Modelos de toma de decisiones de humanos basado en aprendizaje reforzado

Alejandro Weinstein
alejandro.weinstein@uv.cl
@ajweinstein

`https://github.com/aweinstein/evic2017`

Escuela de Ingeniería Biomédica de la Universidad de Valparaíso
Advanced Center for Electrical and Electronics Engineering

EVIC 2017, 13 de diciembre de 2017

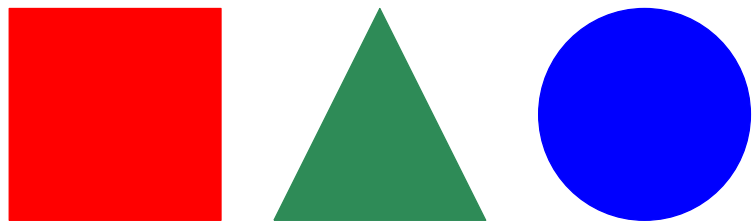
Rescorla y Wagner:

"...organisms only learn when events violate their expectations."

R. A. Rescorla and A. R. Wagner, A theory of Pavlovian conditioning: Variations in the effectiveness of reinforcement and non reinforcement, Classical conditioning II: Current research and theory, vol. 2, pp. 64-99, 1972.

Condición experimental típica

- El sujeto debe elegir entre un conjunto de opciones. Por ejemplo, debe presionar uno de los siguientes botones:



- Según la opción elegida, recibe una recompensa.
- La secuencia se repite un número finito de veces.
- El objetivo es maximizar la recompensa total.

Este problema también se conoce como *Multi-armed bandit*

Condición experimental típica

En cada intento (trial) $t = 1, \ldots, T$, las recompensas se generan según las probabilidades:[1]

$$P(r_t = 1 | a_t = \blacksquare) = 0.8$$
$$P(r_t = -1 | a_t = \blacksquare) = 0.2$$
$$P(r_t = 1 | a_t = \blacktriangle) = 0.5$$
$$P(r_t = -1 | a_t = \blacktriangle) = 0.5$$
$$P(r_t = 1 | a_t = \bullet) = 0.2$$
$$P(r_t = -1 | a_t = \bullet) = 0.8$$

Donde T es el número de intentos disponibles, a_t es la acción realizada en el trial t, y r_t es la recompensa obtenida en el trial t.

[1] Valores de ejemplo. Tanto los valores posibles que puede tener r_t, así como las probabilidades dependen de cada situación.

Condición experimental típica

¡El sujeto no conoce estas probabilidades!

Aprendizaje Reforzado (RL)

El problema se puede modelar como la interacción entre un *agente* y el *ambiente*:

El agente puede ser artificial o un ser vivo.

Los dos problemas:

- ¿Cómo modelamos el mecanismo utilizado por el agente para tomar las decisiones?
- Bajo la suposición de un modelo dado, y de las observaciones

$$(a_1, r_1), (a_2, r_2), \ldots (a_T, r_T)$$

de las interacciones entre un agente y el ambiente, ¿cómo ajustamos los parámetros de dicho modelo?

¿Para qué?

- Las diferencias entre los parámetros de distintos sujetos de una población, permite caracterizar a dichos sujetos.
- Correlacionar alguna variable del modelo con variables fisiológicas (EEG, fMRI, etc.) registradas en forma simultanea a la realización del experimento.
- Comparar distintos modelos.

El modelo “action-value”

Objetivo: Elegir secuencia de acciones de tal forma de que se maximice la recompensa total

$$R = \sum_{t=1}^{T} r_t.$$

El modelo “action-value”

En cada iteración el agente actualiza el *action-value* según la regla

$$Q_{t+1}(a_t) = Q_t(a_t) + \alpha(r_t - Q_t(a_t))$$

y selecciona una acción en forma estocástica usando la regla *softmax* que asigna probabilidades a cada acción según

$$P(a_t = a_j) = \frac{e^{\beta Q_t(a)}}{\sum_{i=1}^{|\mathcal{A}|} e^{\beta Q_t(a_i)}}, \quad j \in \mathcal{A},$$

donde los parámetros del modelo son la tasa de aprendizaje (*learning rate*) α y la temperatura inversa β, y $\mathcal{A}$ es el conjunto de posibles acciones.

