



Pontificia Universidad Católica de Chile  
Escuela de Ingeniería  
Departamento de Ciencias de la Computación

# Aprendizaje Reforzado

**Rodrigo Toro Icarte** ([rodrigo.toro@ing.puc.cl](mailto:rodrigo.toro@ing.puc.cl))

IIC3633 – Sistemas Recomendadores

**13 de Octubre, 2022**

Un poco sobre mí...

# Un poco sobre mí...



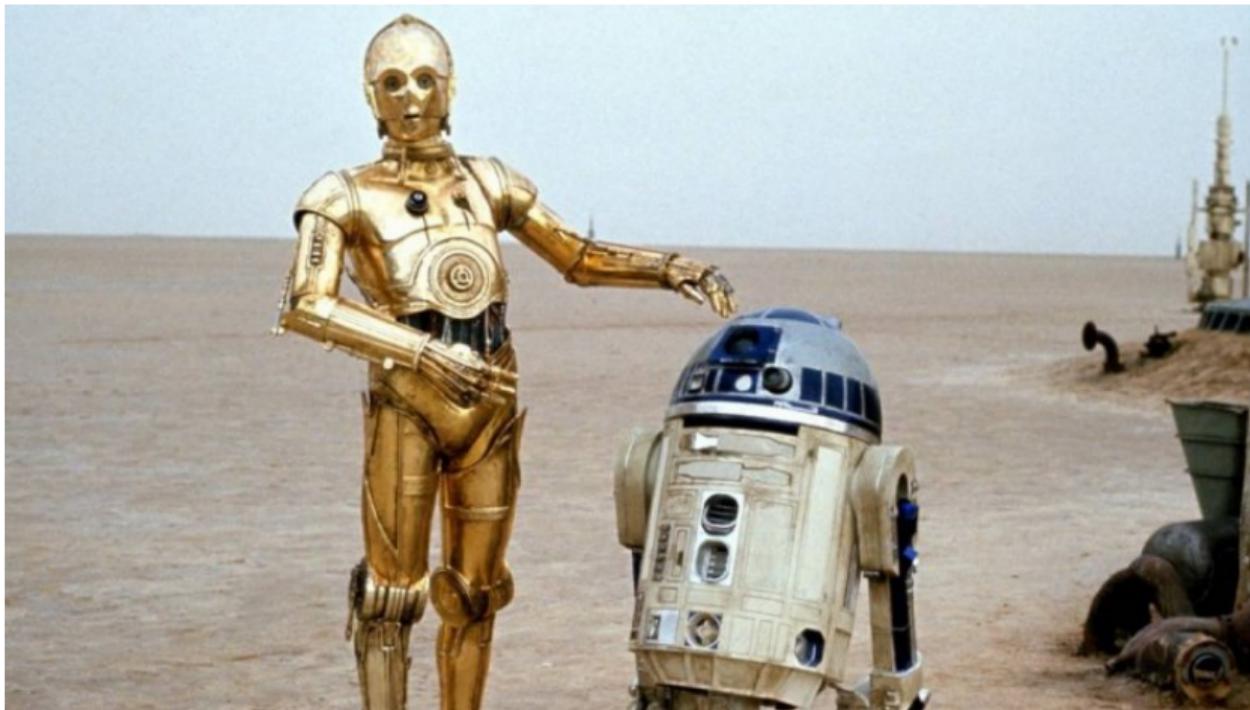
# Un poco sobre mí...



# Un poco sobre mí...



Un poco sobre mí...



# Aprendizaje Reforzado

# Aprendizaje Reforzado

**¿Qué es el aprendizaje reforzado?**

# Aprendizaje Reforzado

## **¿Qué es el aprendizaje reforzado?**

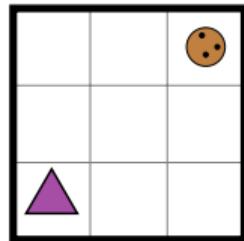
Es una rama de la inteligencia artificial que estudia cómo crear agentes que aprenden a resolver problemas mediante ensayo y error.

# Aprendizaje Reforzado

## ¿Qué es el aprendizaje reforzado?

Es una rama de la inteligencia artificial que estudia cómo crear agentes que aprenden a resolver problemas mediante ensayo y error.

### Ejemplo ilustrativo:

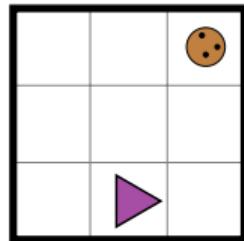


# Aprendizaje Reforzado

## ¿Qué es el aprendizaje reforzado?

Es una rama de la inteligencia artificial que estudia cómo crear agentes que aprenden a resolver problemas mediante ensayo y error.

### Ejemplo ilustrativo:

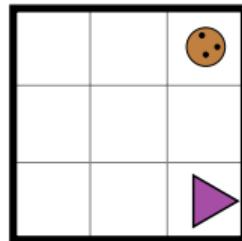


# Aprendizaje Reforzado

## ¿Qué es el aprendizaje reforzado?

Es una rama de la inteligencia artificial que estudia cómo crear agentes que aprenden a resolver problemas mediante ensayo y error.

### Ejemplo ilustrativo:

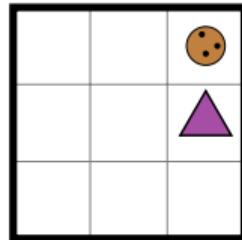


# Aprendizaje Reforzado

## ¿Qué es el aprendizaje reforzado?

Es una rama de la inteligencia artificial que estudia cómo crear agentes que aprenden a resolver problemas mediante ensayo y error.

### Ejemplo ilustrativo:

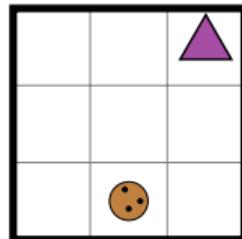


# Aprendizaje Reforzado

## ¿Qué es el aprendizaje reforzado?

Es una rama de la inteligencia artificial que estudia cómo crear agentes que aprenden a resolver problemas mediante ensayo y error.

### Ejemplo ilustrativo:



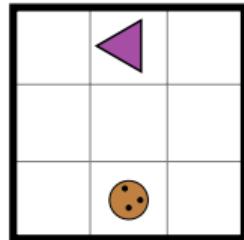
(+1 recompensa)

# Aprendizaje Reforzado

## ¿Qué es el aprendizaje reforzado?

Es una rama de la inteligencia artificial que estudia cómo crear agentes que aprenden a resolver problemas mediante ensayo y error.

### Ejemplo ilustrativo:

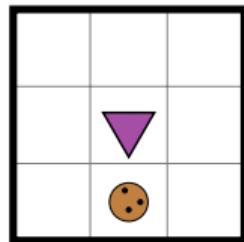


# Aprendizaje Reforzado

## ¿Qué es el aprendizaje reforzado?

Es una rama de la inteligencia artificial que estudia cómo crear agentes que aprenden a resolver problemas mediante ensayo y error.

### Ejemplo ilustrativo:

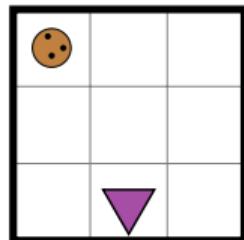


# Aprendizaje Reforzado

## ¿Qué es el aprendizaje reforzado?

Es una rama de la inteligencia artificial que estudia cómo crear agentes que aprenden a resolver problemas mediante ensayo y error.

### Ejemplo ilustrativo:



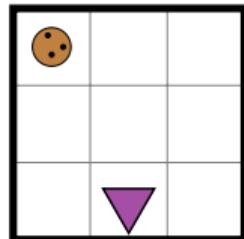
(+1 recompensa)

# Aprendizaje Reforzado

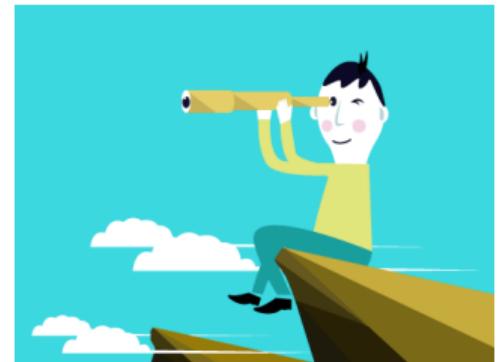
## ¿Qué es el aprendizaje reforzado?

Es una rama de la inteligencia artificial que estudia cómo crear agentes que aprenden a resolver problemas mediante ensayo y error.

### Ejemplo ilustrativo:

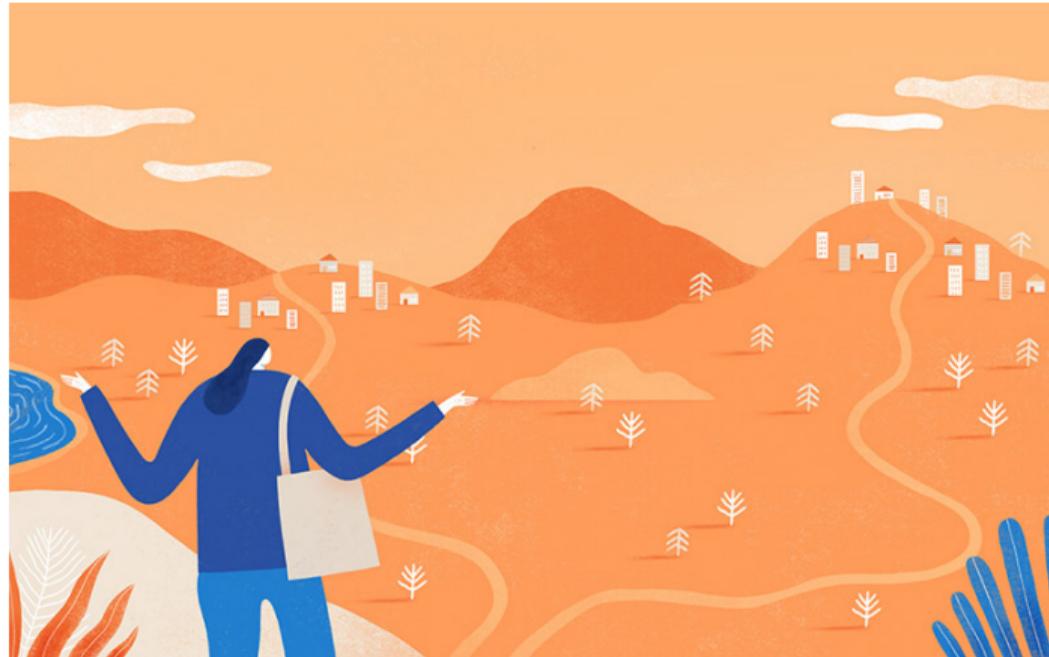


(+1 recompensa)



# Problemas de Decisión

La vida está llena de **problemas de decisión** y el éxito (o fracaso) de nuestros emprendimientos dependen de la calidad de nuestras decisiones.



# Problemas de Decisión

La vida está llena de **problemas de decisión** y el éxito (o fracaso) de nuestros emprendimientos dependen de la calidad de nuestras decisiones.

- ¿Cómo ruteo los camiones de la empresa?



# Problemas de Decisión

La vida está llena de **problemas de decisión** y el éxito (o fracaso) de nuestros emprendimientos dependen de la calidad de nuestras decisiones.

- ¿Cómo ruteo los camiones de la empresa?
- ¿Cómo asigno pacientes a doctores?



# Problemas de Decisión

La vida está llena de **problemas de decisión** y el éxito (o fracaso) de nuestros emprendimientos dependen de la calidad de nuestras decisiones.

- ¿Cómo ruteo los camiones de la empresa?
- ¿Cómo asigno pacientes a doctores?
- ¿Cómo invierto mi dinero?



# Problemas de Decisión

La vida está llena de **problemas de decisión** y el éxito (o fracaso) de nuestros emprendimientos dependen de la calidad de nuestras decisiones.

- ¿Cómo ruteo los camiones de la empresa?
- ¿Cómo asigno pacientes a doctores?
- ¿Cómo invierto mi dinero?
- ¿Cómo combino ingredientes de origen vegetal para recrear alimentos de origen animal?



# Problemas de Decisión

La vida está llena de **problemas de decisión** y el éxito (o fracaso) de nuestros emprendimientos dependen de la calidad de nuestras decisiones.

- ¿Cómo ruteo los camiones de la empresa?
- ¿Cómo asigno pacientes a doctores?
- ¿Cómo invierto mi dinero?
- ¿Cómo combino ingredientes de origen vegetal para recrear alimentos de origen animal?
- ¿Cómo recomiendo items/información a usuarios?



# Problemas de Decisión

La vida está llena de **problemas de decisión** y el éxito (o fracaso) de nuestros emprendimientos dependen de la calidad de nuestras decisiones.

- ¿Cómo ruteo los camiones de la empresa?
- ¿Cómo asigno pacientes a doctores?
- ¿Cómo invierto mi dinero?
- ¿Cómo combino ingredientes de origen vegetal para recrear alimentos de origen animal?
- ¿Cómo recomiendo items/información a usuarios?



El aprendizaje reforzado permite resolver problemas de decisión de manera autónoma.

# **Objetivos de esta clase**

# Objetivos de esta clase

## **Objetivo principal:**

Entender los fundamentos teóricos y prácticos del aprendizaje reforzado.

# Objetivos de esta clase

## **Objetivo principal:**

Entender los fundamentos teóricos y prácticos del aprendizaje reforzado.

## **Objetivos secundarios:**

1. Identificar aspectos en los que el aprendizaje reforzado puede aportar al desarrollo de mejores sistemas recomendadores.
2. Que puedan utilizar aprendizaje reforzado en sus proyectos de curso.

# Objetivos de esta clase

## **Objetivo principal:**

Entender los fundamentos teóricos y prácticos del aprendizaje reforzado.

## **Objetivos secundarios:**

1. Identificar aspectos en los que el aprendizaje reforzado puede aportar al desarrollo de mejores sistemas recomendadores.
2. Que puedan utilizar aprendizaje reforzado en sus proyectos de curso.

## **¿Qué sabemos hasta ahora?**

El aprendizaje reforzado permite resolver **problemas de decisión** de forma automática.

# Objetivos de esta clase

## **Objetivo principal:**

Entender los fundamentos teóricos y prácticos del aprendizaje reforzado.

## **Objetivos secundarios:**

1. Identificar aspectos en los que el aprendizaje reforzado puede aportar al desarrollo de mejores sistemas recomendadores.
2. Que puedan utilizar aprendizaje reforzado en sus proyectos de curso.

## **¿Qué sabemos hasta ahora?**

El aprendizaje reforzado permite resolver **problemas de decisión** de forma automática.

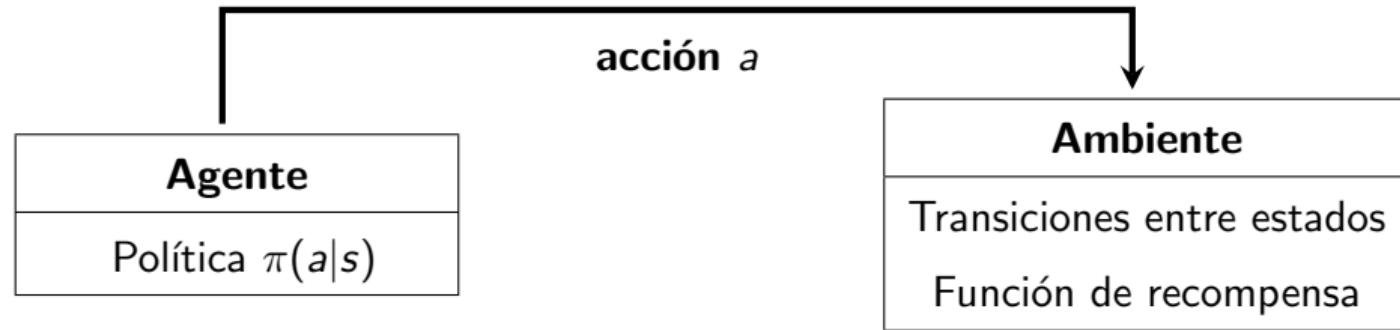
## **¿Cómo lo hace? :O**

# Aprendizaje Reforzado: Descripción de Alto Nivel

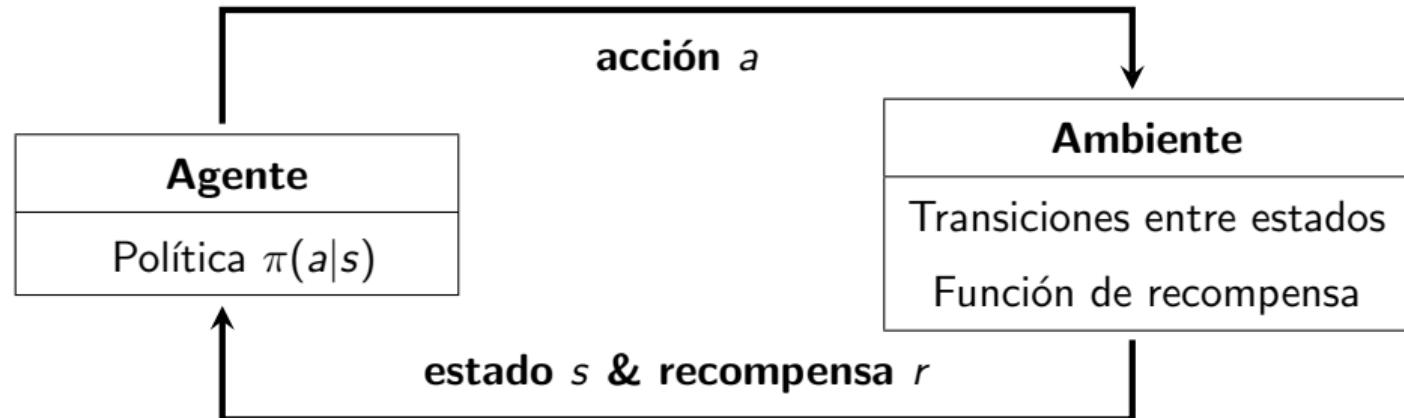
<b>Agente</b>
Política $\pi(a s)$

<b>Ambiente</b>
Transiciones entre estados
Función de recompensa

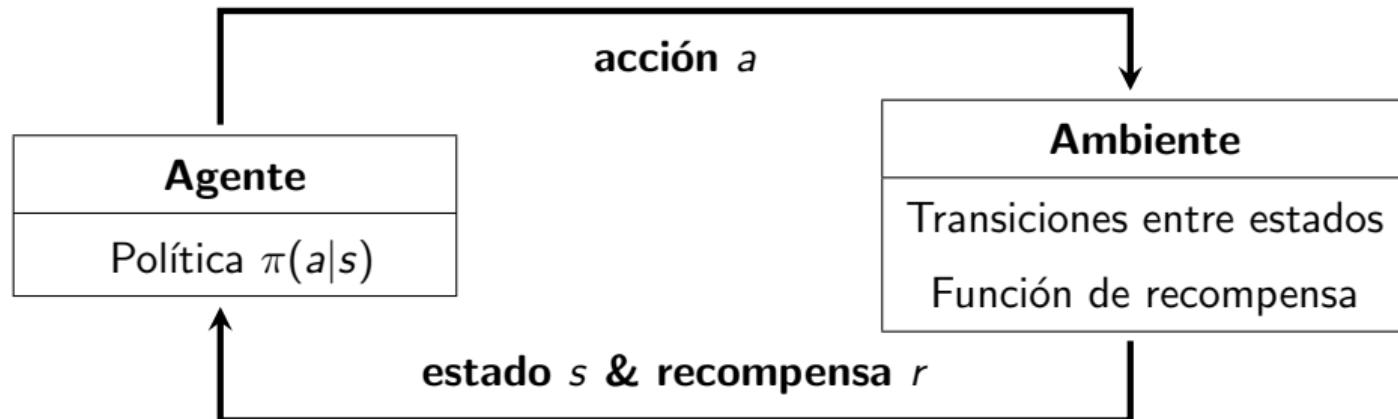
# Aprendizaje Reforzado: Descripción de Alto Nivel



# Aprendizaje Reforzado: Descripción de Alto Nivel



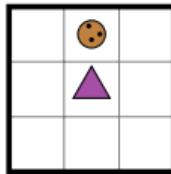
# Aprendizaje Reforzado: Descripción de Alto Nivel



El objetivo del agente es obtener recompensas del ambiente. Para ello, ajusta su política para seleccionar acciones  $\pi(a|s)$  con el fin de obtener más recompensa.

