

# **APRENDIZAJE POR REFUERZO**

*Bandits*

---

Antonio Manjavacas

[manjavacas@ugr.es](mailto:manjavacas@ugr.es)

# CONTENIDOS

---

1. *Bandits*
2. Exploración vs. Explotación
3. Estimación de *action-values*
4. *Action-values*: cálculo incremental
5. Valores iniciales
6. *Upper-confidence-bound*
7. Trabajo propuesto

- Ya conocemos las diferencias entre aprendizaje **instructivo** y **evaluativo**.
- También hemos visto que el aprendizaje **evaluativo** es la base del **aprendizaje por refuerzo** (RL).

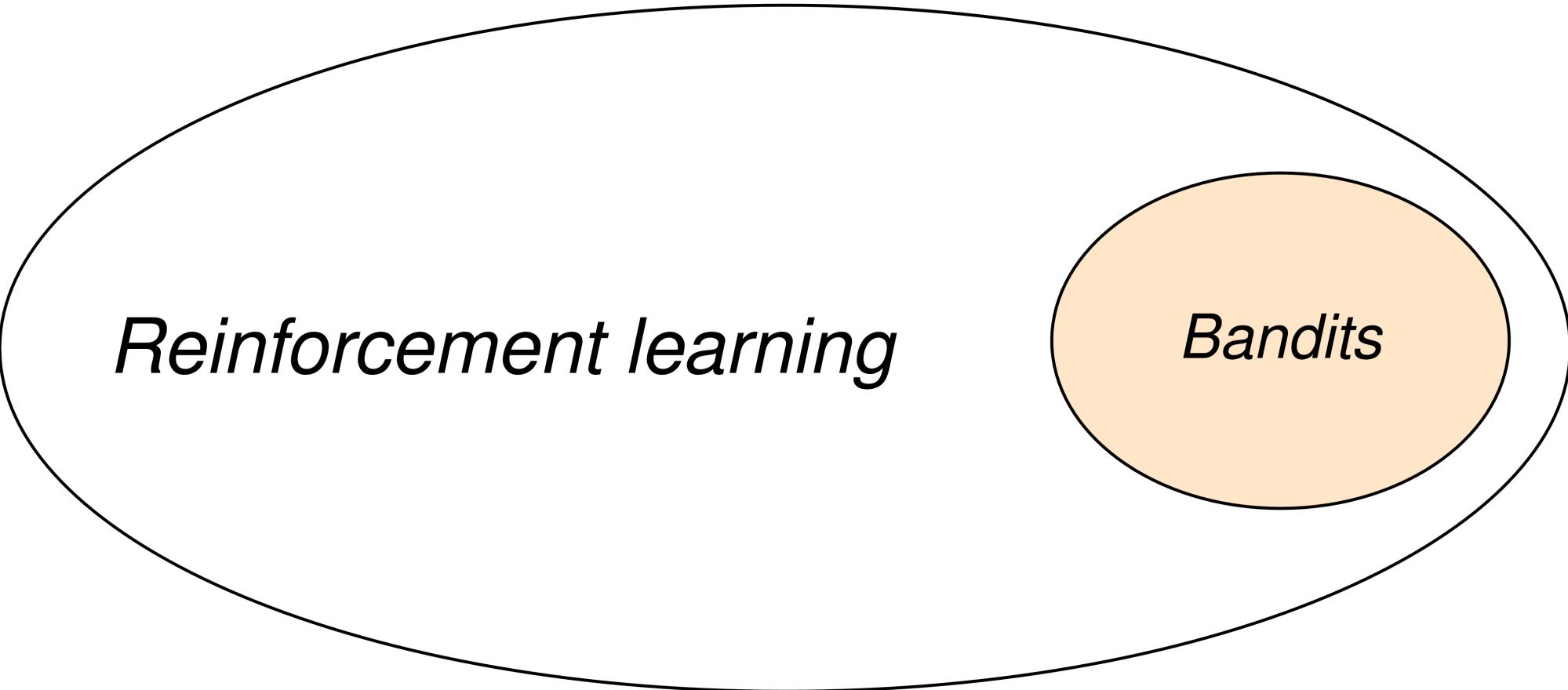
**OBJETIVO:** profundizar sobre RL en un **entorno simplificado**, en el cual no es necesario aprender a actuar en más de una situación.

- Una simplificación de los problemas de *reinforcement learning* «completos».
- Esto nos permitirá introducir algunos conceptos básicos.

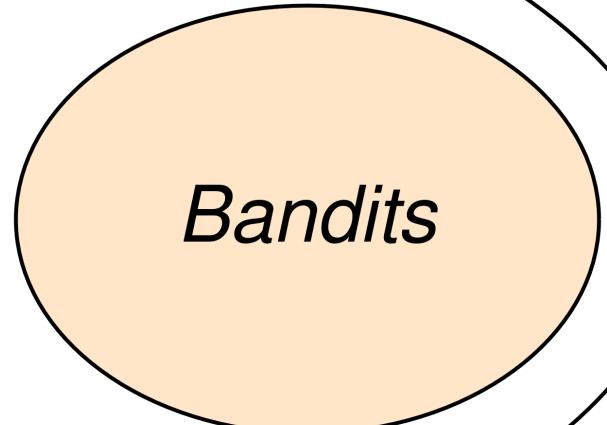
# **BANDITS**

---

- Los problemas de RL pueden entenderse como problemas de **toma de decisiones**.
  - Un agente recibe información del entorno y **decide** qué acción realizar.
- En un problema de RL **completo**, consideramos múltiples estados, así como diferentes acciones a realizar dependiendo de qué estado perciba el agente.
  - Las acciones pueden afectar el estado del entorno y, por tanto, influir en las recompensas futuras.
- Antes de abordar problemas de RL complejos, estudiaremos un **problema simplificado** donde el concepto de *estado* no es tan relevante, y únicamente se tienen en cuenta las *acciones* a realizar por el agente.
  - Es lo que llamamos un **entorno no asociativo**.

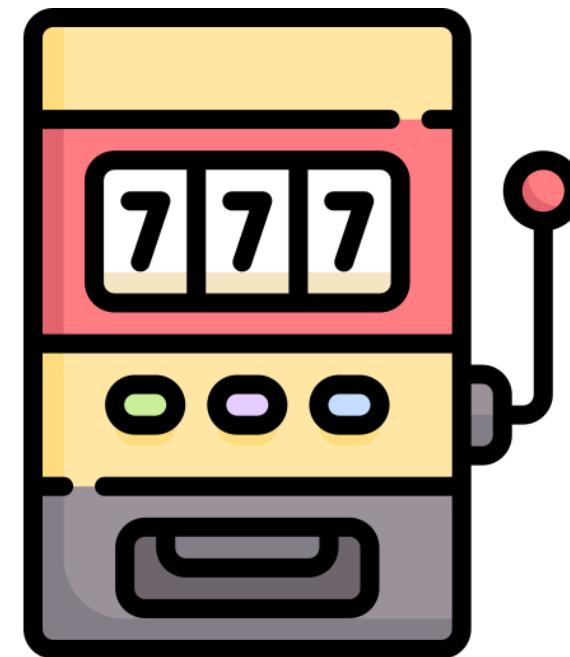


*Reinforcement learning*

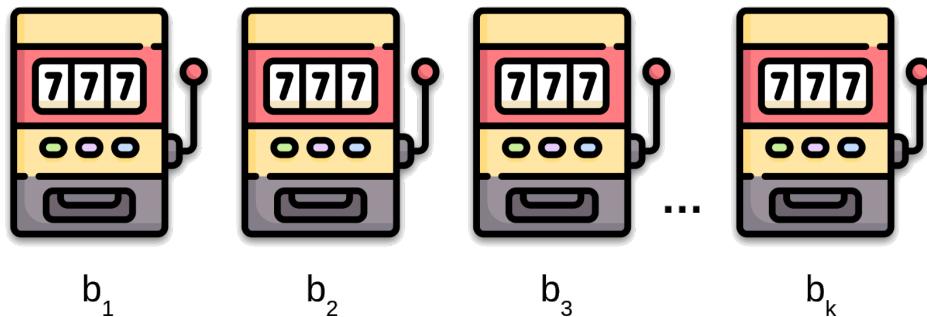


*Bandits*

- ***K-armed bandits***, también llamado ***multi-armed bandit problem***  
«problema de las *tragaperras multi-brazo*» 😊
- Problema clásico en RL y teoría de la probabilidad.
- Extrapolable a campos tan variados como ensayos clínicos, gestión empresarial, economía, *marketing*...
- Existen muchas variantes, pero nos centraremos en la versión más básica del problema.



*Armed bandit*



*K-armed bandits*

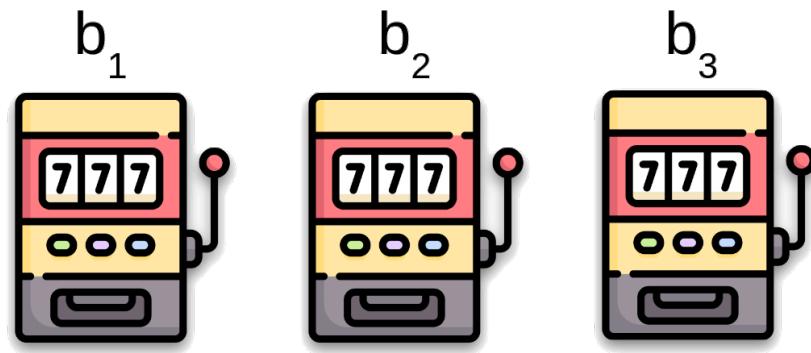
Tenemos un número arbitrario ( $K$ ) de máquinas tragaperras.

