Unidad II - Almacenamiento, Estructura de datos y Clasificación del AA

Germán Braun

Facultado de Informática - Universidad Nacional del Comahue

german.braun@fi.uncoma.edu.ar

11 de septiembre de 2025

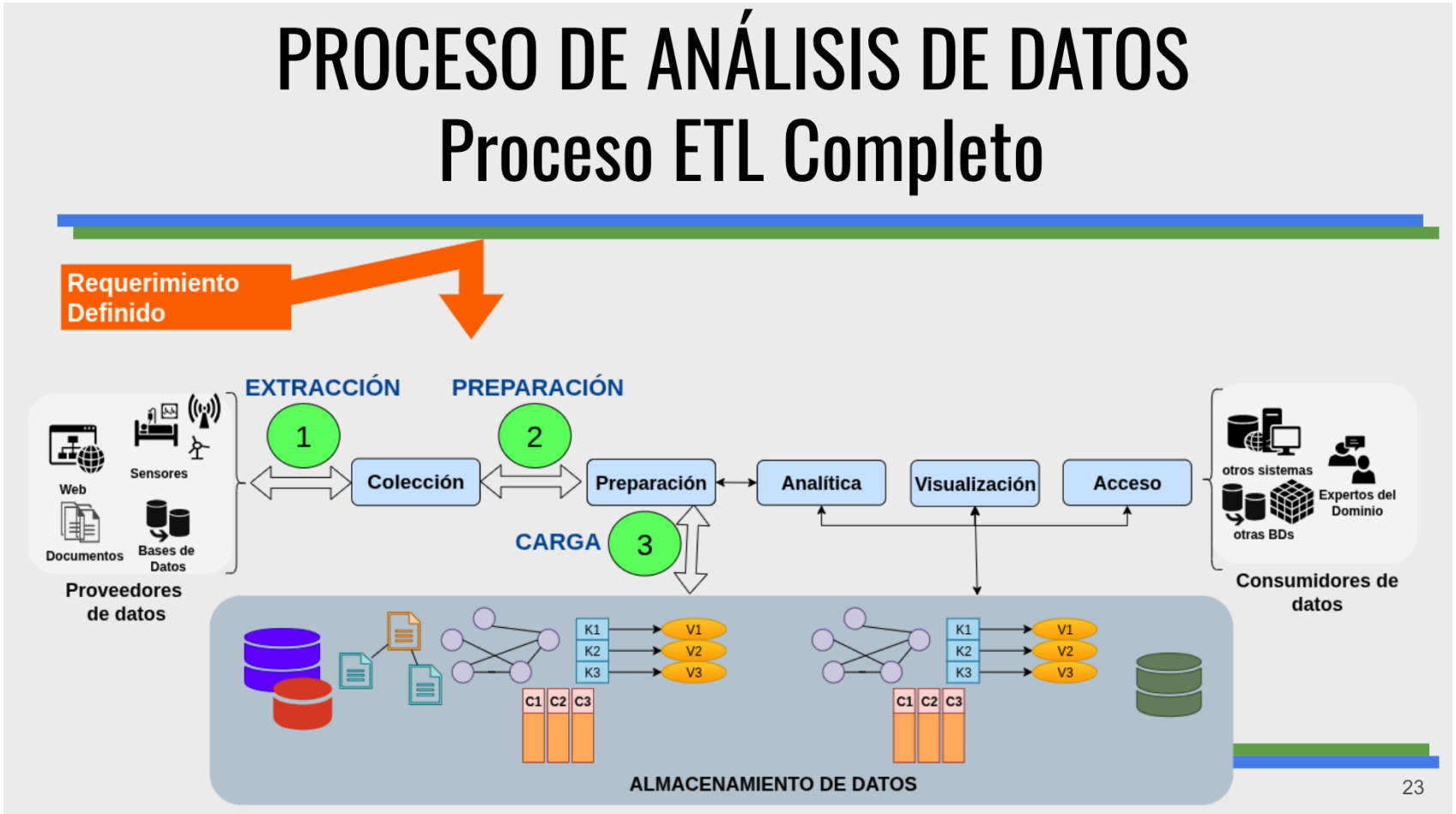
Agenda

ML en la práctica (recap)

¡Recordatorio!

El aprendizaje automático es un proceso de prueba y error.

Almacenamiento y Estructura de datos



(*) del curso EXTRACCIÓN, PREPARACIÓN Y ALMACENAMIENTO DE LOS DATOS. Créditos: Agustina Buccella

Estructuras de datos y tipos de datos para ML

- Variables en un problema de aprendizaje automático puede ser de diversos tipos de datos: numéricas ($\mathbb{R}$ / $\mathbb{Z}$), y de categorías $\in C_1, ..., C_n$. También, manipularemos **strings**.

Estructuras de datos y tipos de datos para ML

- Variables en un problema de aprendizaje automático puede ser de diversos tipos de datos: numéricas ($\mathbb{R}$ / $\mathbb{Z}$), y de categorías $\in C_1, ..., C_n$. También, manipularemos **strings**.
- Las estructuras de datos más comunes que usaremos son: **listas, conjuntos, arreglos, vectores y matrices**

Estructuras y tipos de datos para ML (cont'd)

- **Listas**: permiten almacenar un conjunto arbitrario de elementos. Son ordenadas, se pueden indexar, anidar y son mutables y dinámicas

```
1 # Lista en Python
2 numeros = [1, 2, 3, 4, 5, 'John Doe']
3 frutas = ["manzana", "pera", "naranja"]
4 print(numeros[0])    # 1
5 print(frutas[-1])    # naranja
```

Estructuras y tipos de datos para ML (cont'd)

- **Listas**: permiten almacenar un conjunto arbitrario de elementos. Son ordenadas, se pueden indexar, anidar y son mutables y dinámicas

```
# Lista en Python
numeros = [1, 2, 3, 4, 5, 'John Doe']
frutas = ["manzana", "pera", "naranja"]
print(numeros[0])    # 1
print(frutas[-1])    # naranja
```

- **Conjuntos**: colección no ordenada de elementos heterogéneos y únicos

```
# Conjuntos en Python
numeros = {1, 2, 3, 4, 5}
frutas = {"manzana", "pera", "naranja"}
# Agregar elementos
numeros.add(6)
# Operaciones
pares = {2, 4, 6, 8}
print(numeros & pares)
print(numeros | pares)
```

Estructuras y tipos de datos para ML (cont'd)

Para definición y manipulación de estructuras más complejas en ciencias de datos usaremos la librería Python **NumPy**[1].

