UA

cursos d'estiu | cursos de verano
rafael altamira

Introducción al *Machine Learning*

Aprendizaje supervisado

Profesor: José Javier Valero Mas

Contenidos

- (re-)Introducción al aprendizaje supervisado
- Regresión
  - Definición
  - Técnicas y modelos
- Clasificación
  - Definición
  - Modelos no paramétricos
  - Modelos neuronales
- Evaluación de modelos

Tipos de aprendizaje

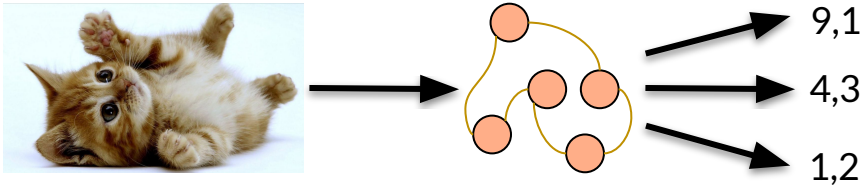
- Aprendizaje supervisado

- Aprendizaje no supervisado

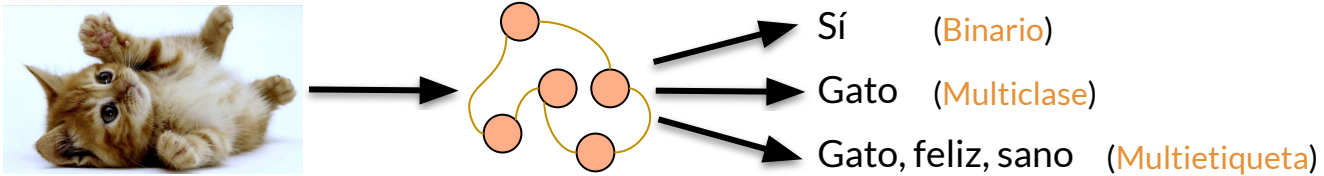
- Aprendizaje por refuerzo

(re-)Introducción al aprendizaje supervisado

- Existe un objetivo concreto a inferir a partir de los datos:
  - **Regresión**: Un valor o valores de naturaleza numérica (valor real)