El modelo “action-value”

Es útil pensar en Q_t como en una tabla que se actualiza en la medida que se interactúa con el ambiente. Por ejemplo:

	■	▲	●
Q_t	0.4	-0.1	-0.5

$$a_t = \blacktriangle, r_t = 1, \alpha = 0.2$$

$$-0.1 + 0.2(-0.1 + 1) = 0.08$$

$$\downarrow$$

	■	▲	●
Q_{t+1}	0.4	0.08	-0.5

Q es una estimación del valor esperado de cada acción

El modelo “action-value”

$$Q_{t+1}(a_t) = Q_t(a_t) + \alpha(\underbrace{r_t - Q_t(a_t)}_{?})$$

El modelo “action-value”

$$Q_{t+1}(a_t) = Q_t(a_t) + \alpha(\underbrace{r_t - Q_t(a_t)}_{?})$$

$$Q_{t+1}(a_t) = Q_t(a_t) + \alpha(\underbrace{r_t - Q_t(a_t)}_{\text{Señal de error}})$$

En la literatura, a esta señal de error se le conoce como “Temporal Difference”.

El modelo “action-value”

$$Q_{t+1}(a_t) = Q_t(a_t) + \alpha(\underbrace{r_t - Q_t(a_t)}_{?})$$

$$Q_{t+1}(a_t) = Q_t(a_t) + \alpha(\underbrace{r_t - Q_t(a_t)}_{\text{Señal de error}})$$

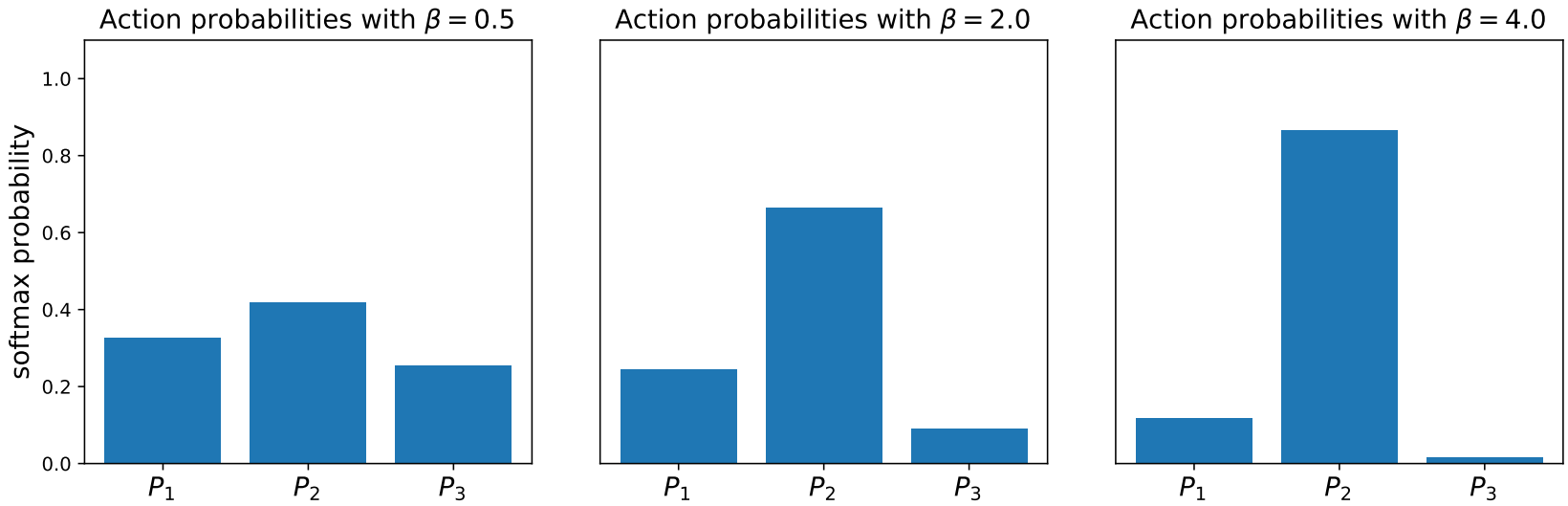
En la literatura, a esta señal de error se le conoce como “Temporal Difference”.