[1] `https://numpy.org/`

Estructuras y tipos de datos para ML (cont'd)

Para definición y manipulación de estructuras más complejas en ciencias de datos usaremos la librería Python **NumPy**[1]. **NumPy** contiene arreglos multidimensionales, y funciones que operan eficientemente sobre esas estructuras. Para importarla y usarla en nuestro código:

```
1 import numpy as np
```

[1] https://numpy.org/

Estructuras y tipos de datos para ML (cont'd)

Para definición y manipulación de estructuras más complejas en ciencias de datos usaremos la librería Python **NumPy**[1]. **NumPy** contiene arreglos multidimensionales, y funciones que operan eficientemente sobre esas estructuras. Para importarla y usarla en nuestro código:

```
1 import numpy as np
```

- **Arreglos**:

```
1 # Crear un array de NumPy
2 numeros = np.array([1, 2, 3, 4, 5])
3 # Acceso e impresion
4 print(numeros[0])  # 1
5 # Operaciones vectorizadas
6 dobles = numeros * 2
7 print(dobles)  # [ 2  4  6  8 10]
8 # Funciones de NumPy
9 print(np.mean(numeros))  # media
```

[1] https://numpy.org/

Estructuras y tipos de datos para ML (cont'd)

- **Vectores**:

```
# Crear un vector fila
v1 = np.array([1, 2, 3])
# Crear un vector columna
v2 = np.array([[4], [5], [6]])
# Producto punto (dot product)
dot = np.dot(v1, [4, 5, 6])
print(dot)  # 32
# Norma del vector
norma = np.linalg.norm(v1)
print(norma)  # 3.741657...
```

Estructuras y tipos de datos para ML (cont'd)

- **Matrices**:

```
# Crear una matriz 2x3
A = np.array([[1, 2, 3],
              [4, 5, 6]])
# Crear otra matriz 3x2
B = np.array([[7, 8],
              [9, 10],
              [11, 12]])
# Multiplicacion de matrices
C = np.dot(A, B)
print(C)
# [[ 58  64]
#  [139 154]]
# Transpuesta
print(A.T)
```

Clasificación del Aprendizaje Automático

AA Landscape

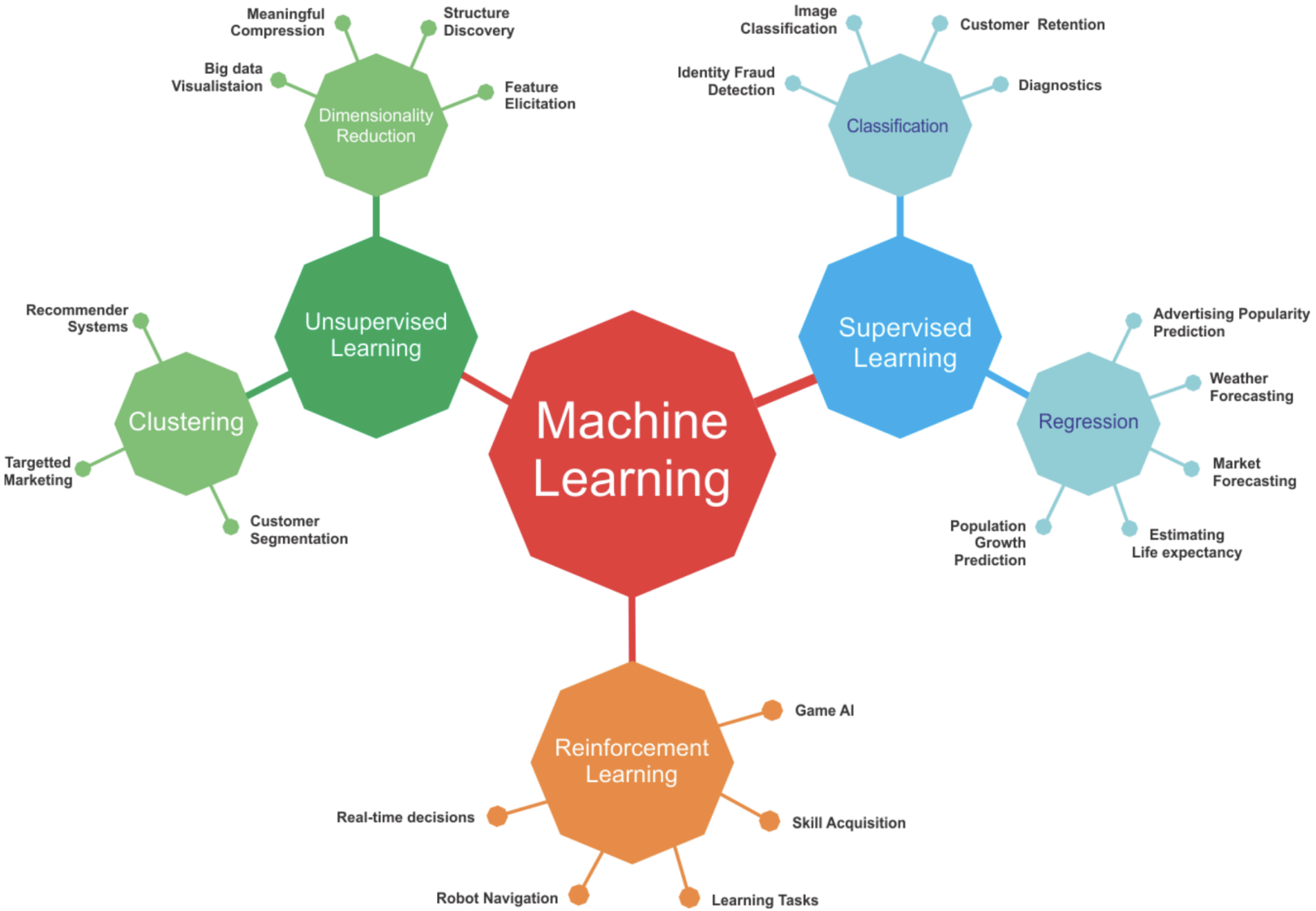


Figura 2.1: Figura de [2]

