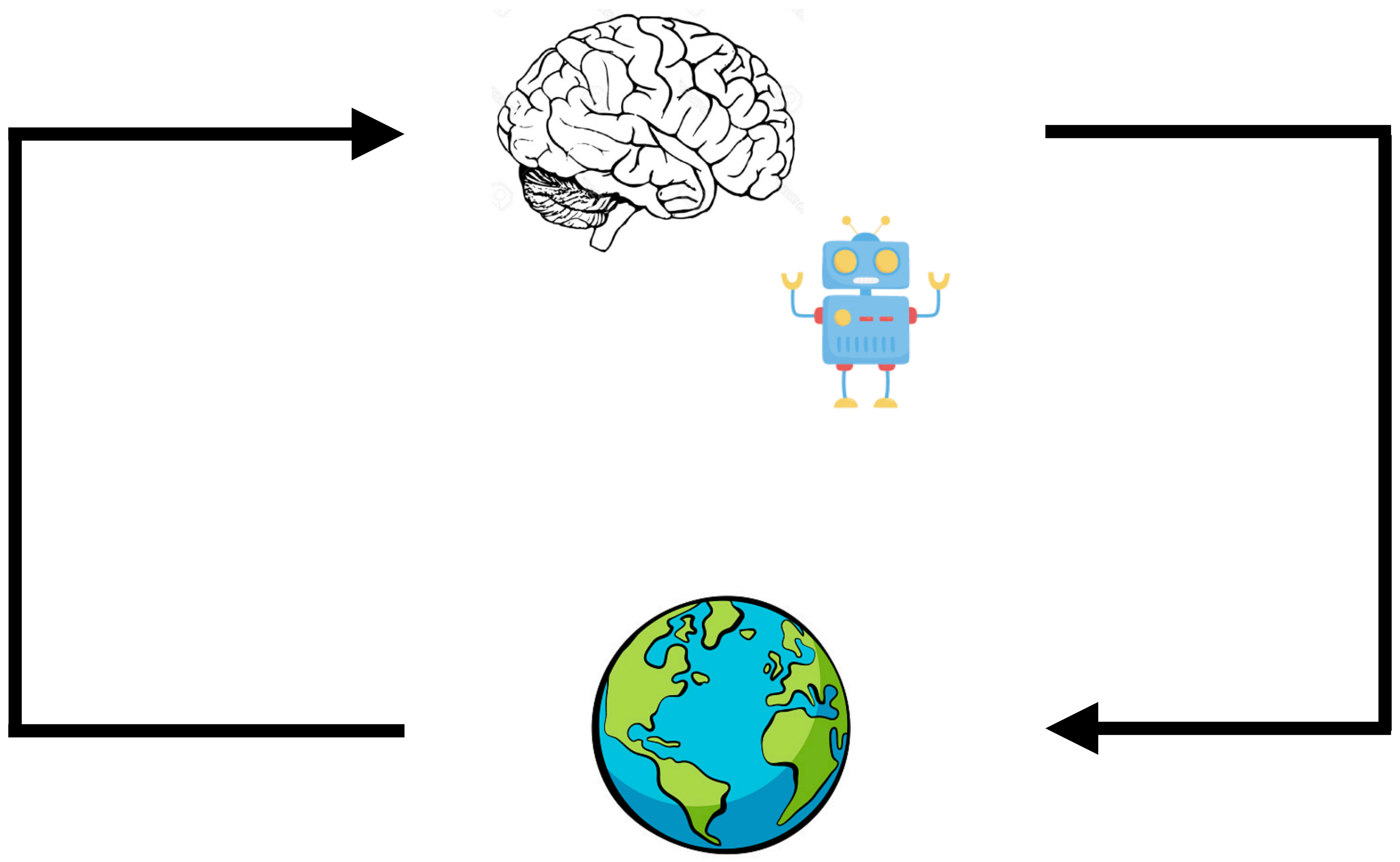
Basic setup: how to agents learn to act?

Based on a reward signal, agents learn **values of actions/states:**

$$V_\pi(s) = \mathbb{E}_\pi[R \,|\, s_0 = s]$$

Action is governed by a **policy**:

$$\pi(a, s) = P(a_t = a \,|\, s_t = s)$$



Reward r_t

Action a_t

State s_t

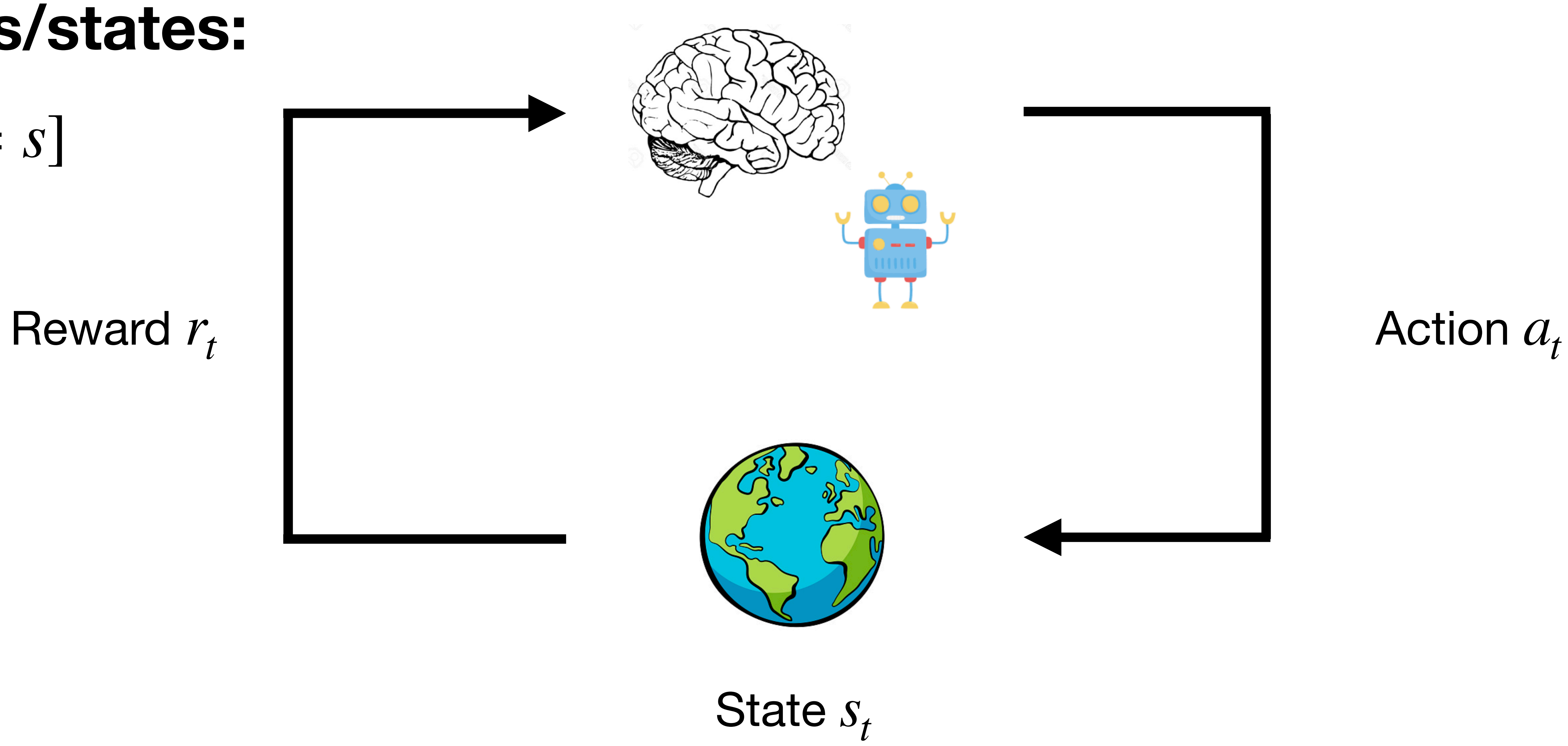
Agents can learn a **model of the environment** to make smarter decisions, e.g.:

$$P(s_{t+1} = s \,|\, s_t = s, a_t = a)$$

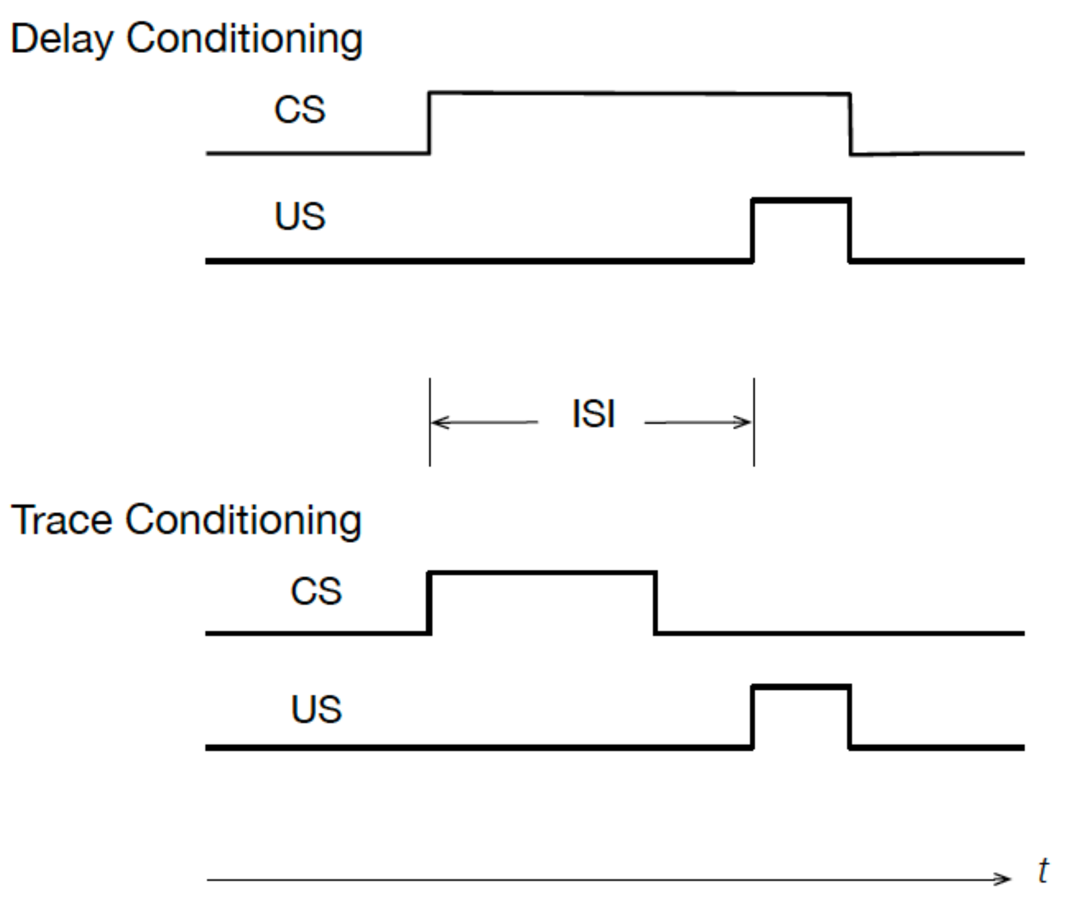
Basic setup: how to agents learn to act?

Based on a reward signal, agents learn **values of actions/states:**

$$V_\pi(s) = \mathbb{E}_\pi[R \mid s_0 = s]$$

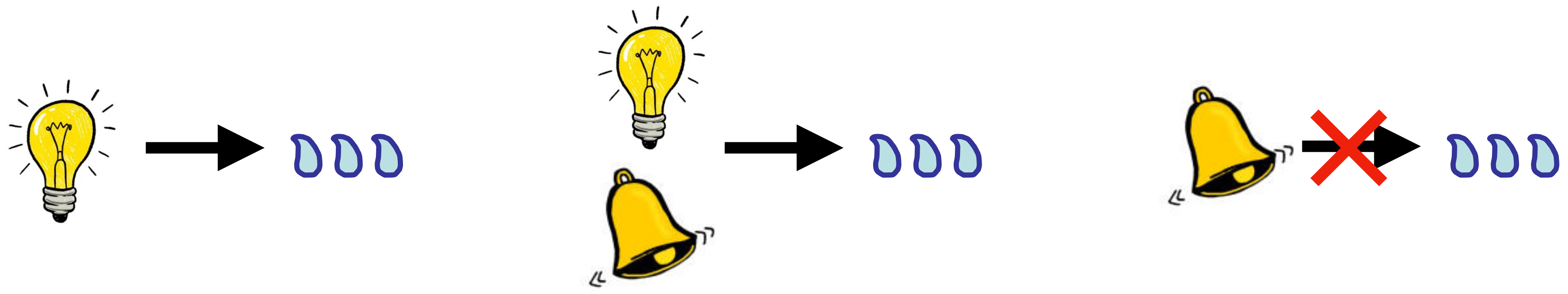


Learning to predict reward



- Two simple learning algorithms:

  - **Rescorla-Wagner Learning** E.g.: blocking

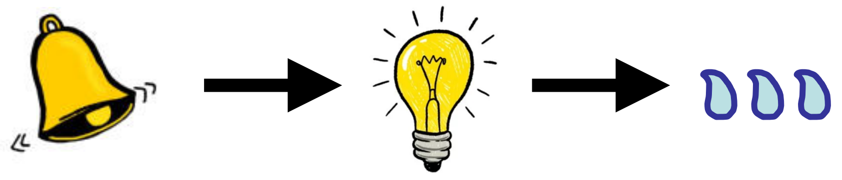


Prediction error

$$V(s_t) \leftarrow V(s_t) + \alpha \cdot (r - V(s_t))$$

Learning rate

  - **Temporal-Difference Learning** E.g.: Higher-order conditioning



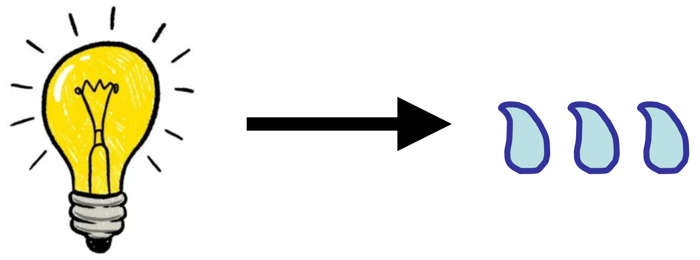
$$V(s_t) \leftarrow V(s_t) + \alpha \cdot (r + \gamma \cdot V(s_{t+1}) - V(s_t))$$

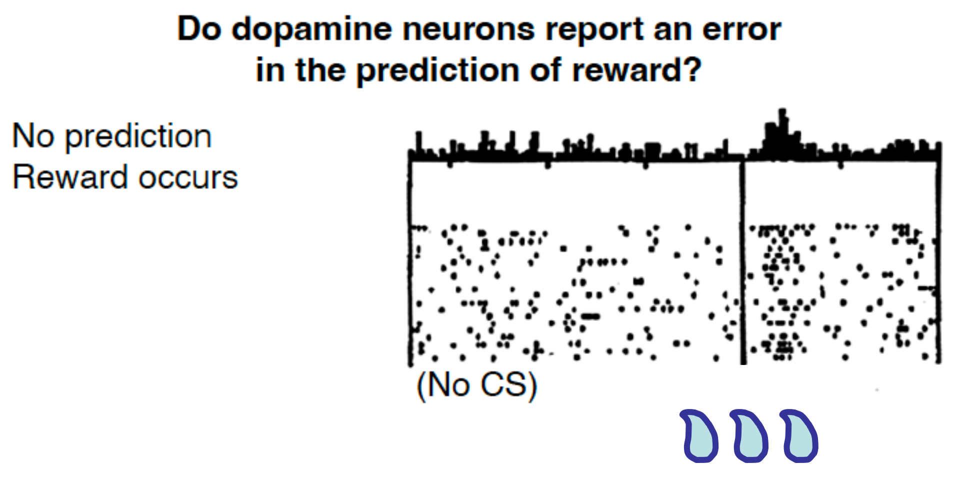
Recap: Temporal Difference Learning

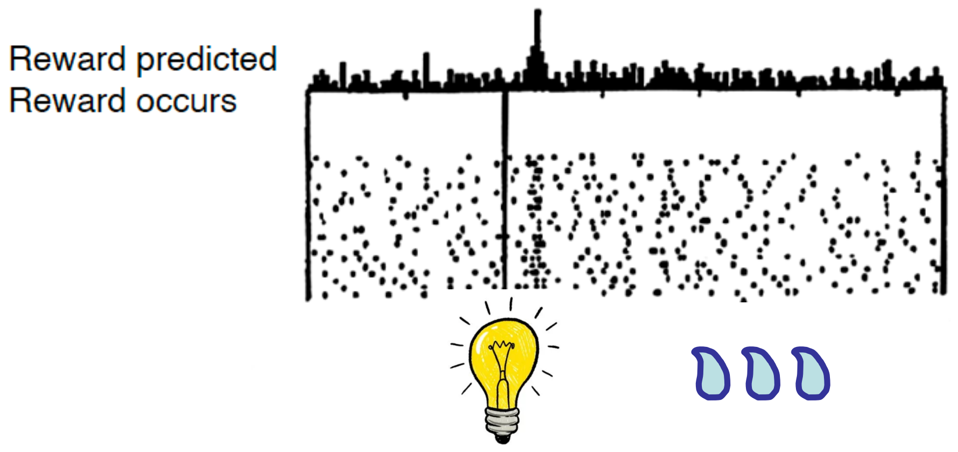
Prediction error

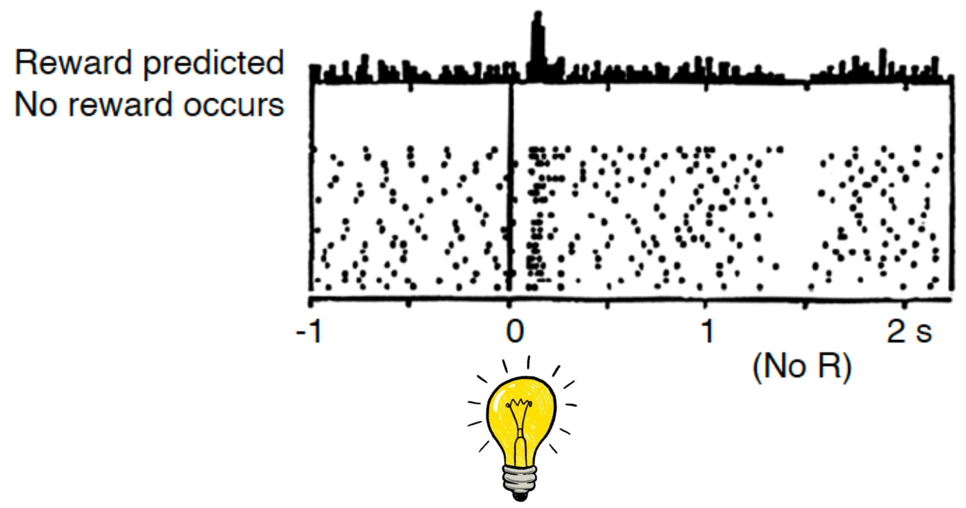
TD Learning: $V(s_t) \leftarrow V(s_t) + \alpha \cdot (r + \gamma \cdot V(s_{t+1}) - V(s_t))$