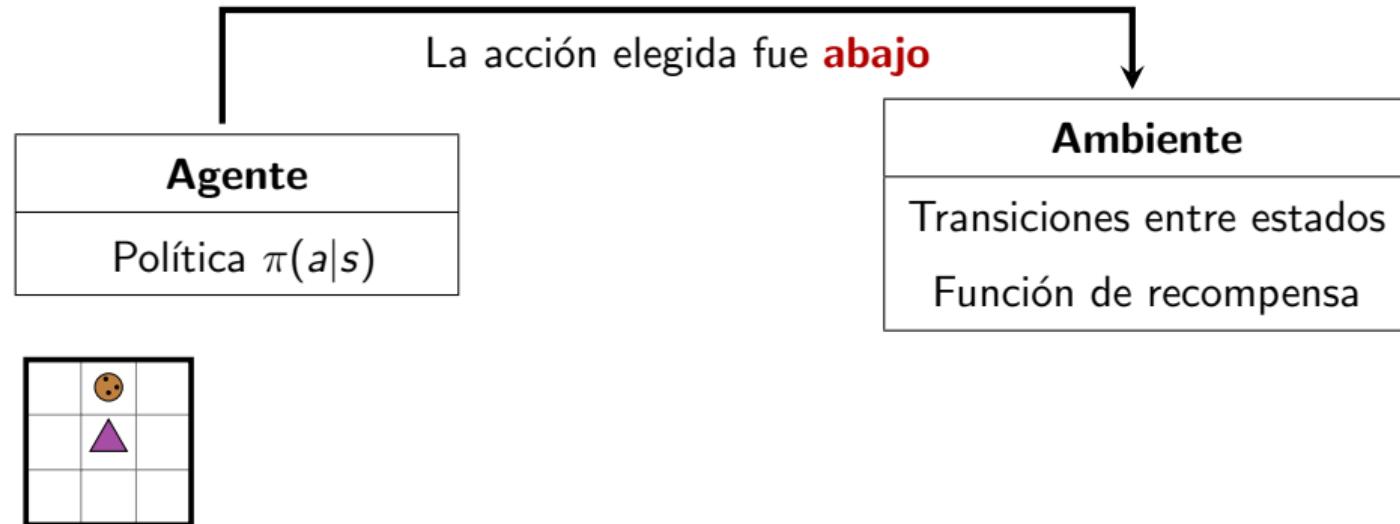
# Aprendizaje Reforzado: Descripción de Alto Nivel

Agente
Política $\pi(a s)$

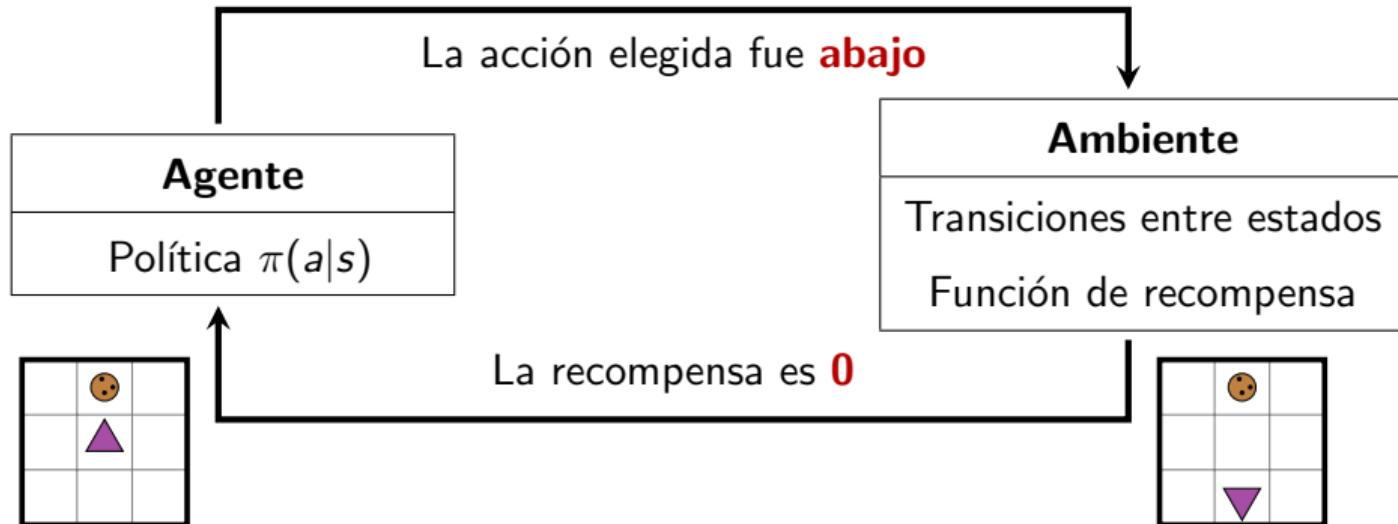


Ambiente
Transiciones entre estados
Función de recompensa

# Aprendizaje Reforzado: Descripción de Alto Nivel

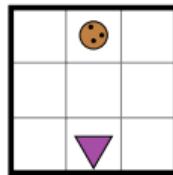


# Aprendizaje Reforzado: Descripción de Alto Nivel



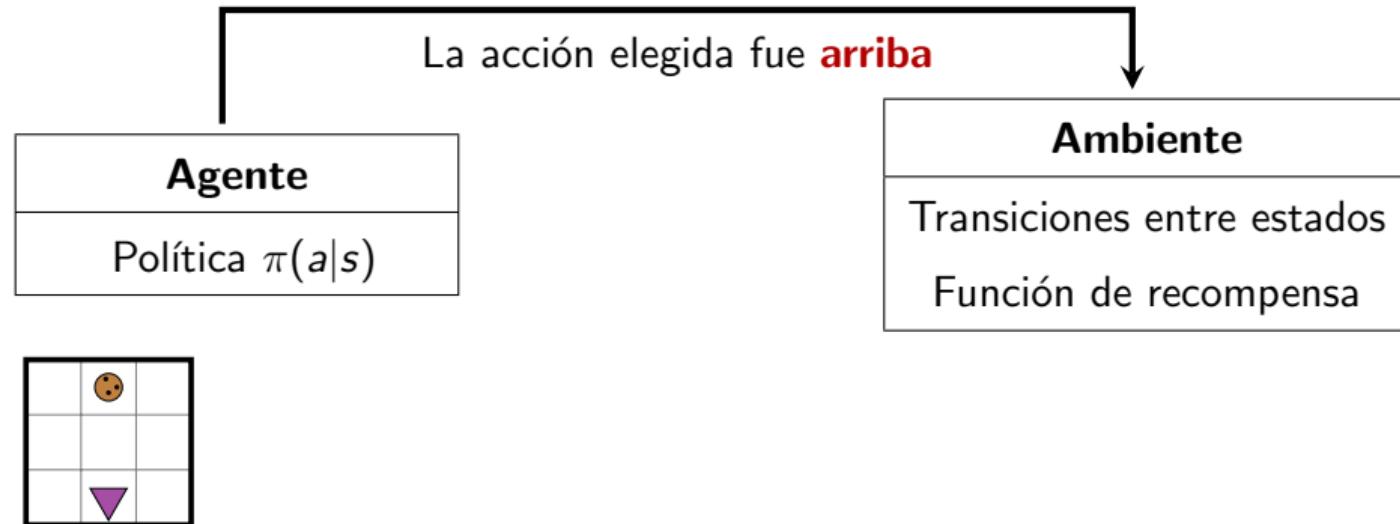
# Aprendizaje Reforzado: Descripción de Alto Nivel

Agente
Política $\pi(a s)$

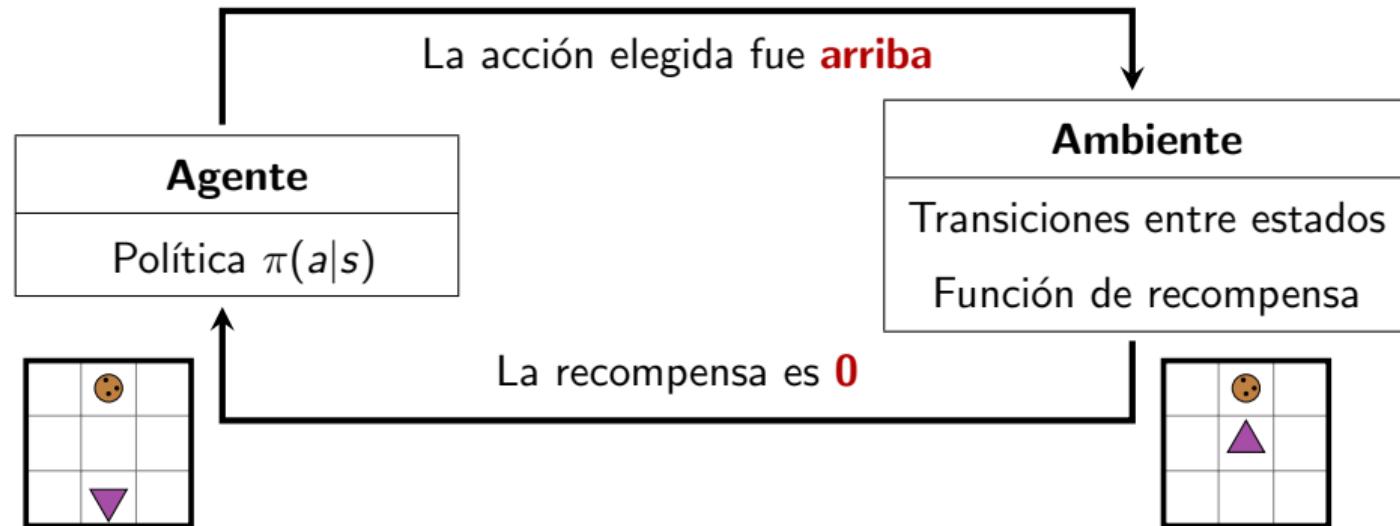


Ambiente
Transiciones entre estados
Función de recompensa

# Aprendizaje Reforzado: Descripción de Alto Nivel

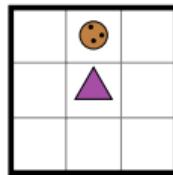


# Aprendizaje Reforzado: Descripción de Alto Nivel



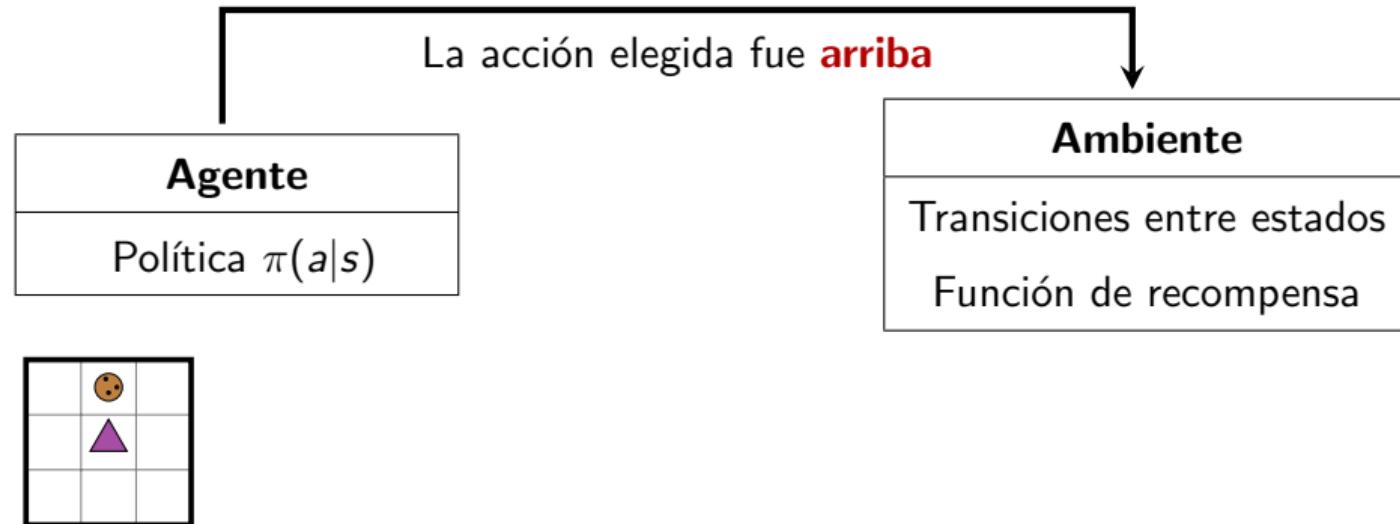
# Aprendizaje Reforzado: Descripción de Alto Nivel

Agente
Política $\pi(a s)$

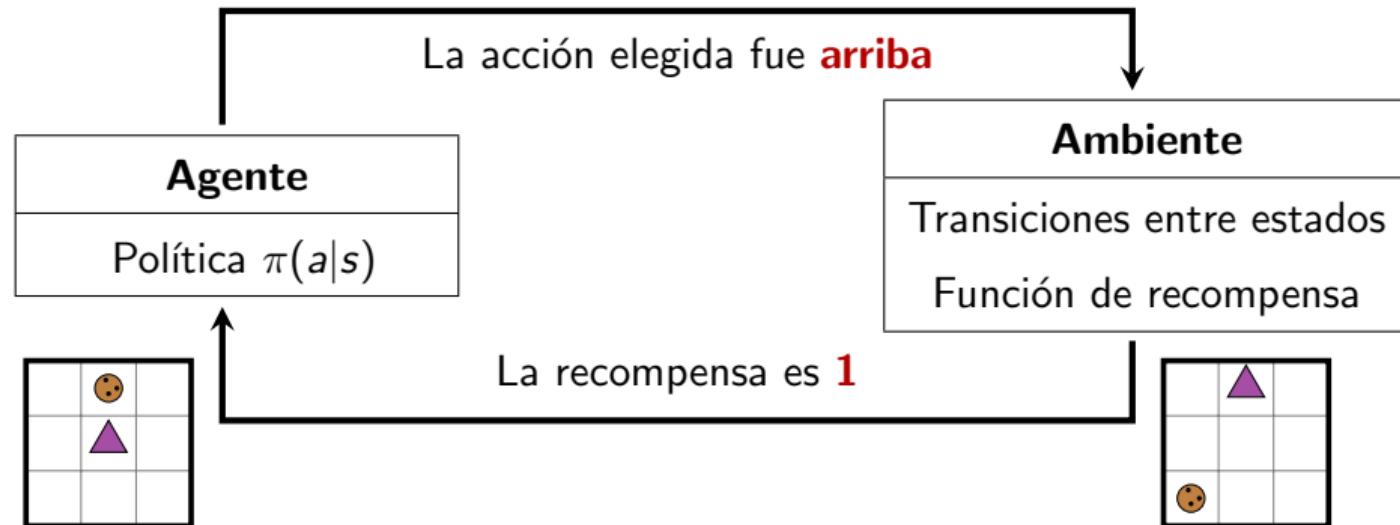


Ambiente
Transiciones entre estados
Función de recompensa

# Aprendizaje Reforzado: Descripción de Alto Nivel

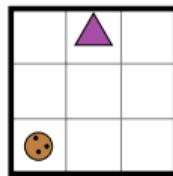


# Aprendizaje Reforzado: Descripción de Alto Nivel



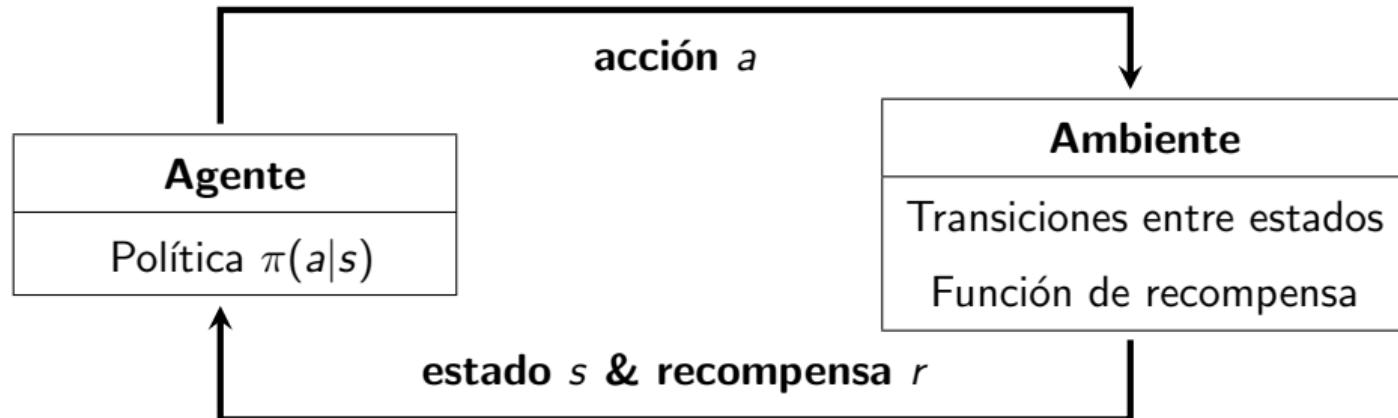
# Aprendizaje Reforzado: Descripción de Alto Nivel

Agente
Política $\pi(a s)$



Ambiente
Transiciones entre estados
Función de recompensa

# Aprendizaje Reforzado: Descripción de Alto Nivel

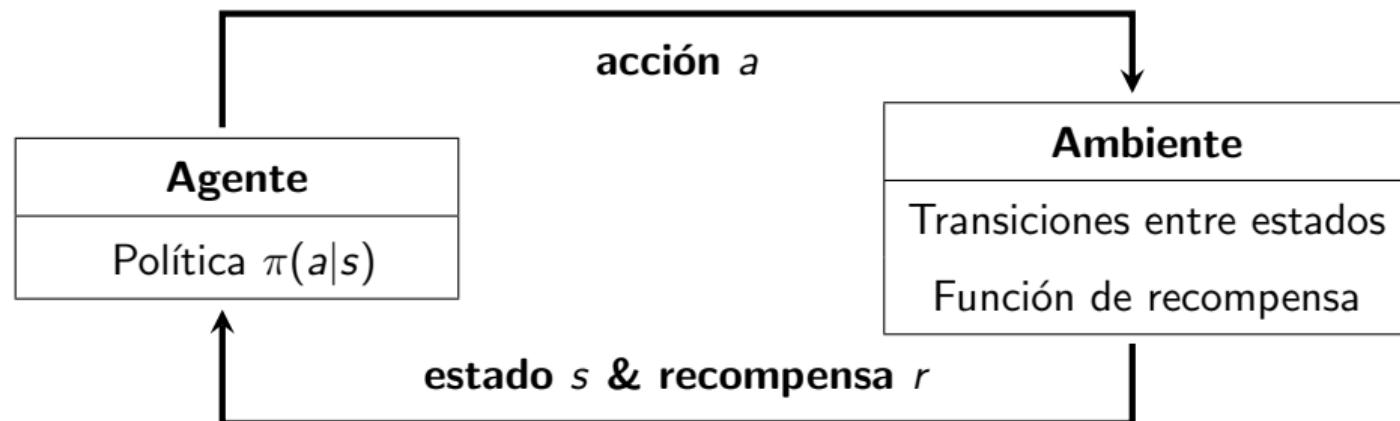


Dividamos la discusión en dos partes: El **ambiente** y el **agente**.

