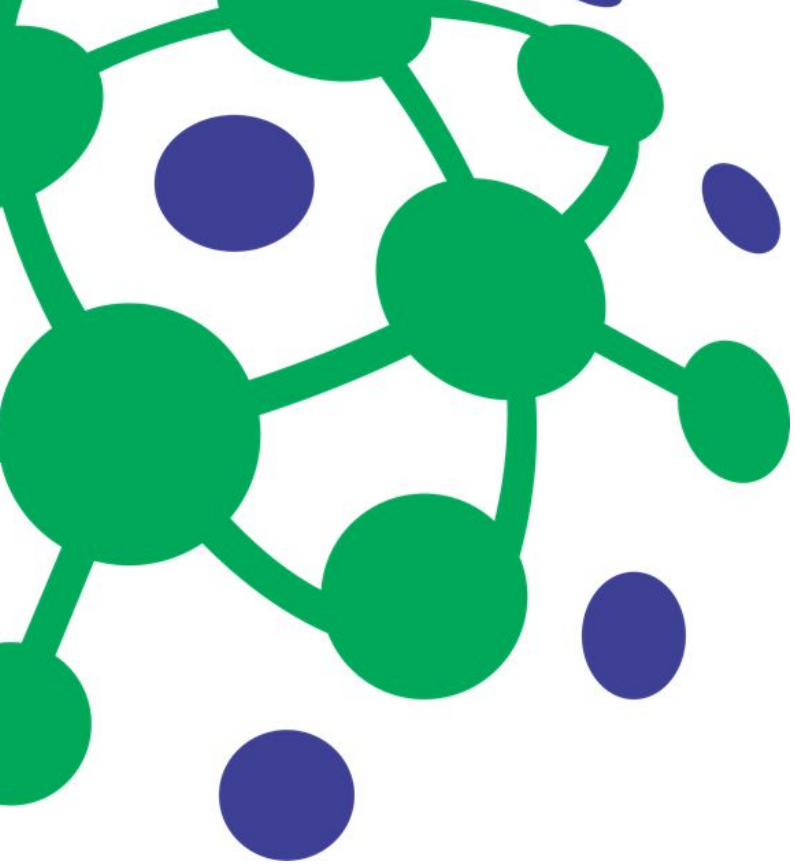


Primer Coloquio de Divulgación de la Comunidad de Ingeniería en Sistemas

Aprendizaje Automático en Python

Gustavo Adolfo Vargas Hákim

José Clemente Hernández Hernández

COVNNEC - App

Research Group on Computer Vision, Neural Networks,

Evolutionary Computation and their Applications

Agenda

1. Algoritmos de aprendizaje no supervisado.
2. Algoritmo de K Medias (K-Means)
3. Algoritmos de aprendizaje por refuerzo
4. Algoritmo de Aprendizaje Q (Q-Learning)

Aprendizaje no Supervisado

Aprendizaje no Supervisado

En este tipo de aprendizaje los datos no están etiquetados. Es decir, no se tiene la componente ***Y*** para supervisar que el modelo aprenda “bien”

¿Qué patrones se pueden encontrar dentro de dichos datos no etiquetados?

Aprendizaje no Supervisado

Aprendizaje no Supervisado

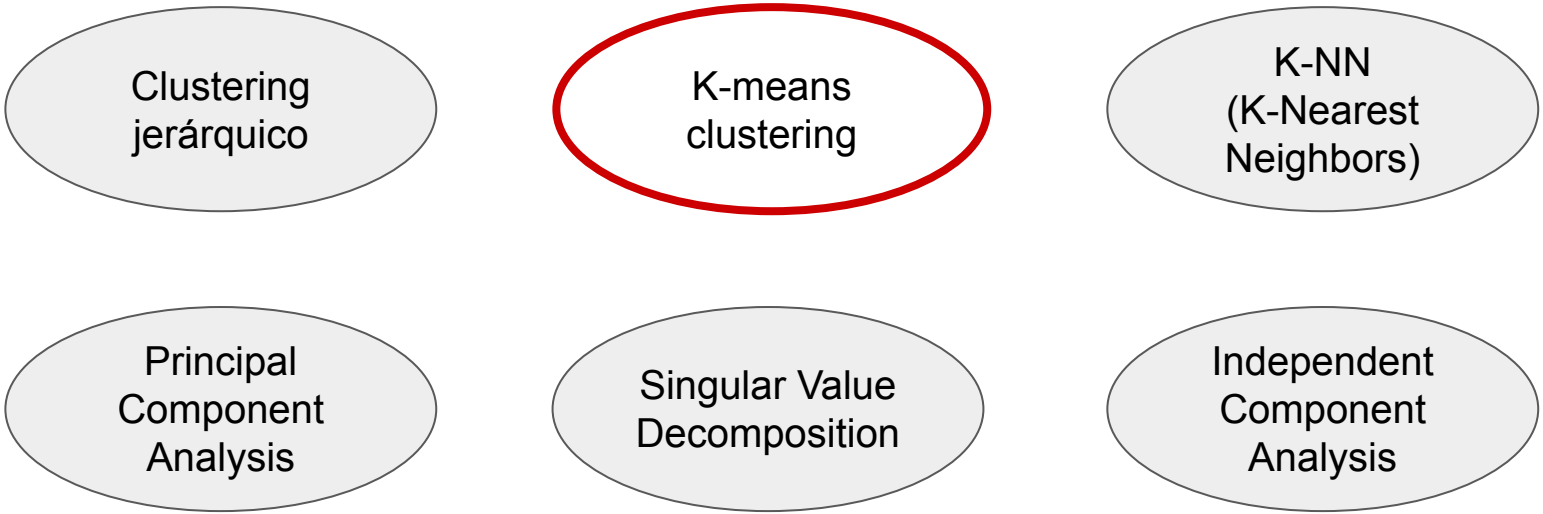
Esencial para:

- Buscar patrones desconocidos en los datos, ya que pueden encontrar características que mejoren el aprendizaje
- El mundo real, porque contiene más datos sin etiquetar que etiquetados. Los datos etiquetados necesitan la intervención humana

Aprendizaje no Supervisado

Clustering	Asociación
- Busca estructuras o patrones dentro de los datos no etiquetados - Procesa los datos buscando *clusters*	- Busca asociaciones dentro de las variables de los datos

Clustering

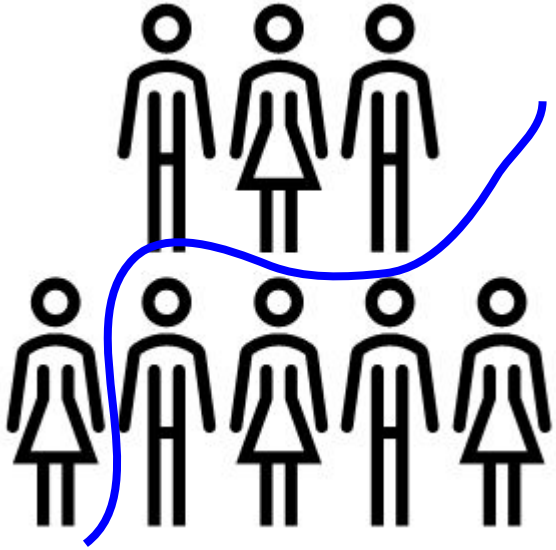

Clustering jerárquico
K-means clustering
K-NN (K-Nearest Neighbors)
Principal Component Analysis
Singular Value Decomposition
Independent Component Analysis

Aplicaciones del no Supervisado

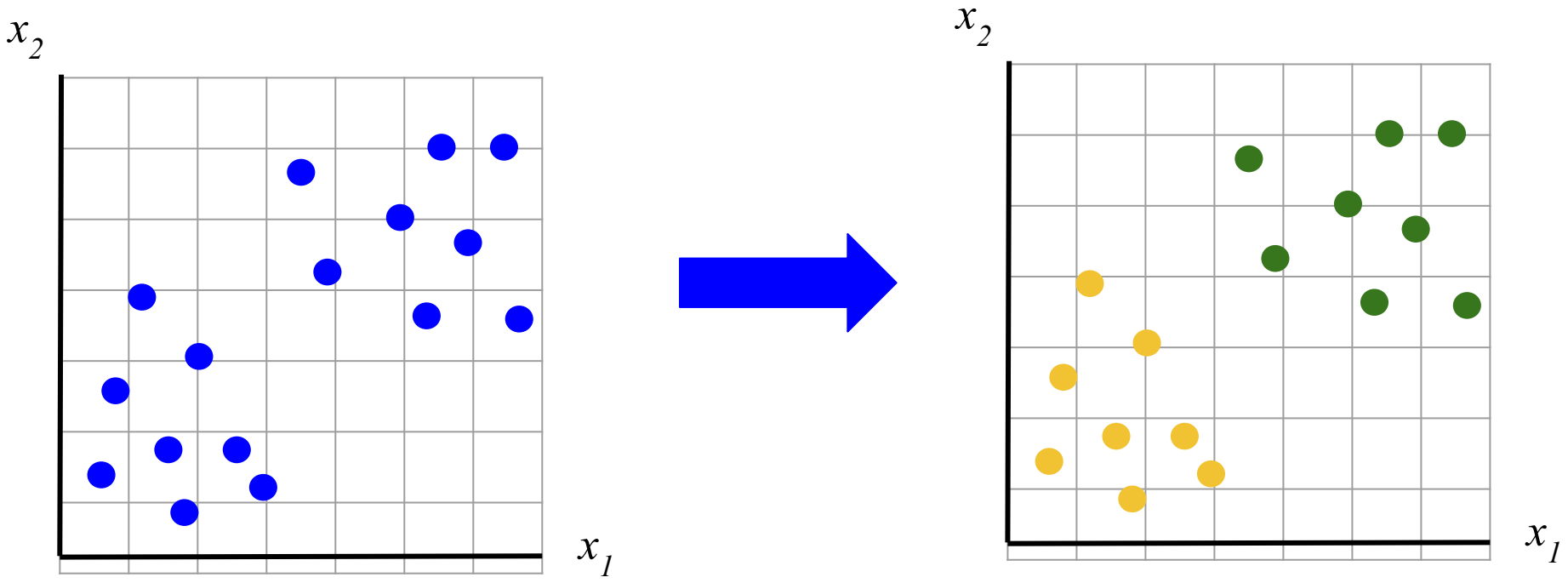
Dhanachandra, N., Manglem, K., Chanu, Y.J.: Image Segmentation Using K-means Clustering Algorithm and Subtractive Clustering Algorithm. Procedia Comput. Sci. 54, 764–771 (2015). https://doi.org/10.1016/j.procs.2015.06.090.