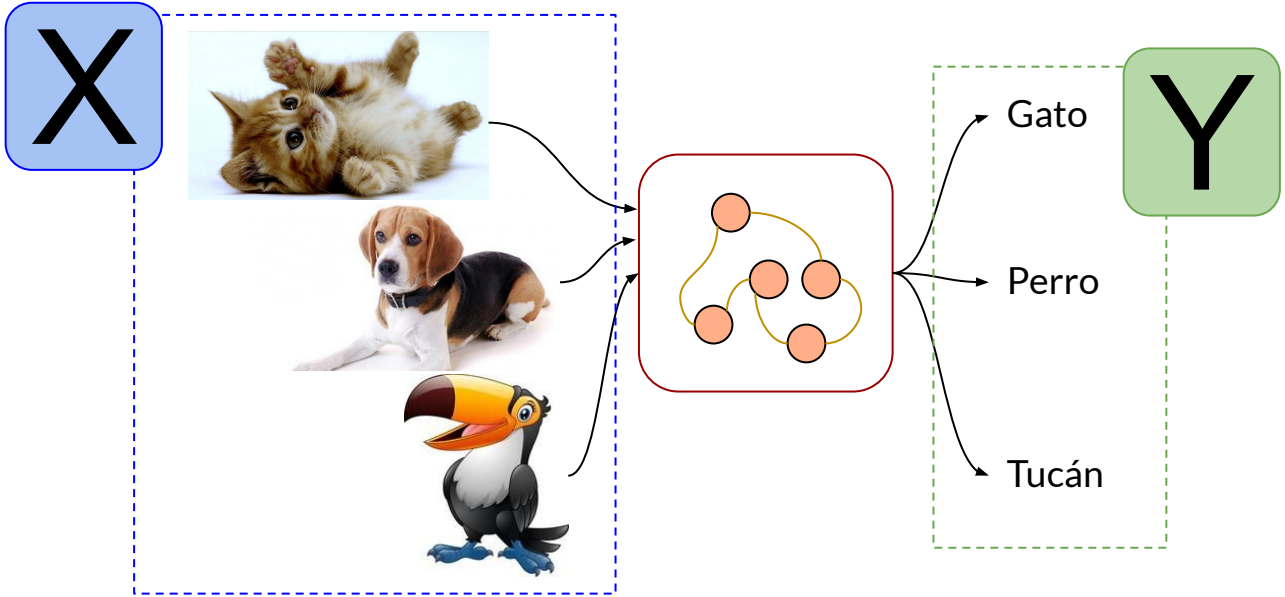


  - **Clasificación**: Una categoría de un conjunto de posibilidades



(re-)Introducción al aprendizaje supervisado

- Necesitamos datos etiquetados para entrenar el modelo

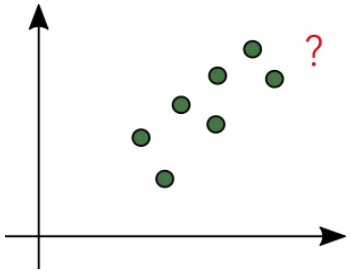


- Por contra, constituye una de sus principales limitaciones

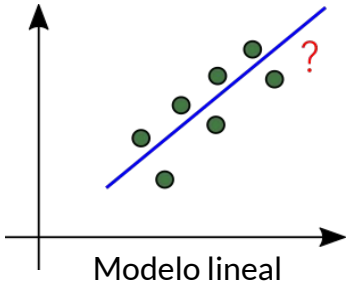
Regresión

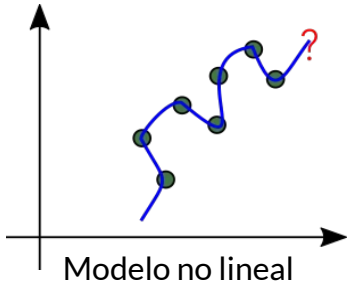
Regresión

- Predecir el valor del punto señalado como ? en:



- ¿Posibilidades?



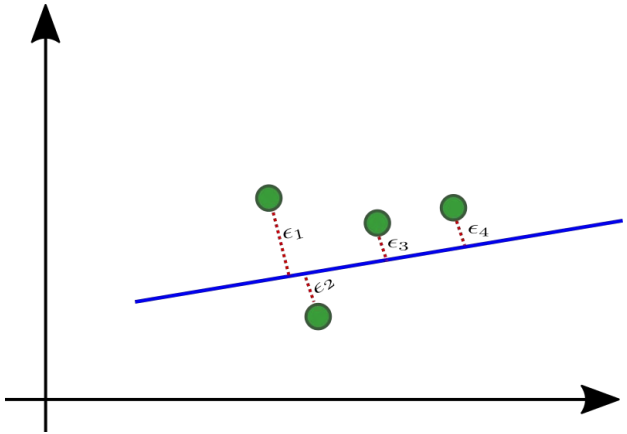


Regresión

- Posibles problemas a modelar:
  - Mercado de valores
  - Predicción de ventas
  - Puntuación de una película
  - ...

Regresión

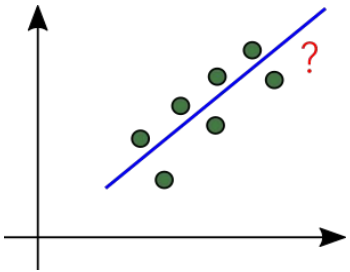
- Objetivo: obtener una función que minimice el error con los datos dados



$$\epsilon_T = \epsilon_1 + \epsilon_2 + \epsilon_3 + \epsilon_4$$

- Medidas típicas de error:
  - Error Absoluto Medio (*Mean Absolute Error*, MAE)
  - Error Cuadrático Medio (*Mean Square Error*, MSE)