# **Ambientes de Aprendizaje**

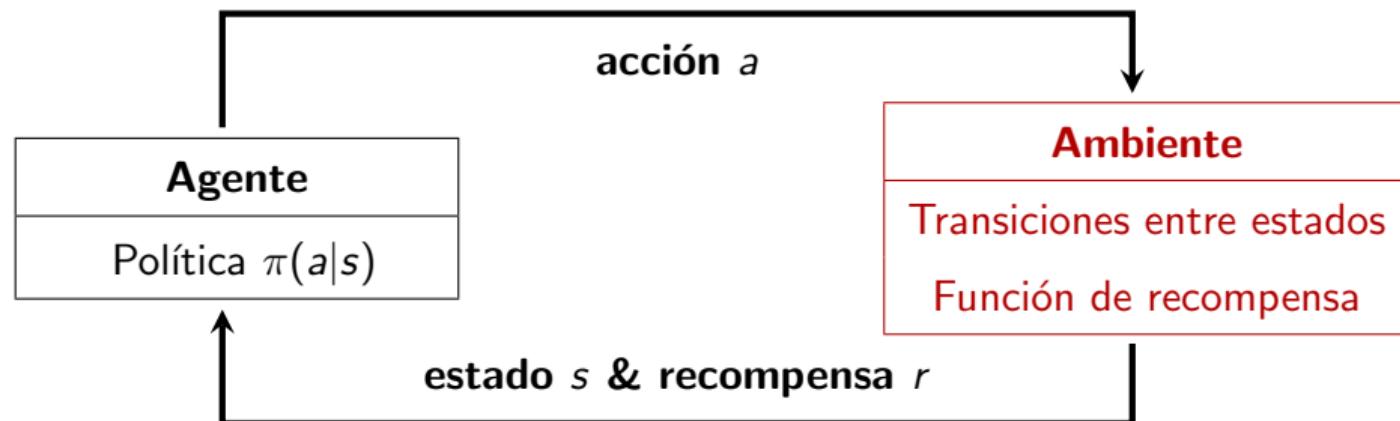
# Problemas de Decisión

Para usar aprendizaje reforzado hay que modelar el problema como un *ambiente*.



# Problemas de Decisión

Para usar aprendizaje reforzado hay que modelar el problema como un *ambiente*.



... pero ¿qué es y cómo se define un ambiente?

## Markov Decision Process (MDP)

Un MDP es una tupla  $\mathcal{M} = (S, A, r, p, \gamma)$  donde

- $S$  es un conjunto finito de estados.
- $A$  es un conjunto finito de acciones.
- $r(s, a, s')$  es la recompensa recibida al ejecutar  $a \in A$  en  $s \in S$  y llegar a  $s' \in S$ .
- $p(s'|s, a)$  es la probabilidad de pasar de  $s \in S$  a  $s' \in S$  si se ejecuta  $a \in A$ .
- $\gamma \in (0, 1]$  es el factor de descuento (generalmente se usa  $\gamma = 0.99$ ).

## Markov Decision Process (MDP)

Un MDP es una tupla  $\mathcal{M} = (S, A, r, p, \gamma)$  donde

- $S$  es un conjunto finito de estados.
- $A$  es un conjunto finito de acciones.
- $r(s, a, s')$  es la recompensa recibida al ejecutar  $a \in A$  en  $s \in S$  y llegar a  $s' \in S$ .
- $p(s'|s, a)$  es la probabilidad de pasar de  $s \in S$  a  $s' \in S$  si se ejecuta  $a \in A$ .
- $\gamma \in (0, 1]$  es el factor de descuento (generalmente se usa  $\gamma = 0.99$ ).

### Ejemplo galletas:

Los estados son las 72 combinaciones posibles de posiciones del agente y la galleta.

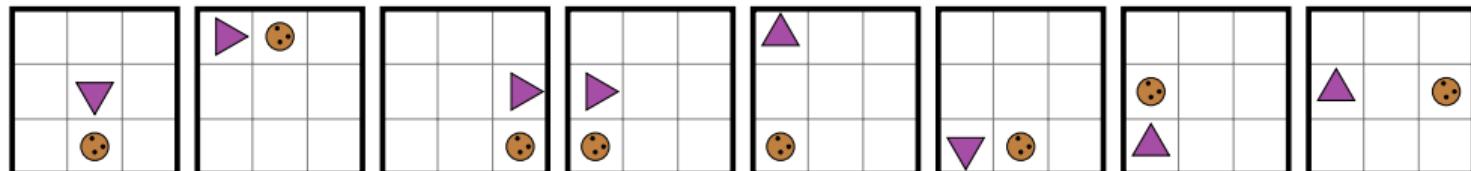
## Markov Decision Process (MDP)

Un MDP es una tupla  $\mathcal{M} = (S, A, r, p, \gamma)$  donde

- $S$  es un conjunto finito de estados.
- $A$  es un conjunto finito de acciones.
- $r(s, a, s')$  es la recompensa recibida al ejecutar  $a \in A$  en  $s \in S$  y llegar a  $s' \in S$ .
- $p(s'|s, a)$  es la probabilidad de pasar de  $s \in S$  a  $s' \in S$  si se ejecuta  $a \in A$ .
- $\gamma \in (0, 1]$  es el factor de descuento (generalmente se usa  $\gamma = 0.99$ ).

### Ejemplo galletas:

Los estados son las 72 combinaciones posibles de posiciones del agente y la galleta.



## Markov Decision Process (MDP)

Un MDP es una tupla  $\mathcal{M} = (S, A, r, p, \gamma)$  donde

- $S$  es un conjunto finito de estados.
- $A$  es un conjunto finito de acciones.
- $r(s, a, s')$  es la recompensa recibida al ejecutar  $a \in A$  en  $s \in S$  y llegar a  $s' \in S$ .
- $p(s'|s, a)$  es la probabilidad de pasar de  $s \in S$  a  $s' \in S$  si se ejecuta  $a \in A$ .
- $\gamma \in (0, 1]$  es el factor de descuento (generalmente se usa  $\gamma = 0.99$ ).

### Ejemplo galletas:

Las acciones posibles son 4: izquierda, derecha, arriba, abajo.

## Markov Decision Process (MDP)

Un MDP es una tupla  $\mathcal{M} = (S, A, r, p, \gamma)$  donde

- $S$  es un conjunto finito de estados.
- $A$  es un conjunto finito de acciones.
- $r(s, a, s')$  es la recompensa recibida al ejecutar  $a \in A$  en  $s \in S$  y llegar a  $s' \in S$ .
- $p(s'|s, a)$  es la probabilidad de pasar de  $s \in S$  a  $s' \in S$  si se ejecuta  $a \in A$ .
- $\gamma \in (0, 1]$  es el factor de descuento (generalmente se usa  $\gamma = 0.99$ ).

### Ejemplo galletas:

La recompensa es **+1** si el agente come una galleta (cero en otro caso).

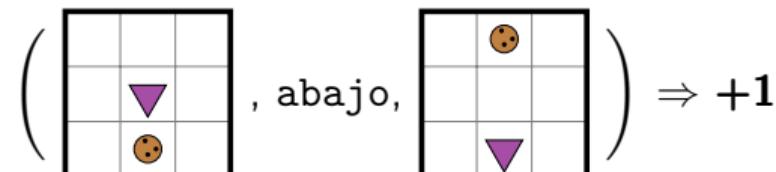
## Markov Decision Process (MDP)

Un MDP es una tupla  $\mathcal{M} = (S, A, r, p, \gamma)$  donde

- $S$  es un conjunto finito de estados.
- $A$  es un conjunto finito de acciones.
- $r(s, a, s')$  es la recompensa recibida al ejecutar  $a \in A$  en  $s \in S$  y llegar a  $s' \in S$ .
- $p(s'|s, a)$  es la probabilidad de pasar de  $s \in S$  a  $s' \in S$  si se ejecuta  $a \in A$ .
- $\gamma \in (0, 1]$  es el factor de descuento (generalmente se usa  $\gamma = 0.99$ ).

### Ejemplo galletas:

La recompensa es **+1** si el agente come una galleta (cero en otro caso).



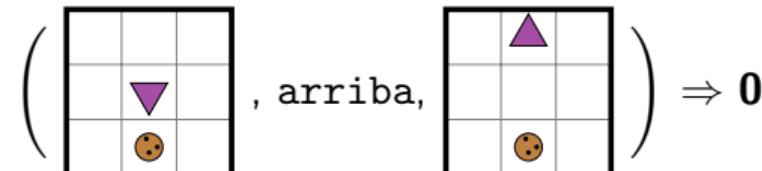
## Markov Decision Process (MDP)

Un MDP es una tupla  $\mathcal{M} = (S, A, r, p, \gamma)$  donde

- $S$  es un conjunto finito de estados.
- $A$  es un conjunto finito de acciones.
- $r(s, a, s')$  es la recompensa recibida al ejecutar  $a \in A$  en  $s \in S$  y llegar a  $s' \in S$ .
- $p(s'|s, a)$  es la probabilidad de pasar de  $s \in S$  a  $s' \in S$  si se ejecuta  $a \in A$ .
- $\gamma \in (0, 1]$  es el factor de descuento (generalmente se usa  $\gamma = 0.99$ ).

### Ejemplo galletas:

La recompensa es **+1** si el agente come una galleta (cero en otro caso).



## Markov Decision Process (MDP)

Un MDP es una tupla  $\mathcal{M} = (S, A, r, p, \gamma)$  donde

- $S$  es un conjunto finito de estados.
- $A$  es un conjunto finito de acciones.
- $r(s, a, s')$  es la recompensa recibida al ejecutar  $a \in A$  en  $s \in S$  y llegar a  $s' \in S$ .
- $p(s'|s, a)$  es la probabilidad de pasar de  $s \in S$  a  $s' \in S$  si se ejecuta  $a \in A$ .
- $\gamma \in (0, 1]$  es el factor de descuento (generalmente se usa  $\gamma = 0.99$ ).

### Ejemplo galletas:

Las probabilidades  $p(s'|s, a)$  modelan las dinámicas del mundo en que vive el agente.

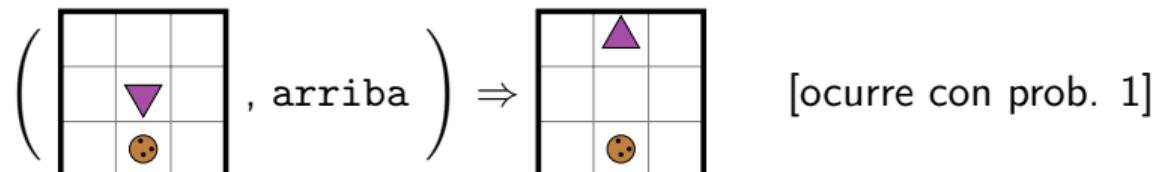
## Markov Decision Process (MDP)

Un MDP es una tupla  $\mathcal{M} = (S, A, r, p, \gamma)$  donde

- $S$  es un conjunto finito de estados.
- $A$  es un conjunto finito de acciones.
- $r(s, a, s')$  es la recompensa recibida al ejecutar  $a \in A$  en  $s \in S$  y llegar a  $s' \in S$ .
- $p(s'|s, a)$  es la probabilidad de pasar de  $s \in S$  a  $s' \in S$  si se ejecuta  $a \in A$ .
- $\gamma \in (0, 1]$  es el factor de descuento (generalmente se usa  $\gamma = 0.99$ ).

### Ejemplo galletas:

Las probabilidades  $p(s'|s, a)$  modelan las dinámicas del mundo en que vive el agente.



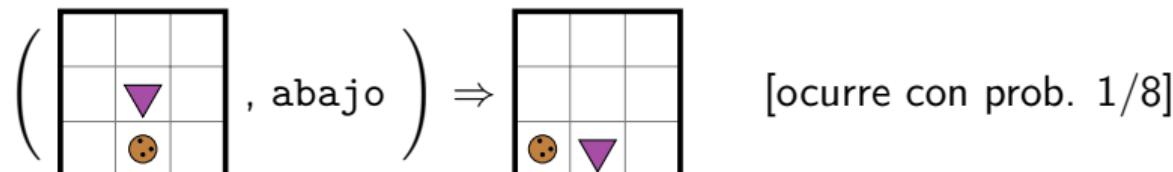
## Markov Decision Process (MDP)

Un MDP es una tupla  $\mathcal{M} = (S, A, r, p, \gamma)$  donde

- $S$  es un conjunto finito de estados.
- $A$  es un conjunto finito de acciones.
- $r(s, a, s')$  es la recompensa recibida al ejecutar  $a \in A$  en  $s \in S$  y llegar a  $s' \in S$ .
- $p(s'|s, a)$  es la probabilidad de pasar de  $s \in S$  a  $s' \in S$  si se ejecuta  $a \in A$ .
- $\gamma \in (0, 1]$  es el factor de descuento (generalmente se usa  $\gamma = 0.99$ ).

### Ejemplo galletas:

Las probabilidades  $p(s'|s, a)$  modelan las dinámicas del mundo en que vive el agente.



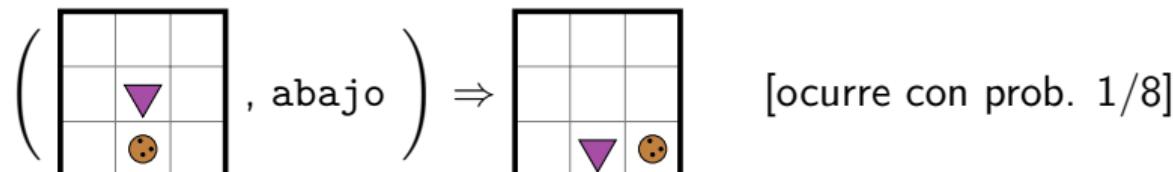
## Markov Decision Process (MDP)

Un MDP es una tupla  $\mathcal{M} = (S, A, r, p, \gamma)$  donde

- $S$  es un conjunto finito de estados.
- $A$  es un conjunto finito de acciones.
- $r(s, a, s')$  es la recompensa recibida al ejecutar  $a \in A$  en  $s \in S$  y llegar a  $s' \in S$ .
- $p(s'|s, a)$  es la probabilidad de pasar de  $s \in S$  a  $s' \in S$  si se ejecuta  $a \in A$ .
- $\gamma \in (0, 1]$  es el factor de descuento (generalmente se usa  $\gamma = 0.99$ ).

### Ejemplo galletas:

Las probabilidades  $p(s'|s, a)$  modelan las dinámicas del mundo en que vive el agente.



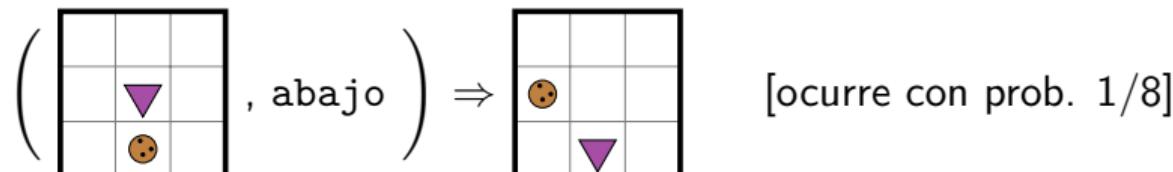
## Markov Decision Process (MDP)

Un MDP es una tupla  $\mathcal{M} = (S, A, r, p, \gamma)$  donde

- $S$  es un conjunto finito de estados.
- $A$  es un conjunto finito de acciones.
- $r(s, a, s')$  es la recompensa recibida al ejecutar  $a \in A$  en  $s \in S$  y llegar a  $s' \in S$ .
- $p(s'|s, a)$  es la probabilidad de pasar de  $s \in S$  a  $s' \in S$  si se ejecuta  $a \in A$ .
- $\gamma \in (0, 1]$  es el factor de descuento (generalmente se usa  $\gamma = 0.99$ ).

### Ejemplo galletas:

Las probabilidades  $p(s'|s, a)$  modelan las dinámicas del mundo en que vive el agente.



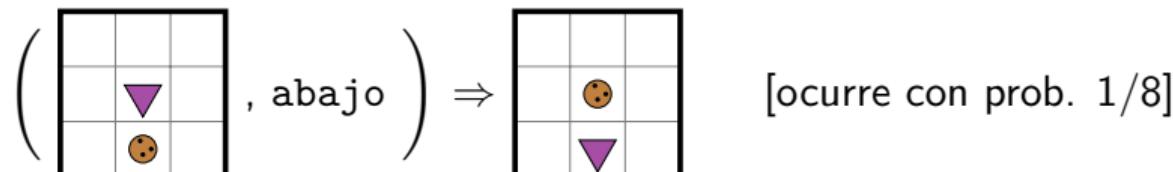
## Markov Decision Process (MDP)

Un MDP es una tupla  $\mathcal{M} = (S, A, r, p, \gamma)$  donde

- $S$  es un conjunto finito de estados.
- $A$  es un conjunto finito de acciones.
- $r(s, a, s')$  es la recompensa recibida al ejecutar  $a \in A$  en  $s \in S$  y llegar a  $s' \in S$ .
- $p(s'|s, a)$  es la probabilidad de pasar de  $s \in S$  a  $s' \in S$  si se ejecuta  $a \in A$ .
- $\gamma \in (0, 1]$  es el factor de descuento (generalmente se usa  $\gamma = 0.99$ ).

### Ejemplo galletas:

Las probabilidades  $p(s'|s, a)$  modelan las dinámicas del mundo en que vive el agente.



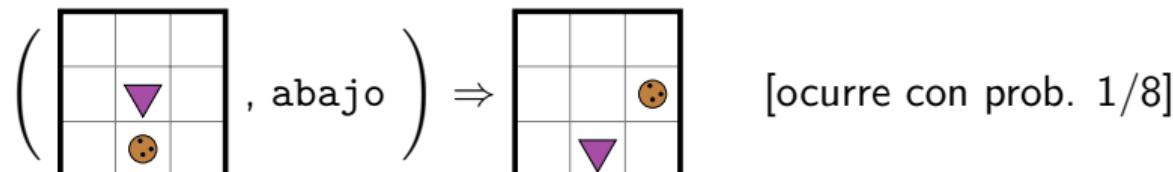
## Markov Decision Process (MDP)

Un MDP es una tupla  $\mathcal{M} = (S, A, r, p, \gamma)$  donde

- $S$  es un conjunto finito de estados.
- $A$  es un conjunto finito de acciones.
- $r(s, a, s')$  es la recompensa recibida al ejecutar  $a \in A$  en  $s \in S$  y llegar a  $s' \in S$ .
- $p(s'|s, a)$  es la probabilidad de pasar de  $s \in S$  a  $s' \in S$  si se ejecuta  $a \in A$ .
- $\gamma \in (0, 1]$  es el factor de descuento (generalmente se usa  $\gamma = 0.99$ ).

### Ejemplo galletas:

Las probabilidades  $p(s'|s, a)$  modelan las dinámicas del mundo en que vive el agente.



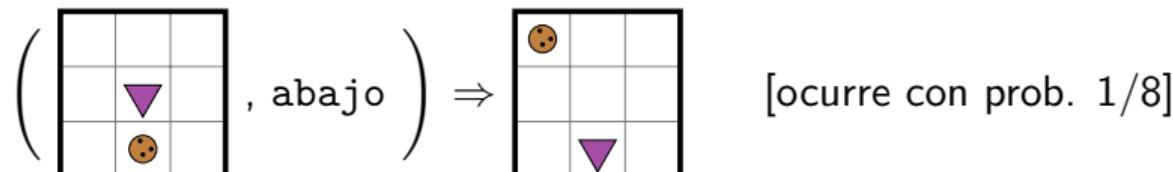
## Markov Decision Process (MDP)

Un MDP es una tupla  $\mathcal{M} = (S, A, r, p, \gamma)$  donde

- $S$  es un conjunto finito de estados.
- $A$  es un conjunto finito de acciones.
- $r(s, a, s')$  es la recompensa recibida al ejecutar  $a \in A$  en  $s \in S$  y llegar a  $s' \in S$ .
- $p(s'|s, a)$  es la probabilidad de pasar de  $s \in S$  a  $s' \in S$  si se ejecuta  $a \in A$ .
- $\gamma \in (0, 1]$  es el factor de descuento (generalmente se usa  $\gamma = 0.99$ ).