Aprendizaje Supervisado

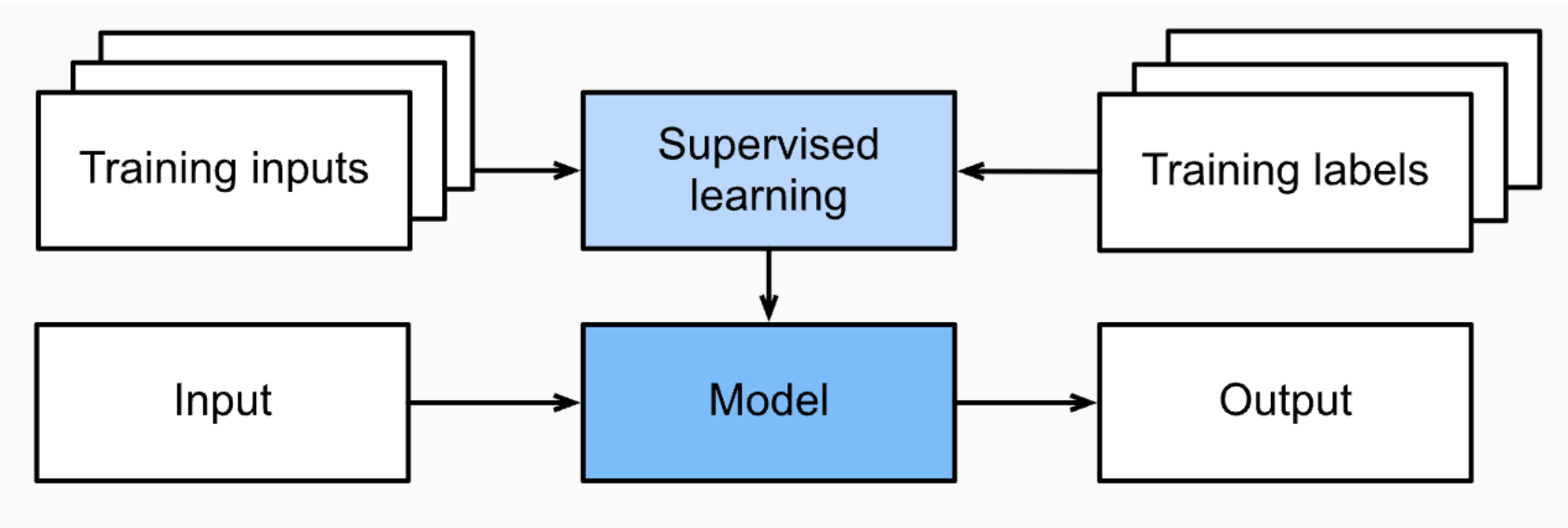


Figura 2.2: Figura de [5]

https://stanford.edu/~shervine/teaching/cs-229/cheatsheet-supervised-learning

Aprendizaje Supervisado (cont'd)

- Está basada sobre un número predefinido de ejemplos ya etiquetados, por ejemplo, $(x^{(1)}, y^{(1)})$ es un spam y otro $(x^{(2)}, y^{(2)})$ no lo es,

[2] En caso de más características (*features*), notaremos $(x_1^{(i)}, y_1^{(i)}),(x_2^{(i)}, y_2^{(i)})$, dónde $x_1^{(i)}$ es el valor de la feature x_1 en el ejemplo i, $x_2^{(i)}$ el valor de la feature x_2 para el mismo i, y asi sucesivamente

Aprendizaje Supervisado (cont'd)

- Está basada sobre un número predefinido de ejemplos ya etiquetados, por ejemplo, $(x^{(1)}, y^{(1)})$ es un spam y otro $(x^{(2)}, y^{(2)})$ no lo es,
- Estos ejemplos son dados a un algoritmo de aprendizaje supervisado, el cual produce una función f como salida (el modelo aprendido),

[2] En caso de más características (*features*), notaremos $(x_1^{(i)}, y_1^{(i)}), (x_2^{(i)}, y_2^{(i)})$, dónde $x_1^{(i)}$ es el valor de la feature x_1 en el ejemplo i, $x_2^{(i)}$ el valor de la feature x_2 para el mismo i, y asi sucesivamente

Aprendizaje Supervisado (cont'd)

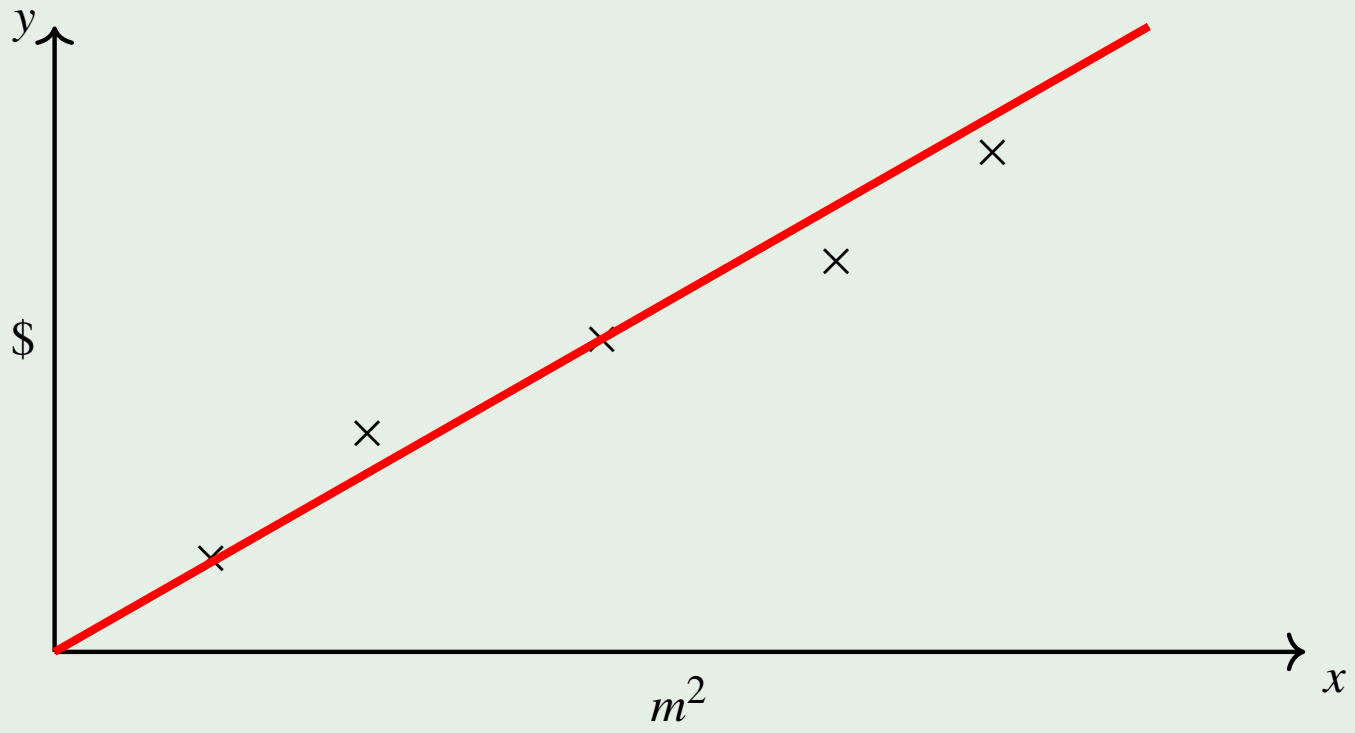
- Está basada sobre un número predefinido de ejemplos ya etiquetados, por ejemplo, $(x^{(1)}, y^{(1)})$ es un spam y otro $(x^{(2)}, y^{(2)})$ no lo es,
- Estos ejemplos son dados a un algoritmo de aprendizaje supervisado, el cual produce una función f como salida (el modelo aprendido),
- El modelo aprendido puede ser usado para evaluar nuevas entradas, las cuales no han sido etiquetadas. El modelo predice a qué clase (label) pertenecen estas entradas: $(x^{(3)}, f(x^{(3)}))$ spam/no spam [2]