Aplicaciones del no Supervisado

Alkhayrat, M., Aljnidi, M. & Aljoumaa, K. A comparative dimensionality reduction study in telecom customer segmentation using deep learning and PCA. J Big Data 7, 9 (2020). https://doi.org/10.1186/s40537-020-0286-0



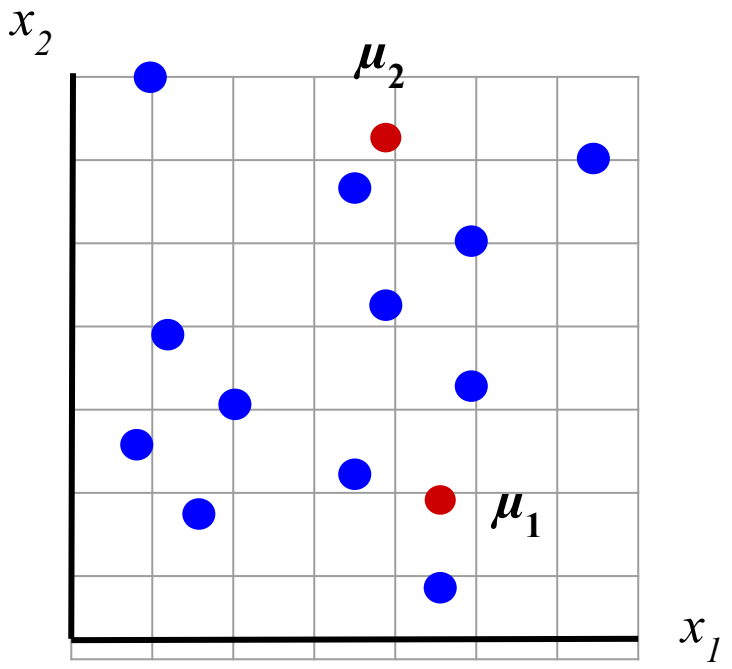
Algoritmo K-Means



Algoritmo K-Means

x_1	x_2	y
0.9	2.5	
1.7	1.8	
1.1	3.9	
2	3	
3.5	2.1	
4.6	0.8	

x_1	x_2	y
1	7	
3.5	5.7	
3.9	4.1	
5	3.1	
5	5	
6.5	6	



Centroides: $\mu_1 = (4.6, 1.8)$,

$\mu_2 = (3.9, 6.2)$

Algoritmo K-Means

$$\min_{\mathbf{X}} E(\boldsymbol{\mu}_i) = \min_{\mathbf{X}} \sum_{i=1}^{k} \sum_{x_j \in X_i} \| x_j - \boldsymbol{\mu}_i \|^2$$

Minimizar la suma de las distancias entre los k centroides y los vectores x_j, que pertenecen a un centroide i denotados por el conjunto X_i que, a su vez provienen del conjunto original de los datos $\mathbf{X}$

Algoritmo K-Means

$$\text{Distancia} = \| x_j - \mu_i \|^2$$

Calcular la distancia de los vectores x_j hacia cada centroide μ_i y elegir los centroides más cercanos para cada vector. ¿A qué centroide pertenece cada vector?

Algoritmo K-Means

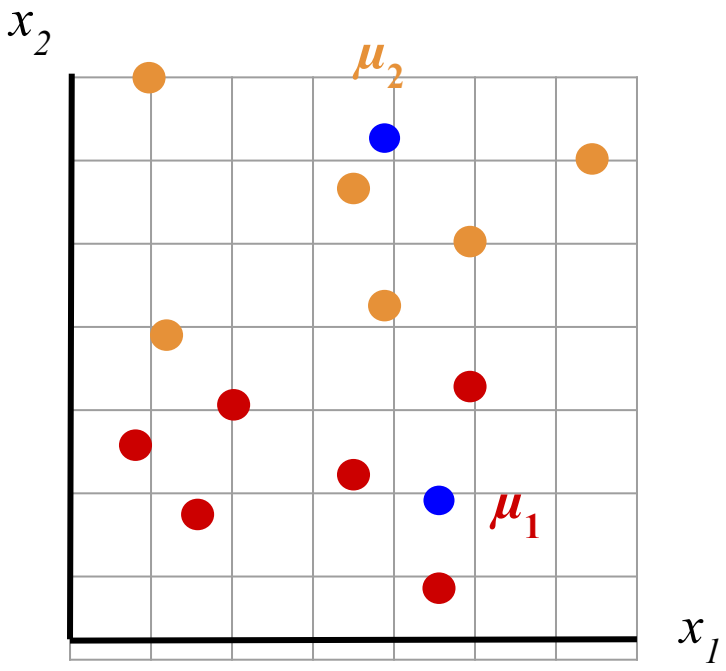
x_1	x_2	y	μ_1	μ_2
0.9	2.5	●	14.18	22.69
1.7	1.8	●	8.41	24.2
1.1	3.9	●	16.66	13.13
2	3	●	8.2	13.85
3.5	2.1	●	1.3	16.97
4.6	0.8	●	1	29.65

x_1	x_2	y	μ_1	μ_2
1	7	●	40	9.05
3.5	5.7	●	16.42	0.41
3.9	4.1	●	5.78	4.41
5	3.1	●	1.85	10.82
5	5	●	10.4	2.65
6.5	6	●	21.25	6.8