Regresión lineal

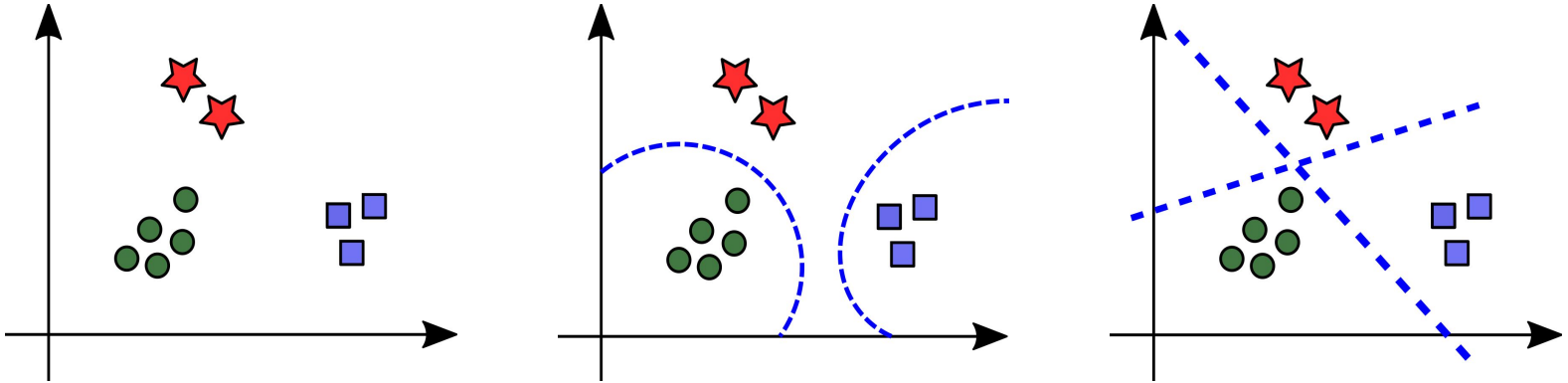


- **Características**:
  - Valor de salida → suma ponderada de todos los descriptores
  - Sencillo de implementar
- **Limitaciones**:
  - Problemas de tipo lineal
  - Todos los descriptores → ¿todas las características son útiles?
- **Alternativas**:
  - LASSO (*Least Absolute Shrinkage and Selection Operator*)
  - Modelos no lineales → Los veremos en clasificación

Clasificación

Frontera de decisión

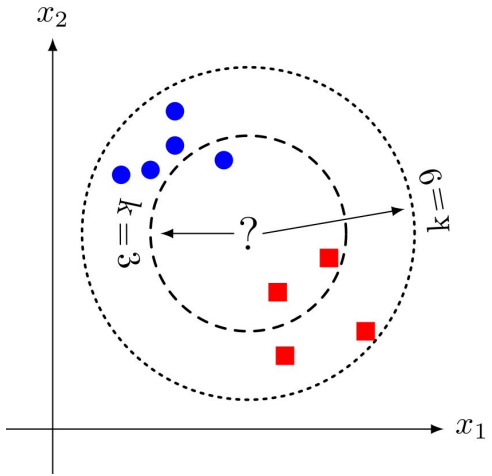
- **Clasificador**: mecanismos para modelar fronteras de decisión
- Estas fronteras dividen el espacio de características en categorías

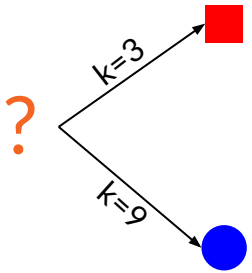


- Estudiaremos cómo obtienen estas fronteras diferentes clasificadores

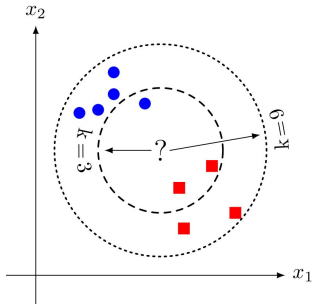
k-Nearest Neighbor (*k*NN)