[2] En caso de más características (*features*), notaremos $(x_1^{(i)}, y_1^{(i)}), (x_2^{(i)}, y_2^{(i)})$, dónde $x_1^{(i)}$ es el valor de la feature x_1 en el ejemplo i, $x_2^{(i)}$ el valor de la feature x_2 para el mismo i, y asi sucesivamente

Aprendizaje Supervisado (cont'd)

Predecir el valor de una casa (adaptado de Tengyu Ma and Chris Re)

Supongamos que tenemos un dataset con n muestras, $(x^{(1)}, y^{(1)})$ ... $(x^{(n)}, y^{(n)})$, donde $y^{(i)}$ es el valor de una casa dado la cantidad de metros cuadrados $x^{(i)}$ para el ejemplo i

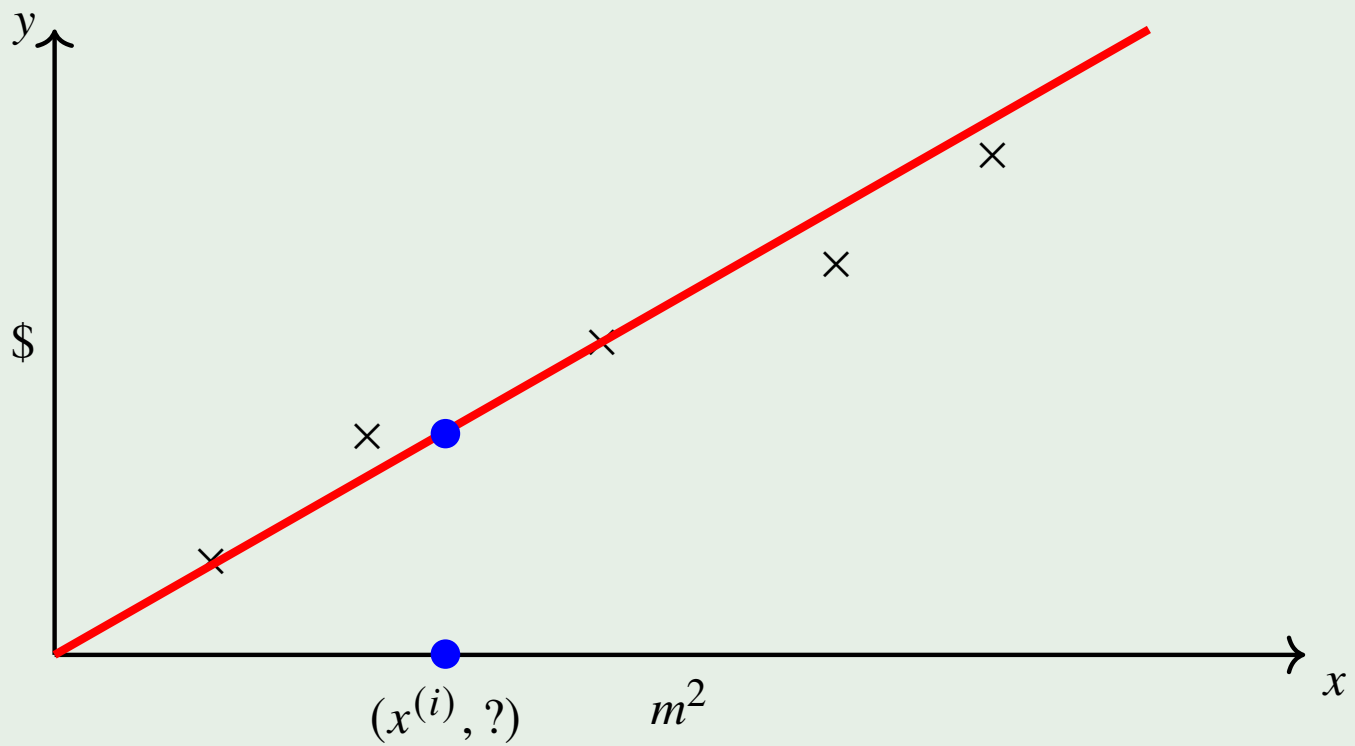


Tarea de aprendizaje: predecir el valor ($) de una casa dada su superficie en m^2

Aprendizaje Supervisado (cont'd)

Predecir el valor de una casa (adaptado de Tengyu Ma and Chris Re)

Supongamos que tenemos un dataset con n muestras, $(x^{(1)}, y^{(1)})$ ... $(x^{(n)}, y^{(n)})$, donde $y^{(i)}$ es el valor de una casa dado la cantidad de metros cuadrados $x^{(i)}$ para el ejemplo i



Tarea de aprendizaje: predecir el valor ($) de una casa dada su superficie en m^2

Aprendizaje Supervisado (cont'd)

- La tareas de aprendizaje pueden ser descriptas como la estimación probabilística de una clase/valor desconocido, dada un dataset con una clasificación previa

Aprendizaje Supervisado (cont'd)

- La tareas de aprendizaje pueden ser descriptas como la estimación probabilística de una clase/valor desconocido, dada un dataset con una clasificación previa
- El aprendizaje supervisado funciona "bien" en tareas como clasificación de imágenes, análisis predictivo, detección de spam, motores de recomendaciónes, ...

Aprendizaje Supervisado (cont'd)

- La tareas de aprendizaje pueden ser descriptas como la estimación probabilística de una clase/valor desconocido, dada un dataset con una clasificación previa
- El aprendizaje supervisado funciona "bien" en tareas como clasificación de imágenes, análisis predictivo, detección de spam, motores de recomendaciónes, ...
- pero ...

Aprendizaje Supervisado (cont'd)

- La tareas de aprendizaje pueden ser descriptas como la estimación probabilística de una clase/valor desconocido, dada un dataset con una clasificación previa
- El aprendizaje supervisado funciona "bien" en tareas como clasificación de imágenes, análisis predictivo, detección de spam, motores de recomendaciónes, ...
- pero ...
- Datasets tienen que ser etiquetados manualmente

Aprendizaje Supervisado (cont'd)

- La tareas de aprendizaje pueden ser descriptas como la estimación probabilística de una clase/valor desconocido, dada un dataset con una clasificación previa
- El aprendizaje supervisado funciona "bien" en tareas como clasificación de imágenes, análisis predictivo, detección de spam, motores de recomendaciónes, ...
- pero ...
- Datasets tienen que ser etiquetados manualmente
- Necesitan intervención humana (no aprenden por sí mismos) → Bias

Aprendizaje Supervisado (cont'd)

- La tareas de aprendizaje pueden ser descriptas como la estimación probabilística de una clase/valor desconocido, dada un dataset con una clasificación previa
- El aprendizaje supervisado funciona "bien" en tareas como clasificación de imágenes, análisis predictivo, detección de spam, motores de recomendaciónes, ...
- pero ...
- Datasets tienen que ser etiquetados manualmente
- Necesitan intervención humana (no aprenden por sí mismos) → Bias
- Ovefitting

Aprendizaje Supervisado (cont'd)

Dependiendo de si la predicción es numérica o basada en categorías, podemos identificar dos tipos básicos de tareas de aprendizaje (supervisado):