Rescorla y Wagner:

“...organisms only learn when events violate their expectations.”

Sobre softmax

Para todos los casos, $Q(\blacksquare) = 5, Q(\blacktriangle) = 5.5, Q(\bullet) = 4.5$



En general,

$$\lim_{\beta \to 0} \frac{e^{\beta x_j}}{\sum e^{\beta x_i}} = \frac{1}{|\mathcal{A}|}$$

$$\lim_{\beta \to \infty} \frac{e^{\beta x_j}}{\sum e^{\beta x_i}} = \begin{cases} 1 & \text{si } x_j > x_i \\ 0 & \text{si } x_j < x_i \end{cases} \qquad j \neq i.$$

Notebook demo...

Implementando el modelo “action-value”

```
class Environment(object):
    def __init__(self):
        self.actions = ('square',
                        'triangle',
                        'circle')
        self.prob_win = {'square': 0.8,
                         'triangle': 0.5,
                         'circle': 0.2}
        self.n = len(self.actions)

    def reward(self, action):
        p = self.prob_win[action]
        if np.random.rand() < p:
            r = 1
        else:
            r = -1
        return r
```

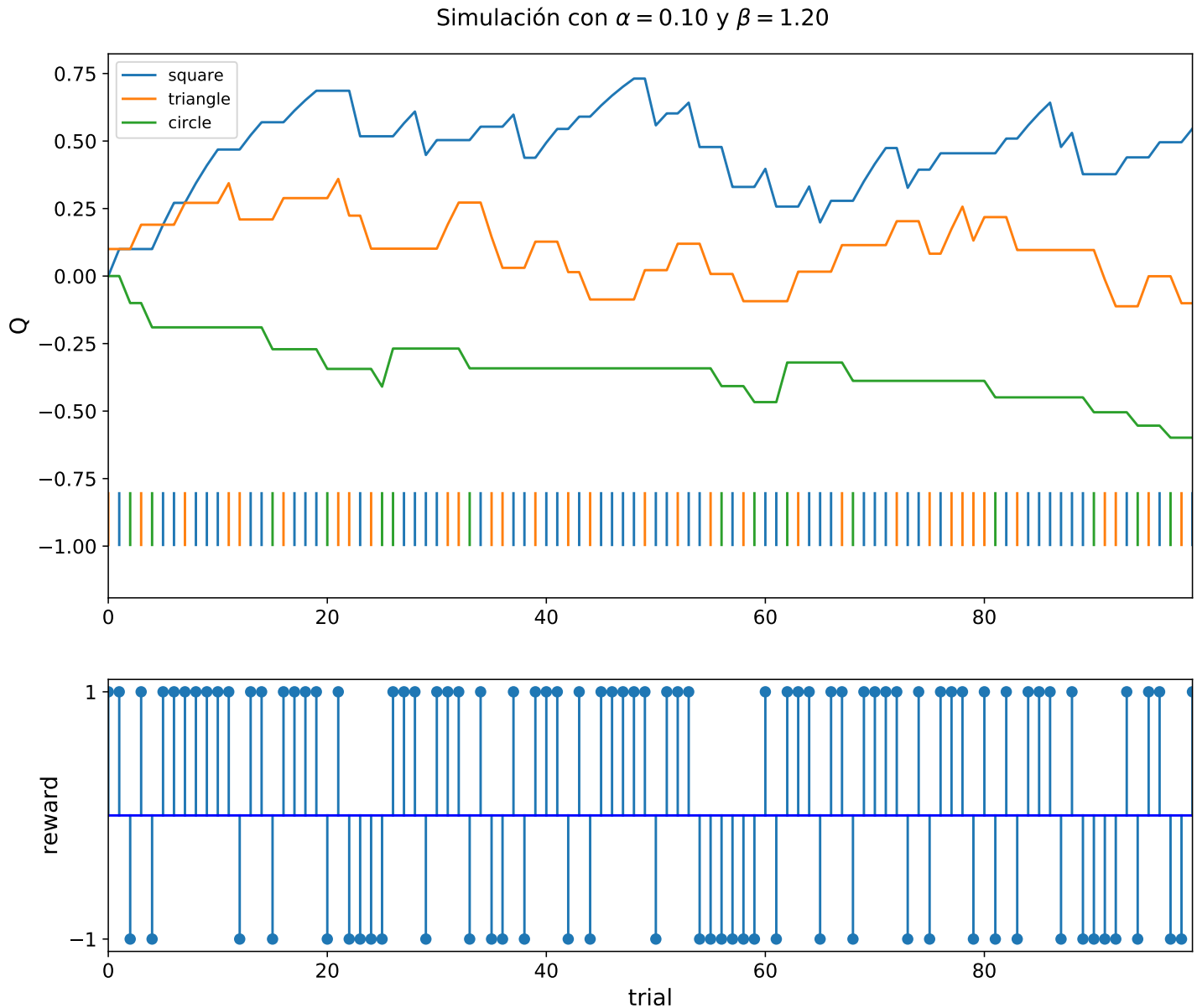
```
class Agent(object):
  def __init__(self, environment, alpha, beta):
      # ...
  def run(self):
      action = self.choose_action()
      action_l = self.actions[action])
      reward = self.environment.reward(action_l)
      self.update_action_value(action, reward)

  def choose_action(self):
      p = softmax(self.Q, self.beta)
      actions = range(self.n)
      action = np.random.choice(actions, p=p)
      return action