Centroides: $\mu_1 = (4.6, 1.8)$,

$\mu_2 = (3.9, 6.2)$

Algoritmo K-Means



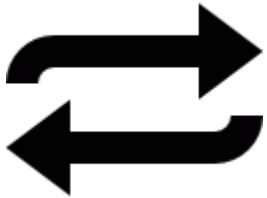
Algoritmo K-Means

$$\mu_i^{(t+1)} = \frac{1}{|X_i^{(t)}|} \sum_{x_j \in X_i^{(t)}} x_j$$

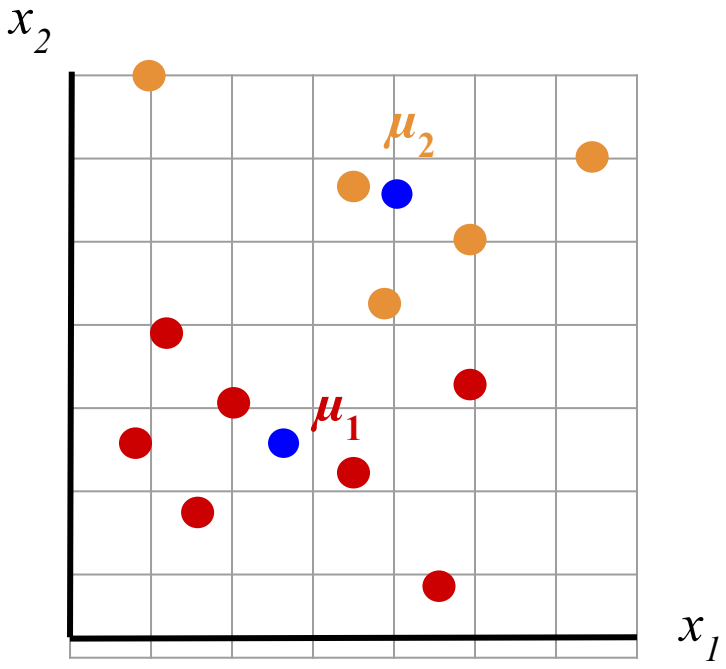
La actualización de los centroides en la siguiente iteración (t + 1), viene dada por el promedio de los vectores pertenecientes al conjunto X_i de ese centroide

Algoritmo K-Means

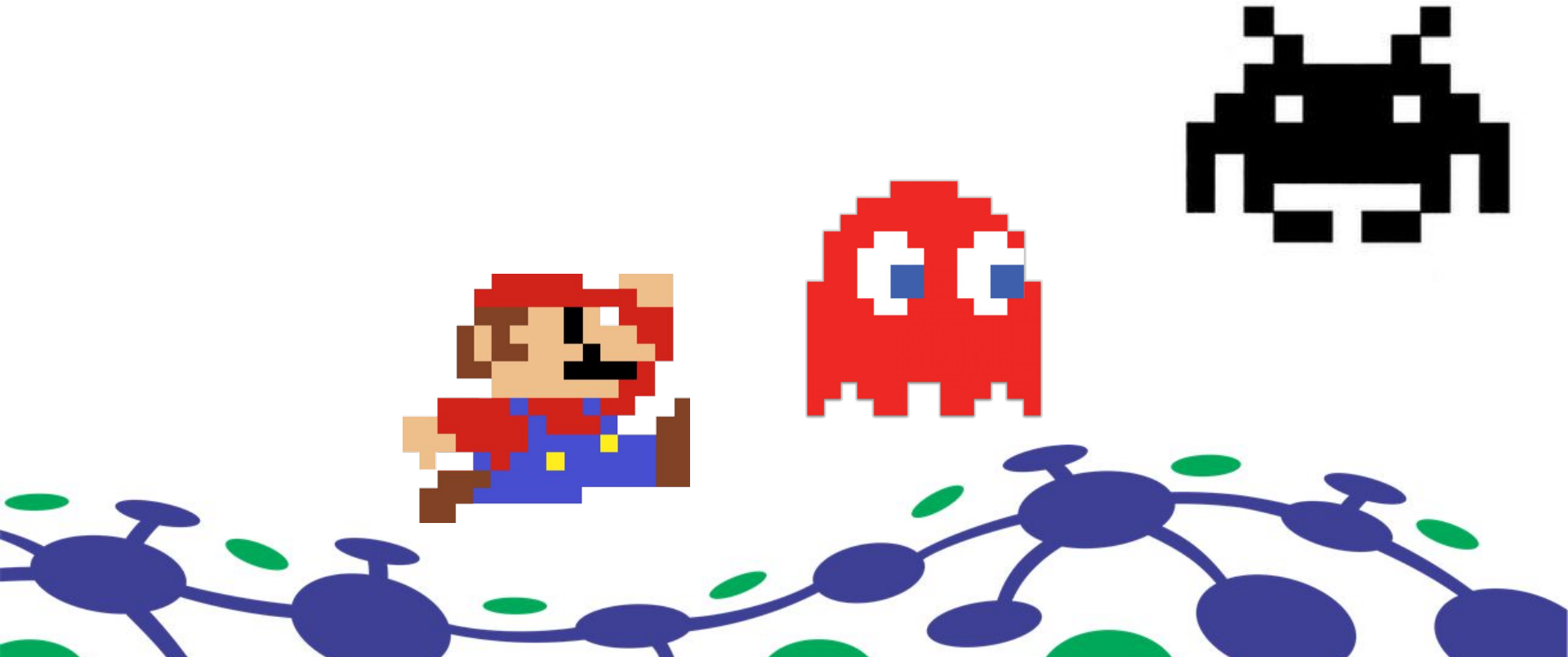
Se repite el procedimiento, calculando distancias a los nuevos centroides y actualizando con respecto a los nuevos conjuntos, hasta llegar a un número dado de iteraciones, umbral o hasta que los centroides no se muevan