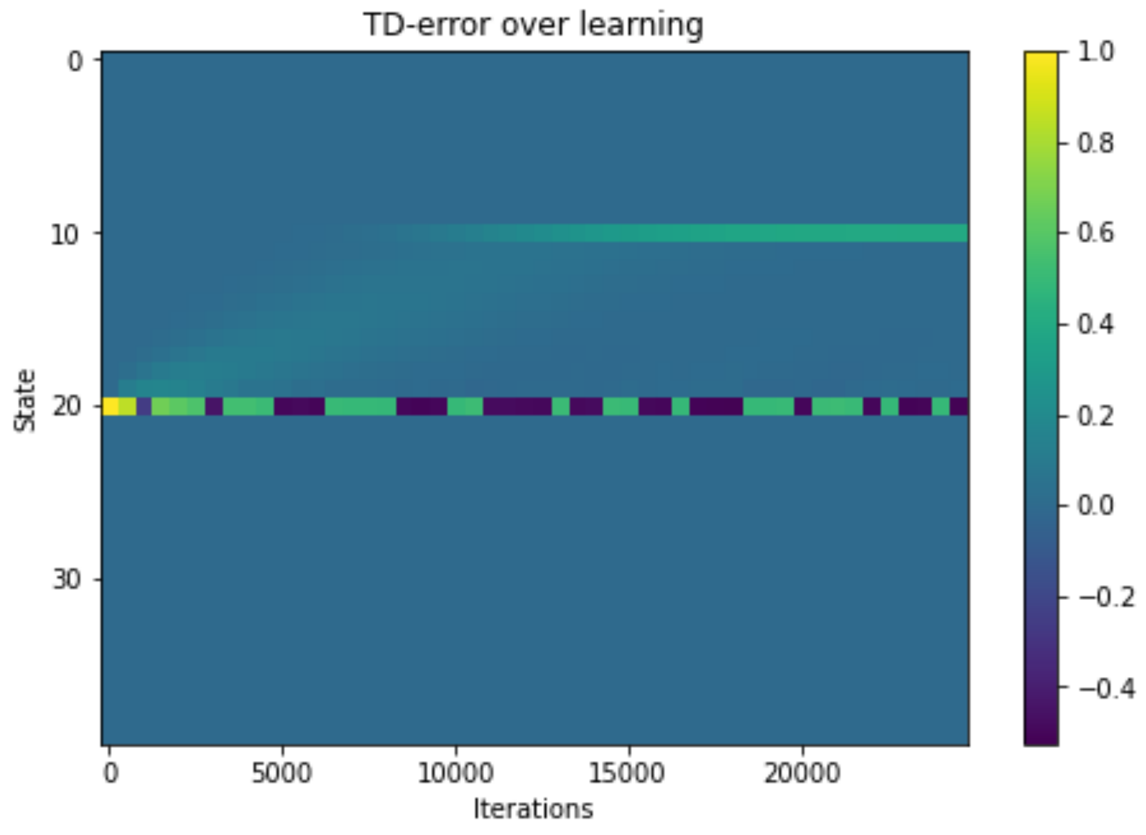
Learning rate   Discount rate









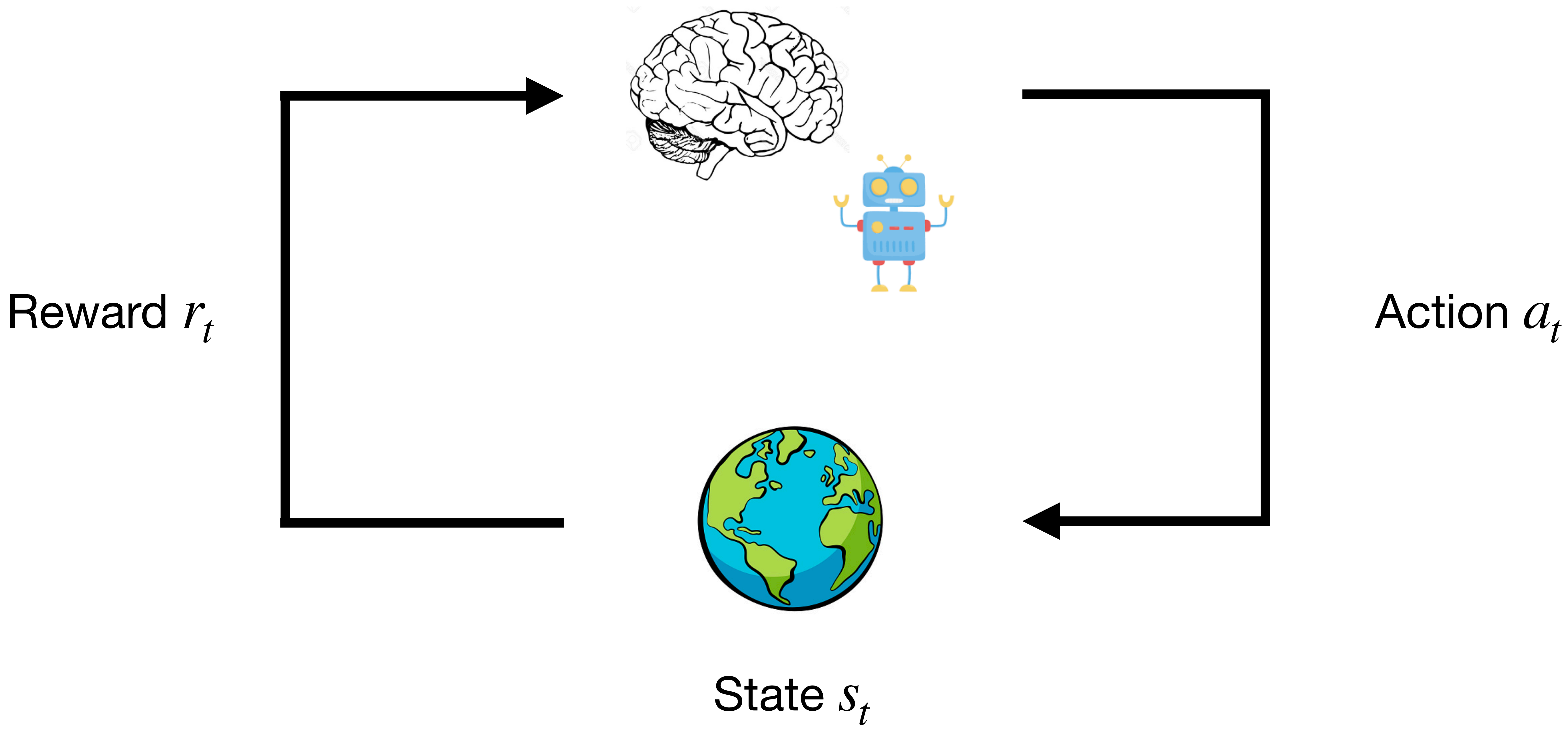


But what about actions?

Basic setup: how to agents learn to act?

Action is governed by a **policy**:

$$\pi(a, s) = P(a_t = a \,|\, s_t = s)$$



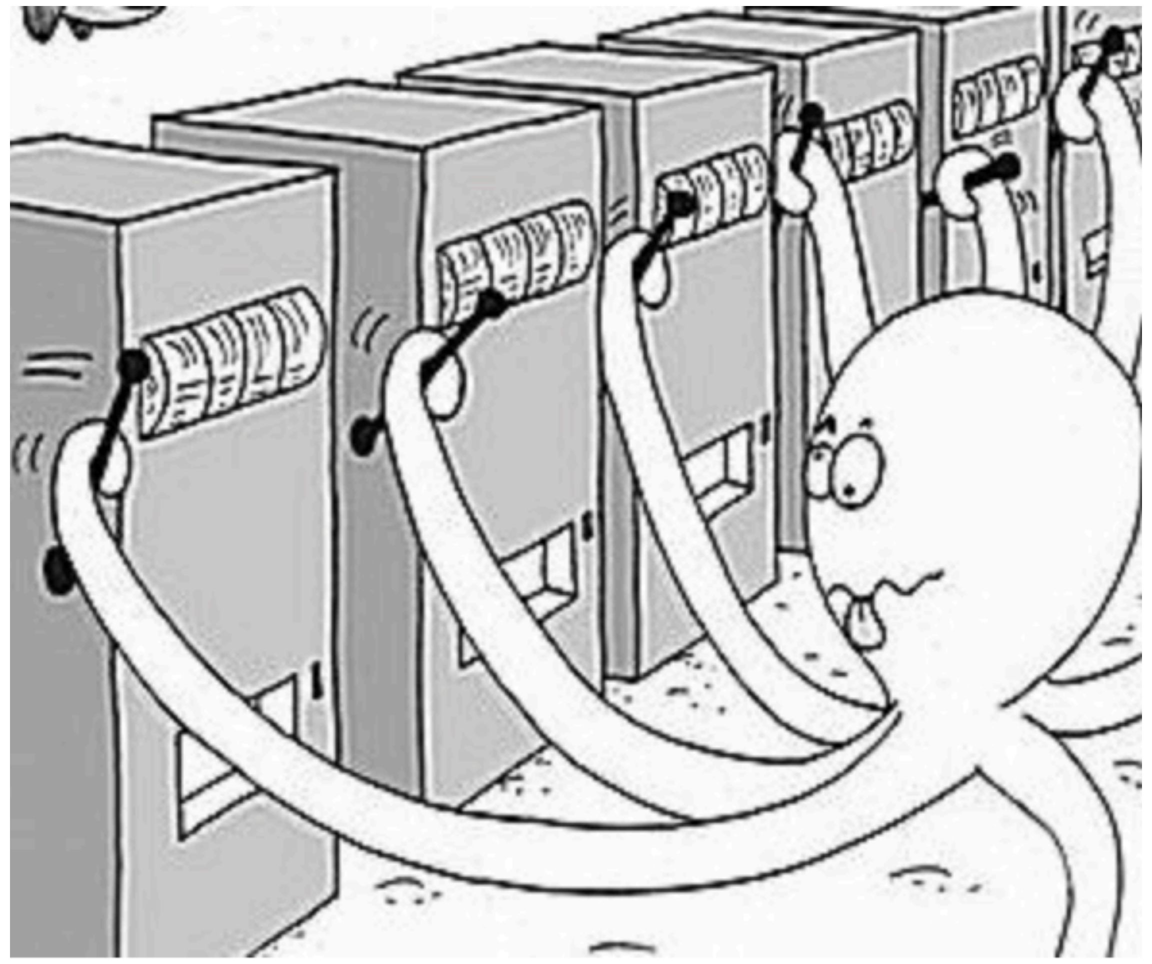
Multi-armed bandits

Greedy action selection:

$$P(a_t = a) = \begin{cases} 1 & \text{if } a_t = \text{argmax}_a \ V_t(a) \\ 0 & \text{otherwise} \end{cases}$$

Epsilon-greedy action selection:

$$P(a_t = a) = \begin{cases} 1 - \epsilon & \text{if } a_t = \text{argmax}_a \ V_t(a) \\ \epsilon / N & \text{otherwise} \end{cases}$$



Action is governed by a **policy**:

$$\pi(a, s) = P(a_t = a \,|\, s_t = s)$$

Softmax action selection:

$$P(a_t = a) = \frac{e^{V_t(a) \cdot \beta}}{\sum_{i=1}^{N} e^{V_t(a_i) \cdot \beta}}$$

Upper-confidence-bound (UCB) action selection:

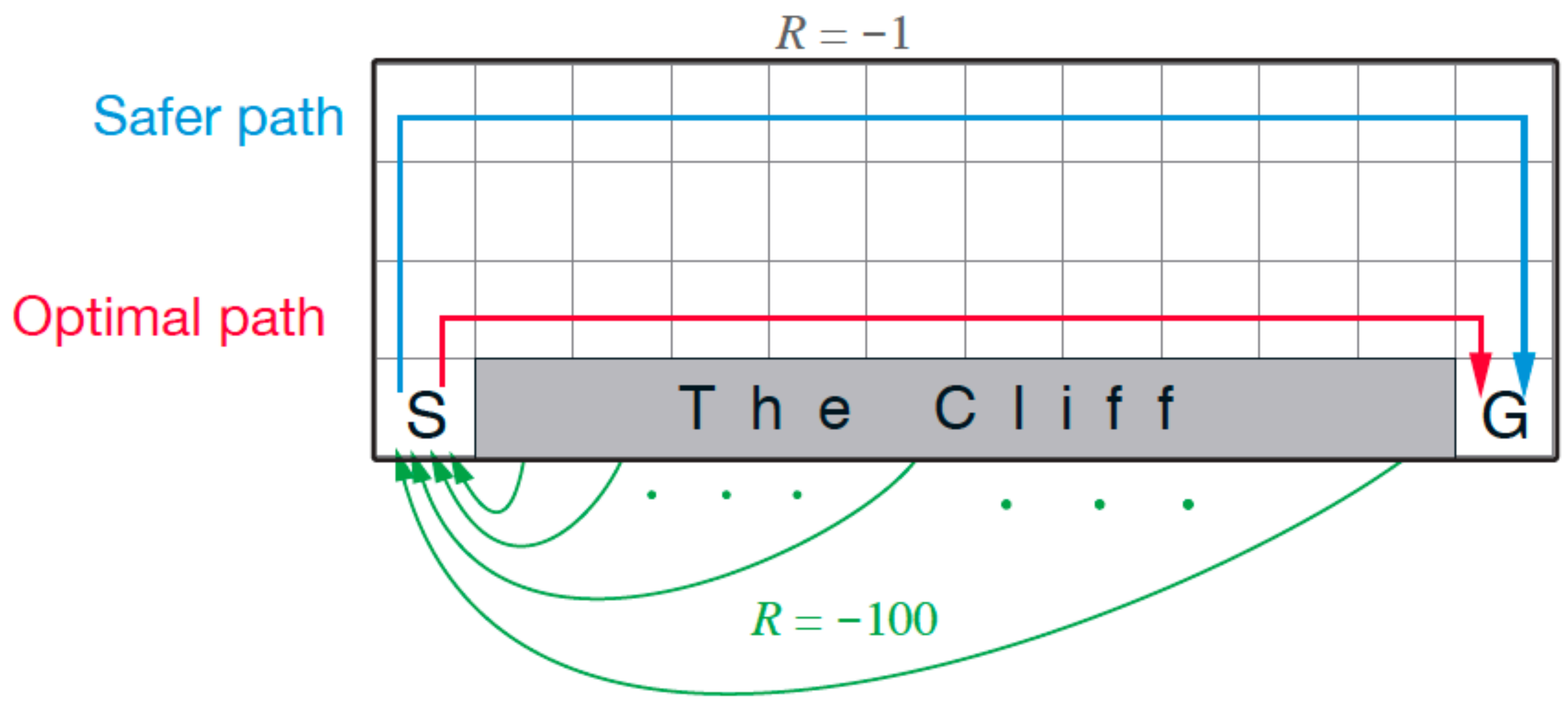
$$P(a_t = a) = \text{argmax}_a [V_t(a) + c \cdot \sqrt{\frac{\ln t}{N_t(a)}}]$$

Limitation of multi-armed bandit problems

Your current action does not influence what happens next!!

How can we solve sequential problems?

The textbook problem:
‘Cliff-World’



From classical to instrumental learning

TD Learning:

Prediction error

$$V(s_t) \leftarrow V(s_t) + \alpha \cdot (r + \gamma \cdot V(s_{t+1}) - V(s_t))$$

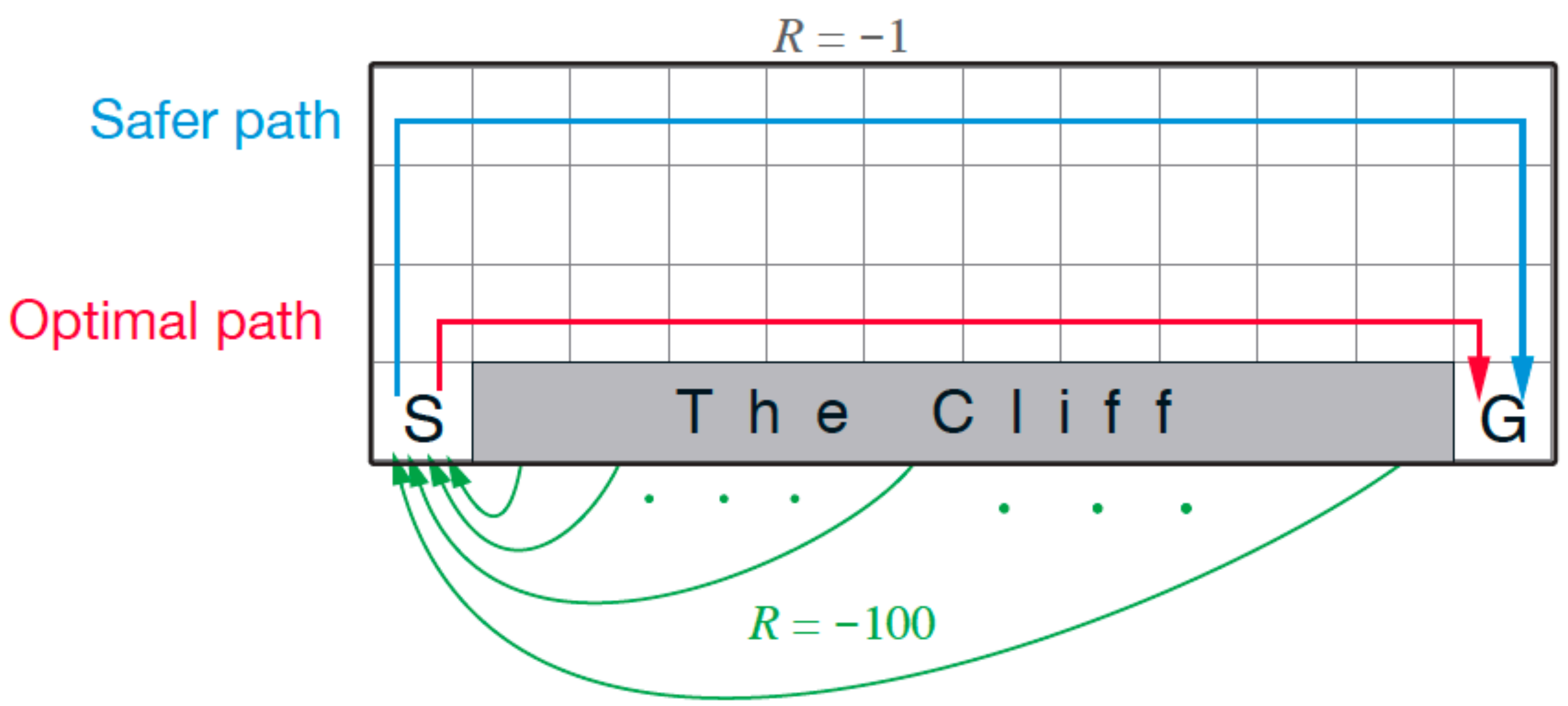
Learning rate Discount rate

Q-Learning:

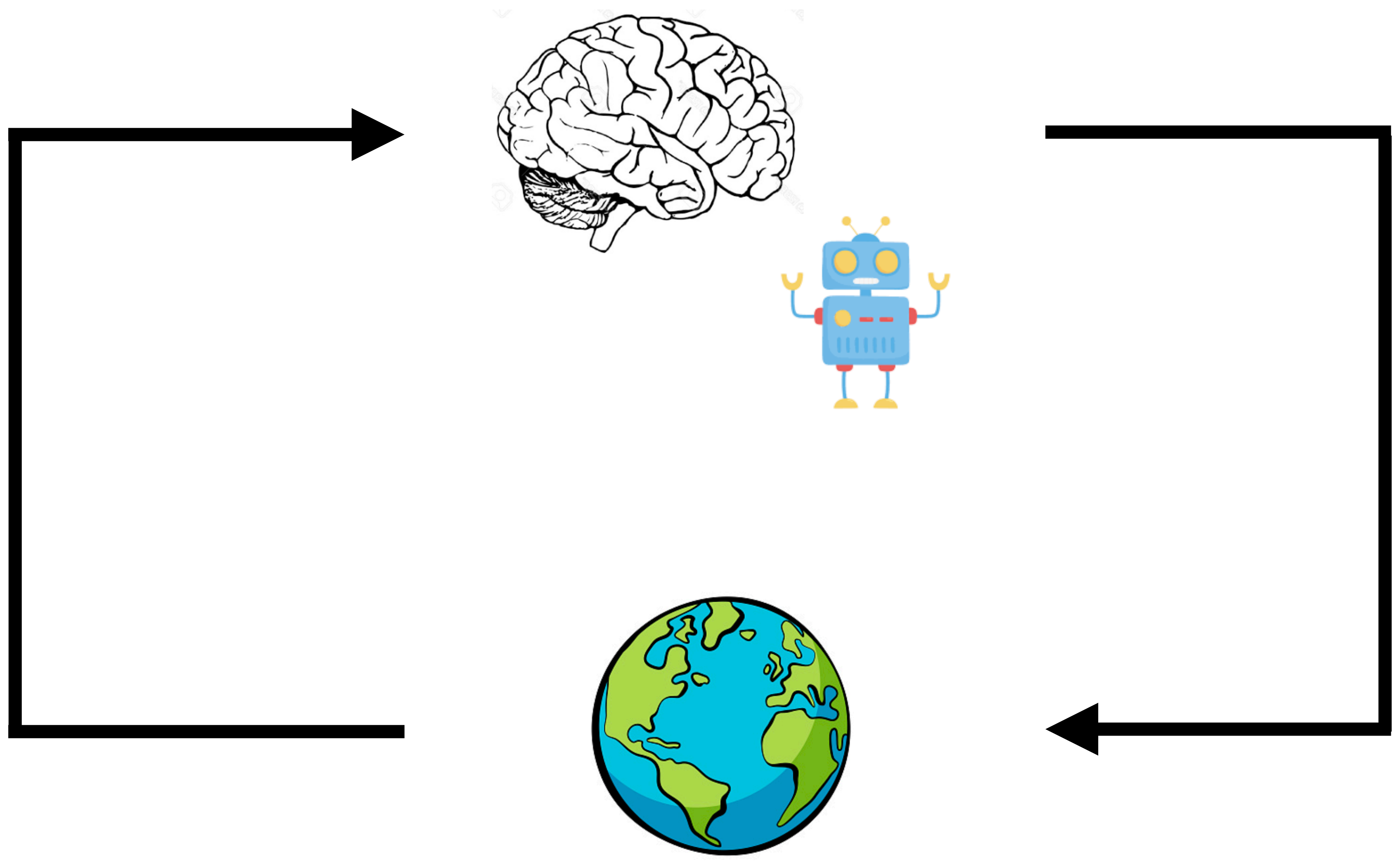
Prediction error

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \cdot (r + \gamma \cdot max_a Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t))$$

Learning rate Discount rate



Basic setup: how do agents learn to act?

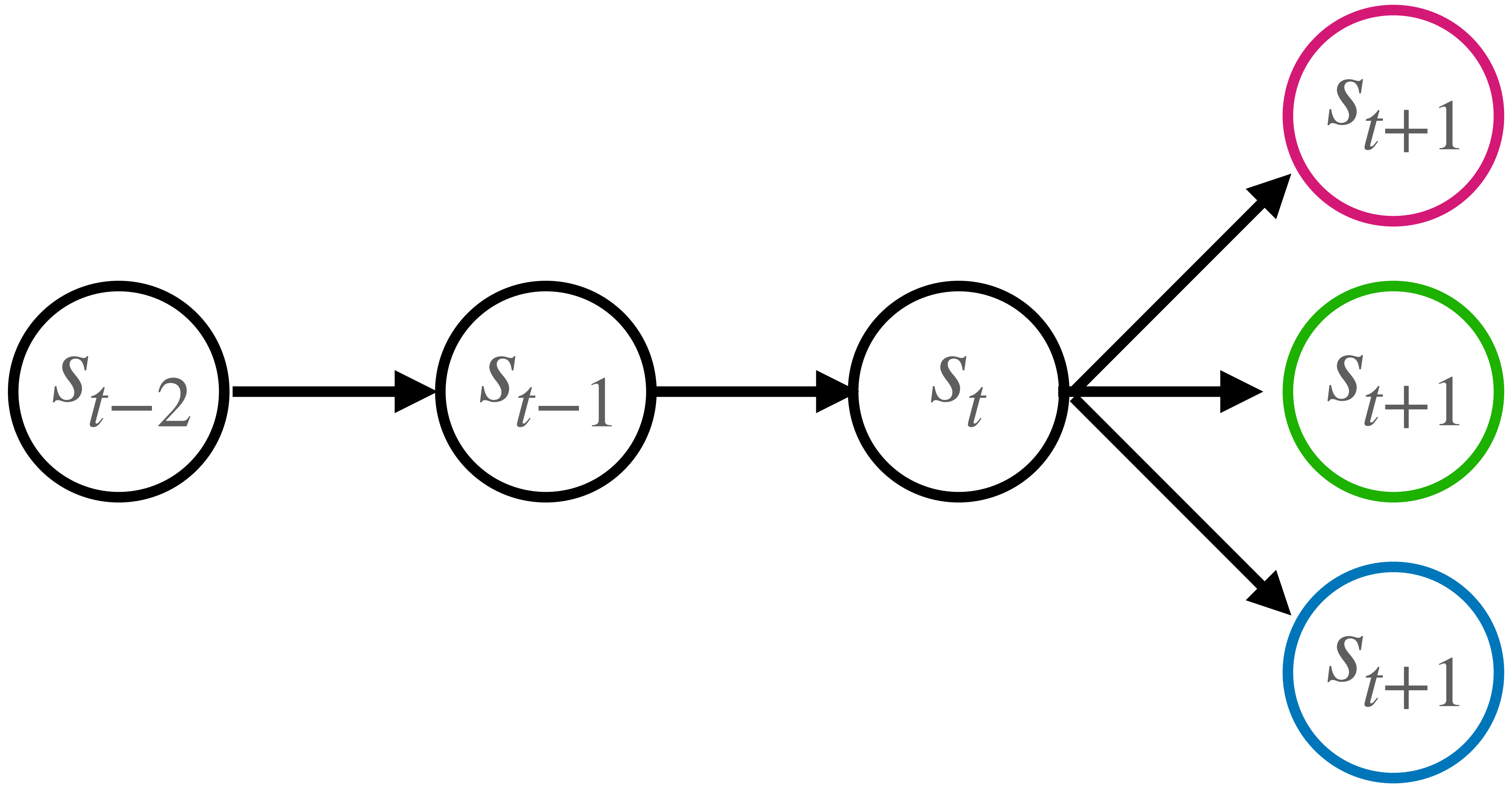


Agents can learn a **model of the environment** to make smarter decisions, e.g.:

$$P(s_{t+1} = s \,|\, s_t = s, a_t = a)$$

Markov Process

Most RL problems are problems where agents face sequences of states:

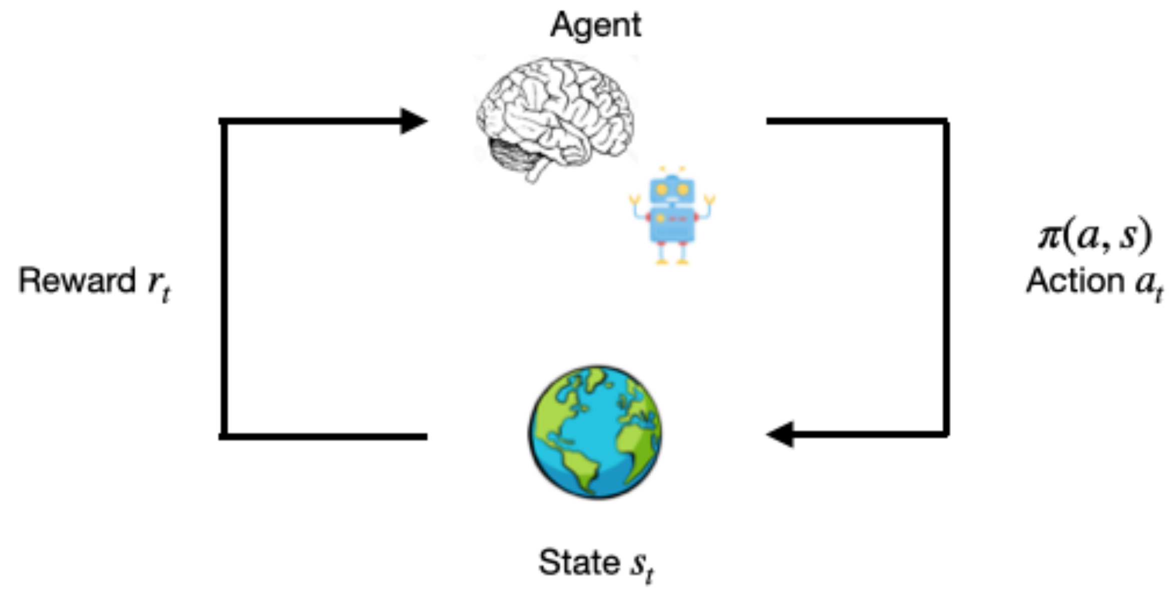


Fundamental property: **Markov property**

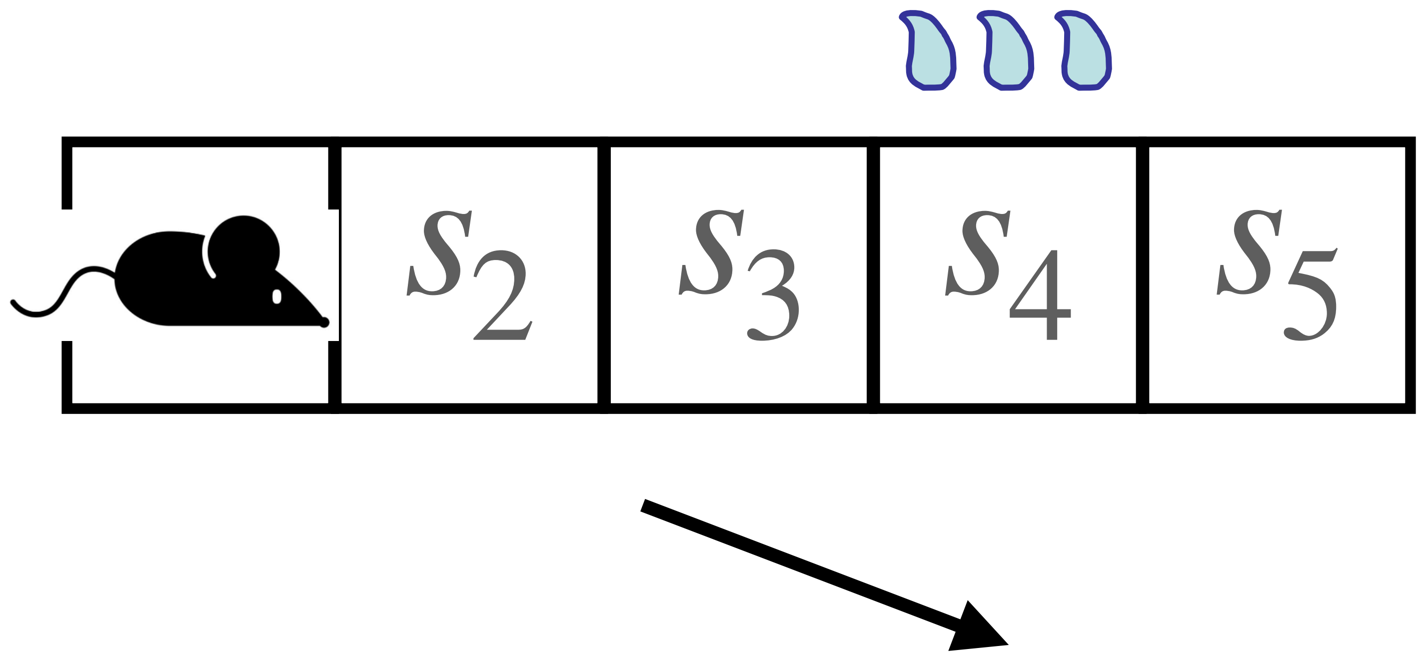
$$P(s_{t+1} = s \mid s_t, s_{t-1}, s_{t-2}, \dots) = P(s_{t+1} = s \mid s_t)$$

"The future is independent of the past given the present"

Markov Process

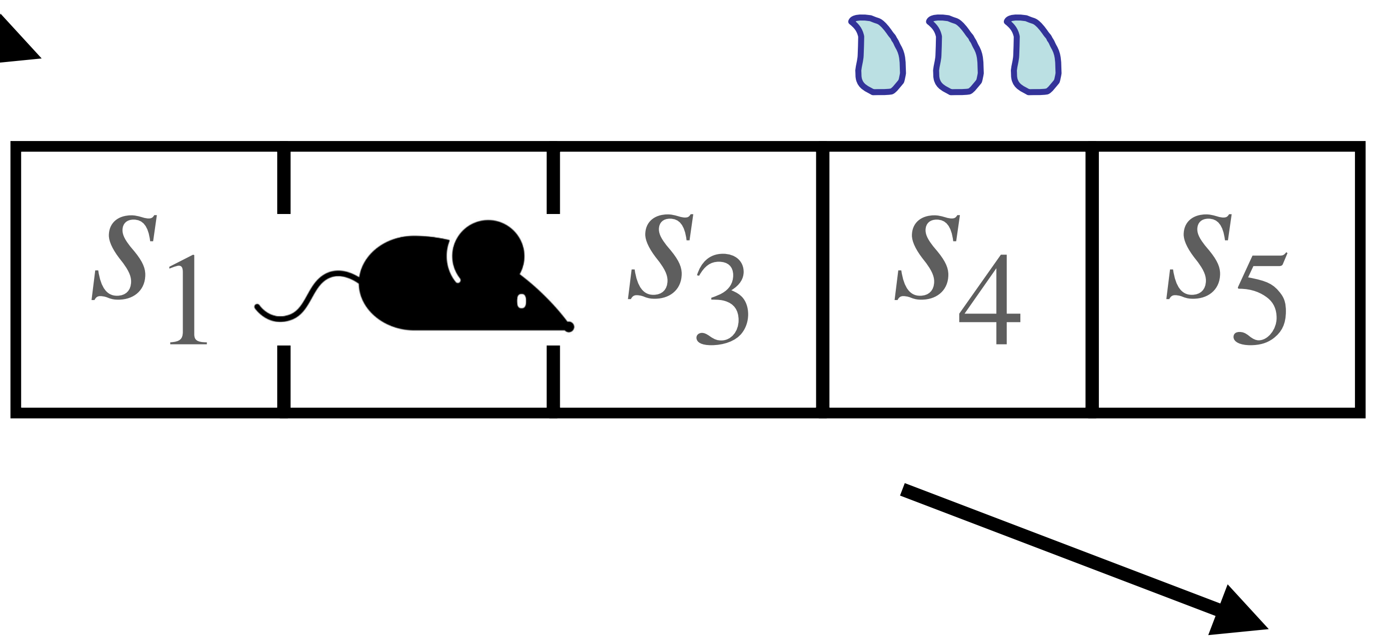


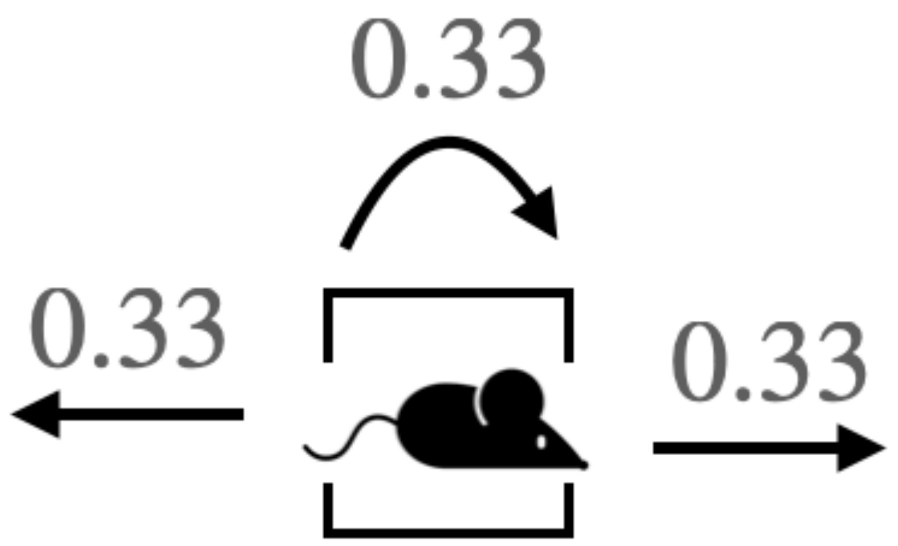
Does this problem have the **Markov Property?**