  def update_action_value(self, action, reward):
      error = reward - self.Q[action]
      self.Q[action] += self.alpha * error
```

```
if __name__ == '__main__':
    env = Environment()
    agent = Agent(env, 0.1, 1.2)
    T = 100
    for _ in range(T):
        agent.run()
```

Implementando el modelo “action-value”



Estimación de los parámetros del modelo

Problema: A partir de la secuencia de pares acción-recompensa

$$(a_1, r_1), (a_2, r_2), \ldots (a_T, r_T)$$

observados durante la interacción de un agente con un ambiente dado, estimar los parámetros del modelo que se asume utiliza dicho agente. Por ejemplo, estimar α y β del modelo "action-value".

Daw, Nathaniel D. "Trial-by-trial data analysis using computational models." Decision making, affect, and learning: Attention and performance XXIII 23 (2011): 3-38.

Estimación de los parámetros del modelo

Los parámetros del modelo se obtienen utilizando el principio de máxima verosimilitud (máximum likelihood). La verosimilitud de los parámetros está dada por

$$\mathcal{L}(\alpha, \beta) = \prod_{t=1}^{T} P(a_t, c_t).$$

Si asumimos el modelo "action-value", la probabilidad $P(a_t, c_t)$ se calcula usando las reglas de actualización del action-value y de softmax. Luego,

$$\widehat{\alpha}, \widehat{\beta} = \underset{0 \leq \alpha \leq 1, 0 \leq \beta}{\operatorname{argmin}} \; -\log(\mathcal{L}(\alpha, \beta)).$$

En la práctica es conveniente asignar un límite superior a β, por ejemplo, $0 \leq \beta \leq 2$.