- En cada instante de tiempo  $t$ , accionamos una máquina y recibimos una recompensa (💰).
- Cada máquina puede comportarse de forma diferente.
- Desconocemos la **distribución de recompensas** de cada máquina.

**OBJETIVO:** obtener la mayor recompensa posible (**recompensa acumulada**).

# El problema de los *K-armed bandits*

9/65



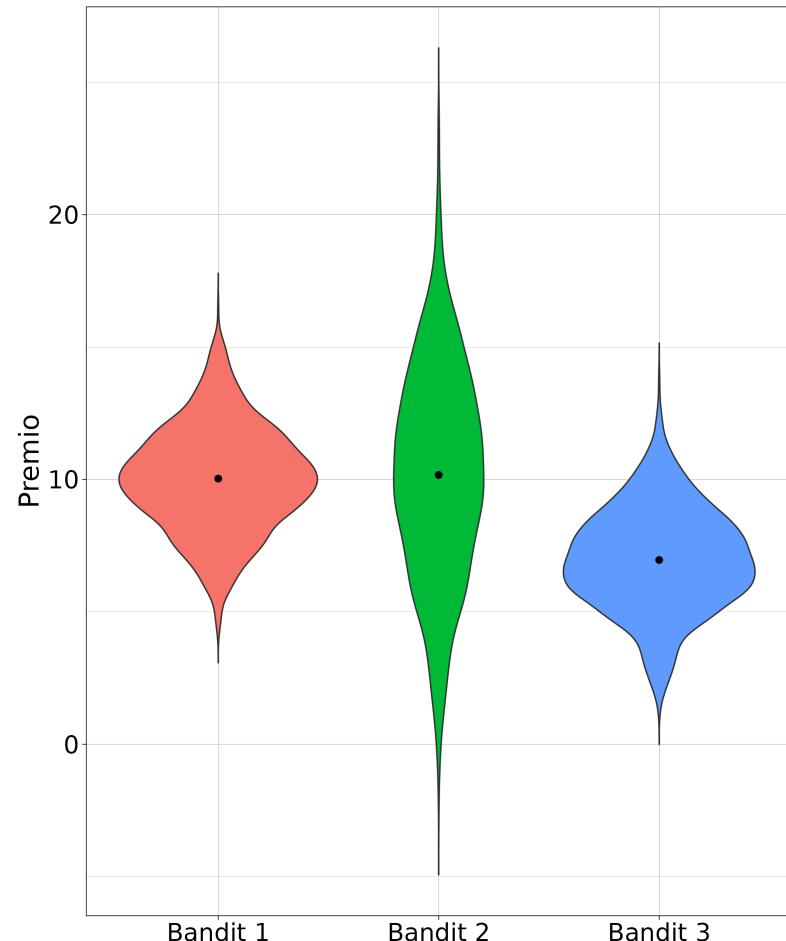
*3-armed bandits*

$$b_1 : \mu = 10, \sigma = 2$$

$$b_2 : \mu = 10, \sigma = 4$$

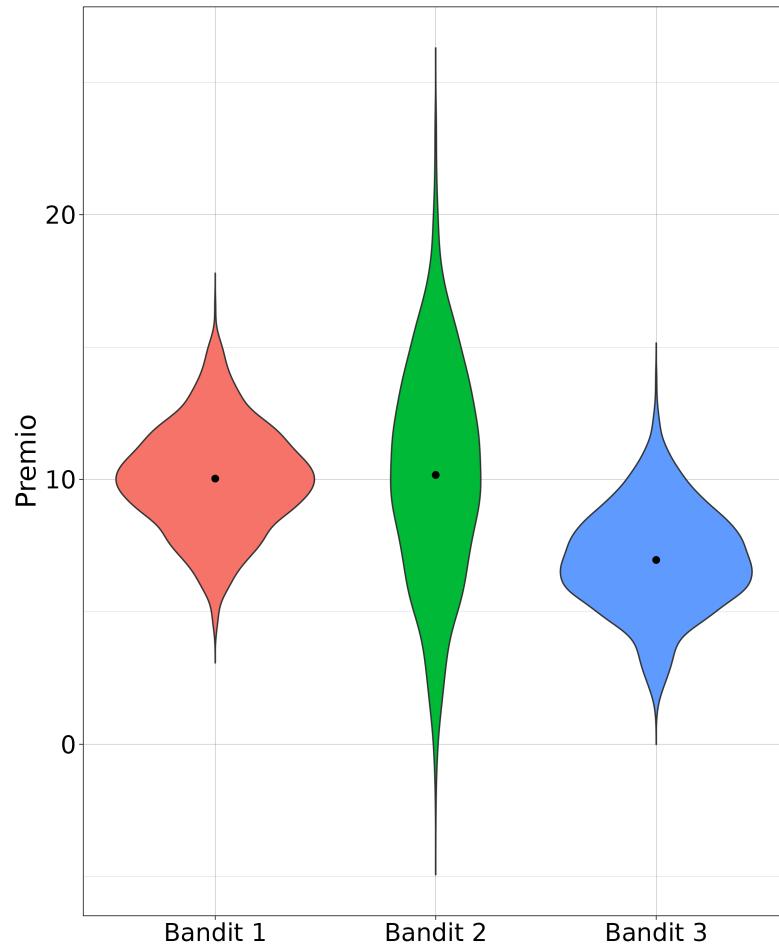
$$b_3 : \mu = 7, \sigma = 2$$

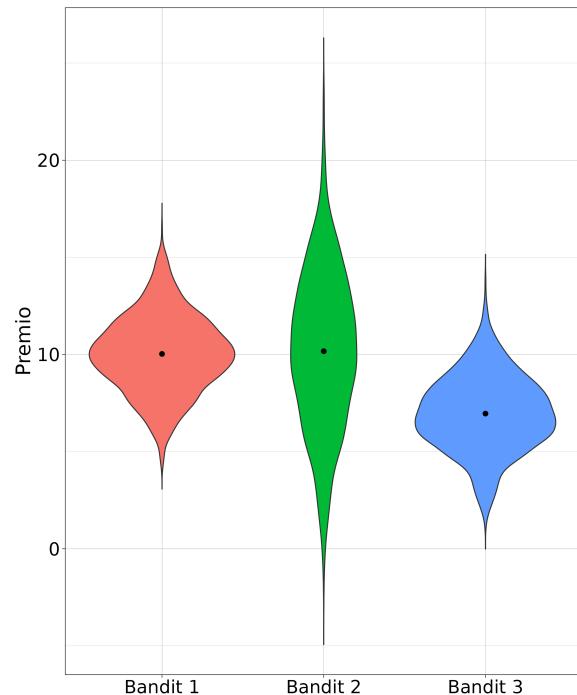
Distribuciones de recompensas: medias y desviaciones típicas



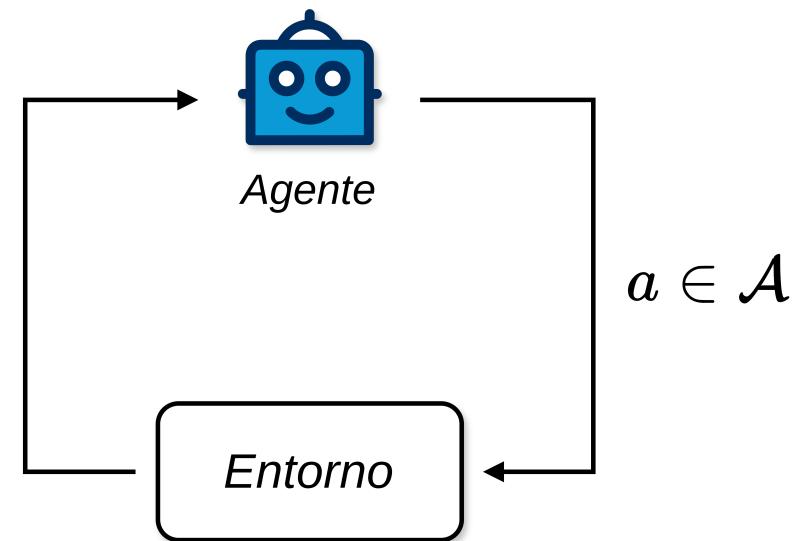
Buscamos maximizar la **recompensa acumulada** concentrando nuestras acciones en las mejores acciones.

- Las recompensas medias de  $b_1$  y  $b_2$  son similares. No obstante,  $b_2$  es más conservador (menor  $\sigma \rightarrow$  valores más próximos a la media).
- Jugar  $b_2$  es más arriesgado (mayor  $\sigma$ ), porque puede darnos mejores recompensas que  $b_2$  y  $b_3$ , pero también recompensas mucho peores.
- Jugar  $b_3$  no parece una buena idea...