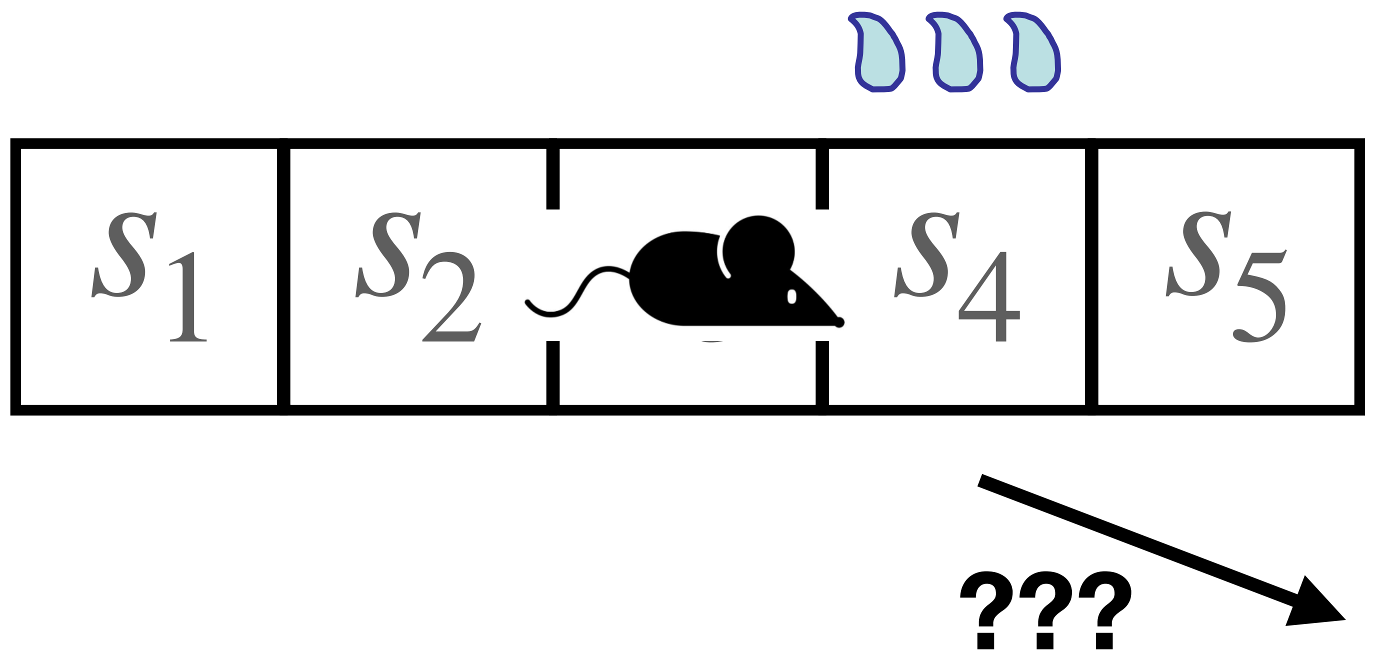


???

$$P(s_{t+1} = s \,|\, s_t, s_{t-1}, s_{t-2}, \ldots) = P(s_{t+1} = s \,|\, s_t)$$







MDPs basis for model-based RL

$$P(s', r \mid s, a) = P(s_{t+1} = s', r_{t+1} = r \mid s_t = s, a_t = a)$$

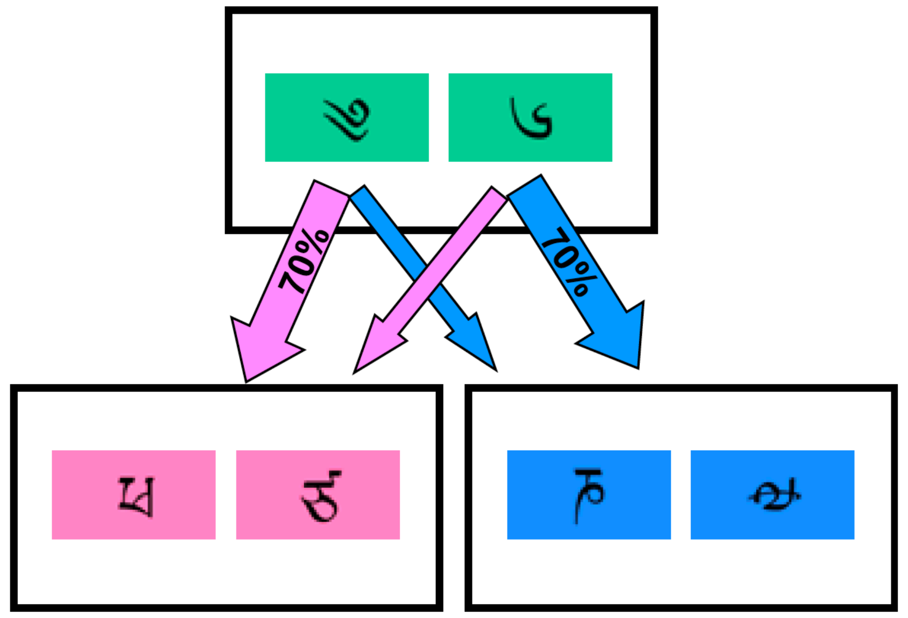
How can we make use of such models of the world?

Planning and **action selection**

Learning

- Key idea: store experiences in world model $P(s', r \mid s, a)$

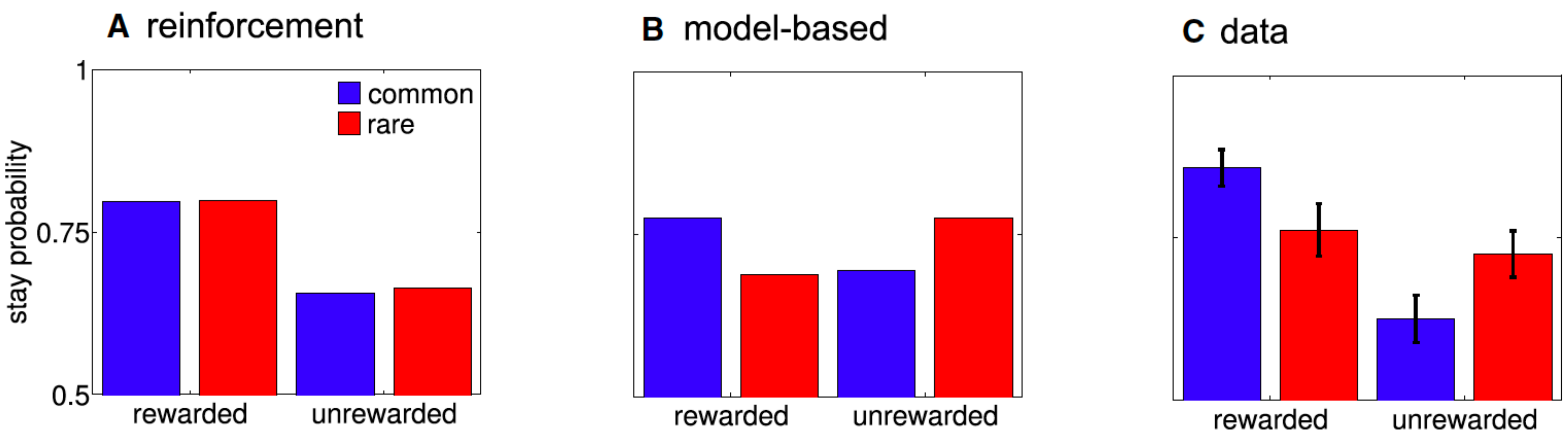
Two-step task: one of the most iconic RL tasks



Distinguish model-free vs. Model-based learning

Trial t+1

?



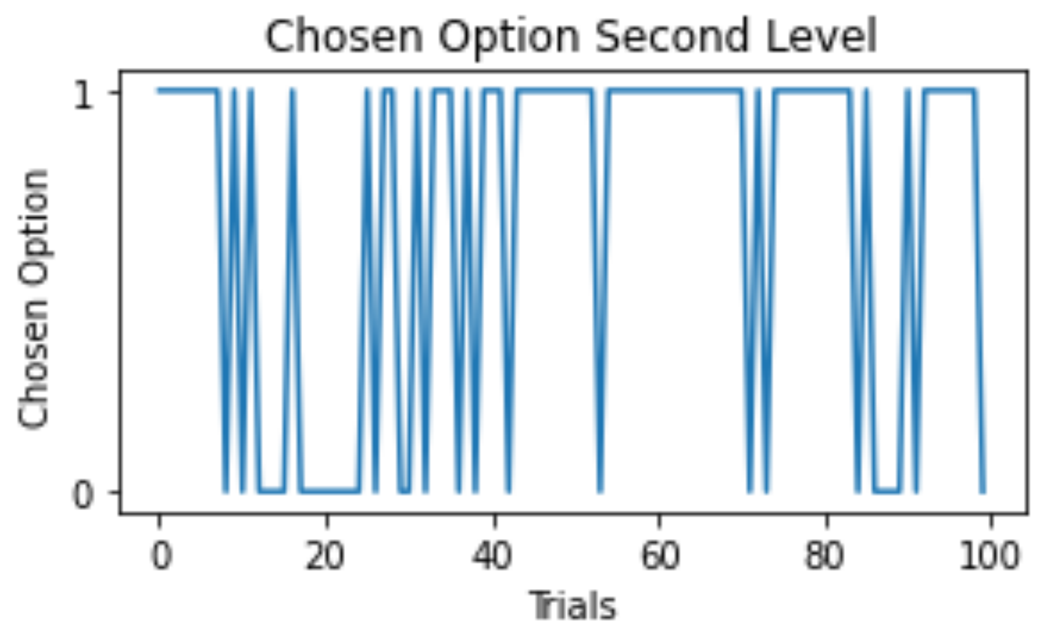
Model-free RL agent: repeat what is rewarding