Cálculo de la función de verosimilitud

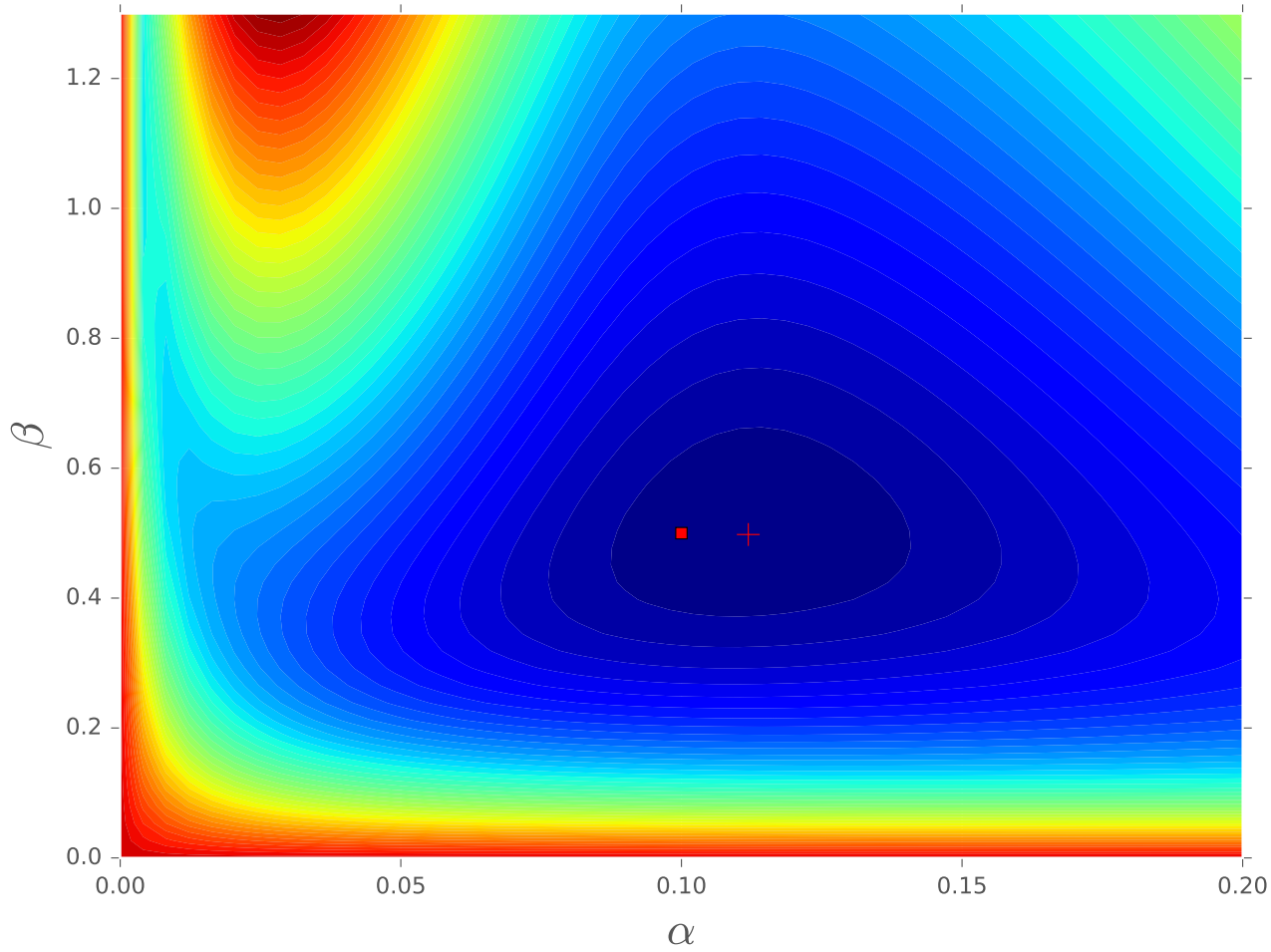
Supongamos que observamos la secuencia $(▲, 1), (●, -1)(●, -1)$. La función Q_t evoluciona como (asumiendo valor inicial 0):

$$Q_1(▲) = 0 + \alpha(1 - 0) = \alpha$$
$$Q_2(●) = 0 + \alpha(-1 - 0) = -\alpha$$
$$Q_3(●) = -\alpha + \alpha(-1 - (-\alpha)) = -2\alpha + \alpha^2.$$