### Ejemplo galletas:

Las probabilidades  $p(s'|s, a)$  modelan las dinámicas del mundo en que vive el agente.



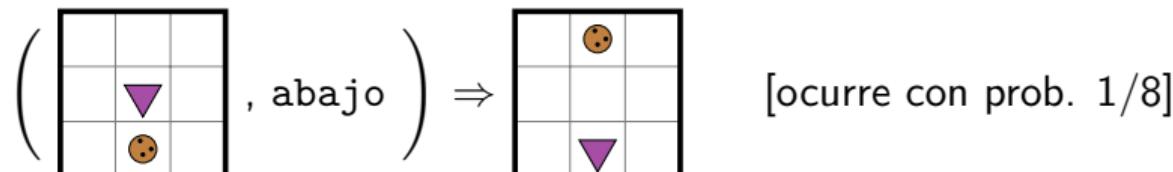
## Markov Decision Process (MDP)

Un MDP es una tupla  $\mathcal{M} = (S, A, r, p, \gamma)$  donde

- $S$  es un conjunto finito de estados.
- $A$  es un conjunto finito de acciones.
- $r(s, a, s')$  es la recompensa recibida al ejecutar  $a \in A$  en  $s \in S$  y llegar a  $s' \in S$ .
- $p(s'|s, a)$  es la probabilidad de pasar de  $s \in S$  a  $s' \in S$  si se ejecuta  $a \in A$ .
- $\gamma \in (0, 1]$  es el factor de descuento (generalmente se usa  $\gamma = 0.99$ ).

### Ejemplo galletas:

Las probabilidades  $p(s'|s, a)$  modelan las dinámicas del mundo en que vive el agente.



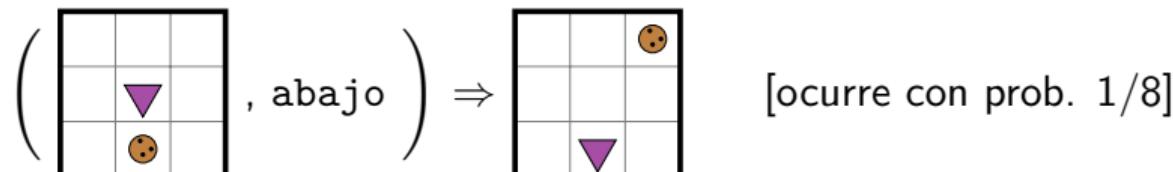
## Markov Decision Process (MDP)

Un MDP es una tupla  $\mathcal{M} = (S, A, r, p, \gamma)$  donde

- $S$  es un conjunto finito de estados.
- $A$  es un conjunto finito de acciones.
- $r(s, a, s')$  es la recompensa recibida al ejecutar  $a \in A$  en  $s \in S$  y llegar a  $s' \in S$ .
- $p(s'|s, a)$  es la probabilidad de pasar de  $s \in S$  a  $s' \in S$  si se ejecuta  $a \in A$ .
- $\gamma \in (0, 1]$  es el factor de descuento (generalmente se usa  $\gamma = 0.99$ ).

### Ejemplo galletas:

Las probabilidades  $p(s'|s, a)$  modelan las dinámicas del mundo en que vive el agente.



## Markov Decision Process (MDP)

Un MDP es una tupla  $\mathcal{M} = (S, A, r, p, \gamma)$  donde

- $S$  es un conjunto finito de estados.
- $A$  es un conjunto finito de acciones.
- $r(s, a, s')$  es la recompensa recibida al ejecutar  $a \in A$  en  $s \in S$  y llegar a  $s' \in S$ .
- $p(s'|s, a)$  es la probabilidad de pasar de  $s \in S$  a  $s' \in S$  si se ejecuta  $a \in A$ .
- $\gamma \in (0, 1]$  es el factor de descuento (generalmente se usa  $\gamma = 0.99$ ).

### Ejemplo galletas:

... y por ahora no nos preocupemos del factor de descuento :)

# Problemas de Decisión

En la práctica, hay que implementar 3 funciones:

# Problemas de Decisión

En la práctica, hay que implementar 3 funciones:

**1) action\_space()**: Retorna las acciones disponibles para el agente.

$$A = [\text{izquierda}, \text{derecha}, \text{arriba}, \text{abajo}]$$

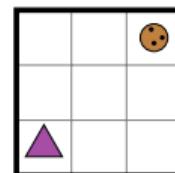
# Problemas de Decisión

En la práctica, hay que implementar 3 funciones:

**1)** `action_space()`: Retorna las acciones disponibles para el agente.

$$A = [\text{izquierda}, \text{derecha}, \text{arriba}, \text{abajo}]$$

**2)** `reset()`: Reinicia el ambiente y retorna el estado inicial.



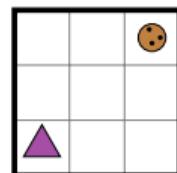
# Problemas de Decisión

En la práctica, hay que implementar 3 funciones:

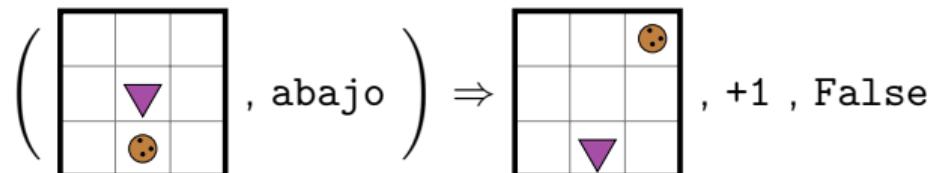
**1)** `action_space()`: Retorna las acciones disponibles para el agente.

$$A = [\text{izquierda}, \text{derecha}, \text{arriba}, \text{abajo}]$$

**2)** `reset()`: Reinicia el ambiente y retorna el estado inicial.



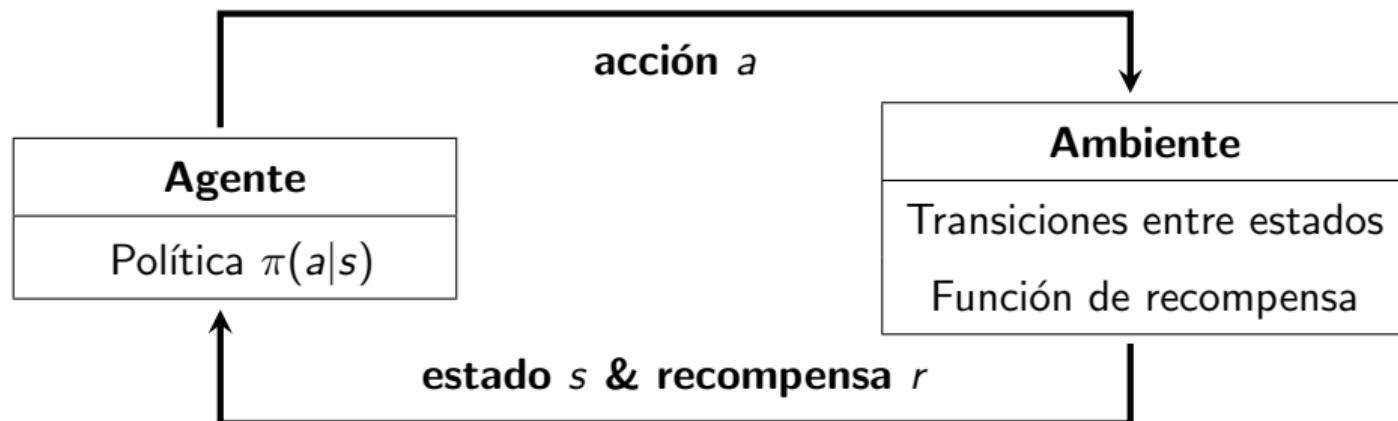
**3)** `step(s,a)`: Retorna el siguiente estado y recompensa luego de ejecutar la acción a en el estado s. Además indica si el episodio ha terminado.



# **Agentes de Aprendizaje Reforzado**

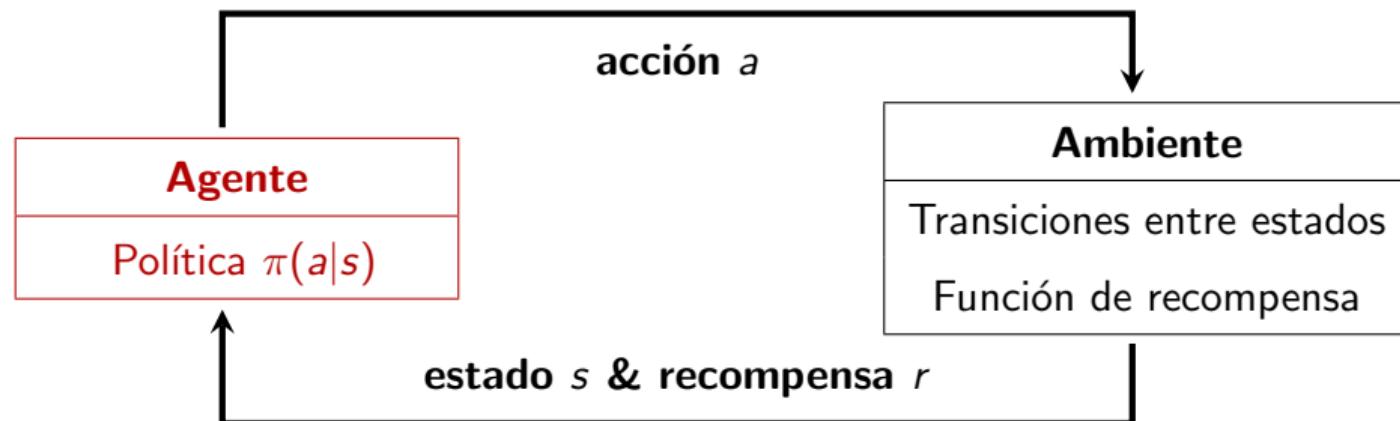
# Agentes de Aprendizaje Reforzado

Una vez definido el ambiente hay que hacer funcionar al agente.



# Agentes de Aprendizaje Reforzado

Una vez definido el ambiente hay que hacer funcionar al agente.



... pero ¿qué es y cómo funciona el agente?

# Agentes de Aprendizaje Reforzado

Todo agente intenta aprender una política *óptima*:

$$\pi^* = \arg \max_{\pi \in \Pi} \mathbb{E}_\pi \left[ \sum_{t=0}^{\infty} \gamma^t R_{t+1} \right]$$

# Agentes de Aprendizaje Reforzado

Todo agente intenta aprender una política *óptima*:

$$\pi^* = \arg \max_{\pi \in \Pi} \mathbb{E}_\pi \left[ \sum_{t=0}^{\infty} \gamma^t R_{t+1} \right]$$

... entendamos qué significa esta función objetivo.

# Agentes de Aprendizaje Reforzado

Todo agente intenta aprender una política *óptima*:

$$\pi^* = \arg \max_{\pi \in \Pi} \mathbb{E}_\pi \left[ \sum_{t=0}^{\infty} \gamma^t R_{t+1} \right]$$

... entendamos qué significa esta función objetivo.

Recordemos que una política  $\pi(a|s)$  simplemente indica la probabilidad con la que el agente elige la acción  $a$  en el estado  $s$ .

# Agentes de Aprendizaje Reforzado

Todo agente intenta aprender una política *óptima*:

$$\pi^* = \arg \max_{\pi \in \Pi} \mathbb{E}_\pi \left[ \sum_{t=0}^{\infty} \gamma^t R_{t+1} \right]$$

... entendamos qué significa esta función objetivo.

Recordemos que una política  $\pi(a|s)$  simplemente indica la probabilidad con la que el agente elige la acción  $a$  en el estado  $s$ . Para todo problema interesante, existirán políticas que son *mejores* que otras.

# Agentes de Aprendizaje Reforzado

Todo agente intenta aprender una política *óptima*:

$$\pi^* = \arg \max_{\pi \in \Pi} \mathbb{E}_{\pi} \left[ \sum_{t=0}^{\infty} \gamma^t R_{t+1} \right]$$

... entendamos qué significa esta función objetivo.

Recordemos que una política  $\pi(a|s)$  simplemente indica la probabilidad con la que el agente elige la acción  $a$  en el estado  $s$ . Para todo problema interesante, existirán políticas que son *mejores* que otras. En la ecuación, el término  $\arg \max_{\pi \in \Pi}$  hace referencia a encontrar **la mejor política** que existe dentro del universo de políticas posibles (representadas por  $\Pi$ ).

# Agentes de Aprendizaje Reforzado

Todo agente intenta aprender una política *óptima*:

$$\pi^* = \arg \max_{\pi \in \Pi} \mathbb{E}_\pi \left[ \sum_{t=0}^{\infty} \gamma^t R_{t+1} \right]$$

... entendamos qué significa esta función objetivo.

Recordemos que una política  $\pi(a|s)$  simplemente indica la probabilidad con la que el agente elige la acción  $a$  en el estado  $s$ . Para todo problema interesante, existirán políticas que son *mejores* que otras. En la ecuación, el término  $\arg \max_{\pi \in \Pi}$  hace referencia a encontrar **la mejor política** que existe dentro del universo de políticas posibles (representadas por  $\Pi$ ).

Entonces, el problema de fondo es cómo definir que una política es mejor que otra.

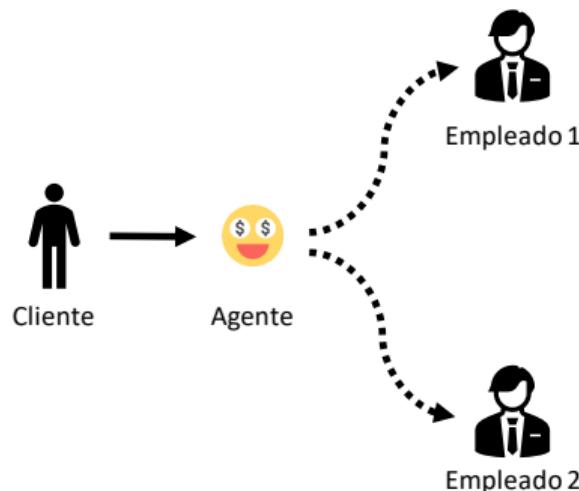
## Ejemplo

Una empresa tiene un módulo de servicio al cliente que cuenta con 2 empleados. Cada cliente que llega debe ser asignado a un empleado. Luego de ser atendidos, los clientes reportan su nivel de satisfacción dando entre 1 y 5 estrellas. El objetivo es asignar clientes a empleados tal que se maximice la satisfacción total de los clientes.

# Agentes de Aprendizaje Reforzado

## Ejemplo

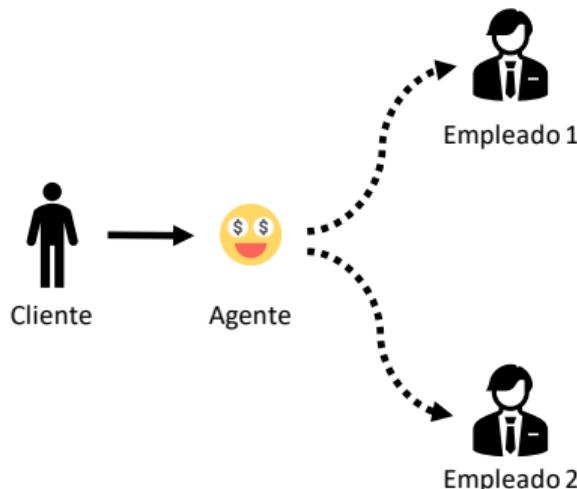
Una empresa tiene un módulo de servicio al cliente que cuenta con 2 empleados. Cada cliente que llega debe ser asignado a un empleado. Luego de ser atendidos, los clientes reportan su nivel de satisfacción dando entre 1 y 5 estrellas. El objetivo es asignar clientes a empleados tal que se maximice la satisfacción total de los clientes.



# Agentes de Aprendizaje Reforzado

## Ejemplo

Una empresa tiene un módulo de servicio al cliente que cuenta con 2 empleados. Cada cliente que llega debe ser asignado a un empleado. Luego de ser atendidos, los clientes reportan su nivel de satisfacción dando entre 1 y 5 estrellas. El objetivo es asignar clientes a empleados tal que se maximice la satisfacción total de los clientes.



### Experiencia:

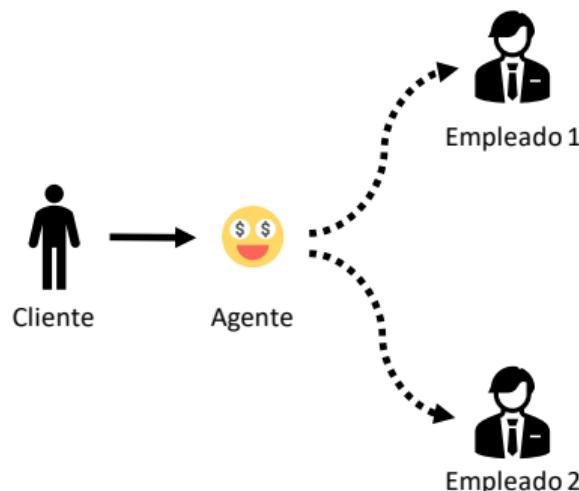
- Asignamos un cliente a  $e_1$  y obtuvimos 2 estrellas.
- Asignamos un cliente a  $e_2$  y obtuvimos 3 estrellas.

¿Es esto suficiente para inferir que  $\pi_1(e_1|c) < \pi_2(e_2|c)$ ?

# Agentes de Aprendizaje Reforzado

## Ejemplo

Una empresa tiene un módulo de servicio al cliente que cuenta con 2 empleados. Cada cliente que llega debe ser asignado a un empleado. Luego de ser atendidos, los clientes reportan su nivel de satisfacción dando entre 1 y 5 estrellas. El objetivo es asignar clientes a empleados tal que se maximice la satisfacción total de los clientes.



Digamos que obtenemos datos de 1000 clientes atendidos por  $e_1$  y  $e_2$  y calculamos su satisfacción:

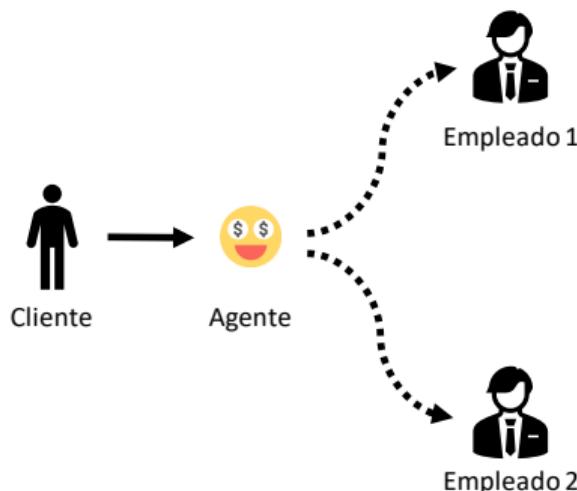
- Satisfacción clientes de  $e_1$  es **4.12** en promedio.
- Satisfacción clientes de  $e_2$  es **3.04** en promedio.

¿Es esto suficiente para inferir que  $\pi_1(e_1|c) > \pi_2(e_2|c)$ ?

# Agentes de Aprendizaje Reforzado

## Ejemplo

Una empresa tiene un módulo de servicio al cliente que cuenta con 2 empleados. Cada cliente que llega debe ser asignado a un empleado. Luego de ser atendidos, los clientes reportan su nivel de satisfacción dando entre 1 y 5 estrellas. El objetivo es asignar clientes a empleados tal que se maximice la satisfacción total de los clientes.