- **Premisa**: dada una consulta ?, se le asigna la clase más común entre los *k* elementos más cercanos
- **Cercanía**: se necesita una medida de (di)similitud para poder estimarla





k-Nearest Neighbor (*k*NN)

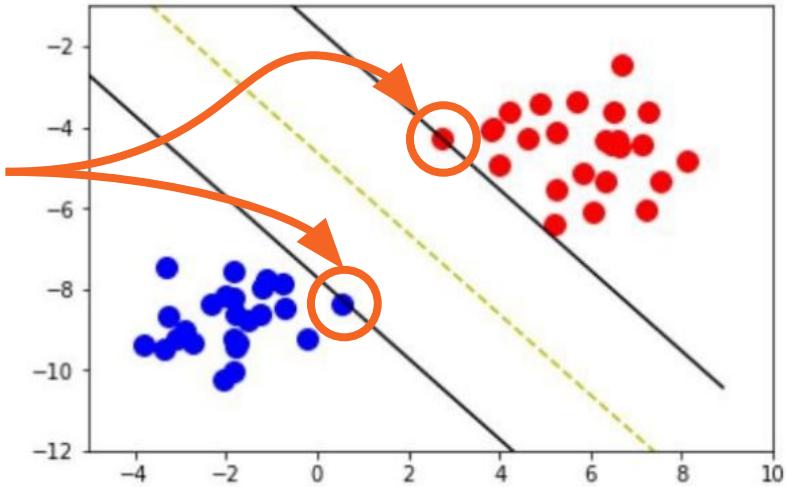


- **Ventajas**:
  - Sólo requiere una medida de disimilitud → aplicable a datos de tipo estadístico y estructurados
  - No requiere entrenamiento
  - Aplicable a casos binarios y multiclase; fácilmente adaptable a multietiqueta
- **Desventajas**:
  - Requiere examinar de manera exhaustiva el conjunto de entrenamiento para cada consulta

Support Vector Machines (SVM)

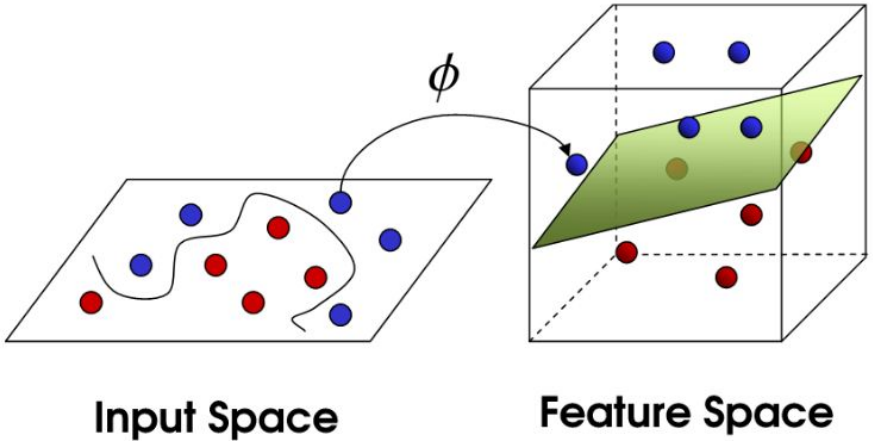
- **Premisa**: clasificador binario que crea un hiperplano que separa el espacio de características en dos zonas, una por categoría
- **Entrenamiento**: sitúa el hiperplano de manera que la distancia a los puntos más cercanos de cada clase sea máxima

Support Vectors



Support Vector Machines (SVM)

- SVM asume que las clases son linealmente separables → Muy infrecuente
  - ***Kernel trick*:** Cambiar el espacio de representación (normalmente a uno de mayor dimensionalidad) que cumpla la premisa de linealidad

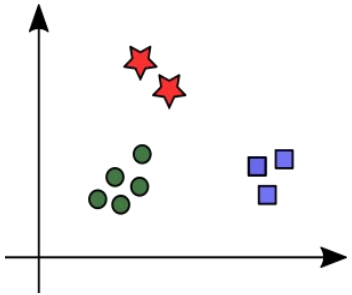


  - ***Kernels* típicos**: *Polynomial*, *Radial Basis Function* (RBF)