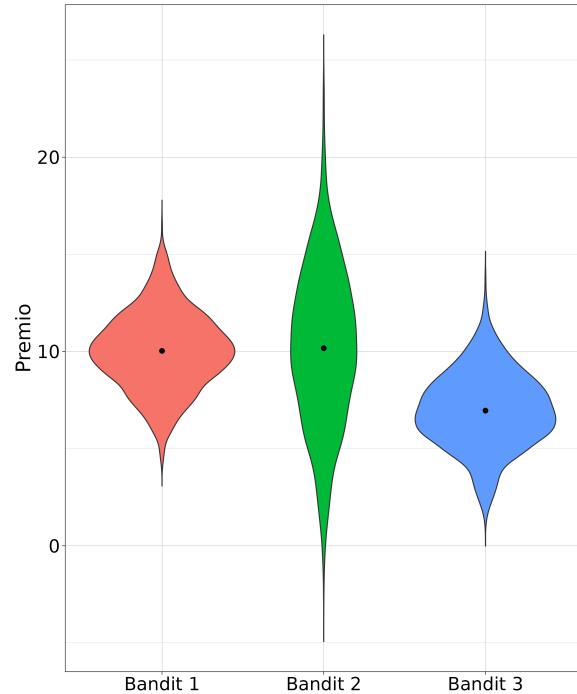
$$r \in \mathbb{R}$$



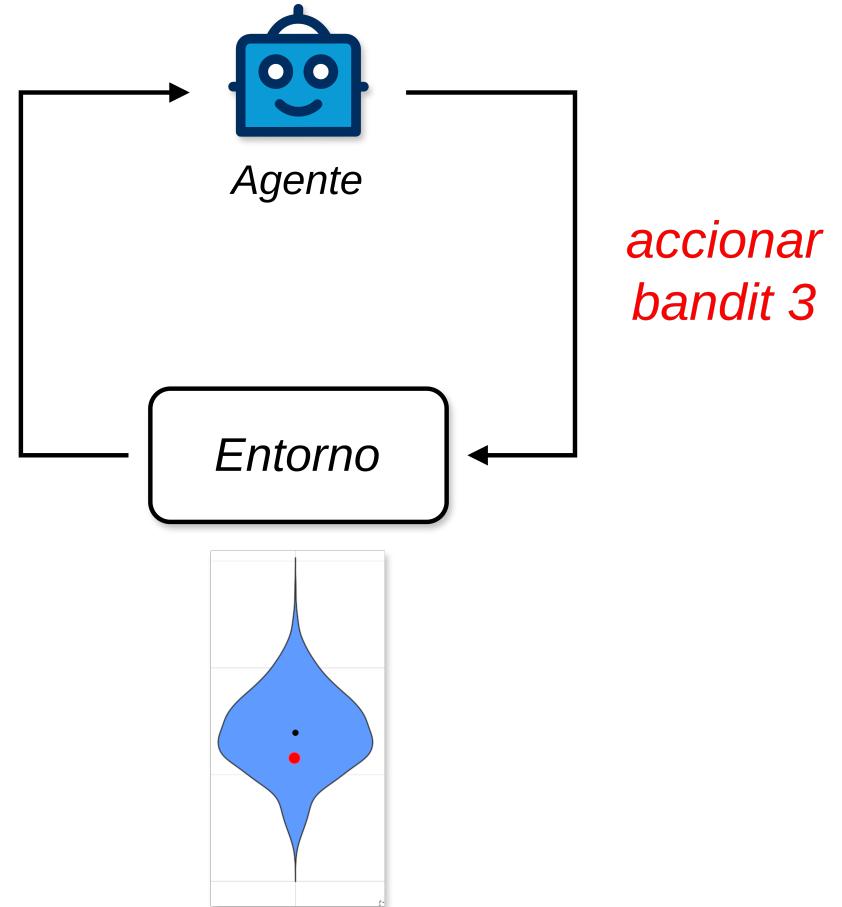
**Espacio de acciones:**  $\mathcal{A} = \{a_1, a_2, a_3\}$

# El problema de los K-armed bandits

12/65



5



Espacio de acciones:  $\mathcal{A} = \{a_1, a_2, a_3\}$

El **valor de una acción** (*action-value*) es la recompensa que esperamos obtener al realizarla:

$$q_*(a) = \underbrace{\mathbb{E}[R_t \mid A_t = a]}_{\substack{\text{Valor de} \\ \text{la acción}}} \quad \underbrace{\mathbb{E}}_{\substack{\text{Recompensa esperada} \\ \text{al realizar dicha acción}}}$$

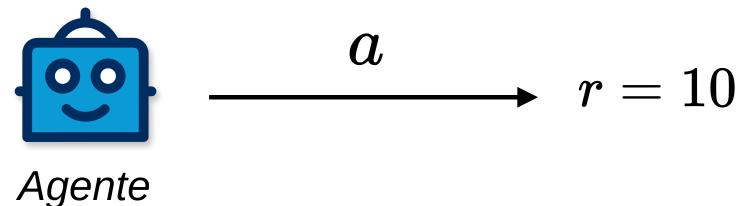
$\mathbb{E}$  representa el **valor esperado**:

$$\mathbb{E}(x) = \sum x P(X = x)$$

- Suma ponderada de los posibles valores de  $x$  por sus probabilidades.
- Relevante en **problemas estocásticos** donde  $R_t$  viene dada por una distribución de probabilidad.

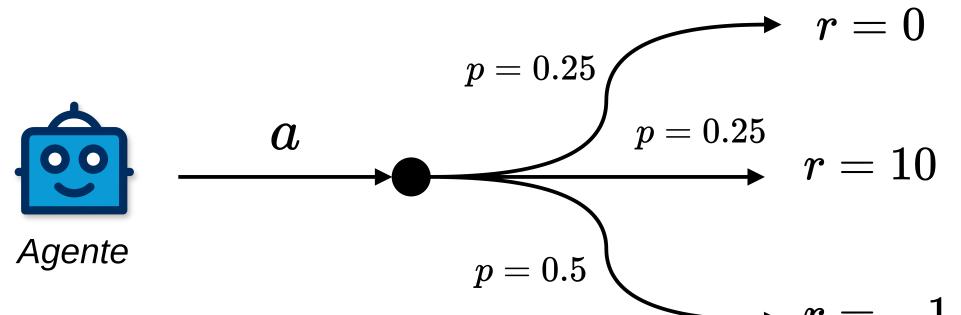
## Problema determinista

Toda acción  $a \in \mathcal{A}$  siempre tiene las mismas consecuencias.



## Problema estocástico

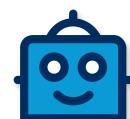
Realizar una acción  $a \in \mathcal{A}$  puede conducirnos a diferentes recompensas o estados a partir de situaciones similares.



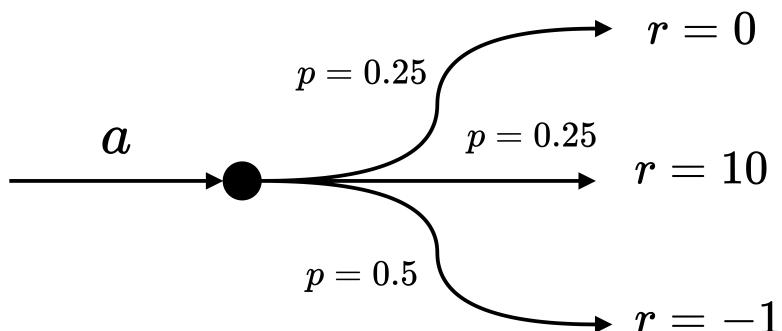
Si desarrollamos la fórmula:

$$q_*(a) = \mathbb{E}[R_t \mid A_t = a] = \sum_r r p(r|a)$$

Suma ponderada de las recompensas por sus probabilidades



Agente



$$\begin{aligned} q_*(a) &= 0 \cdot 0.25 + 10 \cdot 0.25 + \cdot (-1) \cdot 0.5 \\ &= 0 + 2.5 - 0.5 = 2 \end{aligned}$$

Como, a priori, no conocemos los valores reales de cada acción, consideramos valores estimados:

$$Q_t(a) \approx q_*(a)$$

Estos valores se irán aproximando a los reales a medida que ampliemos nuestra **experiencia**.

Pero...