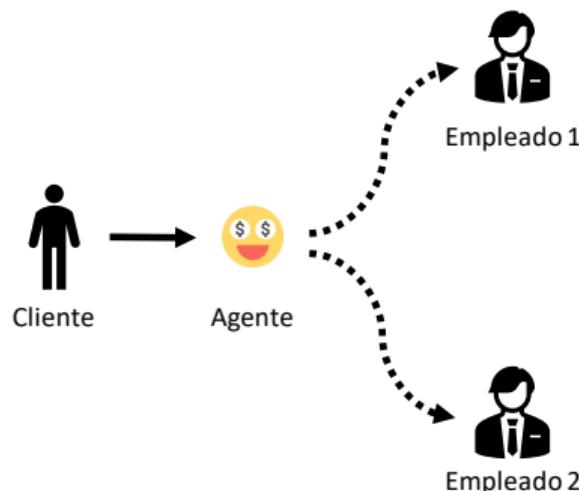
Ahora consideremos una versión simplificada de la ecuación de una política óptima:

$$\pi^* = \arg \max_{\pi \in \Pi} \mathbb{E}_{\pi}[R_{t+1}]$$

# Agentes de Aprendizaje Reforzado

## Ejemplo

Una empresa tiene un módulo de servicio al cliente que cuenta con 2 empleados. Cada cliente que llega debe ser asignado a un empleado. Luego de ser atendidos, los clientes reportan su nivel de satisfacción dando entre 1 y 5 estrellas. El objetivo es asignar clientes a empleados tal que se maximice la satisfacción total de los clientes.

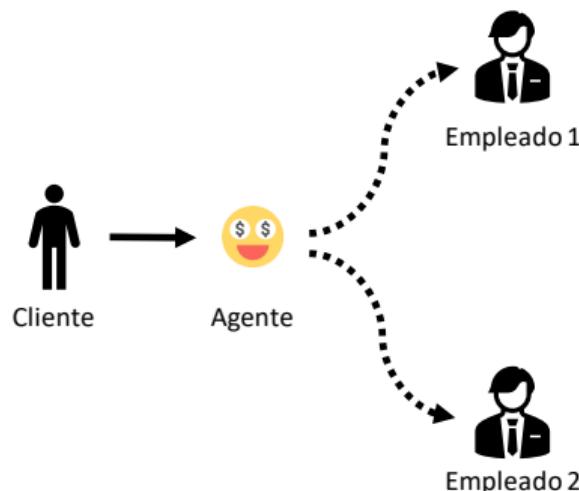


Sin embargo, el objetivo **no** es maximizar la recompensa de **un** cliente, sino que maximizar la recompensa de los  $n$  clientes que llegan a la sucursal.

# Agentes de Aprendizaje Reforzado

## Ejemplo

Una empresa tiene un módulo de servicio al cliente que cuenta con 2 empleados. Cada cliente que llega debe ser asignado a un empleado. Luego de ser atendidos, los clientes reportan su nivel de satisfacción dando entre 1 y 5 estrellas. El objetivo es asignar clientes a empleados tal que se maximice la satisfacción total de los clientes.



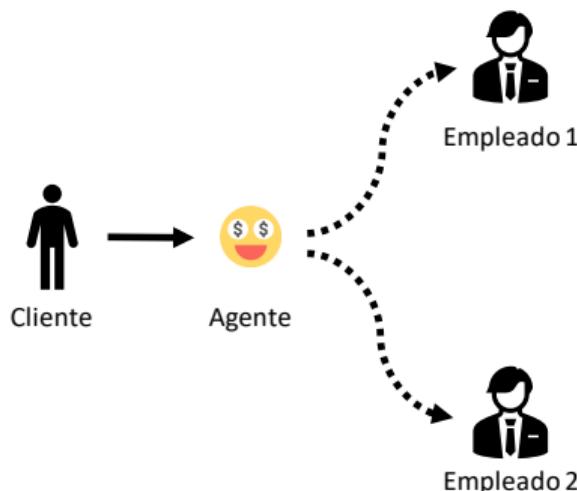
Sin embargo, el objetivo **no** es maximizar la recompensa de **un** cliente, sino que maximizar la recompensa de los  $n$  clientes que llegan a la sucursal.

$$\pi^* = \arg \max_{\pi \in \Pi} \mathbb{E}_{\pi}[R_{t+1}] \Rightarrow \pi^* = \arg \max_{\pi \in \Pi} \mathbb{E}_{\pi} \left[ \sum_{t=0}^n R_{t+1} \right]$$

# Agentes de Aprendizaje Reforzado

## Ejemplo

Una empresa tiene un módulo de servicio al cliente que cuenta con 2 empleados. Cada cliente que llega debe ser asignado a un empleado. Luego de ser atendidos, los clientes reportan su nivel de satisfacción dando entre 1 y 5 estrellas. El objetivo es asignar clientes a empleados tal que se maximice la satisfacción total de los clientes.



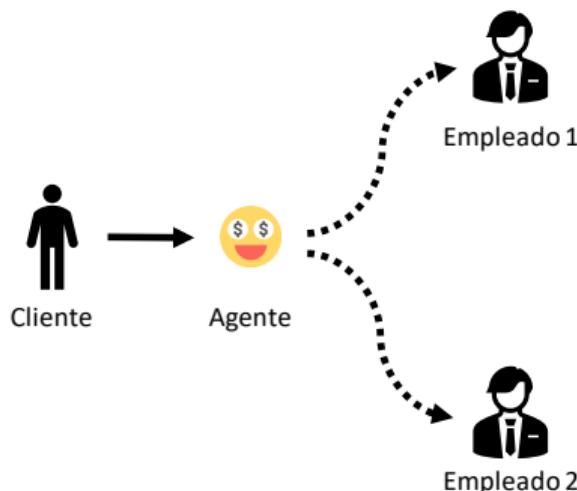
En la definición general se asume que queremos maximizar la recompensa recibida hasta el infinito.

$$\pi^* = \arg \max_{\pi \in \Pi} \mathbb{E}_{\pi} \left[ \sum_{t=0}^{\infty} R_{t+1} \right]$$

# Agentes de Aprendizaje Reforzado

## Ejemplo

Una empresa tiene un módulo de servicio al cliente que cuenta con 2 empleados. Cada cliente que llega debe ser asignado a un empleado. Luego de ser atendidos, los clientes reportan su nivel de satisfacción dando entre 1 y 5 estrellas. El objetivo es asignar clientes a empleados tal que se maximice la satisfacción total de los clientes.



En la definición general se asume que queremos maximizar la recompensa recibida hasta el infinito.

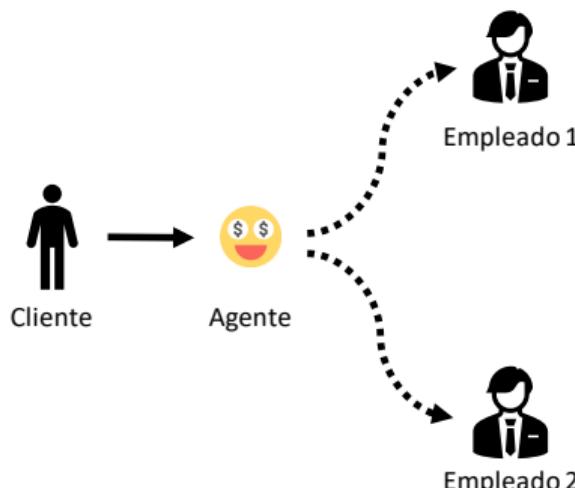
$$\pi^* = \arg \max_{\pi \in \Pi} \mathbb{E}_{\pi} \left[ \sum_{t=0}^{\infty} R_{t+1} \right]$$

El problema es que la suma anterior puede *divergir*.

# Agentes de Aprendizaje Reforzado

## Ejemplo

Una empresa tiene un módulo de servicio al cliente que cuenta con 2 empleados. Cada cliente que llega debe ser asignado a un empleado. Luego de ser atendidos, los clientes reportan su nivel de satisfacción dando entre 1 y 5 estrellas. El objetivo es asignar clientes a empleados tal que se maximice la satisfacción total de los clientes.



En la definición general se asume que queremos maximizar la recompensa recibida hasta el infinito.

$$\pi^* = \arg \max_{\pi \in \Pi} \mathbb{E}_{\pi} \left[ \sum_{t=0}^{\infty} R_{t+1} \right]$$

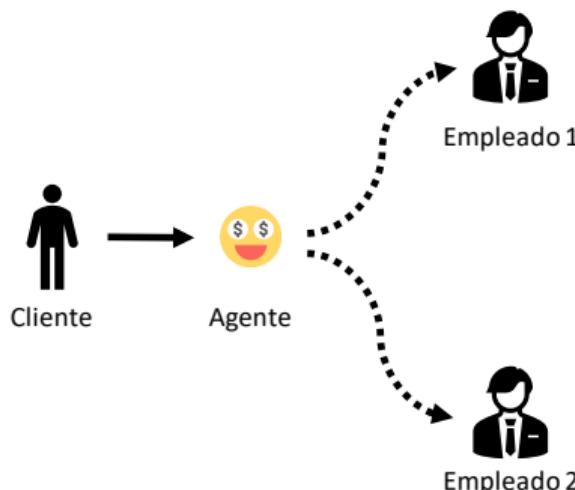
E.j., si  $\pi_{5*}$  siempre obtiene 5 estrella:

$$\sum_{t=0}^{\infty} R_{t+1} = 5 + 5 + 5 + 5 + 5 + 5 + \dots = \infty$$

# Agentes de Aprendizaje Reforzado

## Ejemplo

Una empresa tiene un módulo de servicio al cliente que cuenta con 2 empleados. Cada cliente que llega debe ser asignado a un empleado. Luego de ser atendidos, los clientes reportan su nivel de satisfacción dando entre 1 y 5 estrellas. El objetivo es asignar clientes a empleados tal que se maximice la satisfacción total de los clientes.



En la definición general se asume que queremos maximizar la recompensa recibida hasta el infinito.

$$\pi^* = \arg \max_{\pi \in \Pi} \mathbb{E}_{\pi} \left[ \sum_{t=0}^{\infty} R_{t+1} \right]$$

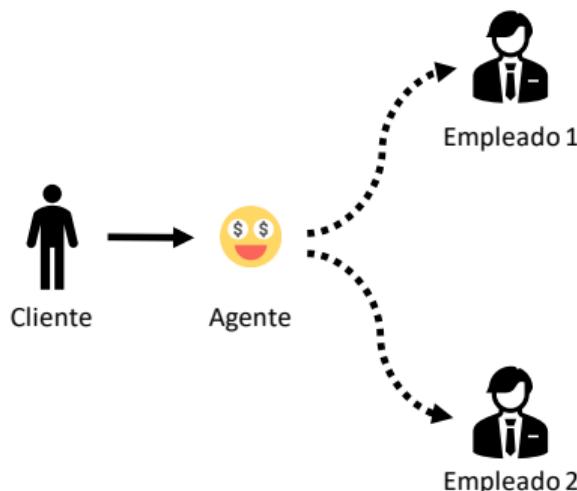
E.j., si  $\pi_{1*}$  siempre obtiene 1 estrella:

$$\sum_{t=0}^{\infty} R_{t+1} = 1 + 1 + 1 + 1 + 1 + 1 + \dots = \infty$$

# Agentes de Aprendizaje Reforzado

## Ejemplo

Una empresa tiene un módulo de servicio al cliente que cuenta con 2 empleados. Cada cliente que llega debe ser asignado a un empleado. Luego de ser atendidos, los clientes reportan su nivel de satisfacción dando entre 1 y 5 estrellas. El objetivo es asignar clientes a empleados tal que se maximice la satisfacción total de los clientes.



En la definición general se asume que queremos maximizar la recompensa recibida hasta el infinito.

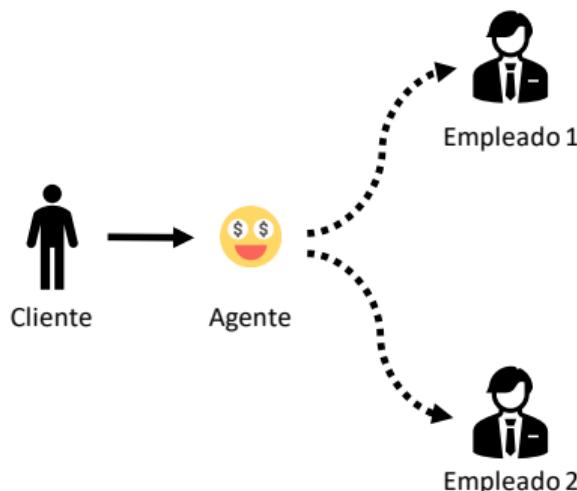
$$\pi^* = \arg \max_{\pi \in \Pi} \mathbb{E}_{\pi} \left[ \sum_{t=0}^{\infty} R_{t+1} \right]$$

Por lo tanto, **no** podemos inferir que  $\pi_{5*} > \pi_{1*}$  :(

# Agentes de Aprendizaje Reforzado

## Ejemplo

Una empresa tiene un módulo de servicio al cliente que cuenta con 2 empleados. Cada cliente que llega debe ser asignado a un empleado. Luego de ser atendidos, los clientes reportan su nivel de satisfacción dando entre 1 y 5 estrellas. El objetivo es asignar clientes a empleados tal que se maximice la satisfacción total de los clientes.



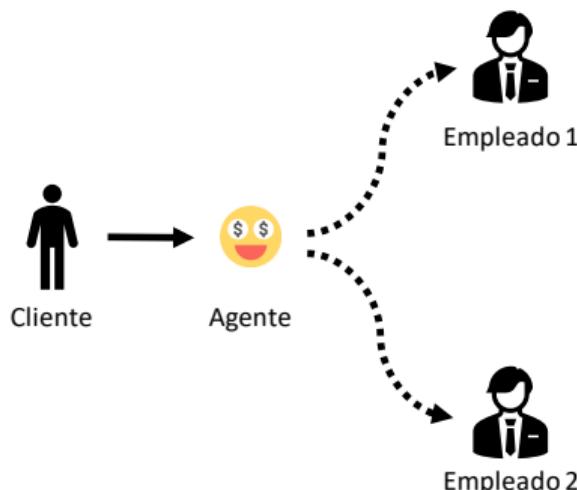
$\gamma \in (0, 1)$  se introduce para evitar que la suma diverja.

$$\pi^* = \arg \max_{\pi \in \Pi} \mathbb{E}_\pi \left[ \sum_{t=0}^{\infty} \gamma^t R_{t+1} \right]$$

# Agentes de Aprendizaje Reforzado

## Ejemplo

Una empresa tiene un módulo de servicio al cliente que cuenta con 2 empleados. Cada cliente que llega debe ser asignado a un empleado. Luego de ser atendidos, los clientes reportan su nivel de satisfacción dando entre 1 y 5 estrellas. El objetivo es asignar clientes a empleados tal que se maximice la satisfacción total de los clientes.



$\gamma \in (0, 1)$  se introduce para evitar que la suma diverja.

$$\pi^* = \arg \max_{\pi \in \Pi} \mathbb{E}_{\pi} \left[ \sum_{t=0}^{\infty} \gamma^t R_{t+1} \right]$$

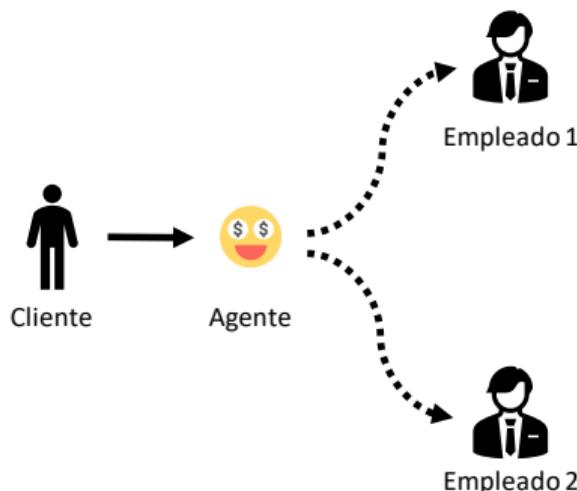
E.j., si  $\gamma = 0.99$  y  $\pi_{5*}$  siempre obtiene 5 de recompensa:

$$\sum_{t=0}^{\infty} \gamma^t R_{t+1} = 5 + 5 \cdot 0.99 + 5 \cdot 0.99^2 + \dots = 500$$

# Agentes de Aprendizaje Reforzado

## Ejemplo

Una empresa tiene un módulo de servicio al cliente que cuenta con 2 empleados. Cada cliente que llega debe ser asignado a un empleado. Luego de ser atendidos, los clientes reportan su nivel de satisfacción dando entre 1 y 5 estrellas. El objetivo es asignar clientes a empleados tal que se maximice la satisfacción total de los clientes.



$\gamma \in (0, 1)$  se introduce para evitar que la suma diverja.

$$\pi^* = \arg \max_{\pi \in \Pi} \mathbb{E}_{\pi} \left[ \sum_{t=0}^{\infty} \gamma^t R_{t+1} \right]$$

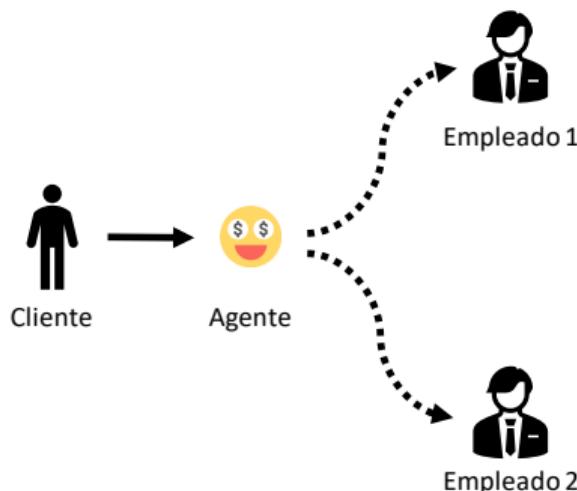
E.j., si  $\gamma = 0.99$  y  $\pi_{1*}$  siempre obtiene 1 de recompensa:

$$\sum_{t=0}^{\infty} \gamma^t R_{t+1} = 1 + 0.99 + 0.99^2 + 0.99^3 + \dots = 100$$

# Agentes de Aprendizaje Reforzado

## Ejemplo

Una empresa tiene un módulo de servicio al cliente que cuenta con 2 empleados. Cada cliente que llega debe ser asignado a un empleado. Luego de ser atendidos, los clientes reportan su nivel de satisfacción dando entre 1 y 5 estrellas. El objetivo es asignar clientes a empleados tal que se maximice la satisfacción total de los clientes.



$\gamma \in (0, 1)$  se introduce para evitar que la suma diverja.