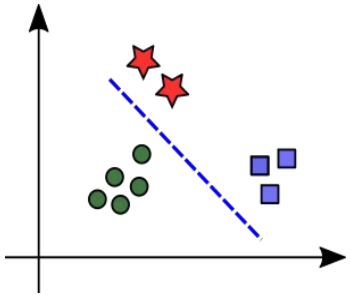
Support Vector Machines (SVM)

- ¿Sólo vale para problemas binarios? Se puede adaptar a multiclase:
  - *One-VS-All*: Enfrentar una clase contra una agrupación del resto
  - *One-VS-One*: Enfrentar pares de clases

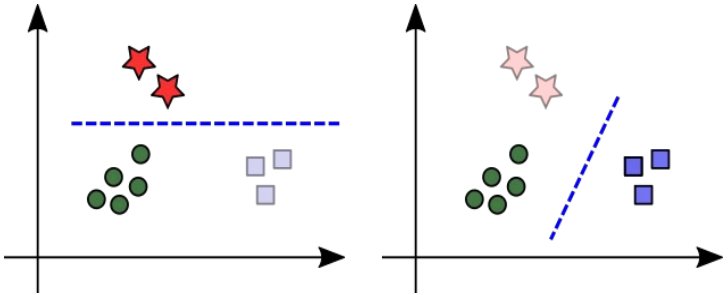
Tarea



SVM-OVA (



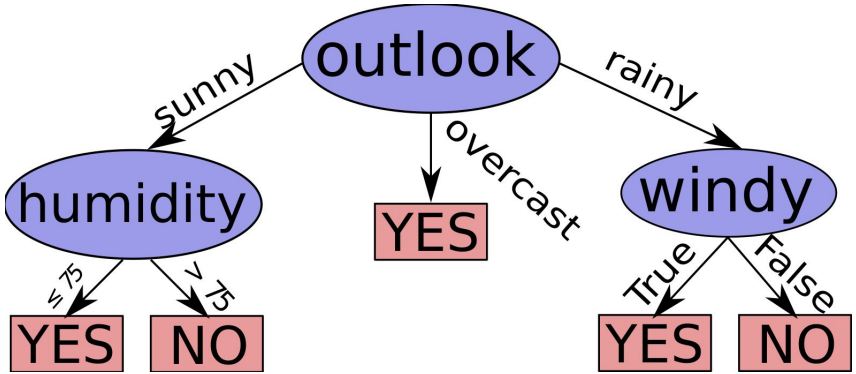
SVM-OVO (



- También puede darse que no haya manera de separar las clases de manera lineal en ningún espacio de representación
  - Se relaja la condición de separación total de clases (parámetro C)

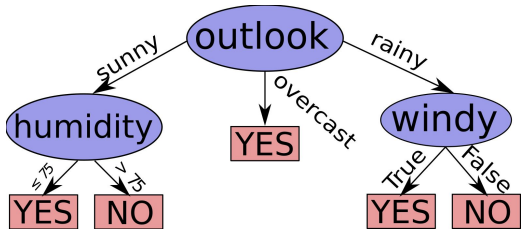
Árboles de decisión

- **Premisa**: dada una consulta **?**, se le asigna la clase obtenida recorriendo un árbol construido en base al conjunto de entrenamiento



- **Árbol**:
  - Nodos: Atributos (**X**)
  - Hojas: Categorías/etiquetas (**Y**)