*¿Cómo aproximamos progresivamente los valores de las acciones?*

# **EXPLORACIÓN VS. EXPLOTACIÓN**

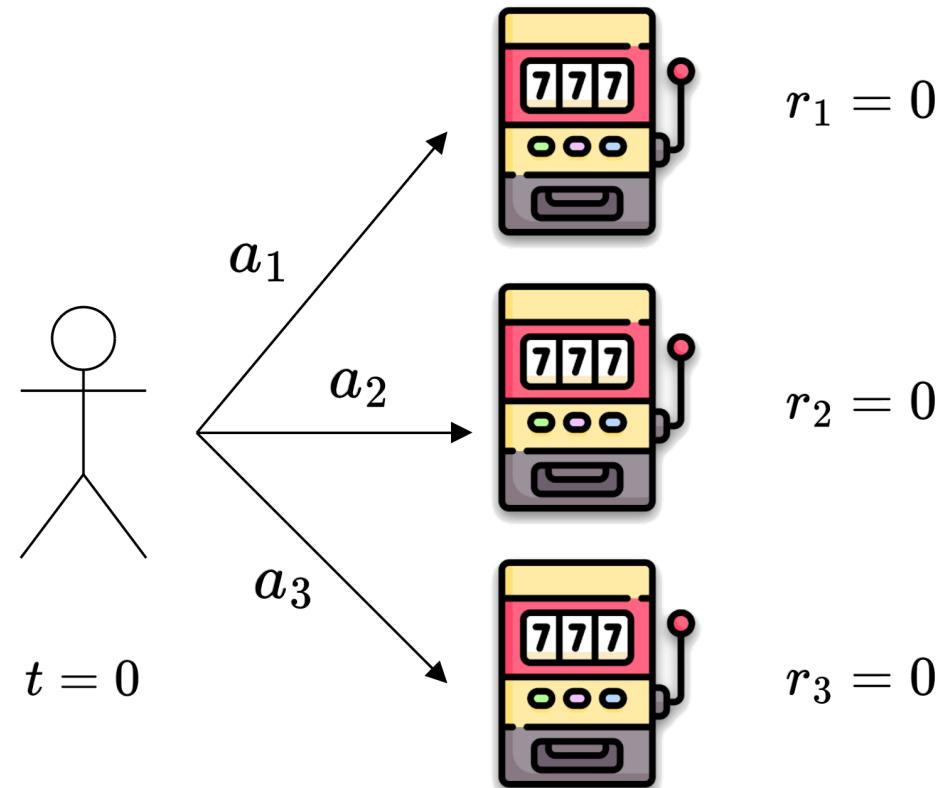
---

Consideremos el siguiente caso:

- Tenemos 3 *bandits* y, por tanto, 3 posibles acciones.
- Inicialmente, desconocemos el valor de cada acción:

$$q_*(a) = 0, \forall a \in \mathcal{A}$$

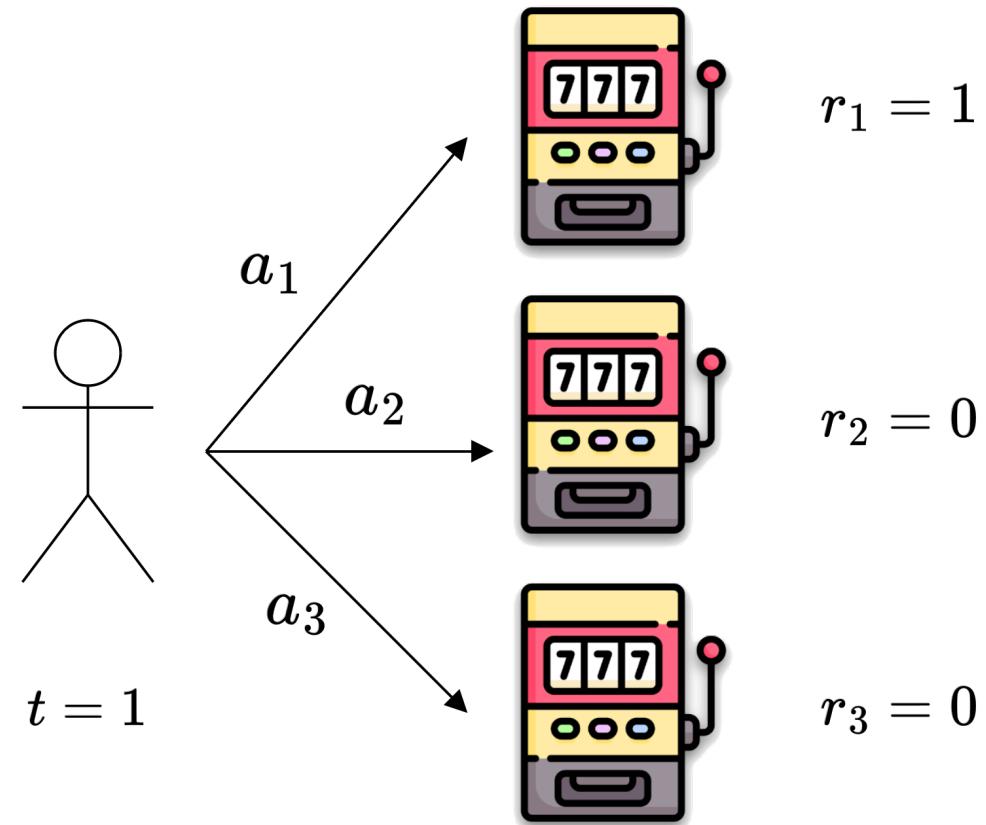
- Por tanto, comenzamos eligiendo una acción arbitraria.
  - Es decir, **exploramos**.



- El agente elige  $a_1$  y recibe una recompensa  $r_1 = 1$ .
- Posteriormente, convendrá explorar  $a_2$  y  $a_3$  para comprobar si son mejores opciones.

**Explorar** implica sacrificar recompensas inmediatas conocidas para probar alternativas hasta el momento no contempladas.

Esta inversión podría llevarnos a obtener mayores recompensas *a largo plazo*.

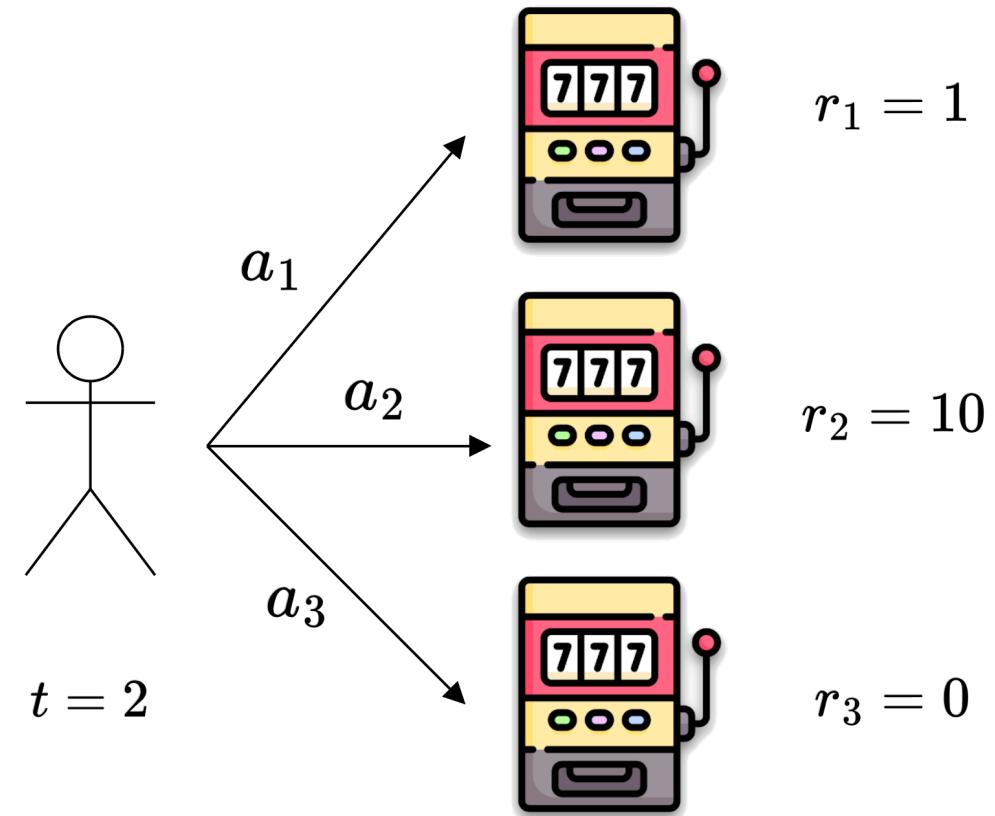


- Es una idea similar a elegir entre un **restaurante** de confianza o uno que no has visitado nunca... 
- ...o entre tu género de **películas** favorito y otro que no sueles ver... 
- ...o entre un **producto** que sueles comprar y otro que podría interesarte... 

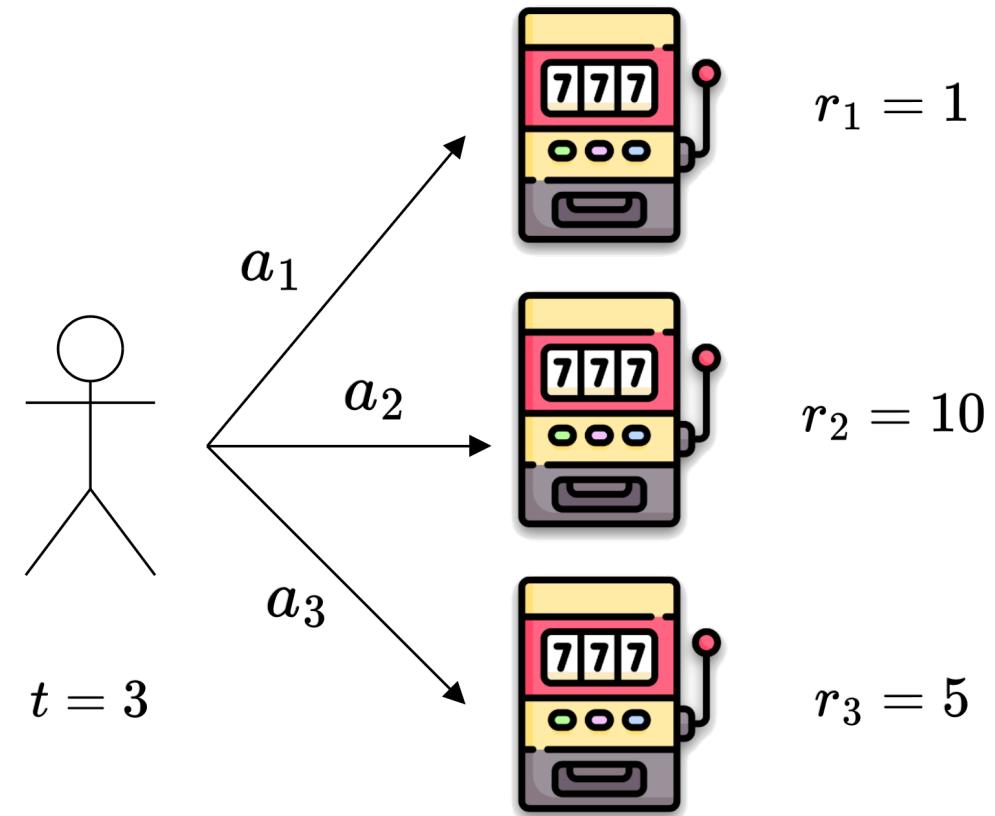
*¿Intuyes las posibles aplicaciones?*



- El agente elige  $a_2$  y recibe una recompensa  $r_2 = 10$ .