- **Regresión**
- **Clasificación**

Aprendizaje Supervisado (cont'd)

Dependiendo de si la predicción es numérica o basada en categorías, podemos identificar dos tipos básicos de tareas de aprendizaje (supervisado):

- **Regresión**
- **Clasificación**

	Regresión	**Clasificación**[a]
Salida	$\mathcal{Y} \subseteq \mathbb{R}$	$\mathcal{Y} = \{C_1, ...C_n\}$
Algoritmos	Regresión Lineal	SVM, Regresión Logística
Ejemplos	$\$/m^2$	Spam/No spam

[a] Adaptada de `https://stanford.edu/~shervine/teaching/cs-229/cheatsheet-supervised-learning`

Aprendizaje No Supervisado

- El dataset no está etiquetado: $x^{(1)} \ldots x^{(n)}$

Aprendizaje No Supervisado

- El dataset no está etiquetado: $x^{(1)} \ldots x^{(n)}$
- El objetivo es encontrar estructuras “interesantes” en los datos $\rightarrow$ *clusters*

Aprendizaje No Supervisado

- El dataset no está etiquetado: $x^{(1)} \ldots x^{(n)}$
- El objetivo es encontrar estructuras “interesantes” en los datos $\rightarrow$ *clusters*
- Los algoritmos descubren patrones ocultos or agrupan datos sin necesidad de intervención humana.

Aprendizaje No Supervisado

- El dataset no está etiquetado: $x^{(1)} \ldots x^{(n)}$
- El objetivo es encontrar estructuras “interesantes” en los datos $\rightarrow$ *clusters*
- Los algoritmos descubren patrones ocultos or agrupan datos sin necesidad de intervención humana.
- A diferencia del supervisado, el aprendizaje no supervisado puede ser útil para la detección de anomalías, segmentación de clientes, entre otros ...

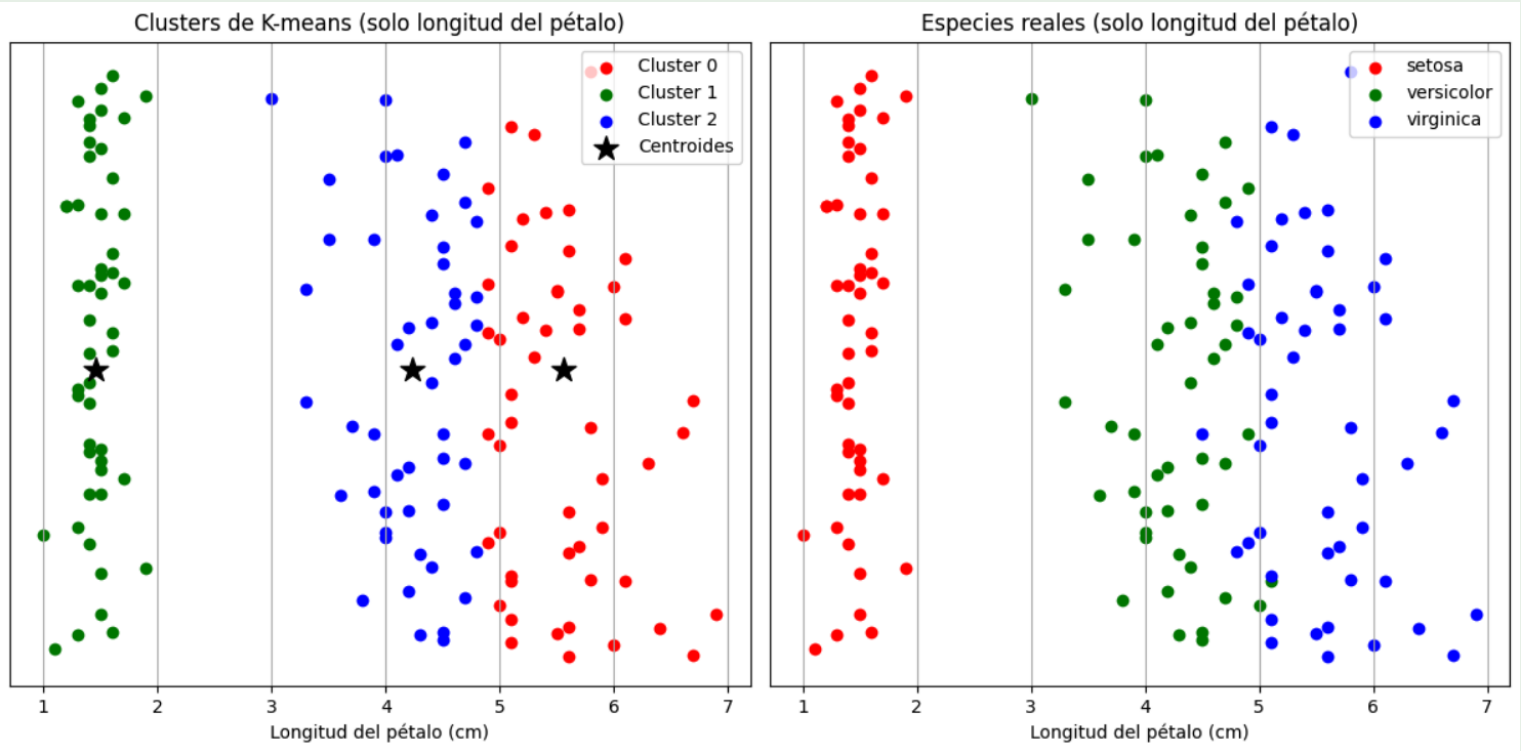
Aprendizaje No Supervisado

- El dataset no está etiquetado: $x^{(1)} \ldots x^{(n)}$
- El objetivo es encontrar estructuras “interesantes” en los datos $\rightarrow$ *clusters*
- Los algoritmos descubren patrones ocultos or agrupan datos sin necesidad de intervención humana.
- A diferencia del supervisado, el aprendizaje no supervisado puede ser útil para la detección de anomalías, segmentación de clientes, entre otros ...
- Son computacionalmente más complejos debido a que requieren de un dataset más grande para producir una salida satisfactoria

Aprendizaje No Supervisado (cont'd)

Agrupar flores de acuerdo a sus características

Supongamos que queremos agrupar las flores de iris analizando sus características: *longitud del sépalo, ancho del sépalo, longitud del pétalo, ancho del pétalo*



El gráfico muestra los 3 clusters generados, uno para cada flor del dataset: *iris setosa, versicolor y virginica*, usando el algoritmo **k-means**.

