

Aprendizaje Automático, introducción

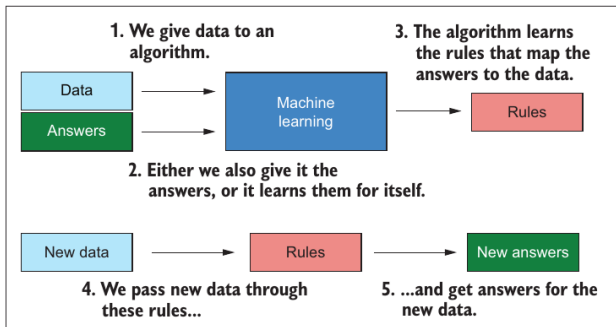
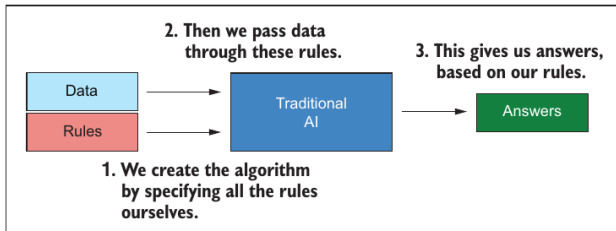
Inteligencia Artificial e Ingeniería del Conocimiento

Constantino Antonio García Martínez

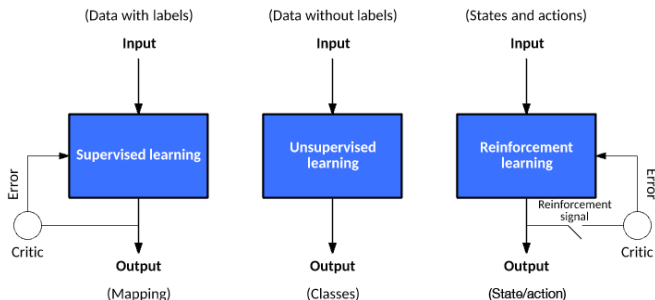
Universidad San Pablo Ceu

Introducción

Aprendizaje automático (Machine Learning, ML), un nuevo paradigma



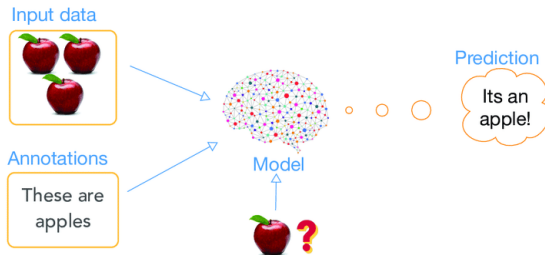
3 + 1 ramas del ML



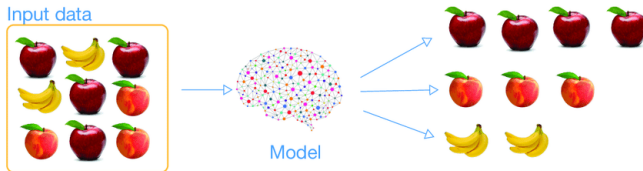
- **Aprendizaje supervisado** (Supervised L): Dado un conjunto de ejemplos, consiste en aprender a mapear datos de entrada a etiquetas conocidas (también llamadas anotaciones).
- **Aprendizaje no supervisado** (Unsupervised L): Consiste en encontrar transformaciones interesantes de los datos de entrada sin la ayuda de objetivos, generalmente para agrupar los datos.
- **Aprendizaje por refuerzo** (Reinforcement L): Un agente recibe información sobre su entorno y aprende a elegir acciones que maximizarán alguna recompensa.
- **Aprendizaje auto-supervisado** (Self-supervised L): es aprendizaje supervisado sin etiquetas anotadas por humanos.

Aprendizaje Supervisado Vs. No Supervisado

supervised learning



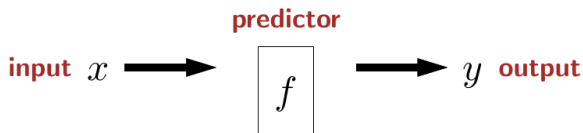
unsupervised learning



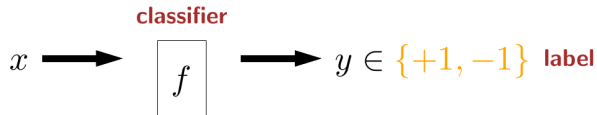
ML Supervisado

Tipos de ML Supervisado

Los algoritmos de ML supervisado se pueden dividir en diferentes tipos según la salida y .



- Clasificación
 - Binaria.
 - Multiclase.
- Regresión.
- Predicción estructurada.