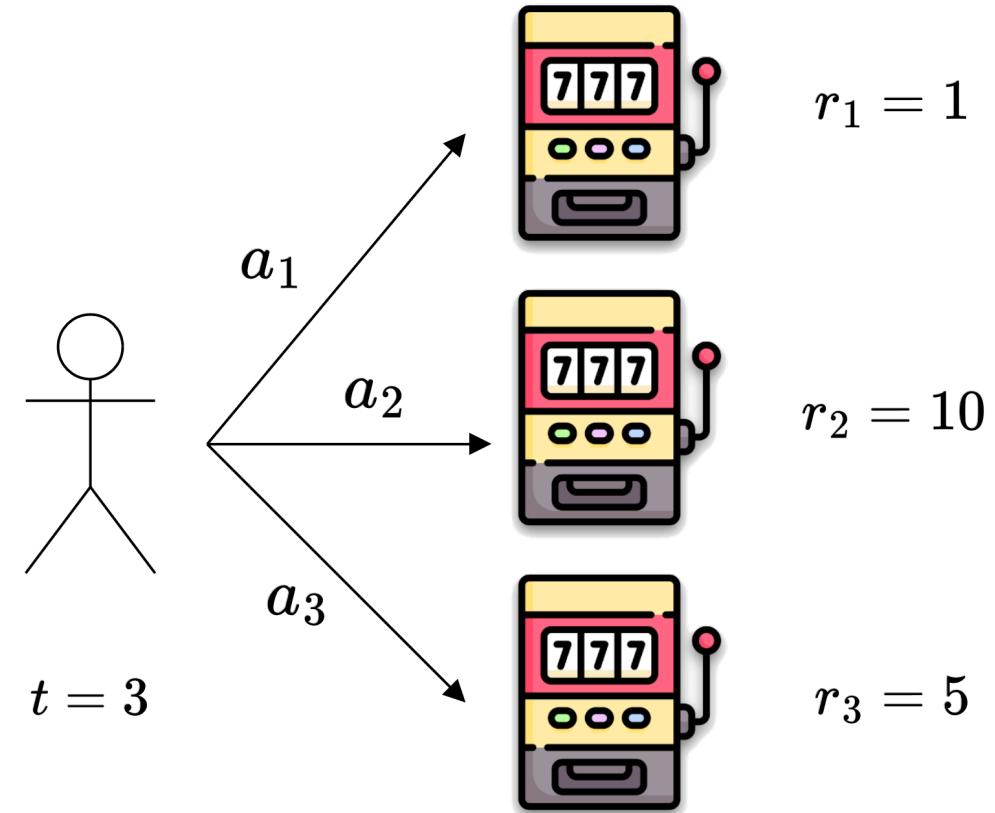
- El agente elige  $a_3$  y recibe una recompensa  $r_3 = 5$ .

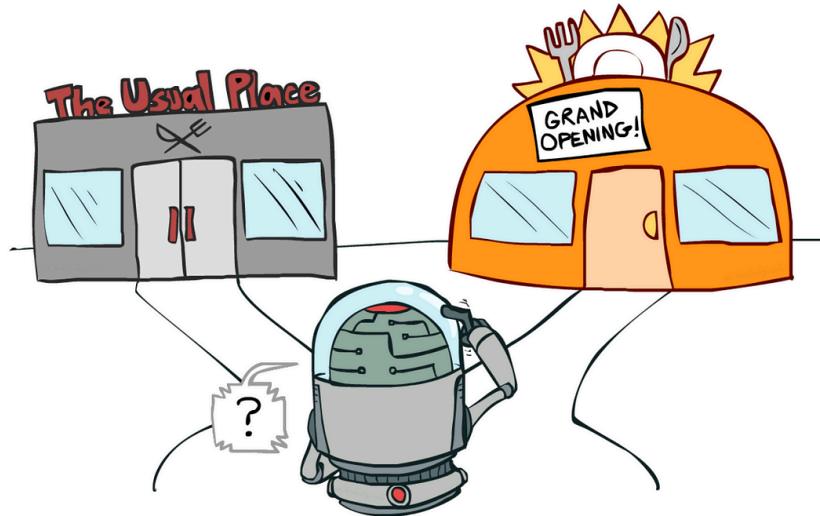


- En  $t = 3$  hemos realizado todas las acciones posibles y recibidas sus correspondientes recompensas.

A partir de aquí podemos:

- Actuar de forma **greedy** («voraz»), **explotando** indefinidamente la mejor acción ( $a_2$ ).
- Mantener un comportamiento aleatorio, **explorando** de forma continua las distribuciones de recompensa asociadas a cada acción.





No es posible **explorar** y **explotar** a la vez, lo que conduce a un **conflicto**.

- ***Exploration-exploitation trade-off***

Elegir una estrategia u otra dependerá de diferentes factores: incertidumbre, estimaciones, tiempo restante...

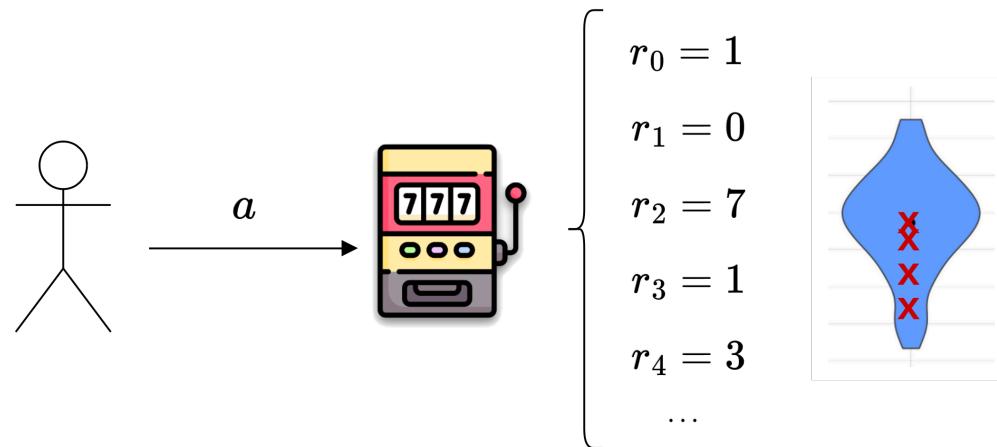
Trataremos de buscar un **balance** entre exploración y explotación, siguiendo un comportamiento  **$\epsilon$ -greedy**.

La estrategia  $\varepsilon$ -greedy consiste en combinar **exploración** y **explotación**:

- **Explotación** (elegir de forma *greedy* la mejor acción) con probabilidad  $1 - \varepsilon$ .
- **Exploración** (elegir una acción aleatoriamente) con probabilidad  $\varepsilon$ .



- Para poder actuar de forma *greedy* necesitamos saber qué acción es la mejor.
- Pero una acción puede no darnos siempre la misma recompensa...



¿Cómo determinamos el valor de una acción?

# **ESTIMACIÓN DE ACTION-VALUES**

---

Existen diferentes formas de estimar el valor de una acción.

Un método a considerar es la **media muestral** (*sample-average method*):

$$Q_t(a) = \frac{\sum_{i=1}^{t-1} R_{i,a}}{\sum_{i=1}^{t-1} n_{i,a}}$$

$$t \rightarrow \infty, \quad Q_t(a) = q_*(a)$$

- El valor estimado de una acción es la suma de las recompensas ofrecidas hasta el momento entre el número de veces que se ha elegido.
- Si  $t$  tiende a  $\infty$ , el valor estimado  $Q_t(a)$  convergerá en el valor real  $q_*(a)$ .

¿Cómo emplear el valor estimado para elegir una acción?

Selección de acciones **greedy**:

$$A_t = \operatorname{argmax}_a Q_t(a)$$

- Seleccionar la acción con el mayor valor estimado.
- Si varias acciones tienen el mismo valor, podemos fijar un criterio (ej. selección aleatoria, la primera, etc.).

¿Cómo emplear el valor estimado para elegir una acción?