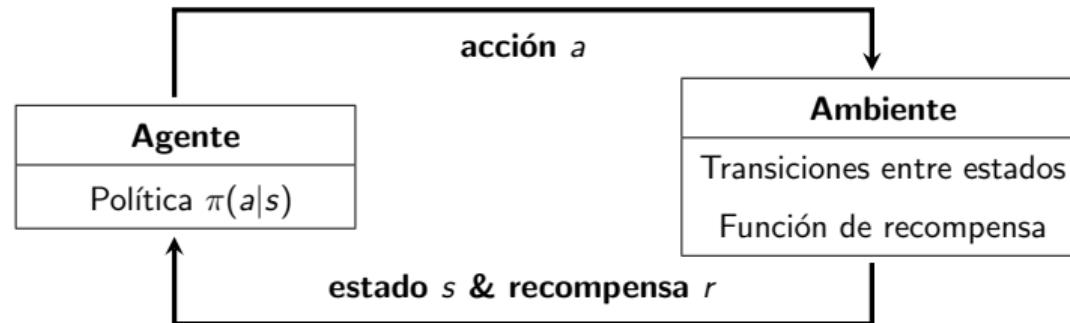
$$\pi^* = \arg \max_{\pi \in \Pi} \mathbb{E}_{\pi} \left[ \sum_{t=0}^{\infty} \gamma^t R_{t+1} \right]$$

Por lo tanto, agregar  $\gamma$  nos permite inferir que  $\pi_{5*} > \pi_{1*}$

# **Agentes de Aprendizaje Reforzado (recapitulando)**

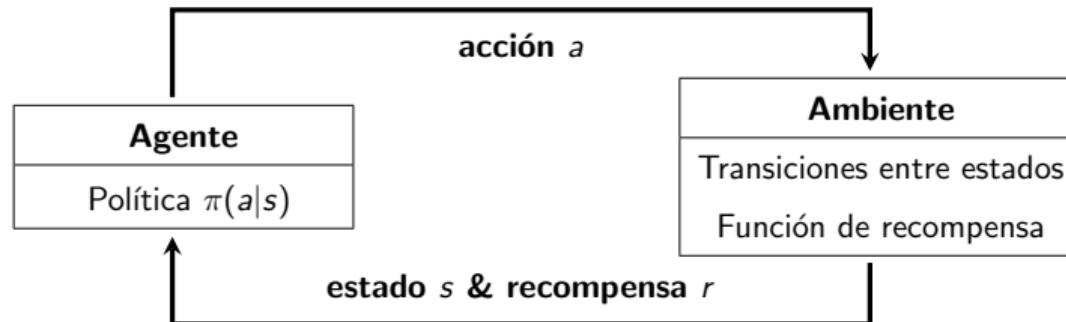
# Agentes de Aprendizaje Reforzado

**Objetivo:** Mediante interactuar con un ambiente



# Agentes de Aprendizaje Reforzado

**Objetivo:** Mediante interactuar con un ambiente

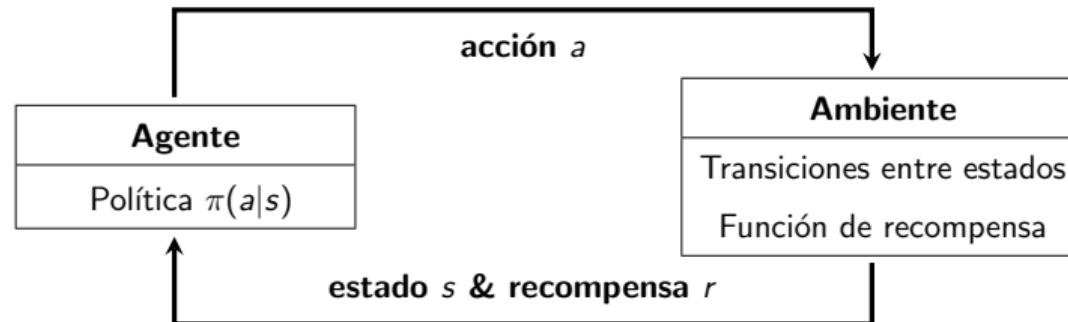


Queremos encontrar una política óptima  $\pi^*$ :

$$\pi^* = \arg \max_{\pi \in \Pi} \mathbb{E}_{\pi} \left[ \sum_{t=0}^{\infty} \gamma^t R_{t+1} \right]$$

# Agentes de Aprendizaje Reforzado

**Objetivo:** Mediante interactuar con un ambiente



Queremos encontrar una política óptima  $\pi^*$ :

$$\pi^* = \arg \max_{\pi \in \Pi} \mathbb{E}_{\pi} \left[ \sum_{t=0}^{\infty} \gamma^t R_{t+1} \right]$$

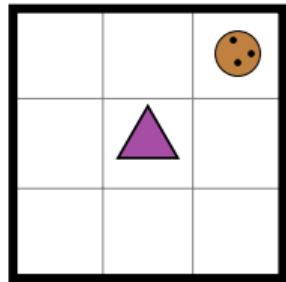
¿Cómo lo hacemos?

# Agentes de Aprendizaje Reforzado

Enfoquemos la discusión en un problema más pequeño.

# Agentes de Aprendizaje Reforzado

Enfoquemos la discusión en un problema más pequeño.

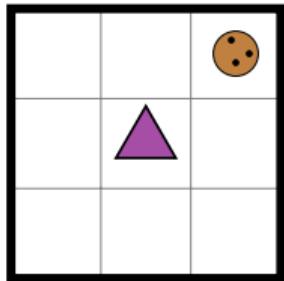


**En esta situación:**

- ¿Qué tan buena es la acción derecha?

# Agentes de Aprendizaje Reforzado

Enfoquemos la discusión en un problema más pequeño.

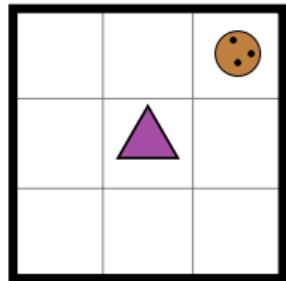


**En esta situación:**

- ¿Qué tan buena es la acción derecha?
- ¿Es mejor la acción izquierda o derecha? ¿Por qué?

# Agentes de Aprendizaje Reforzado

Enfoquemos la discusión en un problema más pequeño.

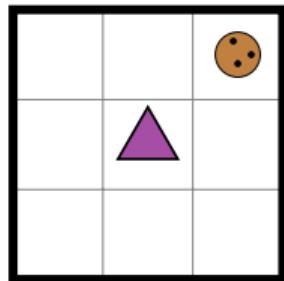


**En esta situación:**

- ¿Qué tan buena es la acción derecha?
- ¿Es mejor la acción izquierda o derecha? ¿Por qué?
- ¿Cómo podemos definir esta intuición matemáticamente?

# Agentes de Aprendizaje Reforzado

Enfoquemos la discusión en un problema más pequeño.



**En esta situación:**

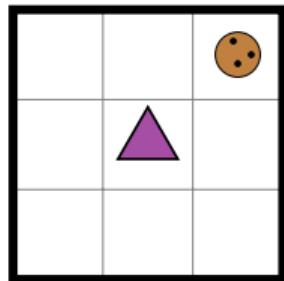
- ¿Qué tan buena es la acción derecha?
- ¿Es mejor la acción izquierda o derecha? ¿Por qué?
- ¿Cómo podemos definir esta intuición matemáticamente?

Los Q-values nos permiten definir qué *tan buena* es una acción en cierto estado.

$$q_*(s, a) = \max_{\pi \in \Pi} \mathbb{E}_{\pi} \left[ \sum_{t=0}^{\infty} \gamma^t R_{t+1} \middle| S_0 = s, A_0 = a \right]$$

# Agentes de Aprendizaje Reforzado

Enfoquemos la discusión en un problema más pequeño.



**En esta situación:**

- ¿Qué tan buena es la acción derecha?
- ¿Es mejor la acción izquierda o derecha? ¿Por qué?
- ¿Cómo podemos definir esta intuición matemáticamente?

Los Q-values nos permiten definir qué *tan buena* es una acción en cierto estado.

$$q_*(s, a) = \max_{\pi \in \Pi} \mathbb{E}_{\pi} \left[ \sum_{t=0}^{\infty} \gamma^t R_{t+1} \middle| S_0 = s, A_0 = a \right]$$

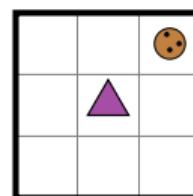
... para ilustrar esta idea consideremos una variante del problema de la galleta.

# Agentes de Aprendizaje Reforzado

**Definición:**

$$q_*(s, a) = \max_{\pi \in \Pi} \mathbb{E}_{\pi} \left[ \sum_{t=0}^{\infty} \gamma^t R_{t+1} \middle| S_0 = s, A_0 = a \right]$$

Si ejecutamos la acción derecha y luego seguimos una política óptima:

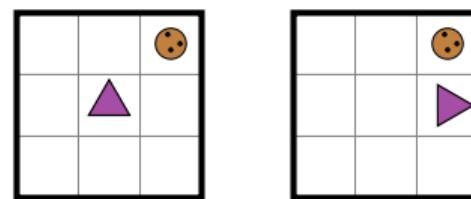


# Agentes de Aprendizaje Reforzado

## Definición:

$$q_*(s, a) = \max_{\pi \in \Pi} \mathbb{E}_{\pi} \left[ \sum_{t=0}^{\infty} \gamma^t R_{t+1} \middle| S_0 = s, A_0 = a \right]$$

Si ejecutamos la acción derecha y luego seguimos una política óptima:



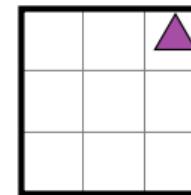
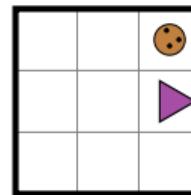
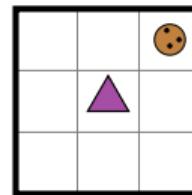
$$R_1 = 0$$

# Agentes de Aprendizaje Reforzado

**Definición:**

$$q_*(s, a) = \max_{\pi \in \Pi} \mathbb{E}_{\pi} \left[ \sum_{t=0}^{\infty} \gamma^t R_{t+1} \middle| S_0 = s, A_0 = a \right]$$

Si ejecutamos la acción derecha y luego seguimos una política óptima:



$$R_1 = 0$$

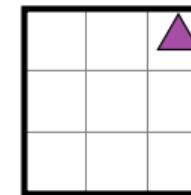
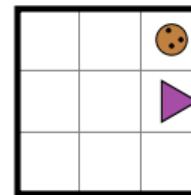
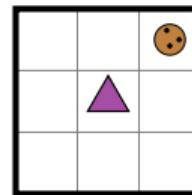
$$R_2 = 1$$

# Agentes de Aprendizaje Reforzado

## Definición:

$$q_*(s, a) = \max_{\pi \in \Pi} \mathbb{E}_{\pi} \left[ \sum_{t=0}^{\infty} \gamma^t R_{t+1} \middle| S_0 = s, A_0 = a \right]$$

Si ejecutamos la acción derecha y luego seguimos una política óptima:



$$R_1 = 0$$

$$R_2 = 1$$

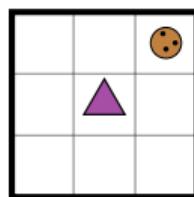
$$q_*(s, \text{derecha}) = R_1 + \gamma \cdot R_2 + \sum_{t=2}^{\infty} \gamma^t R_{t+1} = 0 + \gamma \cdot 1 = \gamma = 0.99.$$

# Agentes de Aprendizaje Reforzado

**Definición:**

$$q_*(s, a) = \max_{\pi \in \Pi} \mathbb{E}_{\pi} \left[ \sum_{t=0}^{\infty} \gamma^t R_{t+1} \middle| S_0 = s, A_0 = a \right]$$

Si ejecutamos la acción izquierda y luego seguimos una política óptima:

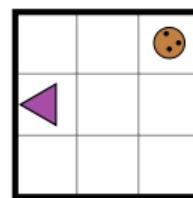
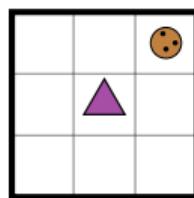


# Agentes de Aprendizaje Reforzado

**Definición:**

$$q_*(s, a) = \max_{\pi \in \Pi} \mathbb{E}_{\pi} \left[ \sum_{t=0}^{\infty} \gamma^t R_{t+1} \middle| S_0 = s, A_0 = a \right]$$

Si ejecutamos la acción izquierda y luego seguimos una política óptima:



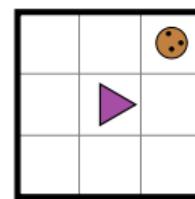
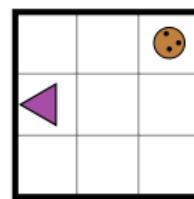
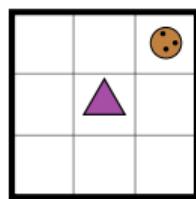
$$R_1 = 0$$

# Agentes de Aprendizaje Reforzado

**Definición:**

$$q_*(s, a) = \max_{\pi \in \Pi} \mathbb{E}_{\pi} \left[ \sum_{t=0}^{\infty} \gamma^t R_{t+1} \middle| S_0 = s, A_0 = a \right]$$

Si ejecutamos la acción izquierda y luego seguimos una política óptima:



$$R_1 = 0$$

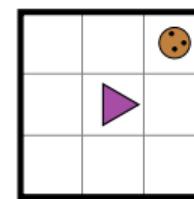
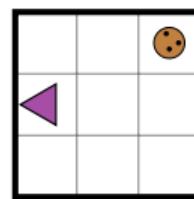
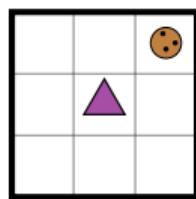
$$R_2 = 0$$

# Agentes de Aprendizaje Reforzado

**Definición:**

$$q_*(s, a) = \max_{\pi \in \Pi} \mathbb{E}_{\pi} \left[ \sum_{t=0}^{\infty} \gamma^t R_{t+1} \middle| S_0 = s, A_0 = a \right]$$

Si ejecutamos la acción izquierda y luego seguimos una política óptima:



$$R_1 = 0$$

$$R_2 = 0$$

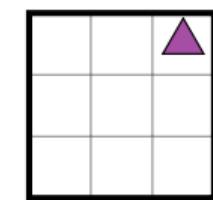
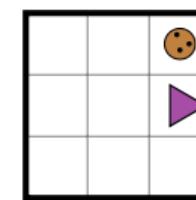
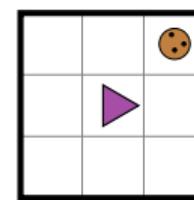
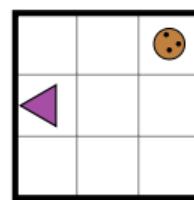
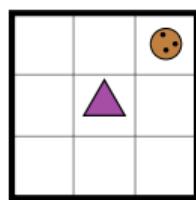
$$R_3 = 0$$

# Agentes de Aprendizaje Reforzado

**Definición:**

$$q_*(s, a) = \max_{\pi \in \Pi} \mathbb{E}_{\pi} \left[ \sum_{t=0}^{\infty} \gamma^t R_{t+1} \middle| S_0 = s, A_0 = a \right]$$

Si ejecutamos la acción izquierda y luego seguimos una política óptima:



$$R_1 = 0$$

$$R_2 = 0$$

$$R_3 = 0$$

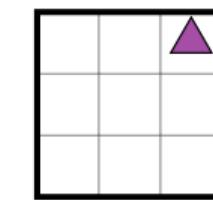
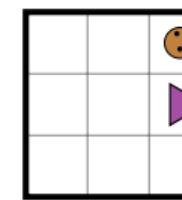
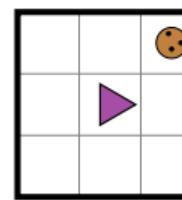
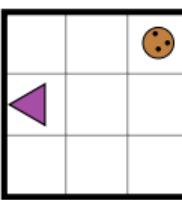
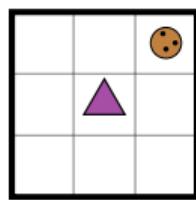
$$R_4 = 1$$

# Agentes de Aprendizaje Reforzado

**Definición:**

$$q_*(s, a) = \max_{\pi \in \Pi} \mathbb{E}_{\pi} \left[ \sum_{t=0}^{\infty} \gamma^t R_{t+1} \middle| S_0 = s, A_0 = a \right]$$

Si ejecutamos la acción izquierda y luego seguimos una política óptima:



$$R_1 = 0$$

$$R_2 = 0$$

$$R_3 = 0$$

$$R_4 = 1$$

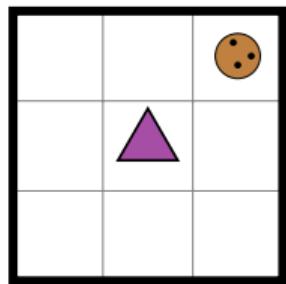
$$\begin{aligned} q_*(s, \text{izquierda}) &= R_1 + \gamma \cdot R_2 + \gamma^2 \cdot R_3 + \gamma^3 \cdot R_4 + \sum_{t=4}^{\infty} \gamma^t R_{t+1} \\ &= 0 + \gamma \cdot 0 + \gamma^2 \cdot 0 + \gamma^3 \cdot 1 = \gamma^3 = 0.99^3 \approx 0.97. \end{aligned}$$

# Agentes de Aprendizaje Reforzado

Acabamos de demostrar los siguiente:

# Agentes de Aprendizaje Reforzado

Acabamos de demostrar los siguiente:

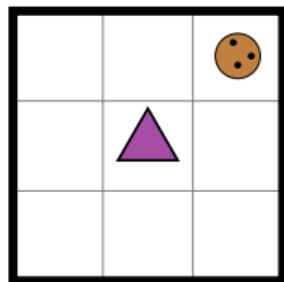


**En esta situación:**

- $q_*(s, \text{derecha}) = 0.99$
- $q_*(s, \text{izquierda}) = 0.97$

# Agentes de Aprendizaje Reforzado

Acabamos de demostrar los siguiente:



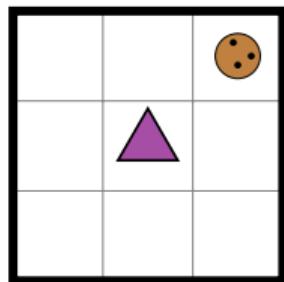
**En esta situación:**

- $q_*(s, \text{derecha}) = 0.99$
- $q_*(s, \text{izquierda}) = 0.97$

Como  $q_*(s, \text{derecha}) > q_*(s, \text{izquierda})$ , podemos inferir que la acción derecha es *mejor* que la acción izquierda.

# Agentes de Aprendizaje Reforzado

Acabamos de demostrar los siguiente:



**En esta situación:**

- $q_*(s, \text{derecha}) = 0.99$
- $q_*(s, \text{izquierda}) = 0.97$

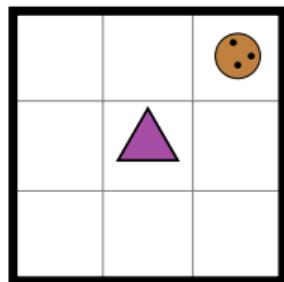
Como  $q_*(s, \text{derecha}) > q_*(s, \text{izquierda})$ , podemos inferir que la acción derecha es *mejor* que la acción izquierda.

De hecho, si conocemos  $q_*$  podemos extraer una política óptima de la siguiente forma:

$$\pi^*(a|s) = \arg \max_{a \in A} q_*(s, a)$$

# Agentes de Aprendizaje Reforzado

Acabamos de demostrar los siguiente:



**En esta situación:**

- $q_*(s, \text{derecha}) = 0.99$
- $q_*(s, \text{izquierda}) = 0.97$

Como  $q_*(s, \text{derecha}) > q_*(s, \text{izquierda})$ , podemos inferir que la acción derecha es *mejor* que la acción izquierda.

De hecho, si conocemos  $q_*$  podemos extraer una política óptima de la siguiente forma:

$$\pi^*(a|s) = \arg \max_{a \in A} q_*(s, a)$$

... entonces, para encontrar  $\pi^*$  podemos intentar aprender  $q_*(s, a)$ .

# **Q-Learning**

# Q-Learning

Los Q-values óptimos cumplen con la siguiente propiedad (si el ambiente es un MDP):

$$q_*(s, a) = \sum_{s' \in S} p(s'|s, a) \left[ r(s, a, s') + \gamma \max_{a' \in A} q_*(s', a') \right] \quad \text{para todo } s \in S, a \in A.$$

# Q-Learning

Los Q-values óptimos cumplen con la siguiente propiedad (si el ambiente es un MDP):

$$q_*(s, a) = \sum_{s' \in S} p(s'|s, a) \left[ r(s, a, s') + \gamma \max_{a' \in A} q_*(s', a') \right] \quad \text{para todo } s \in S, a \in A.$$

Es decir, si de alguna forma encontramos un conjunto de valores  $Q(s, a)$  tal que:

$$Q(s, a) = \sum_{s' \in S} p(s'|s, a) \left[ r(s, a, s') + \gamma \max_{a' \in A} Q(s', a') \right] \quad \text{para todo } s \in S, a \in A.$$

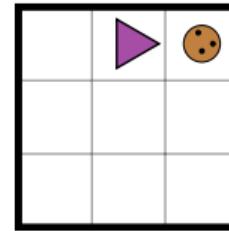
... entonces  $Q$  es igual a  $q_*$  (y de  $Q$  podemos extraer  $\pi^*$ ).