- Detección de fraude: transacción con tarjeta de crédito \rightarrow fraude o no fraude
- Comentarios tóxicos: comentario en línea \rightarrow tóxico o no tóxico
- Bosón de Higgs: mediciones del evento \rightarrow evento de decaimiento o fondo

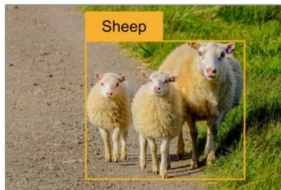
Extensión: clasificación multiclase: $y \in \{1, \dots, K\}$



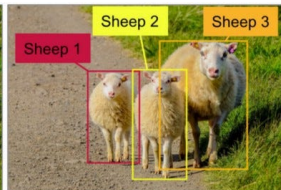
- Mapeo de pobreza: imagen satelital \rightarrow renta media
- Vivienda: información sobre casa \rightarrow precio
- Tiempos de llegada: destino, clima, tiempo \rightarrow hora de llegada

$$x \longrightarrow \boxed{f} \longrightarrow y \text{ is a complex object}$$

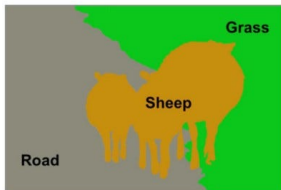
- Traducción automática: frase en inglés \rightarrow frase en japonés
- Descripción de imágenes: imagen \rightarrow frase que describe la imagen
- Segmentación de imágenes: imagen \rightarrow segmentación



Classification + Localization



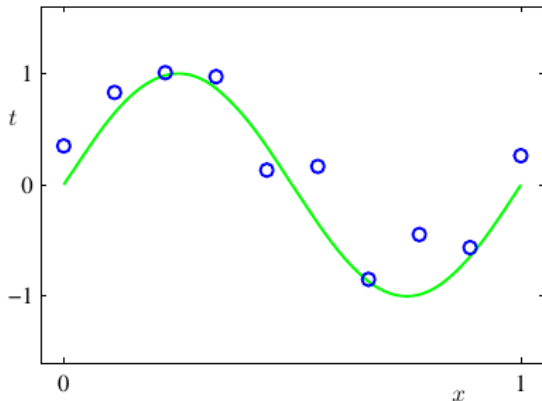
Object Detection



Conceptos básicos de ML: un ejemplo de regresión

Un problema de regresión

- Dado un conjunto de puntos $x = \{x_1, x_2, \dots, x_n\}$, intentemos aprender cómo predecir $y = \{y_1, y_2, \dots, y_n\}$.
- x se denominan **características** o **predictores**. y es la **variable objetivo**.
- Llamamos $\{x, y\}$ los **datos de entrenamiento**.

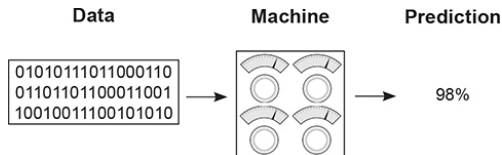


Un problema de regresión: pasos

Para ello debemos:

Receta ML (I)

1. Elegir un modelo. [clasificación o regresión](#)
2. Elegir cómo medir el rendimiento.
3. Entrenar el modelo para intentar maximizar el rendimiento.
4. Medir el rendimiento real.



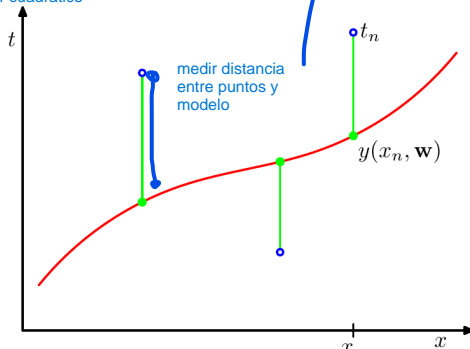
Un problema de regresión: pasos

1. Usemos un **modelo de regresión lineal** simple: $y(\mathbf{w}, x) = w_0 + w_1 \cdot x$,
2. Y usemos el **Error Cuadrático Medio (MSE)** como **función de error o función de pérdida**:

$$E(\mathbf{w}) = \sum_{i=1}^n (y(\mathbf{w}, x_i) - y_i)^2$$

funcion de error o perdida

3. **Entrenaremos** el modelo para aprender \mathbf{w} de modo que minimicen el MSE en los datos de entrenamiento.
4. Medir el rendimiento usando MSE...
usando f error cuadratico



Code Example: Scikit-learn y Regresión Lineal



Un problema de regresión: pasos

Intentemos mejorar los resultados...

Receta ML (II)

1. Preprocesar datos e **ingeniería de características.**

manipular la entrada, x ,
para que sea mas preciso

Raw data: pixel grid		
Better features: clock hands' coordinates	{x1: 0.7, y1: 0.7} {x2: 0.5, y2: 0.0}	{x1: 0.0, y2: 1.0} {x2: -0.38, y2: 0.32}
Even better features: angles of clock hands	theta1: 45 theta2: 0	theta1: 90 theta2: 140

2. Elegir un modelo.

3. ...