$\varepsilon$ -greedy combina la estrategia greedy con la probabilidad  $\varepsilon$  de explorar:

$$A_t = \begin{cases} \operatorname{argmax}_a Q_t(a) & \text{con prob. } 1-\varepsilon \\ a \sim \text{Uniform}(\{a_1, a_2, \dots, a_k\}) & \text{con prob. } \varepsilon \end{cases}$$

- Cuanto menor sea  $\varepsilon$ , más tardaremos en converger en los valores reales.
- Es posible ir reduciendo  $\varepsilon$  con el paso del tiempo (a medida que los valores convergen).

Consideremos el espacio de acciones:  $\mathcal{A} = \{a_1, a_2\}$

¿Cuál es la probabilidad de elegir  $a_2$  siguiendo una estrategia  $\varepsilon$ -greedy con  $\varepsilon = 0.5$ ?

$$P(a_2) = 0.5 \cdot 0.5 = 0.25$$

Consideremos el espacio de acciones:  $\mathcal{A} = \{a_1, a_2\}$

¿Cuál es la probabilidad de elegir  $a_2$  siguiendo una estrategia  $\varepsilon$ -greedy con  $\varepsilon = 0.5$ ?

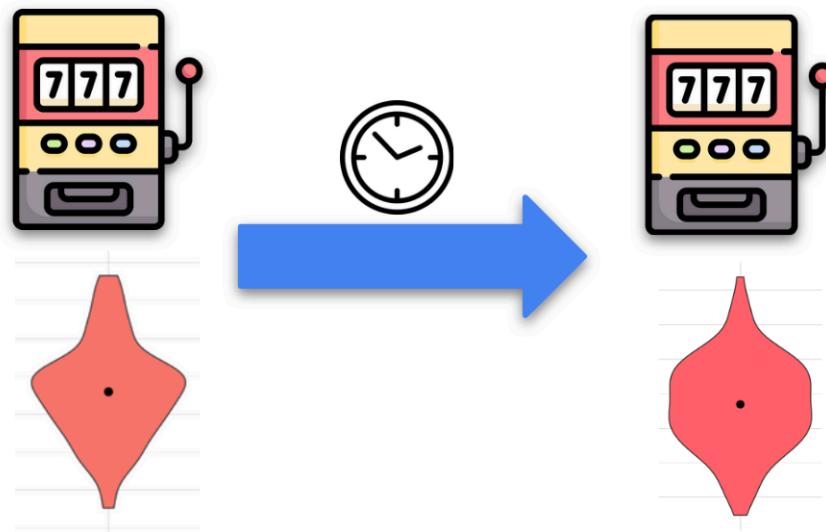
$$P(a_2) = \frac{0.5}{\text{Probabilidad de explorar}} \cdot \frac{0.5}{\text{Probabilidad de que la acción elegida sea } a_2} = 0.25$$

Podemos asumir que la elección de un método u otro se realizará cuando todas las acciones hayan sido probadas, al menos, una vez.

- Si las recompensas son **valores únicos** ( $\sigma = 0$ ), elegiremos siempre la acción con mejores resultados.
  - En este caso, **greedy** es mejor.
- Si las recompensas se corresponden con una **distribución de probabilidad** ( $\sigma > 0$ ), nos interesa no perder la posibilidad de explorar.
  - Por tanto, es mejor  $\epsilon$ -**greedy**.
  - Especialmente en problemas con *noisier rewards* → mayor varianza de las distribuciones.

## Problema no estacionario

Decimos que un problema de decisión es **no estacionario** si las distribuciones de recompensa varían con el tiempo.



- Una acción, *a priori*, mala puede mejorar con el tiempo, y viceversa.
- Es un fenómeno muy común en aprendizaje por refuerzo.

En este tipo de problemas, la mejor estrategia es  $\varepsilon$ -greedy, porque nunca se descarta la posibilidad de explorar y, por tanto, de **reaprender las distribuciones de recompensa**.

# **ACTION-VALUES: CÁLCULO INCREMENTAL**

---

Previamente hemos propuesto estimar el valor de las acciones de la siguiente forma:

$$Q_t(a) = \frac{\sum_{i=1}^{t-1} R_{i,a}}{\sum_{i=1}^{t-1} n_{i,a}}$$

El problema de este cálculo es que requiere mantener en **memoria** todas las recompensas obtenidas para cada acción en el tiempo.

- En problemas con un gran espacio de acciones, o prolongados en el tiempo, este método es inviable en términos de escalabilidad.

**SOLUCIÓN:** **cálculo incremental de la media.**

Si desarrollamos la fórmula para el cálculo del *action-value* medio, podemos hacer que este cálculo sea **incremental**.

No depende de todas las recompensas anteriores, sino únicamente del ***action-value actual*** y de la **última recompensa** obtenida.

$$\begin{aligned} Q_{n+1} &= \frac{1}{n} \sum_{i=1}^n R_i \\ &= \frac{1}{n} \left( R_n + \sum_{i=1}^{n-1} R_i \right) \\ &= \frac{1}{n} \left( R_n + (n-1) \frac{1}{n-1} \sum_{i=1}^{n-1} R_i \right) \\ &= \frac{1}{n} (R_n + (n-1)Q_n) \\ &= \frac{1}{n} (R_n + nQ_n - Q_n) \\ &= Q_n + \frac{1}{n} (R_n - Q_n) \end{aligned}$$

$$Q_{n+1} = \frac{1}{n} \sum_{i=1}^n R_i = Q_n + \frac{1}{n}(R_n - Q_n)$$

Se trata de una regla de actualización incremental (*incremental update rule*) bastante frecuente en RL:

nuevoValor  $\leftarrow$  valorActual + stepSize(objetivo – valorActual)

O bien:

$$v_t \leftarrow v_t + \alpha[G_t - v_t], \alpha \in (0, 1]$$

$$Q(A) \leftarrow Q(A) + \frac{1}{N(A)} \cdot \underbrace{\begin{bmatrix} R_{\text{objetivo}} - Q(A)_{\text{estimación actual}} \end{bmatrix}}_{\text{error de estimación}}$$

step size

- El **error de estimación** se reduce a medida que las **estimaciones** se acercan al **objetivo**.
- Indica la diferencia entre la recompensa obtenida y el valor actual.
  - Determina cuánto nos hemos equivocado en nuestra estimación más reciente.

$$Q(A) \leftarrow Q(A) + \underbrace{\frac{1}{N(A)}}_{\text{step size}} \cdot \underbrace{\begin{bmatrix} R & - Q(A) \\ \text{objetivo} & \text{estimación actual} \end{bmatrix}}_{\text{error de estimación}}$$

- El **step-size** pondera la importancia que damos al error de estimación.
  - Determina el peso de la nueva información recibida.
- Lo que hacemos es añadir un pequeño ajuste al valor anterior de la acción, que depende de la diferencia entre la recompensa obtenida y nuestra estimación anterior del valor de la acción.

# Ejemplo

# Ejemplo: piedra, papel, tijeras

$R = +1$  si se gana

$R = 0$  si se pierde o empata

$$\alpha = \frac{1}{N}$$

| t   | Yo  | Rival   | Recompensa | Actualización  | $Q(\text{tijeras})$ |
|-----|---|---|------------|--|---------------------|
| 0   | -   | -   | -          | -  | 0                   |
| 1   |  |  | 0          | $Q(\text{tijeras}) = 0 + 1(0 - 0)$                   | 0                   |
| 2   |  |  | +1         | $Q(\text{tijeras}) = 0 + \frac{1}{2}(1 - 0)$         | 0.1                 |
| 3   |  |  | 0          | $Q(\text{tijeras}) = 0.5 + \frac{1}{3}(0 - 0.5)$     | 0.09                |
| 4   |  |  | 0          | $Q(\text{tijeras}) = 0.335 + \frac{1}{4}(0 - 0.335)$ | 0.081               |
| ... | ...   | ...   | ...        | ...  | ...                 |
| N-1 | -   | -   | -          | -  | 0.33                |

¿Y si variamos el *step size*?       $\alpha = 0.1$

| <b>t</b> | <b>Yo</b>   | <b>Rival</b>  | <b>Recompensa</b> | <b>Actualización</b>                         | <b><math>Q(\text{tijeras})</math></b> |
|----------|---|---|-------------------|--|---------------------------------------|
| 0        | -   | -   | -                 | -  | 0                                     |
| 1        |  |  | 0                 | $Q(\text{tijeras}) = 0 + 0.1(0 - 0)$         | 0                                     |
| 2        |  |  | +1                | $Q(\text{tijeras}) = 0 + 0.1(1 - 0)$         | 0.5                                   |
| 3        |  |  | 0                 | $Q(\text{tijeras}) = 0.5 + 0.1(0 - 0.5)$     | 0.335                                 |
| 4        |  |  | 0                 | $Q(\text{tijeras}) = 0.335 + 0.1(0 - 0.335)$ | 0.25125                               |
| ...      | ...   | ...   | ...               | ...  | ...                                   |
| N-1      | -   | -   | -                 | -  | 0.33                                  |

La principal diferencia entre *step sizes* es la **velocidad de convergencia**:

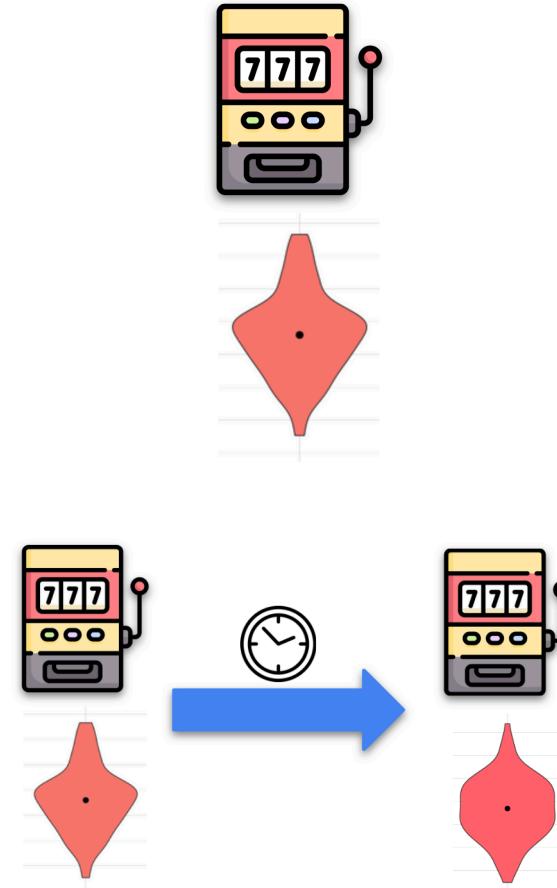
- $\alpha = \frac{1}{N}$  supone una convergencia más lenta a medida que aumenta  $N$ . Esto provoca que actualizaciones más pequeñas y menos impactantes con el paso del tiempo.