Y la función de verosimilitud es

$$\begin{aligned}\mathcal{L}(\alpha, \beta) = &\text{softmax}(▲, (0, 0, 0), \beta)\cdot \\ &\text{softmax}(●, (\alpha, 0, 0), \beta)\cdot \\ &\text{softmax}(●, (\alpha, 0, -\alpha), \beta).\end{aligned}$$

Ejemplo de función de verosimilitud



Los datos corresponden a un agente (artificial) operando con $\alpha = 0.1$ y $\beta = 0.5$ (cuadrado rojo). Los parámetros estimados utilizando el principio de máxima verosimilitud son $\widehat{\alpha} = 0.11$ y $\widehat{\beta} = 0.49$ (cruz roja).

Implementando estimación de parámetros usando ML

```
class ML(object):
    def __init__(self, log):
        self.rewards = log['reward']
        label_to_n = {'square': 0, 'triangle': 1, 'circle': 2}
        self.actions = [label_to_n[a] for a in log['action']]
        self.n_actions = 3

    def neg_log_likelihood(self, alphabeta):
        alpha, beta = alphabeta
        prob_log = 0
        Q = np.zeros(self.n_actions)
        for action, reward in zip(self.actions, self.rewards):
            Q[action] += alpha * (reward - Q[action])
            prob_log += np.log(softmax(Q, beta)[action])

        return -prob_log

    def fit(self):
        bounds = ((0, 1), (0, 2))
        r = minimize(self.neg_log_likelihood, [0.1, 0.1],
                     method='L-BFGS-B',
                     bounds=bounds)
        return r
```

Referencias anotadas

- Sutton, Richard S., and Andrew G. Barto. Reinforcement learning: An introduction. Vol. 1. No. 1. Cambridge: MIT press, Second Edition, 2017 (`http://incompleteideas.net/book/the-book-2nd.html`).

  Ver capítulo 2 para más detalles sobre el modelo "action-value", y capítulos 14 y 15 para relación entre RL con psicología y neurociencia.

- Daw, Nathaniel D. "Trial-by-trial data analysis using computational models." Decision making, affect, and learning: Attention and performance XXIII 23 (2011): 3-38.

  Todos los detalles de estimación de parámetros RL usando ML, incluyendo incerteza en los parámetros, y comparación de modelos.

Referencias anotadas

- Dayan, Peter, and Laurence F. Abbott. Theoretical neuroscience. Vol. 806. Cambridge, MA: MIT Press, 2001.

  Ver capítulo 9 para más detalles sobre el desarrollo del modelo "action-value", partiendo desde el condicionamiento clásico.

- Littman, Michael L. "Reinforcement learning improves behaviour from evaluative feedback." Nature 521.7553 (2015): 445-451.

  Revisión general de RL, con mención sobre el uso de RL como modelo de aprendizaje animal.

- Wiering, Marco, and Martijn van Otterlo, eds. Reinforcement Learning: State-of-the-Art. Vol. 12. Springer Science & Business Media, 2012.

  Ver capítulo 16 para más información sobre la relación entre RL y cognición, incluyendo rol de la dopamina y ganglio basal.

Referencias anotadas

- FitzGerald, Thomas HB, Raymond J. Dolan, and Karl Friston. “Dopamine, reward learning, and active inference.” Frontiers in computational neuroscience 9 (2015).

  Modelo alternativo de aprendizaje, basado en energía libre.