Para 1 + 2 usaremos **regresión lineal polinómica**:

$$y(\mathbf{w}, x) = w_0 + w_1 \cdot x + w_2 \cdot x^2 + \dots = \sum_{i=1}^M w_j x^j$$

Code Example: Ingeniería de Características

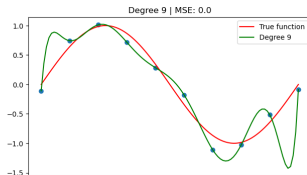
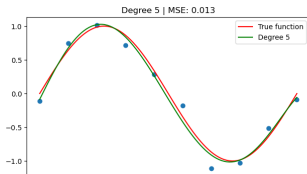
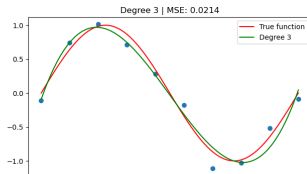
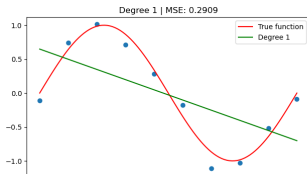
Code Exercise: Comparando modelos...

La necesidad de datos de entrenamiento y prueba

¡Medir el rendimiento en los datos de entrenamiento es incorrecto! Necesitamos los llamados **datos de prueba**.

overfitting
error típico

no podemos
medir
rendimiento
sobre el
mismo
conjunto sobre
el que hemos
aprendido



La figura también es interesante ya que muestra la tensión entre la **complejidad del modelo** y los tres “régimenes”: **underfitting** (ajuste insuficiente), ajuste correcto y **overfitting** (sobreajuste).

Receta ML (III)

1. **Preprocesar** datos e **ingeniería de características**.
2. Elegir un modelo.
3. **Dividir datos en conjuntos de entrenamiento y prueba.**
4. **Entrenar el modelo en el conjunto de entrenamiento** para intentar maximizar el rendimiento.
5. **Medir el rendimiento** real en el **conjunto de prueba**.

Usar mismos datos para cosas distintas no es buena idea

Code Example: División entrenamiento-prueba

Un problema de regresión: pasos

El procedimiento anterior se puede mejorar... En realidad hay **fuga de datos** en nuestro procedimiento. **Estamos usando datos de prueba para dos propósitos diferentes: seleccionar los hiperparámetros del mejor modelo y medir el rendimiento.**

hyperparametros: grado polinomio
Elegir mejor y usar ese IRL

Nuevo problema: estamos usando los datos para comparar modelos y para medir el rendimiento

Receta ML (IV)

1. Preprocesar datos e ingeniería de características.
2. Elegir un modelo.
3. Dividir datos en **conjunto de entrenamiento**, **conjunto de validación** (para comparación de modelos o selección de hiperparámetros), y **conjuntos de prueba**.
4. **Para cada configuración de hiperparámetros/modelo...**
 - Entrenar el modelo en el conjunto de entrenamiento para intentar maximizar el rendimiento.
 - **Medir rendimiento en el conjunto de validación.**
5. **Seleccionar la mejor configuración de hiperparámetros/modelo** basada en las métricas del conjunto de validación.
6. **Medir el rendimiento de todo el procedimiento en el conjunto de prueba.**

Generalmente no usar mismos datos para dos cosas diferentes

Code Example: Procedimiento completo con división entrenamiento-validación-prueba

Regularización

Queremos usar modelos complejos, pero combatir el sobreajuste. Inspeccionar los coeficientes de diferentes modelos produce una idea:

	$M = 0$	$M = 1$	$M = 6$	$M = 9$	
w_0^*	0.19	0.82	0.31	0.35	
w_1^*		-1.27	7.99	232.37	
w_2^*			-25.43	-5321.83	mucha magnitud
w_3^*			17.37	48568.31	
w_4^*				-231639.30	ajustan sus coeficientes con numeros demasiado grandes
w_5^*				640042.26	
w_6^*				-1061800.52	penalizar estos numeros grandes
w_7^*				1042400.18	
w_8^*				-557682.99	
w_9^*				125201.43	Regularizacion

¡Vamos a **penalizar** los pesos grandes! Esto se llama **término de regularización**

$$E(\mathbf{w}) = \sum_{i=1}^n (y(\mathbf{w}, x_i) - y_i)^2 + \underbrace{\lambda \cdot \|\mathbf{w}\|^2}_{\text{penalización}}$$

Durante el entrenamiento se pone un termino que penaliza estos numeros grandes

Esto se llama **Regresión Ridge**.

Cuanto mas grande la magnitud, mas grande la penalizacion

Code Exercise: Pipeline

Code Exercise: Coeficientes del modelo de regresión

Code Exercise: Regresión Ridge

IDEAS A RECORDAR

- ¡No midas el rendimiento en el conjunto de entrenamiento! Debes usar un conjunto de prueba.
- Si quieres ajustar hiperparámetros/comparar modelos también necesitarás un conjunto de validación.
- El underfitting (ajuste insuficiente) y el overfitting (sobreajuste) son dos problemas comunes.
- La regularización es útil para combatir el sobreajuste.