Aprendizaje Por Refuerzo

El objetivo de este aprendizaje es que un agente aprenda a evolucionar en un ambiente

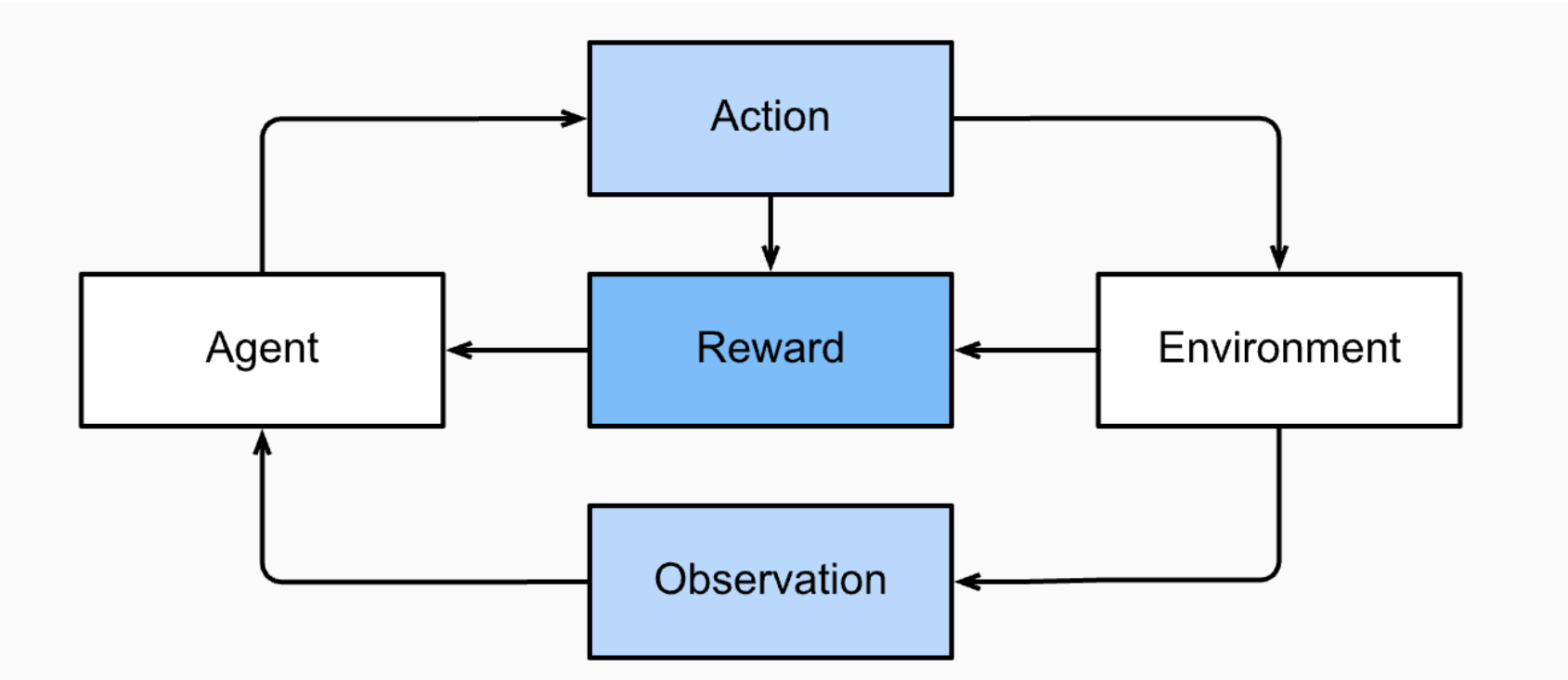


Figura 2.3: Figura de [5]

Aprendizaje Por Refuerzo (cont'd)

- Un agente aprende basado en su interacción con el ambiente (*trail-and-error*)

Aprendizaje Por Refuerzo (cont'd)

- Un agente aprende basado en su interacción con el ambiente (*trail-and-error*)
- Los parámetros del sistema están basados en el *feedback* obtenido del ambiente → *reward*

Aprendizaje Por Refuerzo (cont'd)

- Un agente aprende basado en su interacción con el ambiente (*trail-and-error*)
- Los parámetros del sistema están basados en el *feedback* obtenido del ambiente → *reward*
- No es tan popular..., sin embargo, es un enfoque potente para problemas de toma de decisiones donde los agentes deben aprender interactuando con el ambiente.
- Aplicación en robótica

Aprendizaje Por Refuerzo (cont'd)

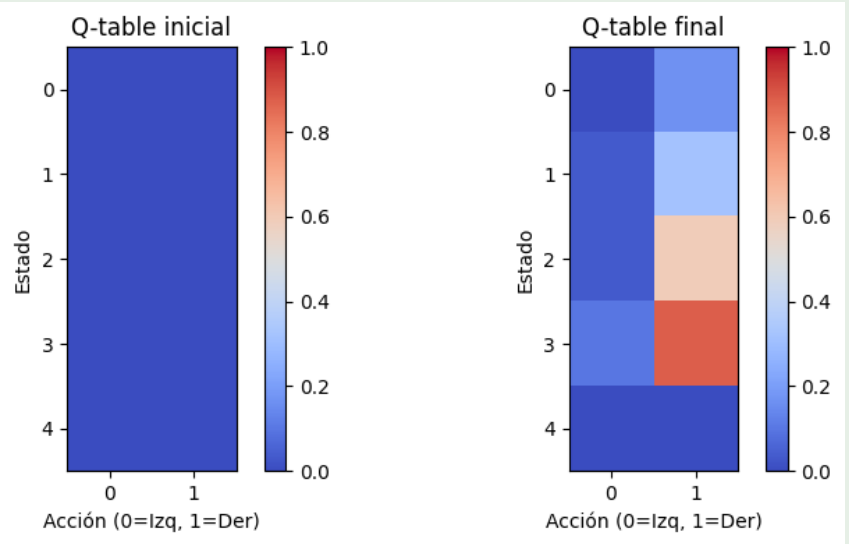
Un Agente que aprende de la experiencia (*Q-learning* [4])

Supongamos que tenemos un agente que tiene que recorre una habitación cuyas posiciones son $[0, 1, 2, 3, 4]$. La posición inicial es la 0 y solo tiene dos acciones posibles que puede ejecutar: $\leftarrow (0)$ y $\rightarrow (1)$. Además gana 1 punto de recompensa si llega a la celda 4.