Deseable si queremos dar más peso a las experiencias tempranas.

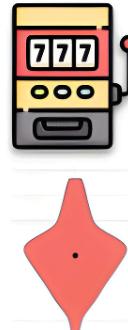
- Si se utiliza un *step size* constante,  $\alpha \in (0, 1]$ , la estimación del *action-value* converge más rápido hacia su valor real, pero es más sensible a experiencias recientes.

Más efectivo cuando se desea dar más peso a las experiencias recientes.

- En **problemas estacionarios**, los métodos basados en media muestral (*average sampling*) son apropiados, porque las distribuciones de probabilidad de las recompensas no varían con el tiempo.
  - Es decir, preferimos  $\alpha = \frac{1}{N}$
- En **problemas no estacionarios**, es más importante dar mayor peso a las recompensas recientes.
  - Por tanto, optamos por  $\alpha \in (0, 1]$



## Estacionario



$$\begin{aligned} r_1 &= 1 \\ r_2 &= 0 \\ r_3 &= 3 \\ r_4 &= 0 \\ \dots & \end{aligned}$$

$$Q_t(a) = \frac{\sum_{i=1}^{t-1} R_{i,a}}{\sum_{i=1}^{t-1} n_{i,a}}$$

$$Q_{n+1} = Q_n + \frac{1}{n}(R_n - Q_n)$$

## No estacionario



$$\begin{aligned} r_1 &= 1 \\ r_2 &= 0 \\ r_3 &= 3 \\ r_4 &= 0 \\ \dots & \end{aligned}$$



$$\begin{aligned} r_{k+1} &= -10 \\ r_{k+2} &= -12 \\ r_{k+3} &= -15 \\ r_{k+4} &= -14 \\ \dots & \end{aligned}$$

$$Q_{n+1} = Q_n + \alpha(R_n - Q_n), \quad \alpha \in (0, 1]$$

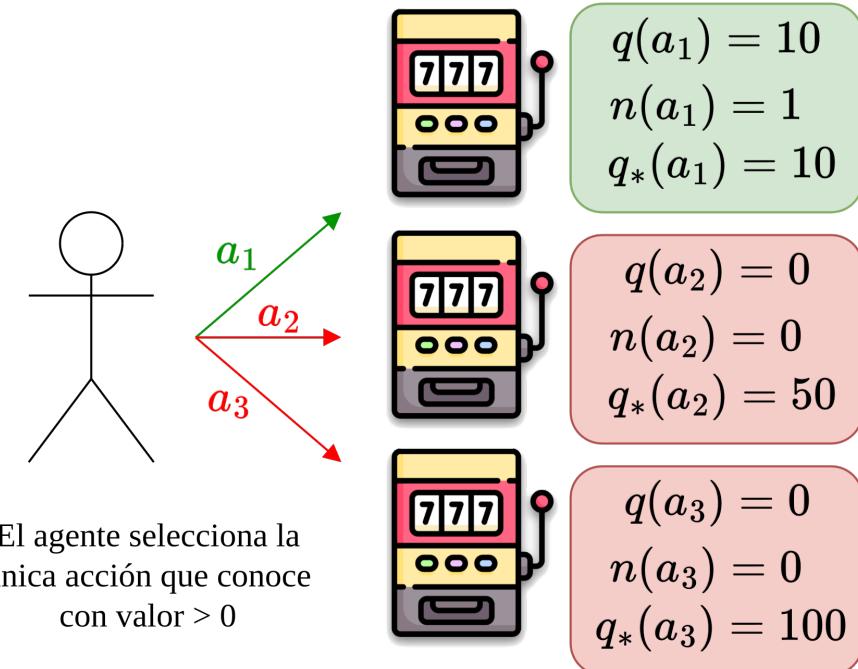
# **VALORES INICIALES**

---

Los métodos que hemos visto dependen en gran medida de las estimaciones iniciales de los *action-values*.

Esto supone un **sesgo** (*bias*).

- En el método de la **media muestral**, si inicialmente todas las acciones tienen valor 0, habrá un sesgo hacia la primera acción de la que se obtenga una recompensa  $> 0$ .
- **El sesgo desaparece una vez hemos seleccionado todas las acciones posibles.**



En métodos con **step size constante**, el sesgo es permanente, aunque decrece con el tiempo:

$$\begin{aligned} Q_{n+1} &= Q_n + \alpha[R_n - Q_n] \\ &= (1 - \alpha)^n Q_1 + \sum_{i=1}^n \alpha(1 - \alpha)^{n-i} R_i \end{aligned}$$

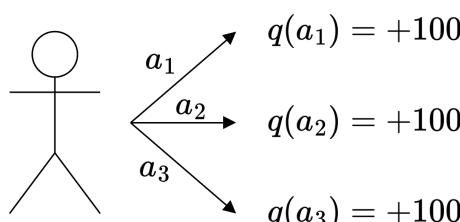
En métodos con **step size constante**, el sesgo es permanente, aunque decrece con el tiempo:

$$\begin{aligned} Q_{n+1} &= Q_n + \alpha[R_n - Q_n] \\ &= (1 - \alpha)^n Q_1 + \underbrace{\sum_{i=1}^n \alpha(1 - \alpha)^{n-i} R_i}_{\text{A mayor número de experiencias pasadas (n) menor peso tienen las recompensas anteriores}} \end{aligned}$$

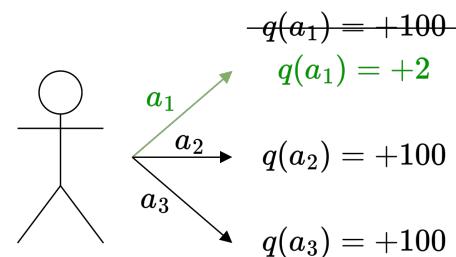
En la práctica, el sesgo no suele ser un problema y a veces puede resultar útil.

- Las estimaciones iniciales pueden proporcionar **conocimiento previo/experto** sobre qué recompensas podemos esperar de cada acción.
- Un inconveniente es que estas estimaciones iniciales se convierten en un **conjunto de parámetros que el usuario debe elegir**, aunque por defecto pueden ser = 0.

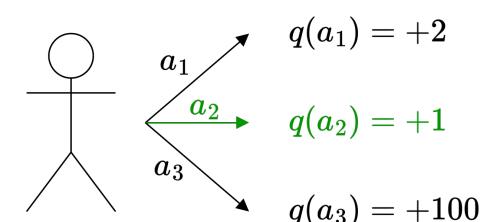
Podemos utilizar el sesgo para guiar la exploración inicial de nuestro agente. Por ejemplo:



Se asignan falsos valores iniciales (+100) a cada acción, a pesar de que los valores reales estén en el intervalo [-2, +2].



Se elige una acción en base a su valor estimado inicial. Podría ser la mejor, pero sigue siendo peor que los valores iniciales del resto de acciones.



De esta forma, se aprovecha el sesgo para favorecer naturalmente la **exploración** inicial de todas las acciones posibles.

Utilizamos el sesgo para provocar la exploración inicial de todas/algunas acciones.

**Esto permite que incluso un método *greedy explore*.**

- Se denomina ***optimistic greedy***, porque emplea valores iniciales optimistas.
  - Puede dar lugar a mejores resultados que un  $\varepsilon$ -*greedy* estándar.
- La principal limitación es que la exploración es simplemente inicial (disminuye con el tiempo hasta desaparecer).
  - Esto hace que no sea útil en problemas **no estacionarios**.

«*The beginning of time occurs only once, and thus we should not focus on it too much*».



Sutton, R. S., & Barto, A. G. (2018). *Reinforcement learning: An introduction* (2nd ed.). MIT press. (p. 35).

# **UPPER-CONFIDENCE-BOUND**

---

$\varepsilon$ -greedy fuerza la selección de acciones *non-greedy* de forma indiscriminada:

$$A_t = \begin{cases} \operatorname{argmax}_a Q_t(a) & \text{con prob. } 1 - \varepsilon \\ a \sim \text{Uniform}(\{a_1, a_2, \dots, a_k\}) & \text{con prob. } \varepsilon \end{cases}$$

- Todas tienen la misma probabilidad.
- No hay preferencia por aquellas más cercanas al valor *greedy*, o aquellas menos visitadas/desconocidas.