Model-based RL agent: repeat what is rewarding, but be clever

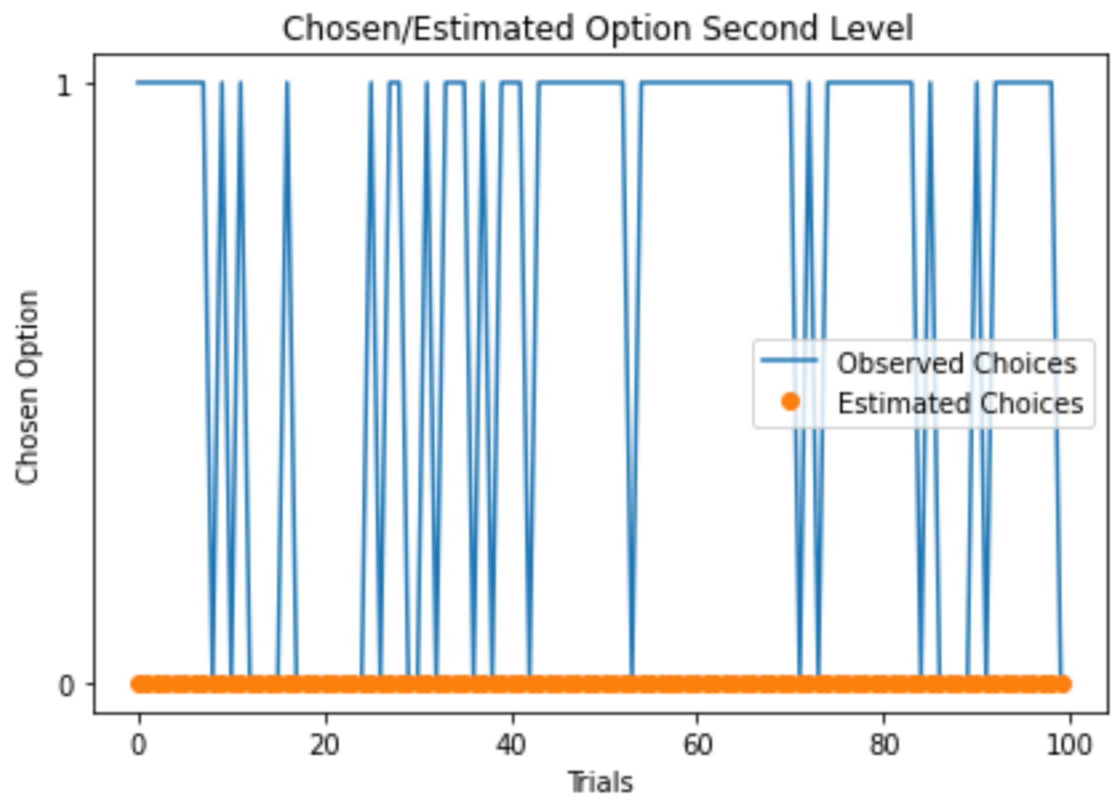
Really data: a mix of both

Problem: how do we find the best parameters for a given model?

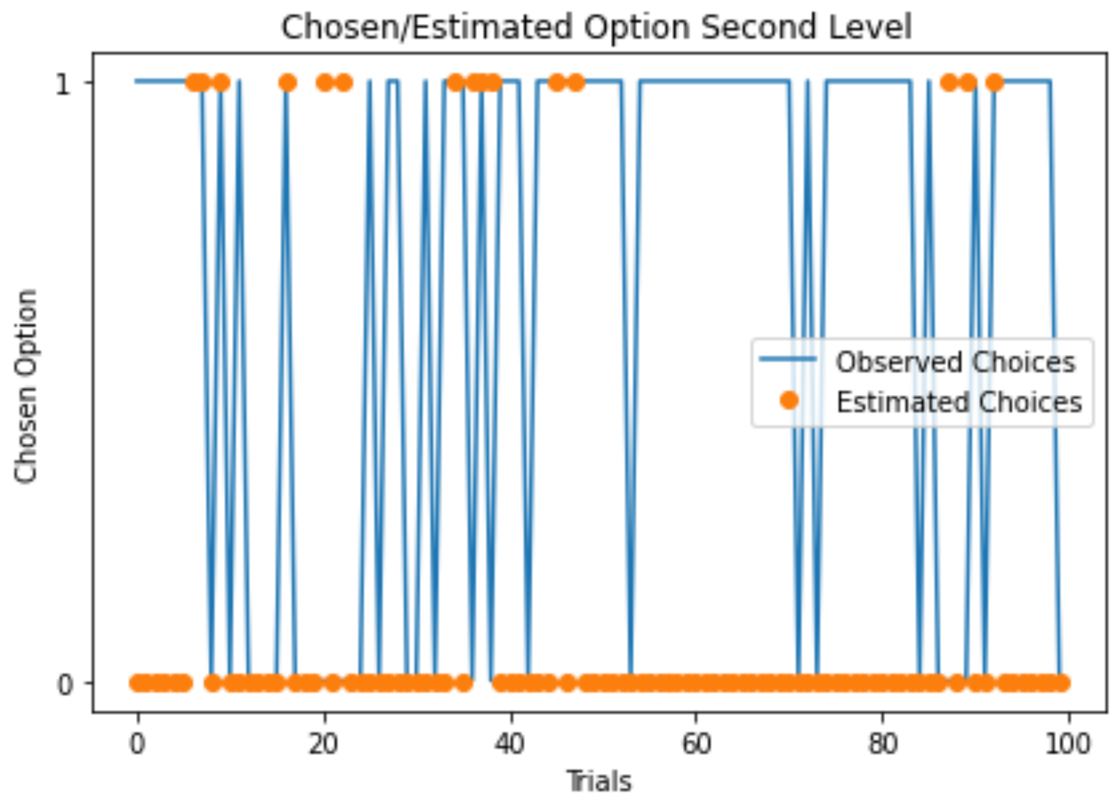
Assume your participant behaves like this:



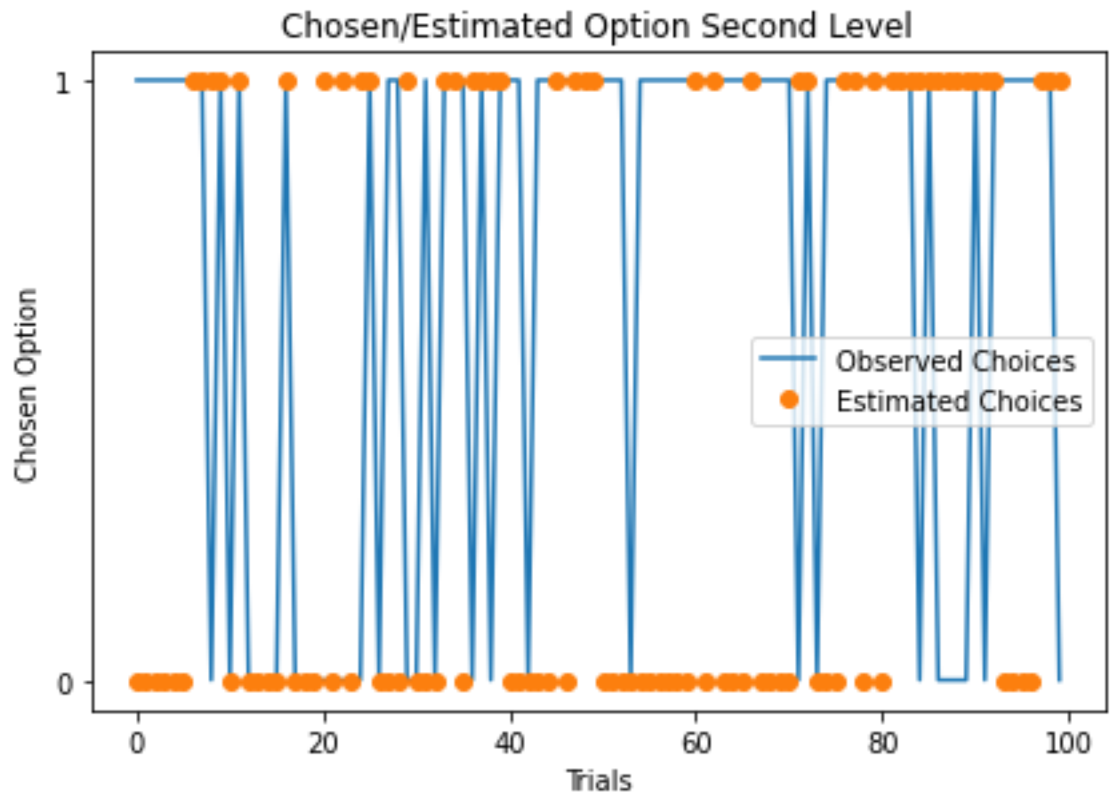
(Here: data generated with $\alpha = 0.5$ and $\beta = 5$)



($\alpha = 0$ and $\beta = 5$)



($\alpha = 1$ and $\beta = 5$)



($\alpha = 0.55$ and $\beta = 5.06$)

Any other Questions?