# Q-Learning

$s$	$a$	$Q(s, a)$
...	...	...
	arriba	0.04
	derecha	0.1
	abajo	0.06
	izquierda	0.01
...	...	...
	arriba	0.0
	derecha	-0.9
	abajo	0.0
	izquierda	-0.1
...	...	...

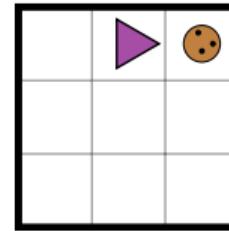
# Q-Learning

$s$	$a$	$Q(s, a)$
...	...	...
	arriba	0.04
	derecha	0.1
	abajo	0.06
	izquierda	0.01
...	...	...
	arriba	0.0
	derecha	-0.9
	abajo	0.0
	izquierda	-0.1
...	...	...



# Q-Learning

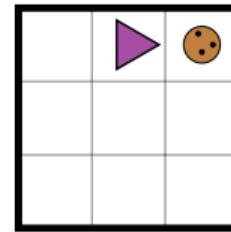
$s$	$a$	$Q(s, a)$
...	...	...
	arriba	0.04
	derecha	0.1
	abajo	0.06
	izquierda	0.01
...	...	...
	arriba	0.0
	derecha	-0.9
	abajo	0.0
	izquierda	-0.1
...	...	...



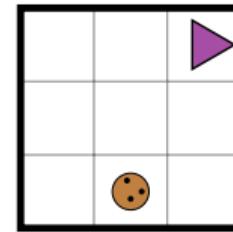
derecha

# Q-Learning

$s$	$a$	$Q(s, a)$
...	...	...
	arriba	0.04
	derecha	0.1
	abajo	0.06
	izquierda	0.01
...	...	...
	arriba	0.0
	derecha	-0.9
	abajo	0.0
	izquierda	-0.1
...	...	...

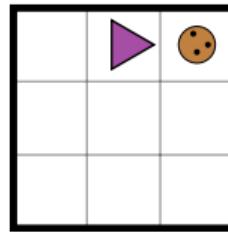


derecha +1

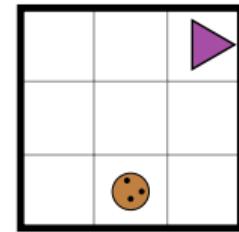


# Q-Learning

$s$	$a$	$Q(s, a)$
...	...	...
	arriba	0.04
	derecha	0.1
	abajo	0.06
	izquierda	0.01
...	...	...
	arriba	0.0
	derecha	-0.9
	abajo	0.0
	izquierda	-0.1
...	...	...



derecha +1

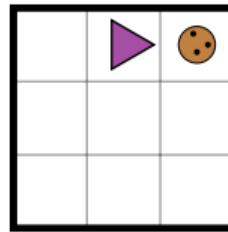


**Experiencia:**  $(s, a, r, s')$

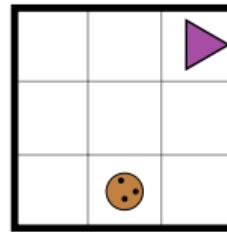
$$Q(s, a) \xleftarrow{\alpha} r + \gamma \max_{a'} Q(s', a')$$

# Q-Learning

$s$	$a$	$Q(s, a)$
...	...	...
	arriba	0.04
	derecha	0.1
	abajo	0.06
	izquierda	0.01
...	...	...
	arriba	0.0
	derecha	-0.9
	abajo	0.0
	izquierda	-0.1
...	...	...



derecha +1



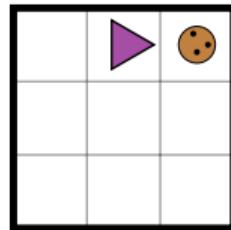
**Experiencia:**  $(s, a, r, s')$

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

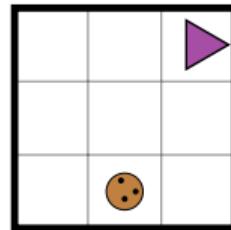
$$Q(s, a) = 0.1 + 0.7(1 - 0.1) = 0.73$$

# Q-Learning

$s$	$a$	$Q(s, a)$
...	...	...
	arriba	0.04
	derecha	0.1
	abajo	0.06
	izquierda	0.01
...	...	...
	arriba	0.0
	derecha	-0.9
	abajo	0.0
	izquierda	-0.1
...	...	...



derecha +1



**Experiencia:**  $(s, a, r, s')$

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

$$Q(s, a) = 0.1 + 0.7(1 - 0.1) = 0.73$$

El objetivo es lograr que  $Q(s, a)$  se parezca cada vez más a  $r + \gamma \max_{a'} Q(s', a')$ . Cuando sean iguales,  $Q = q_*$ .

# Q-Learning

Q-learning (off-policy TD control) for estimating  $\pi \approx \pi_*$

Algorithm parameters: step size  $\alpha \in (0, 1]$ , small  $\varepsilon > 0$

Initialize  $Q(s, a)$ , for all  $s \in \mathcal{S}^+, a \in \mathcal{A}(s)$ , arbitrarily except that  $Q(\text{terminal}, \cdot) = 0$

Loop for each episode:

    Initialize  $S$

    Loop for each step of episode:

        Choose  $A$  from  $S$  using policy derived from  $Q$  (e.g.,  $\varepsilon$ -greedy)

        Take action  $A$ , observe  $R, S'$

$$Q(S, A) \leftarrow Q(S, A) + \alpha [R + \gamma \max_a Q(S', a) - Q(S, A)]$$

$$S \leftarrow S'$$

    until  $S$  is terminal

(Source: "Reinforcement Learning: An Introduction" by Sutton and Barto, 2018)

# Q-Learning

Sin embargo, Q-learning tiene un problema que lo hace sumamente impráctico :(

# Q-Learning

Sin embargo, Q-learning tiene un problema que lo hace sumamente impráctico :(

$s$	$a$	$Q(s, a)$
...	...	...
	arriba	0.04
	derecha	0.1
	abajo	0.06
	izquierda	0.01
...	...	...

Para todo problema interesante esta tabla será **enorme!**

# Q-Learning

Sin embargo, Q-learning tiene un problema que lo hace sumamente impráctico :(

$s$	$a$	$Q(s, a)$
...	...	...
	arriba	0.04
	derecha	0.1
	abajo	0.06
	izquierda	0.01
...	...	...

Para todo problema interesante esta tabla será **enorme!**



# Q-Learning

Sin embargo, Q-learning tiene un problema que lo hace sumamente impráctico :(

$s$	$a$	$Q(s, a)$
...	...	...
	arriba	0.04
	derecha	0.1
	abajo	0.06
	izquierda	0.01
...	...	...

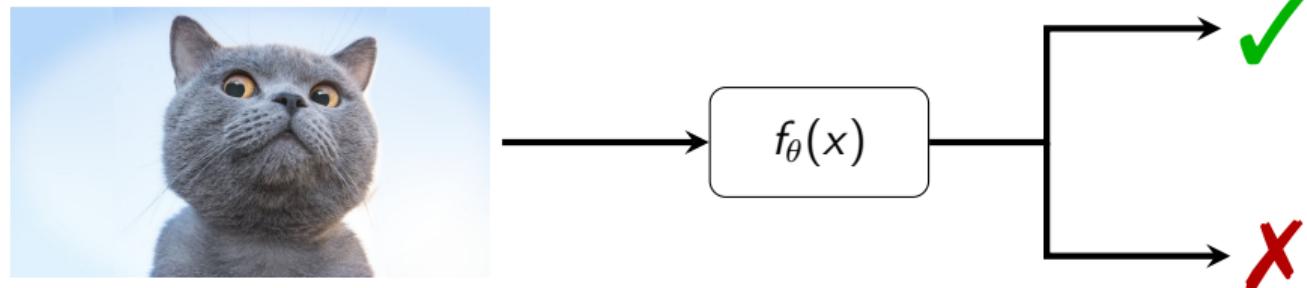
Para todo problema interesante esta tabla será **enorme!**



Este problema se solucionó usando deep learning :)

# Aprendizaje Supervisado (repaso)

# Aprendizaje Supervisado

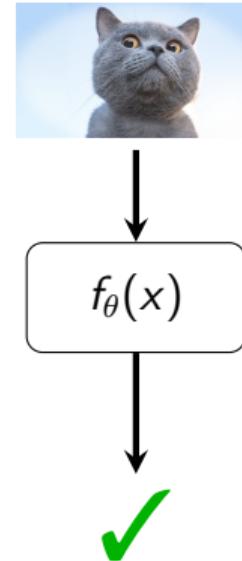


# Aprendizaje Supervisado

Input	Output
	✓
	✓
	✓
	✗
	✗
...	...

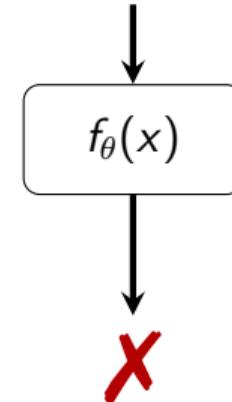
# Aprendizaje Supervisado

Input	Output
	✓
	✓
	✓
	✗
	✗
...	...



# Aprendizaje Supervisado

Input	Output
	✓
	✓
	✓
	✗
	✗
...	...



# Aprendizaje Supervisado

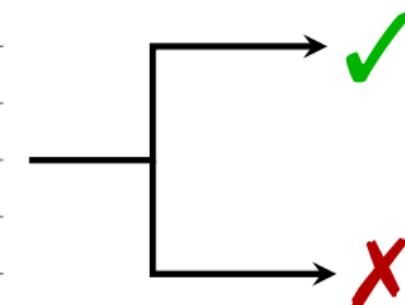
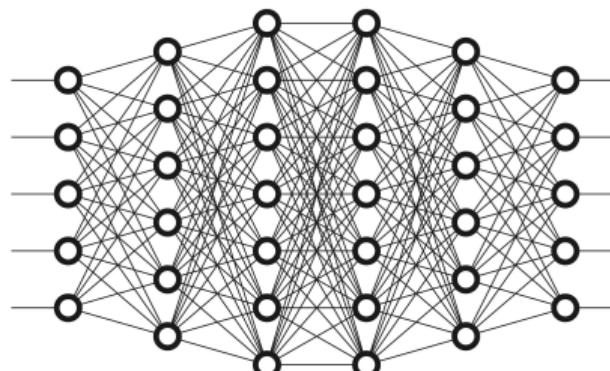
## El problema de aprendizaje

Encontrar un set de parámetros  $\theta$  tal que  $f_{\theta}(x) \approx y$  for all  $(x, y) \in \mathcal{T}$ .

# Aprendizaje Supervisado

## El problema de aprendizaje

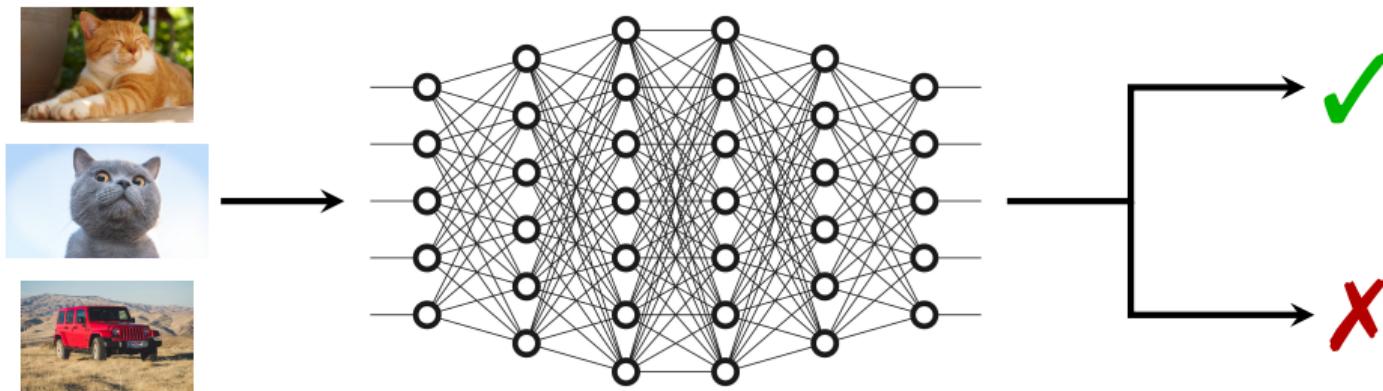
Encontrar un set de parámetros  $\theta$  tal que  $f_{\theta}(x) \approx y$  for all  $(x, y) \in \mathcal{T}$ .



# Aprendizaje Supervisado

## El problema de aprendizaje

Encontrar un set de parámetros  $\theta$  tal que  $f_{\theta}(x) \approx y$  for all  $(x, y) \in \mathcal{T}$ .

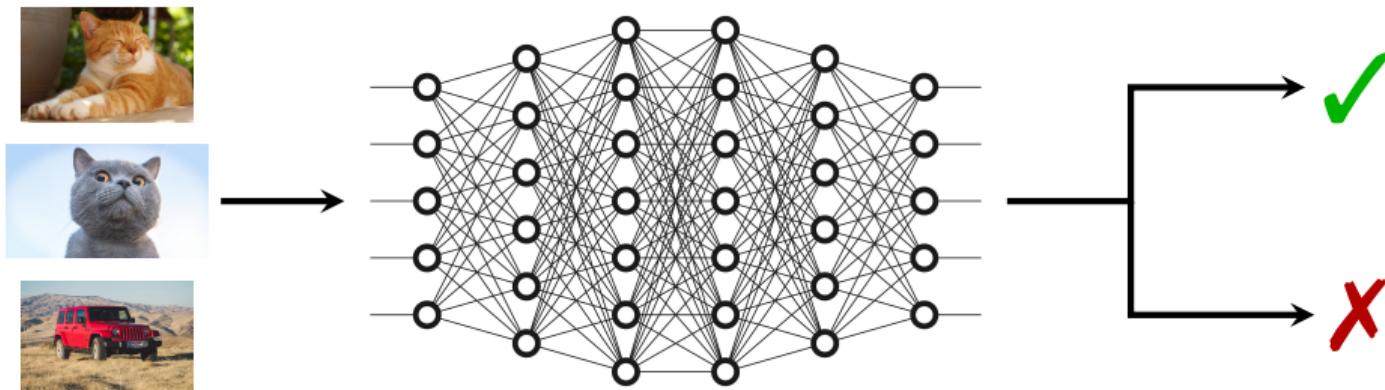


**(1)** Podemos resolver el problema de aprendizaje usando descenso de gradiente.

# Aprendizaje Supervisado

## El problema de aprendizaje

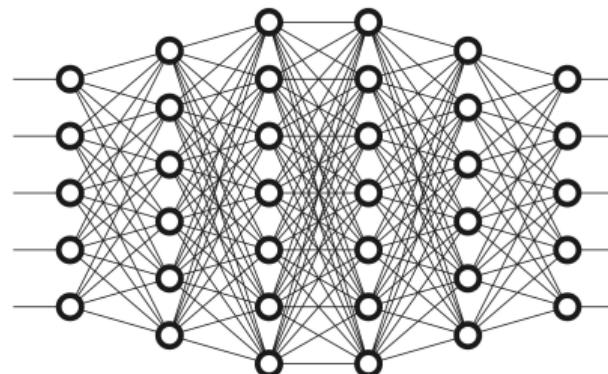
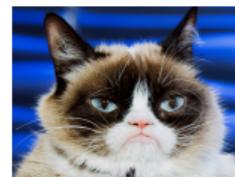
Encontrar un set de parámetros  $\theta$  tal que  $f_{\theta}(x) \approx y$  for all  $(x, y) \in \mathcal{T}$ .



(2)  $f_{\theta}(x)$  generaliza a instancias fuera del conjunto de entrenamiento.

## El problema de aprendizaje

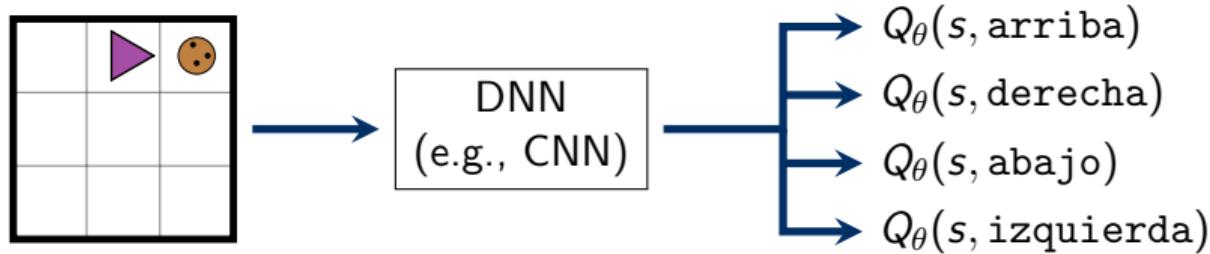
Encontrar un set de parámetros  $\theta$  tal que  $f_{\theta}(x) \approx y$  for all  $(x, y) \in \mathcal{T}$ .



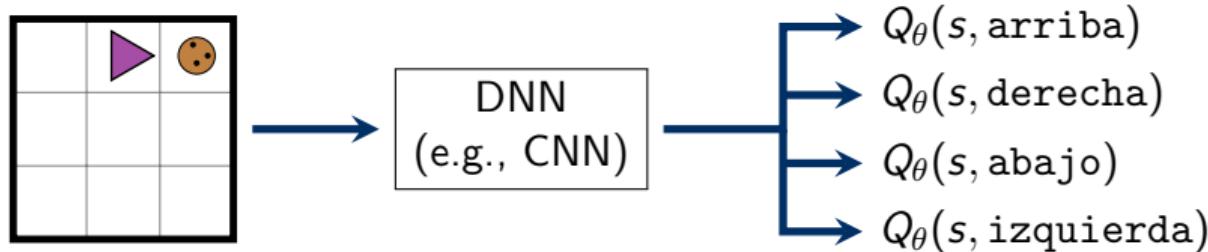
**(2)**  $f_{\theta}(x)$  generaliza a instancias fuera del conjunto de entrenamiento.

# **Deep Q-Networks**

# Deep Q-Networks



# Deep Q-Networks

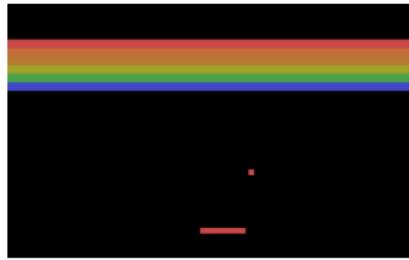
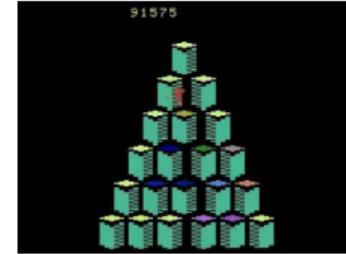


**Experience buffer:**  $\langle \dots, (s, a, r, s'), \dots \rangle$

$$\text{Minimize } (r + \gamma \max_{a'} Q_{\theta'}(s', a') - Q_\theta(s, a))^2$$

# Deep Q-Networks

# Deep Q-Networks<sup>1</sup>



<sup>1</sup>Mnih et al. "Human-level control through deep reinforcement learning." Nature (2015).