Algoritmo K-Means



Aprendizaje por Refuerzo

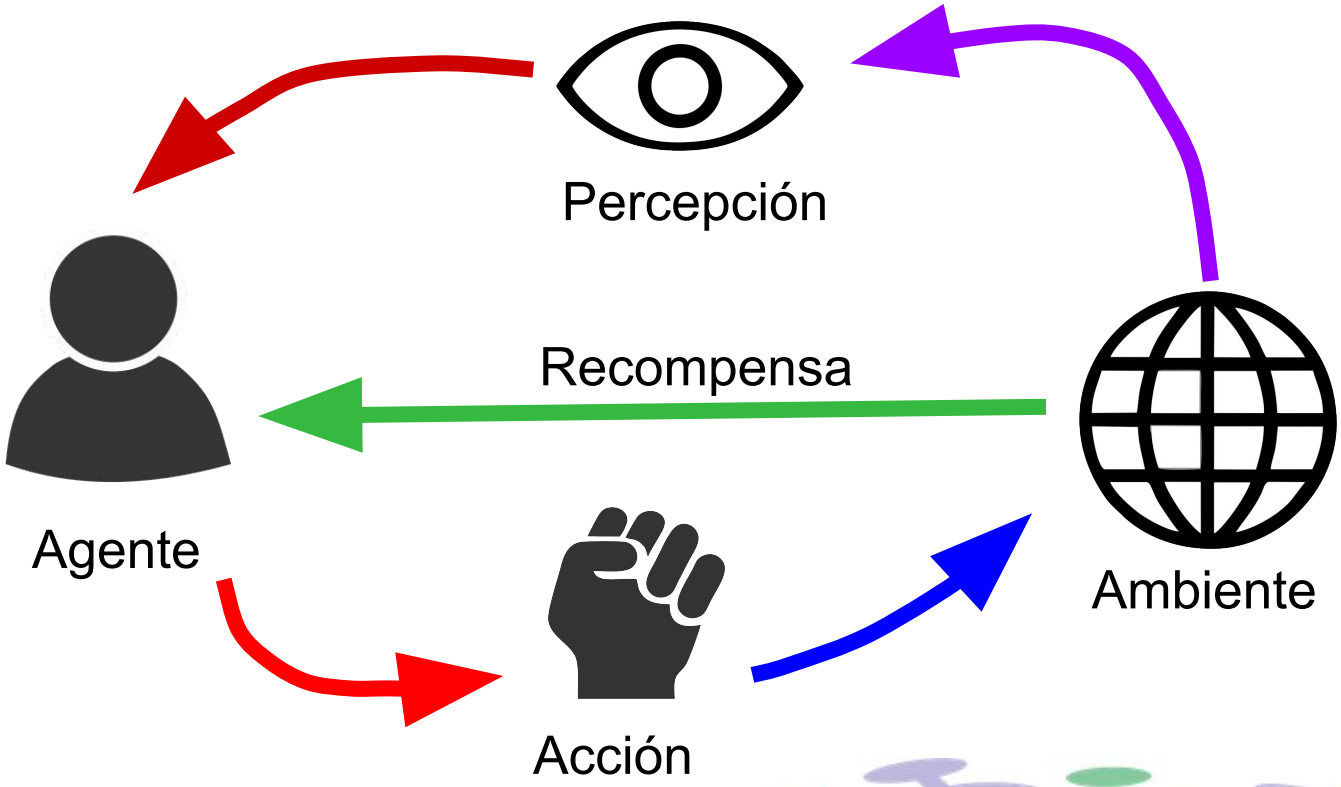


Aprendizaje por refuerzo

Es un tipo de aprendizaje menos *tradicional*: no contamos con **datos ni etiquetas**, ni con **reglas ni procedimientos específicos**.

¿Puede una computadora desarrollar un comportamiento inteligente por sí misma?

Aprendizaje por refuerzo



Aprendizaje por refuerzo

Estado s_t

Estado s_{t+1}

Acción

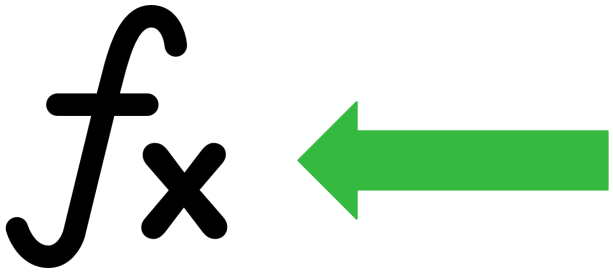
a_t

Recompensa

r_{t+1}

Aprendizaje por refuerzo

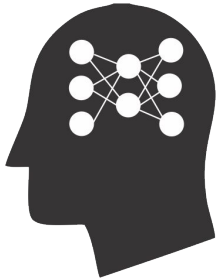
Dos componentes esenciales:



Función de valor

Agente

Comportamiento



Política

π

Aprendizaje por refuerzo

Definamos la función de valor [1]:

Recompensa en el tiempo $t+1$

Recompensa en el tiempo t

$$V^{\pi}(s_t) = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \dots$$

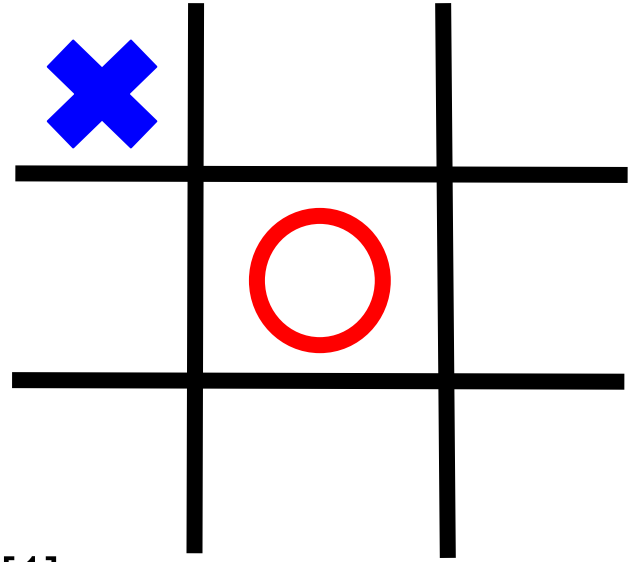
Recompensa en el tiempo $t+2$

Factor de descuento

$$0 \leq \gamma \leq 1$$

Aprendizaje por refuerzo

Un ejemplo:



[1]

Acciones $A = \{\bigcirc, \times\}$

Recompensa $R = \{1, 0, -1\}$

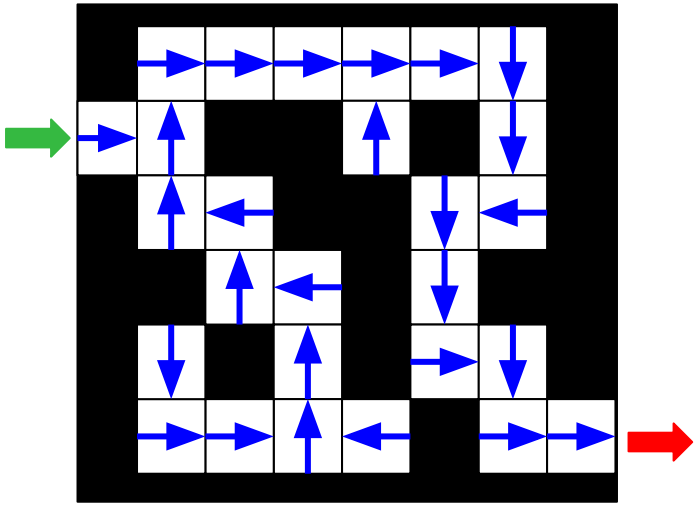
Estados $S = \{$posición de $\bigcirc$ y $\times\}$

Política $\pi = S \times A$

Función de valor predicción de la recompensa a futuro

Aprendizaje por refuerzo

Otro ejemplo:



[1]

Acciones $A = \{\uparrow, \rightarrow, \downarrow, \leftarrow\}$

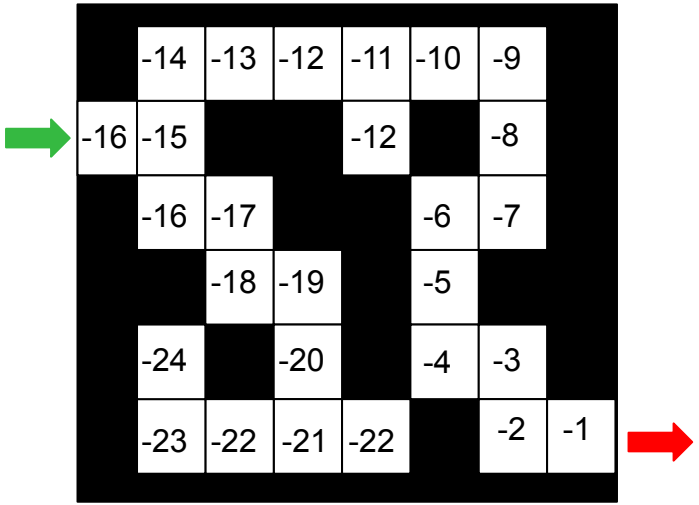
Recompensa $R = \{-t \mid t \text{ es un episodio}\}$

Estados S = localización del agente

Política π (flechas en color azul)

Aprendizaje por refuerzo

Otro ejemplo:



Acciones $A = \{\uparrow, \rightarrow, \downarrow, \leftarrow\}$

Recompensa $R = \{-t \mid t \text{ es un episodio}\}$

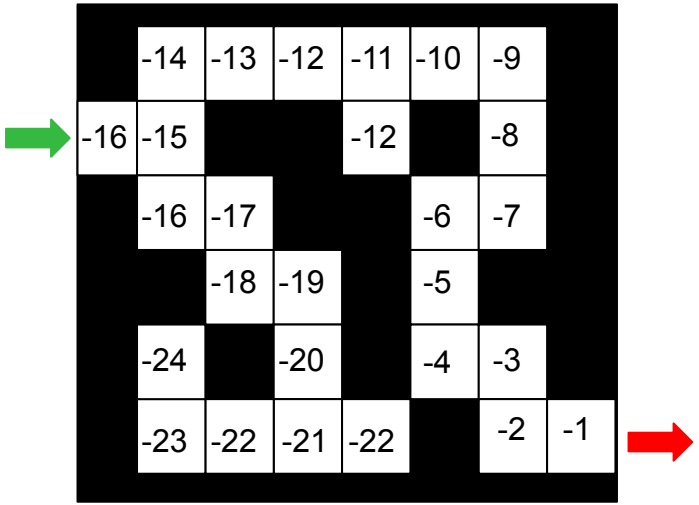
Estados S = localización del agente

Política π (flechas en color azul)

Función de valor predicción de la recompensa a futuro

Aprendizaje por refuerzo

Otro ejemplo:



Acciones $A = \{\uparrow, \rightarrow, \downarrow, \leftarrow\}$

Recompensa $R = \{-t \mid t \text{ es un episodio}\}$

Estados S = localización del agente

Política π (flechas en color azul)

Función de valor predicción de la recompensa a futuro

Aprendizaje por refuerzo

Más optimización:

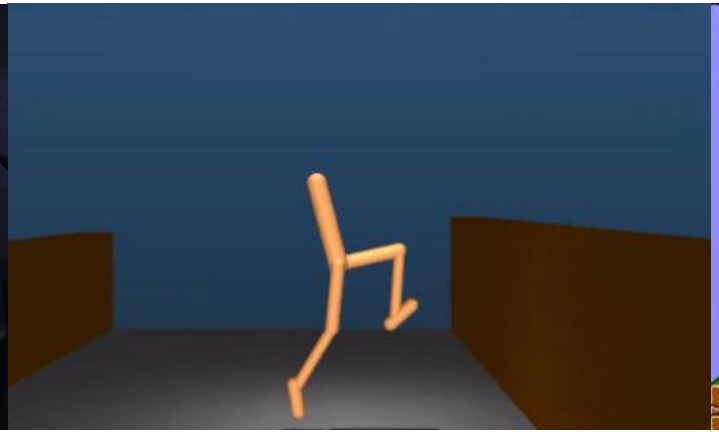
$$\max_{\pi} V^{\pi}(s_t)$$

Maximizar la función de valor en cada estado a partir de la política.

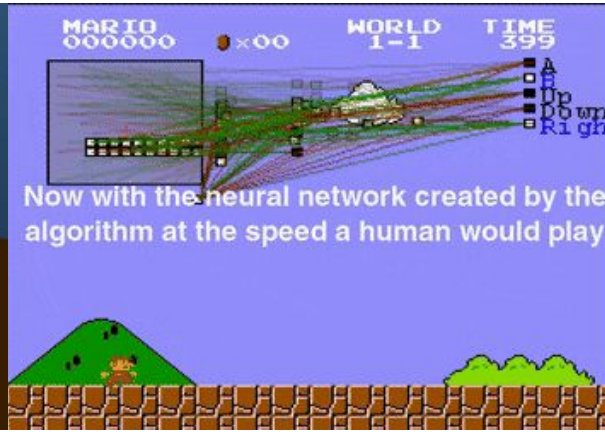
Aprendizaje por refuerzo

Ejemplos muy interesantes:

[i1]



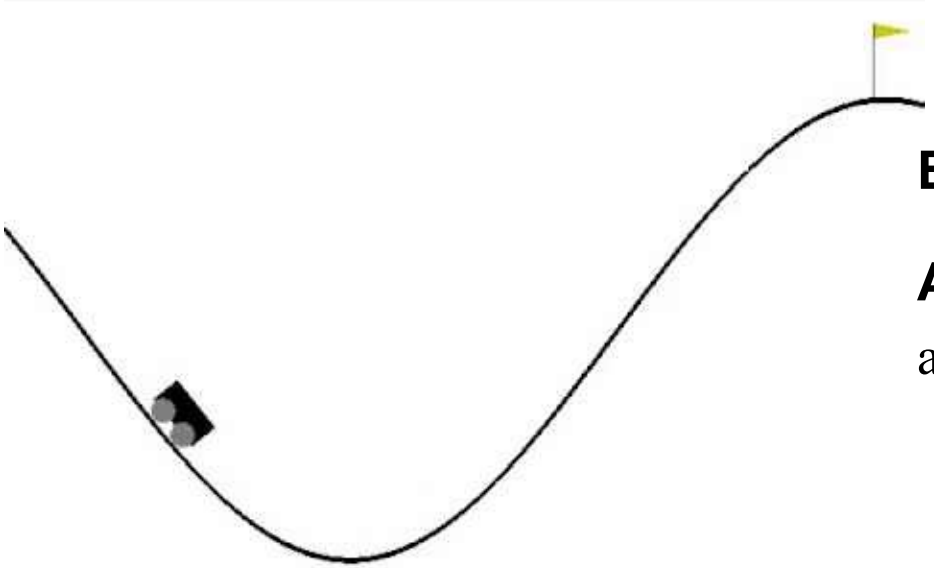
[i2]



[i3]

Q Learning

Una algoritmo de Aprendizaje por Refuerzo muy popular.



Estados S = (posición, rapidez)

Acciones A = {acelerar a la izquierda, acelerar a la derecha, no hacer nada}

Meta: *pos = 0.5*

Recompensa: *-1 en cada instante donde no se llegue a la meta*

Q Learning

Primero definimos los límites para las variables de estado:

Variable	Min	Max
Posición	-1.2	0.6
Rapidez	-0.07	0.07

Q Learning

Para cada estado s_t → Una acción a_t

Recompensa $R(s_t, a_t) = r_t$

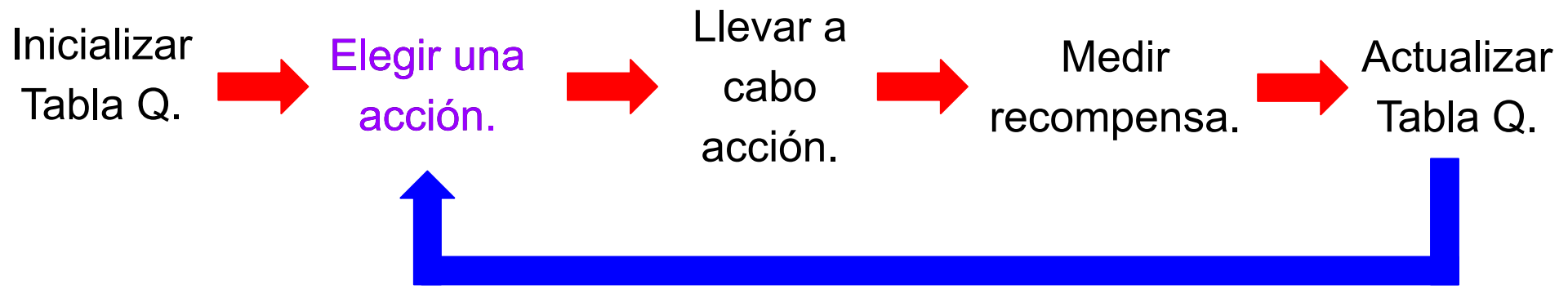
Q Learning

Si tenemos una **cantidad finita de estados**, habrá una c**antidad finita de acciones**. Podemos organizar las recompensas en la **Tabla Q** para cada instante t:

	a_1	a_2	...	a_m
s_1	0	1	...	1
s_2	1	0	...	999
...	...	...	...	...
s_n	0	1	...	0

El valor de $Q(s_i, a_j)$ es la máxima recompensa esperada si el agente toma la acción a_j en el estado s_i.

Q Learning


Inicializar Tabla Q.
Elegir una acción.
Llevar a cabo acción.
Medir recompensa.
Actualizar Tabla Q.

Q Learning

Cuando elegimos una nueva acción para el nuevo episodio $t + 1$, tenemos que actualizar la Tabla Q:

La actualización de $Q(s , a)$

Coeficiente de aprendizaje

Recompensa inmediata de aplicar a en s

Tasa de descuento

$$Q(s , a) = Q(s , a) + \propto R(s, a) + \gamma \max Q^{t+1}(s^{t+1}, a^{t+1}) - Q(s, a))$$

El valor anterior de $Q(s , a)$

El valor máximo esperado de cualquier futura acción después de haber elegido a en este episodio

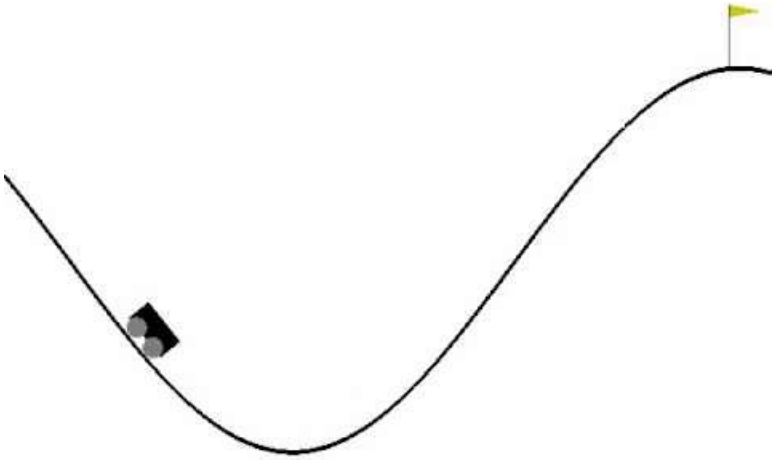
Q Learning

Después de una cantidad **predeterminada** de repeticiones, la Tabla Q estará actualizada con los valores máximos esperados de recompensa.

La política π será la propia Tabla Q; una vez que el agente está entrenado, resolver el problema consiste en que, para cada posible estado s_i, seleccionemos la acción a_m que devuelva el mayor valor $Q(s_i, a_m)$.

Q Learning

En nuestro ejemplo, elegiremos la siguiente acción de la siguiente forma:



Si $\varepsilon \geq U(0,1)$:

$$a^* = \max Q(s, a)$$

Si no:

$$a^* = \text{random } Q(s, a)$$

El parámetro ε se reduce ligeramente durante cada episodio.

Referencias

1. Zemel, R. Ursatun, R., Fidler, S. (2016). *CSC 411: Lecture 19: Reinforcement Learning*. Recuperado de: https://www.cs.toronto.edu/~urtasun/courses/CSC411_Fall16/19_rl.pdf
2. n.a. (2018). *An introduction to Q-Learning: reinforcement learning.* Recuperado de: https://www.freecodecamp.org/news/an-introduction-to-q-learning-reinforcement-learning-14ac0b4493cc/
3. n.a. (2019). *Mountain_Car.py.* Recuperado de: https://gist.github.com/gkhayes/3d154e0505e31d6367be22ed3da2e955

Imágenes

i1. Rus, C. (2018). *'AlphaGo' es el documental de Netflix que explica lo que supuso la victoria de la IA de Google al campeón de Go.* Recuperado de: https://www.xataka.com/cine-y-tv/alphago-es-el-documental-de-netflix-que-mejor-explica-lo-que-supuso-la-victoria-de-la-ia-de-google-al-campeon-de-go

i2. Heess, N., Merel, J., Wang, Z. (2017). *Producing flexible behaviours in simulated environments*. Recuperado de: https://deepmind.com/blog/article/producing-flexible-behaviours-simulated-environments

i3. Sethi, V. (2019). *Reinforcement Learning: a Subtle Introduction.* Recuperado de: https://towardsdatascience.com/reinforcement-learning-a-subtle-introduction-7a150e37960e