Aprendizaje Por Refuerzo (cont'd)

Un Agente que aprende de la experiencia (*Q-learning* [4])

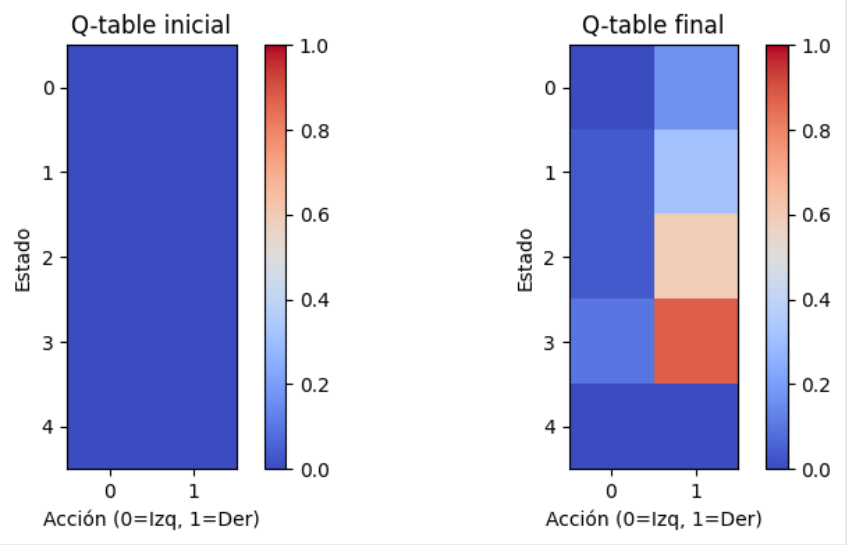
Supongamos que tenemos un agente que tiene que recorre una habitación cuyas posiciones son $[0, 1, 2, 3, 4]$. La posición inicial es la 0 y solo tiene dos acciones posibles que puede ejecutar: $\leftarrow (0)$ y $\rightarrow (1)$. Además gana 1 punto de recompensa si llega a la celda 4.



Aprendizaje Por Refuerzo (cont'd)

Un Agente que aprende de la experiencia (*Q-learning* [4])

Supongamos que tenemos un agente que tiene que recorre una habitación cuyas posiciones son $[0, 1, 2, 3, 4]$. La posición inicial es la 0 y solo tiene dos acciones posibles que puede ejecutar: $\leftarrow (0)$ y $\rightarrow (1)$. Además gana 1 punto de recompensa si llega a la celda 4.



iteración 1: [[0. 0.] [0. 0.] [0. 0.] [0. 0.1] [0. 0.]]

iteración 2: [[0. 0.] [0. 0.] [0. 0.01] [0. 0.19] [0. 0.]]

iteración n: [[0. 0.17] [0.03 0.32] [0.03 0.59] [0.1 0.88] [0. 0.]]

Aprendizaje Por Refuerzo (cont'd)

- Un problema de aprendizaje por refuerzo consta de cuatro componentes:
  1. **Política**: mapeo de estados-acciones (determinístico vs. estocástico)

Aprendizaje Por Refuerzo (cont'd)

- Un problema de aprendizaje por refuerzo consta de cuatro componentes:
  1. **Política**: mapeo de estados-acciones (determinístico vs. estocástico)
  2. **Recompensa**: las acciones de agente llevan a una recomensa o no y dependen del problema a resolver. Para un vehículo autónomo, la recomensa podría asociarse a la reducción del tiempo de viaje.

Aprendizaje Por Refuerzo (cont'd)

- Un problema de aprendizaje por refuerzo consta de cuatro componentes:
  1. **Política**: mapeo de estados-acciones (determinístico vs. estocástico)
  2. **Recompensa**: las acciones de agente llevan a una recomensa o no y dependen del problema a resolver. Para un vehículo autónomo, la recomensa podría asociarse a la reducción del tiempo de viaje.
  3. **Función de valor**: estados deseables. Para el vehículo autónomo, la reducción del tiempo de viaje no debería implicar estados indebidos.

Aprendizaje Por Refuerzo (cont'd)

- Un problema de aprendizaje por refuerzo consta de cuatro componentes:
  1. **Política**: mapeo de estados-acciones (determinístico vs. estocástico)
  2. **Recompensa**: las acciones de agente llevan a una recomensa o no y dependen del problema a resolver. Para un vehículo autónomo, la recomensa podría asociarse a la reducción del tiempo de viaje.
  3. **Función de valor**: estados deseables. Para el vehículo autónomo, la reducción del tiempo de viaje no debería implicar estados indebidos.
  4. **Modelo**: (opcional) un modelo permite al agente predecir la *conducta* del ambiente para ejecutar posibles acciones. Para el caso del vehículo autónomo, un modelo podría ayudar a predecir posición de otros vehículos.

Algoritmos de Aprendizaje

Componentes de los algoritmos de aprendizaje

Representation	**Evaluation**	**Optimization**
Instances	Accuracy/Error rate	Combinatorial optimization
K-nearest neighbor	Precision and recall	Greedy search
Support vector machines	Squared error	Beam search
Hyperplanes	Likelihood	Branch-and-bound
Naive Bayes	Posterior probability	Continuous optimization
Logistic regression	Information gain	Unconstrained
Decision trees	K-L divergence	Gradient descent
Sets of rules	Cost/Utility	Conjugate gradient
Propositional rules	Margin	Quasi-Newton methods
Logic programs		Constrained
Neural networks		Linear programming
Graphical models		Quadratic programming
Bayesian networks		
Conditional random fields		

Figura 3.1: Table de [1]

Algoritmos de aprendizaje (cont'd)

- **Representación**: lenguaje formal con el cual un clasificador puede ser representado → espacio de hipótesis

Algoritmos de aprendizaje (cont'd)

- **Representación**: lenguaje formal con el cual un clasificador puede ser representado → espacio de hipótesis
- **Evaluación**: una función de evaluación es necesaria para poder distinguir entre "buenos y malos" clasificadores

Algoritmos de aprendizaje (cont'd)

- **Representación**: lenguaje formal con el cual un clasificador puede ser representado → espacio de hipótesis
- **Evaluación**: una función de evaluación es necesaria para poder distinguir entre "buenos y malos" clasificadores
- **Optimización**: involucra técnicas para buscar en el espacio de hipótesis → en la práctica
  - comenzar con una hipótesis simple
  - refinar parámetros gradualmente

Algoritmos de aprendizaje (cont'd)

- **Representación**: lenguaje formal con el cual un clasificador puede ser representado → espacio de hipótesis
- **Evaluación**: una función de evaluación es necesaria para poder distinguir entre "buenos y malos" clasificadores
- **Optimización**: involucra técnicas para buscar en el espacio de hipótesis → en la práctica
  - comenzar con una hipótesis simple
  - refinar parámetros gradualmente
- Representaciones discretas son combinadas con optimización combinatoria. De manera similar ocurre con las continuas

Algoritmos de aprendizaje en la práctica

Regresión Lineal

- **Representación**: Los métodos basados en hiperplanos forman una combinación lineal de las features por clase y predicen la clase con la combinación valuada más alta.