# Policy gradient methods<sup>2</sup>

---

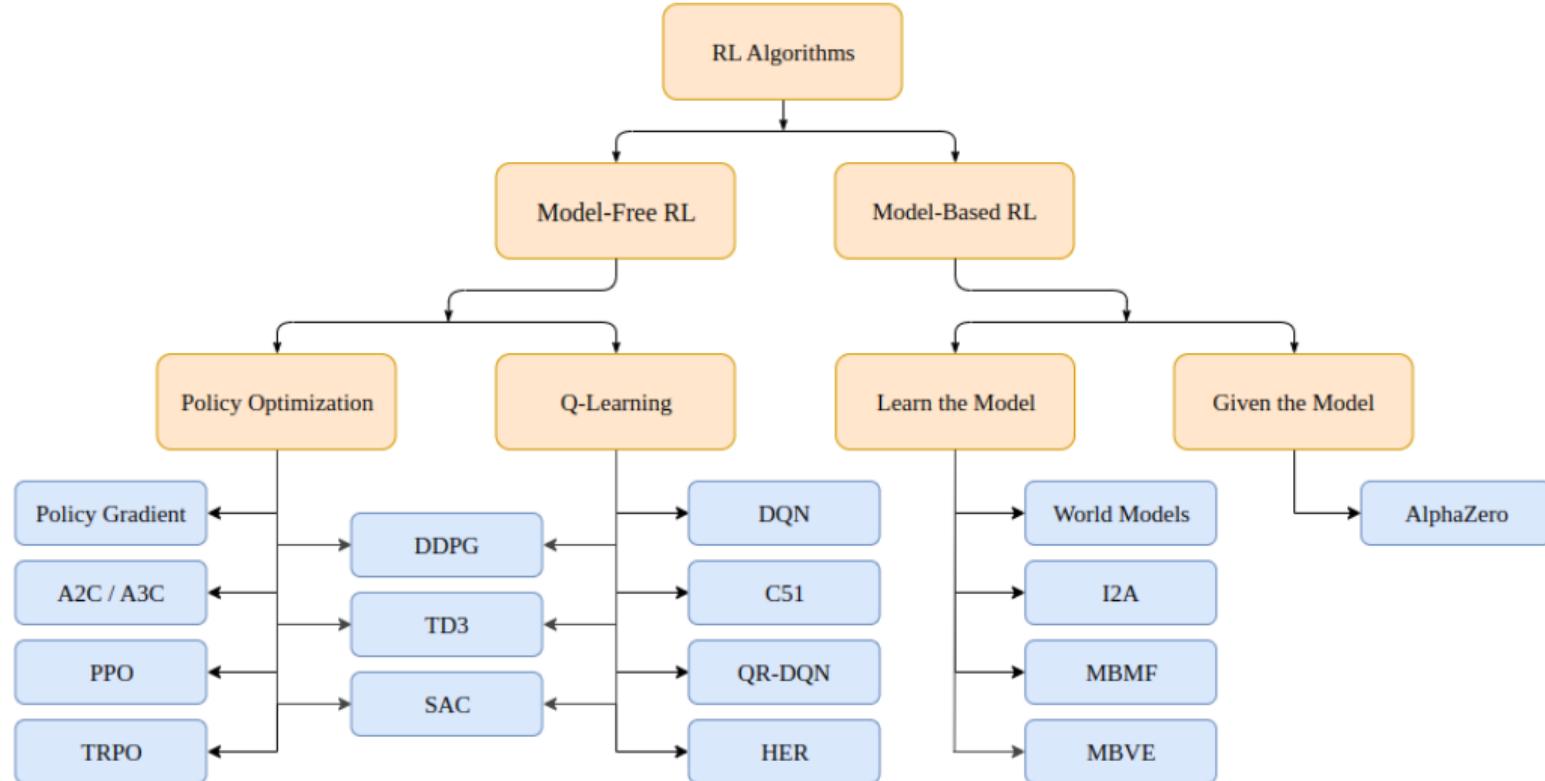
<sup>2</sup>Heess et al. "Emergence of Locomotion Behaviours in Rich Environments." ArXiv (2017).

# Policy gradient methods<sup>3</sup>

---

<sup>3</sup>Heess et al. "Emergence of Locomotion Behaviours in Rich Environments." ArXiv (2017).

# Algoritmos de aprendizaje reforzado



Source: [https://spinningup.openai.com/en/latest/spinningup/rl\\_intro2.html](https://spinningup.openai.com/en/latest/spinningup/rl_intro2.html)

**¿Qué puede aportar RL a RecSys?**

# ¿Qué puede aportar RL a RecSys?

## Respuesta Típica: Online Learning

Un agente de aprendizaje reforzado puede *adaptarse* a como evoluciona un usuario en el tiempo y las decisiones que va tomando.

# ¿Qué puede aportar RL a RecSys?

## Respuesta Típica: Online Learning

Un agente de aprendizaje reforzado puede *adaptarse* a como evoluciona un usuario en el tiempo y las decisiones que va tomando.

... a mí no me satisface del todo esta respuesta :/

# ¿Qué puede aportar RL a RecSys?

## Respuesta Típica: Online Learning

Un agente de aprendizaje reforzado puede *adaptarse* a como evoluciona un usuario en el tiempo y las decisiones que va tomando.

... a mí no me satisface del todo esta respuesta :/

RL no es mágico... igual necesita datos para aprender  $\pi^*$ .

# ¿Qué puede aportar RL a RecSys?

## Respuesta Típica: Online Learning

Un agente de aprendizaje reforzado puede *adaptarse* a como evoluciona un usuario en el tiempo y las decisiones que va tomando.

... a mí no me satisface del todo esta respuesta :/

RL no es mágico... igual necesita datos para aprender  $\pi^*$ . En ese sentido, RL no es necesariamente mejor que seguir entrenando nuestro modelo de basado SL con los nuevos datos que obtengamos.

# ¿Qué puede aportar RL a RecSys?

## Online learning con aprendizaje supervisado:

- 1 Usando muchos datos entrenamos  $f_{\theta}(u, i) \approx p(u, i)$ , donde  $u$  incluye información del usuario (y sus compras) e  $i$  incluye información del ítem.
- 2 En tiempo  $t$ , al usuario  $u_t$  le recomendamos  $i^* = \arg \max_{i \in I} f_{\theta}(u_t, i_t)$ .
- 3 Luego de cierto tiempo, reentrenamos  $f_{\theta}$  usando los nuevos datos obtenidos.

# ¿Qué puede aportar RL a RecSys?

## Online learning con aprendizaje supervisado:

- 1 Usando muchos datos entrenamos  $f_\theta(u, i) \approx p(u, i)$ , donde  $u$  incluye información del usuario (y sus compras) e  $i$  incluye información del ítem.
- 2 En tiempo  $t$ , al usuario  $u_t$  le recomendamos  $i^* = \arg \max_{i \in I} f_\theta(u_t, i_t)$ .
- 3 Luego de cierto tiempo, reentrenamos  $f_\theta$  usando los nuevos datos obtenidos.

## Online learning con aprendizaje reforzado:

- 1 Usando muchos datos entrenamos  $\pi_\theta(u, i) \approx \pi^*(i|u)$ , donde  $u$  incluye información del usuario (y sus compras) e  $i$  incluye información del ítem.
- 2 En tiempo  $t$ , al usuario  $u_t$  le recomendamos  $i^* = \arg \max_{i \in I} \pi_\theta(u_t, i_t)$ .
- 3 Luego de cierto tiempo, reentrenamos  $\pi_\theta$  usando los nuevos datos obtenidos.

# ¿Qué puede aportar RL a RecSys?

## Online learning con aprendizaje supervisado:

- 1 Usando muchos datos entrenamos  $f_\theta(u, i) \approx p(u, i)$ , donde  $u$  incluye información del usuario (y sus compras) e  $i$  incluye información del ítem.
- 2 En tiempo  $t$ , al usuario  $u_t$  le recomendamos  $i^* = \arg \max_{i \in I} f_\theta(u_t, i_t)$ .
- 3 Luego de cierto tiempo, reentrenamos  $f_\theta$  usando los nuevos datos obtenidos.

## Online learning con aprendizaje reforzado:

- 1 Usando muchos datos entrenamos  $\pi_\theta(u, i) \approx \pi^*(i|u)$ , donde  $u$  incluye información del usuario (y sus compras) e  $i$  incluye información del ítem.
- 2 En tiempo  $t$ , al usuario  $u_t$  le recomendamos  $i^* = \arg \max_{i \in I} \pi_\theta(u_t, i_t)$ .
- 3 Luego de cierto tiempo, reentrenamos  $\pi_\theta$  usando los nuevos datos obtenidos.

¿Qué nos hace pensar que el algoritmo de abajo sea mejor que el de arriba? 🤔

## ¿Qué puede aportar RL a RecSys?

Existe una diferencia sutil (pero importante) entre  $f_\theta(u, i)$  y  $\pi_\theta(u, i)$ .

## ¿Qué puede aportar RL a RecSys?

Existe una diferencia sutil (pero importante) entre  $f_\theta(u, i)$  y  $\pi_\theta(u, i)$ .

- $f_\theta(u_t, i_t)$ : Busca recomendar el ítem con mayor probabilidad de ser comprado inmediatamente por el usuario.
- $\pi_\theta(u_t, i_t)$ : Busca recomendar el ítem que maximiza la probabilidad de recomendar ítems que sean comprados por el usuario (en el corto, mediano y largo plazo).

## ¿Qué puede aportar RL a RecSys?

Existe una diferencia sutil (pero importante) entre  $f_\theta(u, i)$  y  $\pi_\theta(u, i)$ .

- $f_\theta(u_t, i_t)$ : Busca recomendar el ítem con mayor probabilidad de ser comprado inmediatamente por el usuario.
- $\pi_\theta(u_t, i_t)$ : Busca recomendar el ítem que maximiza la probabilidad de recomendar ítems que sean comprados por el usuario (en el corto, mediano y largo plazo).

La política  $\pi_\theta(u_t, i_t)$  está dispuesta a hacer una “*mala*” recomendación inmediata con tal de ganar información que nos permita hacer mejores recomendaciones a futuro.

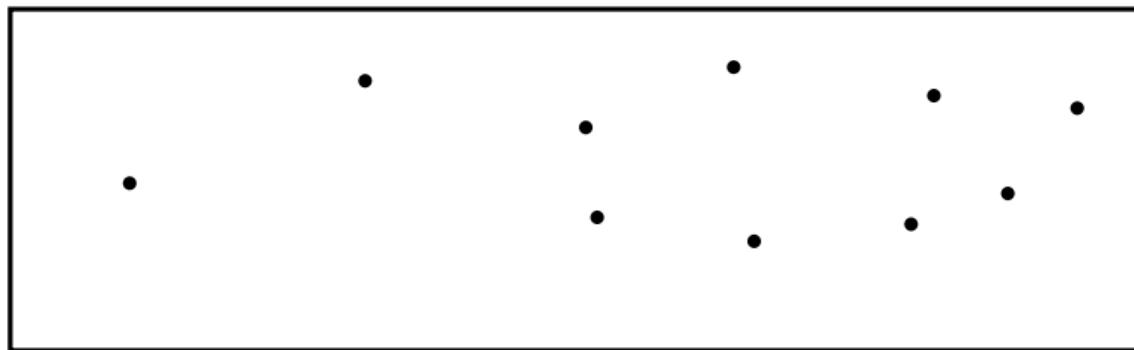
## ¿Qué puede aportar RL a RecSys?

Consideremos un problema de recomendación que tiene la siguiente estructura (desconocida por nosotros):

# ¿Qué puede aportar RL a RecSys?

Consideremos un problema de recomendación que tiene la siguiente estructura (desconocida por nosotros):

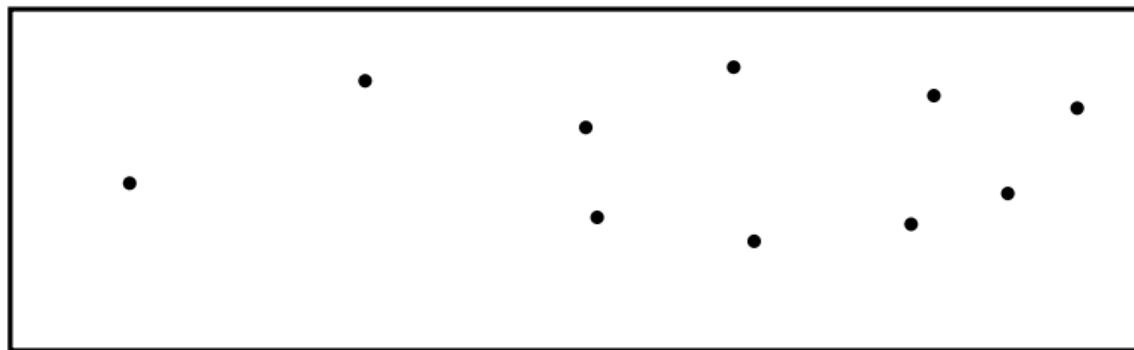
Existen items que en general a todos les gustan.



# ¿Qué puede aportar RL a RecSys?

Consideremos un problema de recomendación que tiene la siguiente estructura (desconocida por nosotros):

Existen items que en general a todos les gustan.

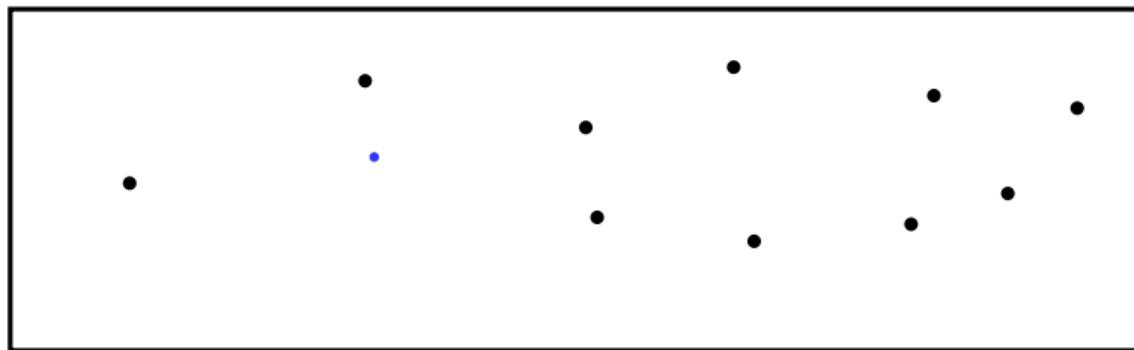


... pero por lo mismo, nos entregan poca información del usuario.

# ¿Qué puede aportar RL a RecSys?

Consideremos un problema de recomendación que tiene la siguiente estructura (desconocida por nosotros):

Existen items que en general a todos les gustan.

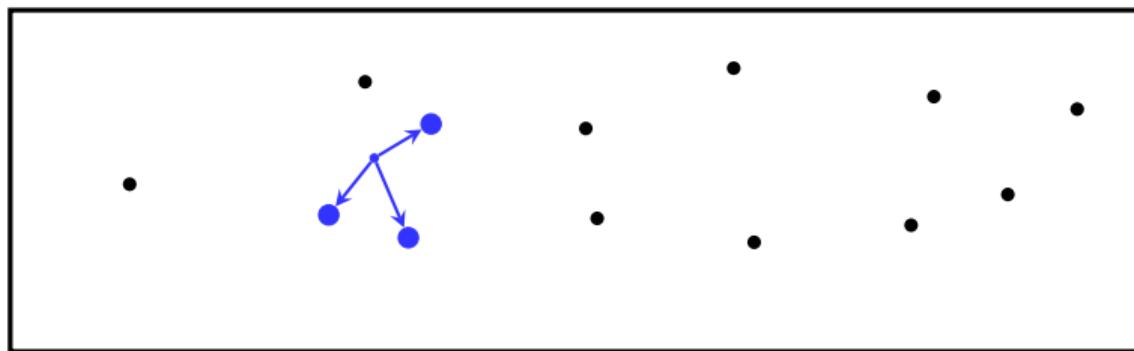


... pero por lo mismo, nos entregan poca información del usuario.

# ¿Qué puede aportar RL a RecSys?

Consideremos un problema de recomendación que tiene la siguiente estructura (desconocida por nosotros):

Existen items que en general a todos les gustan.

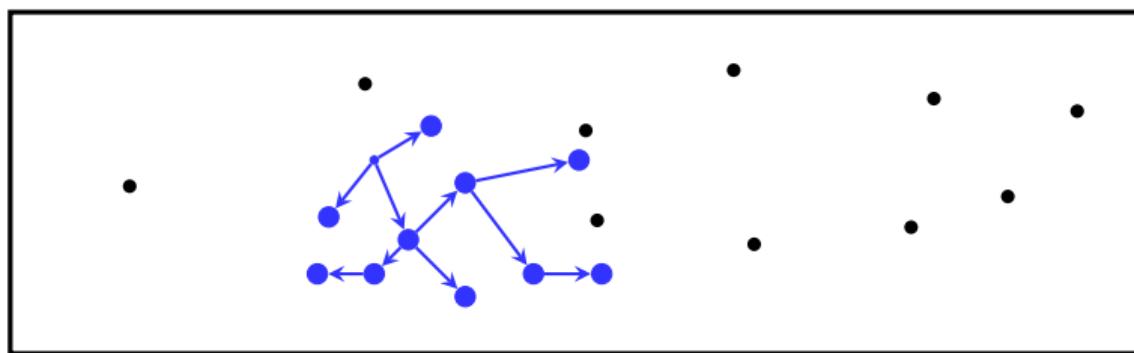


... pero por lo mismo, nos entregan poca información del usuario.

# ¿Qué puede aportar RL a RecSys?

Consideremos un problema de recomendación que tiene la siguiente estructura (desconocida por nosotros):

Existen items que en general a todos les gustan.

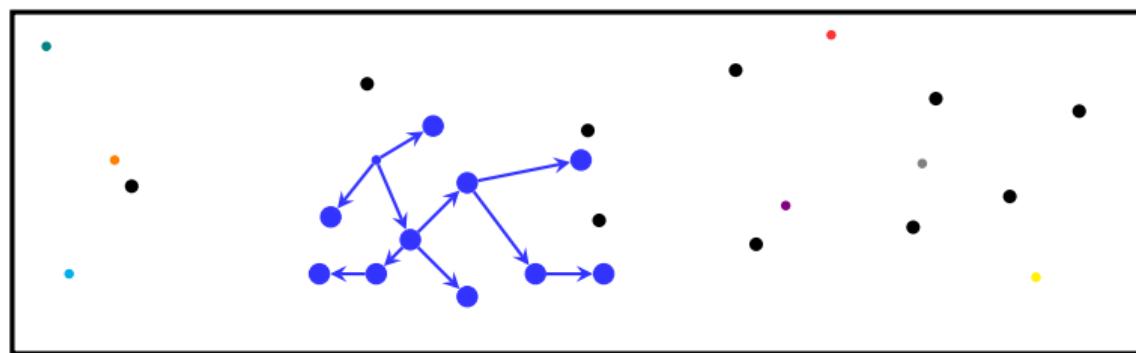


... pero por lo mismo, nos entregan poca información del usuario.

# ¿Qué puede aportar RL a RecSys?

Consideremos un problema de recomendación que tiene la siguiente estructura (desconocida por nosotros):

Existen items que en general a todos les gustan.



... pero por lo mismo, nos entregan poca información del usuario.

# ¿Qué puede aportar RL a RecSys?

## Resumen:

Desde un punto de vista teórico, creo que intentar aprender  $\pi_\theta(i_t|u_t) \approx \pi^*(i_t|u_t)$  es un mejor objetivo que  $f_\theta(u_t, i_t) \approx p(u_t, i_t)$  para entrenar un sistema recomendador.

# ¿Qué puede aportar RL a RecSys?

## Resumen:

Desde un punto de vista teórico, creo que intentar aprender  $\pi_\theta(i_t|u_t) \approx \pi^*(i_t|u_t)$  es un mejor objetivo que  $f_\theta(u_t, i_t) \approx p(u_t, i_t)$  para entrenar un sistema recomendador.

Desde un punto de vista práctico, el aprendizaje **supervisado** resuelve un problema *menos ambicioso* que el aprendizaje **reforzado**... por lo que  $f_\theta(u_t, i_t)$  es más fácil de entrenar.

*¿Preguntas?*