Sería interesante explorar acciones ***non-greedy*** de acuerdo a su **potencial** para ser óptimas.

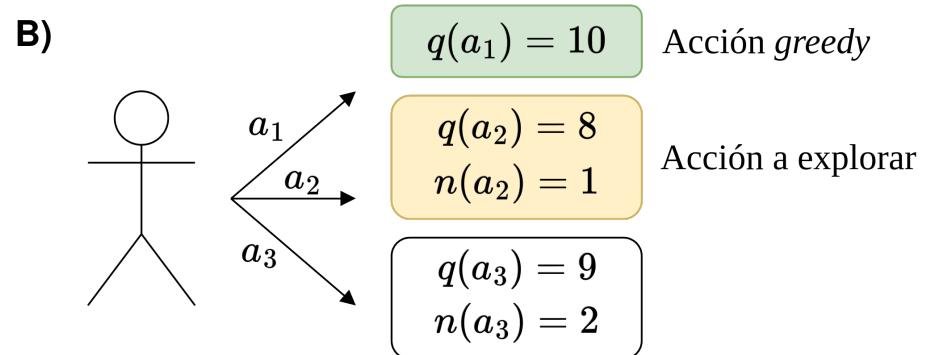
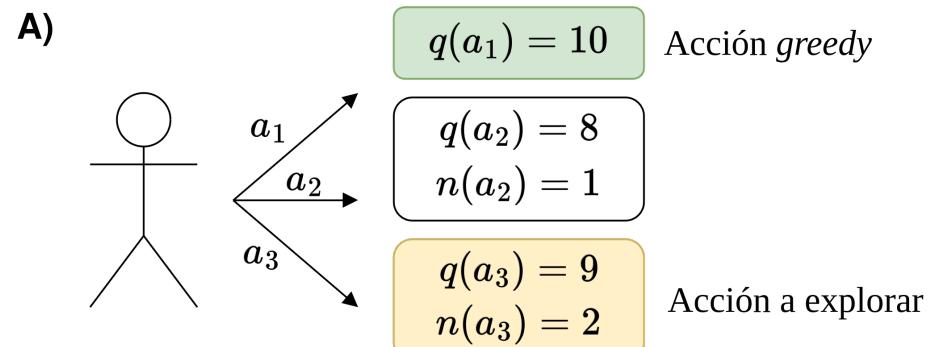
Para decidir qué acción explorar, podemos considerar:

## A) La cercanía al **valor máximo actual**

- El valor de la acción *greedy*

## B) La **incertidumbre** en las estimaciones.

- Qué acciones se han realizado menos ( $n_i$ )



**Si los combinamos...**

El método del **límite superior de confianza**, o *Upper-confidence-bound (UCB)*, nos permite balancear **valor** e **incertidumbre** a la hora de seleccionar acciones:

$$A_t = \operatorname{argmax}_a \left[ Q_t(a) + c \sqrt{\frac{\ln t}{N_t(a)}} \right]$$

El método del **límite superior de confianza**, o *Upper-confidence-bound (UCB)*, nos permite balancear **valor** e **incertidumbre** a la hora de seleccionar acciones:

$$A_t = \operatorname{argmax}_a \left[ Q_t(a) + c \sqrt{\frac{\ln t}{N_t(a)}} \right]$$

- $c > 0$  controla cuánto explorar.
- $N_t(a)$  indica el número de selecciones previas de la acción  $a$ .
- Si  $N_t(a) = 0$ , se considera  $a$  como la acción más preferible

$$A_t = \operatorname{argmax}_a \left[ \underbrace{Q_t(a)}_{\text{valor estimado}} + \underbrace{c \sqrt{\frac{\ln t}{N_t(a)}}}_{\text{incertidumbre}} \right]$$

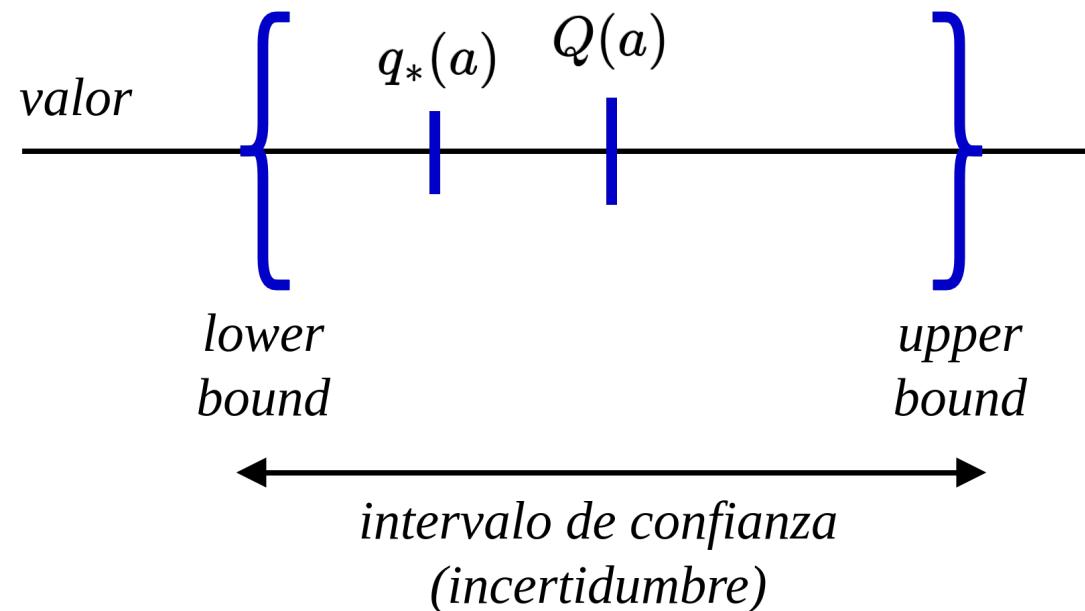
La selección de una acción depende de:

1. Su **valor estimado** hasta el momento.
2. La **incertidumbre** sobre dicha acción.
  - Cada vez que una acción se selecciona, su incertidumbre se reduce.
  - Según pasa el tiempo, la incertidumbre sobre una acción vuelve a aumentar poco a poco.

El coeficiente  $c$  pondera la importancia que damos a la exploración.

UCB reduce la exploración con el tiempo (el término de incertidumbre tiende a 0).

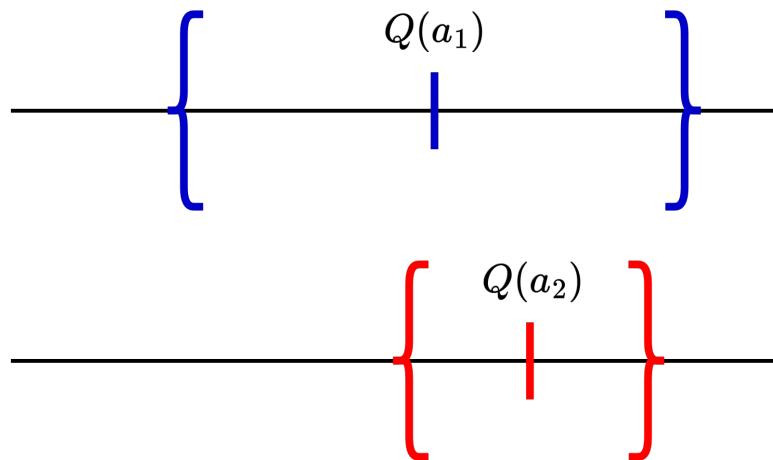
Definimos **intervalos de confianza** dentro de los cuales se encuentran los valores originales de las acciones y, por tanto, sus estimaciones:



**Opción 1.** Elegir la acción con mayor incertidumbre (**exploración**).

- A mayor incertidumbre, mayor creencia de que es bueno (*optimismo en presencia de incertidumbre*).
- Elegimos la acción  $a_1$  en base a

$$c \sqrt{\frac{\ln t}{N_t(a)}}$$



**Opción 2.** Elegir la acción con mayor valor estimado (**explotación**).

- Elegimos la acción  $a_2$  en base a  $Q_t(a)$ .

# **TRABAJO PROPUESTO**

---

- **Implementación y comparativa** de los métodos *greedy*,  $\varepsilon$ -*greedy* y UCB para un problema formalizado como *K*-armed bandits.
- **Thompson sampling**
  - Definición y características.
  - Diferencias y similitudes con los métodos vistos.
- **Contextual bandits**
  - ¿Qué son?
  - ¿Qué relación podrían tener con las próximas lecciones?

## Bibliografía y recursos

- **Capítulo 2** de Sutton, R. S., & Barto, A. G. (2018). Reinforcement learning: An introduction.
- <https://www.ma.imperial.ac.uk/~cpikebur/trybandits/trybandits.html>
- <https://rlplaygrounds.com/reinforcement/learning/Bandits.html>
- [https://youtu.be/bkw6hWvh\\_3k](https://youtu.be/bkw6hWvh_3k)

# **APRENDIZAJE POR REFUERZO**

*Bandits*

---

Antonio Manjavacas

[manjavacas@ugr.es](mailto:manjavacas@ugr.es)