Algoritmos de aprendizaje en la práctica

Regresión Lineal

- **Representación**: Los métodos basados en hiperplanos forman una combinación lineal de las features por clase y predicen la clase con la combinación valuada más alta.
- **Evaluación**: error cuadrático (*squared-error*)

Algoritmos de aprendizaje en la práctica

Regresión Lineal

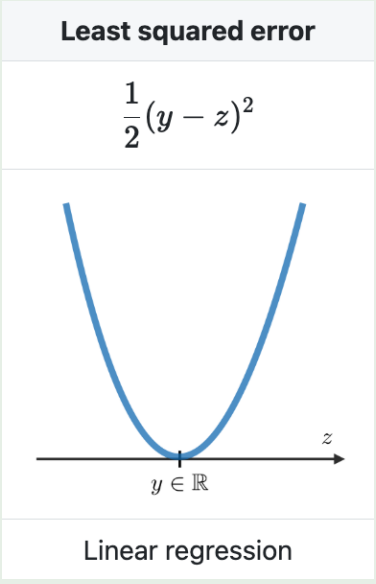
- **Representación**: Los métodos basados en hiperplanos forman una combinación lineal de las features por clase y predicen la clase con la combinación valuada más alta.
- **Evaluación**: error cuadrático (*squared-error*)
- **Optimización**: descenso por gradiente

Algoritmos de aprendizaje en la práctica

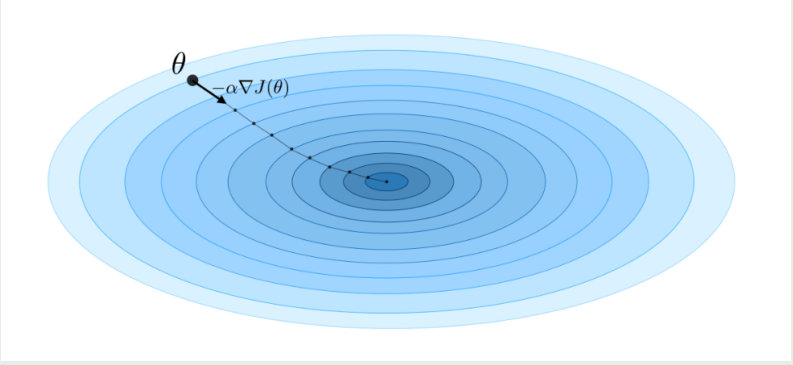
Regresión Lineal

- **Representación**: Los métodos basados en hiperplanos forman una combinación lineal de las features por clase y predicen la clase con la combinación valuada más alta.
- **Evaluación**: error cuadrático (*squared-error*)
- **Optimización**: descenso por gradiente

Diferencia cuadrática entre predicción y valor real



Ajuste de *learning rate* y costo



Créditos: `https://stanford.edu/~shervine/teaching/cs-229/cheatsheet-supervised-learning`

ML en la práctica (recap)

¡Recordatorio!

- Entender el dominio, conocimiento previo y metas.
- Pre-procesar datos (intregar, seleccionar, limpiar, dividir dataset)
- Entrenar modelos (comenzando por el más simple posible)
- Interpretar resultados
- Consolidar y deplegar conocimiento descubierto
- Ciclar sobre estos pasos anteriores

Conclusiones

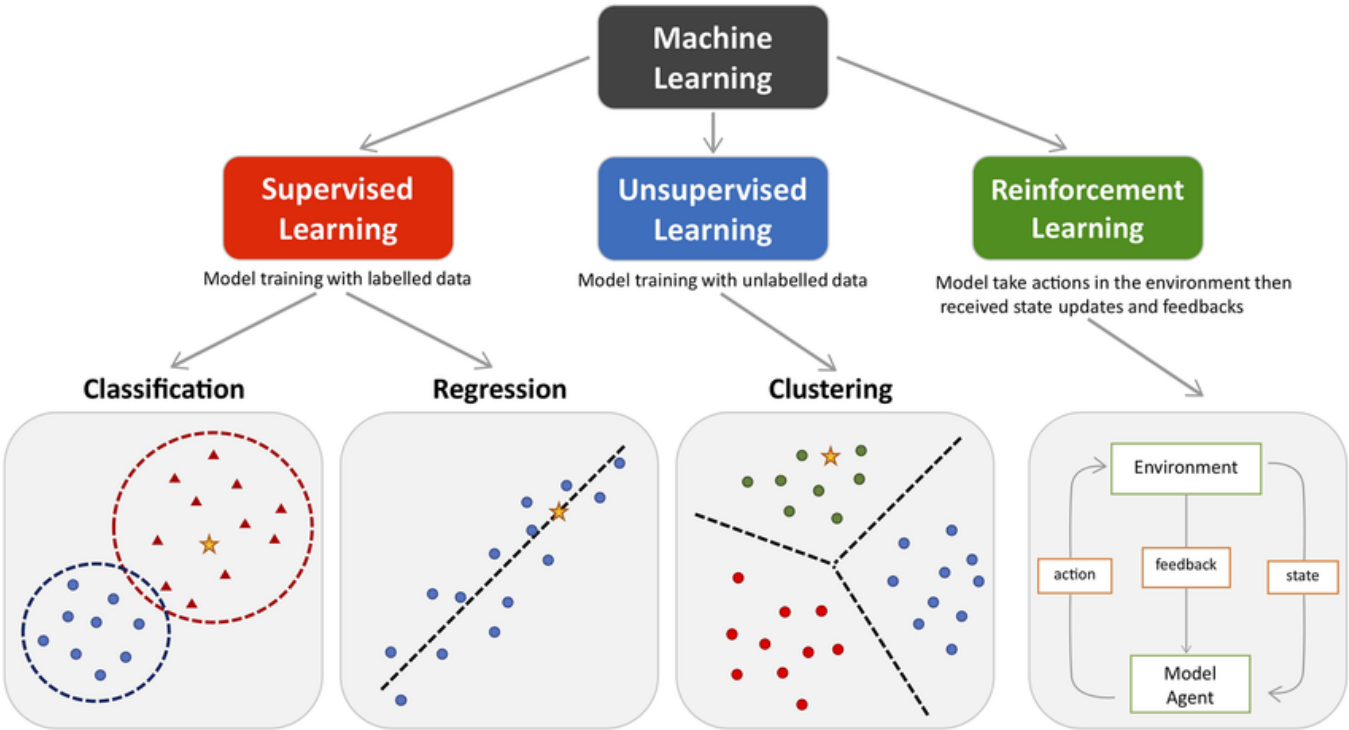


Figura 3.2: Resumen tipos de ML [3]

Próximas clases

- Preprocesamiento y generación de características
- Regresión
- Máquinas de soporte vectorial
- Redes Neuronales
- Aprendizaje no supervisado

¡Gracias!

DOMINGOS, P.

A few useful things to know about machine learning.

Communications of the ACM 55, 10 (Oct. 2012), 78–87.

KARIM, M. R.

Java Deep Learning Projects: Implement 10 Real-World Deep Learning Applications Using Deeplearning4j and Open Source APIs.

Packt Publishing Ltd, Birmingham; Mumbai, 2018.

PENG, J., JURY, E. C., DÖNNES, P., AND CIURTIN, C.

Machine learning techniques for personalised medicine approaches in immune-mediated chronic inflammatory diseases: Applications and challenges.

Frontiers in Pharmacology 12 (2021), 720694.

WATKINS, C. J., AND DAYAN, P.

Q-learning.

Machine Learning 8, 3-4 (1992), 279–292.

ZHANG, A., LIPTON, Z. C., LI, M., AND SMOLA, A. J.
Dive into Deep Learning.
Cambridge University Press, 2023.
https://D2L.ai.