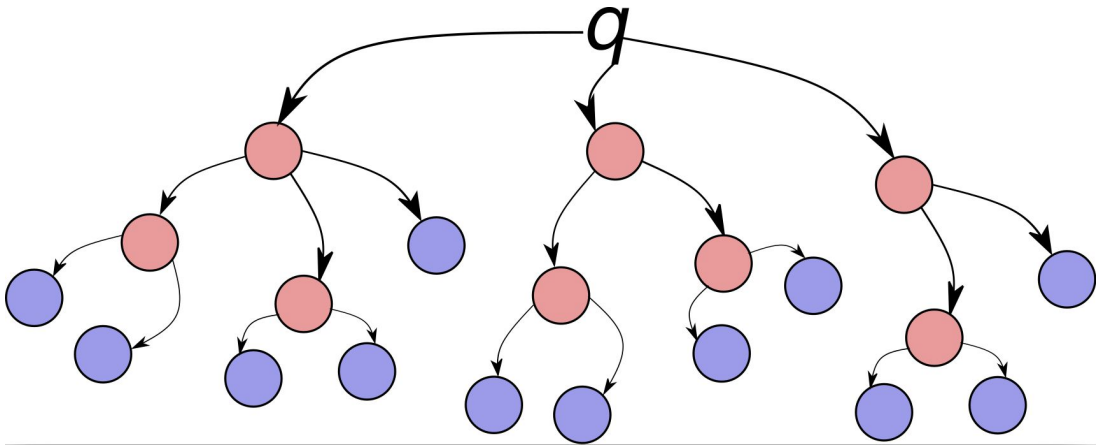
Árboles de decisión



- **Entrenamiento**:
  - Ganancia de información → Capacidad de los atributos de separar las clases
  - Sobreajusta de manera natural → Poda del árbol (*pruning*)
- **Características**:
  - Altamente eficiente
  - Interpretable → Se puede obtener un conjunto de reglas
  - Aplicable a casos binarios y multiclase; fácilmente adaptable a multietiqueta

Random Forest

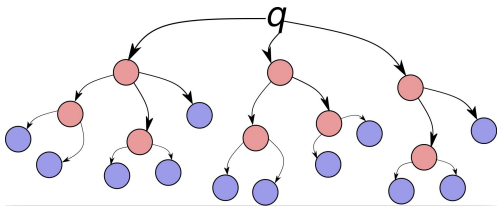
- Caso particular en el que se crean diferentes árboles de decisión y se unen sus predicciones individuales

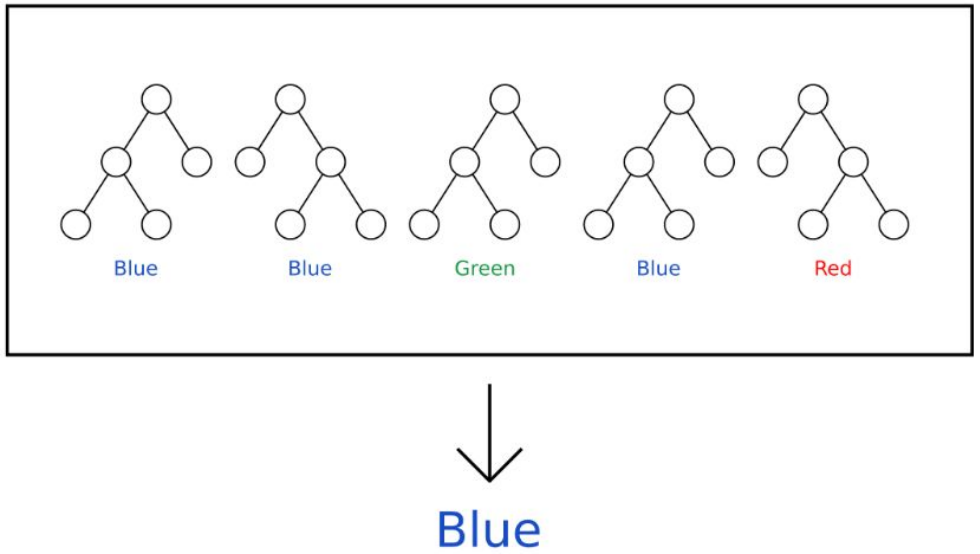


- Añade robustez a la clasificación
- Realmente pertenece a la familia de *ensemble learning*

Random Forest

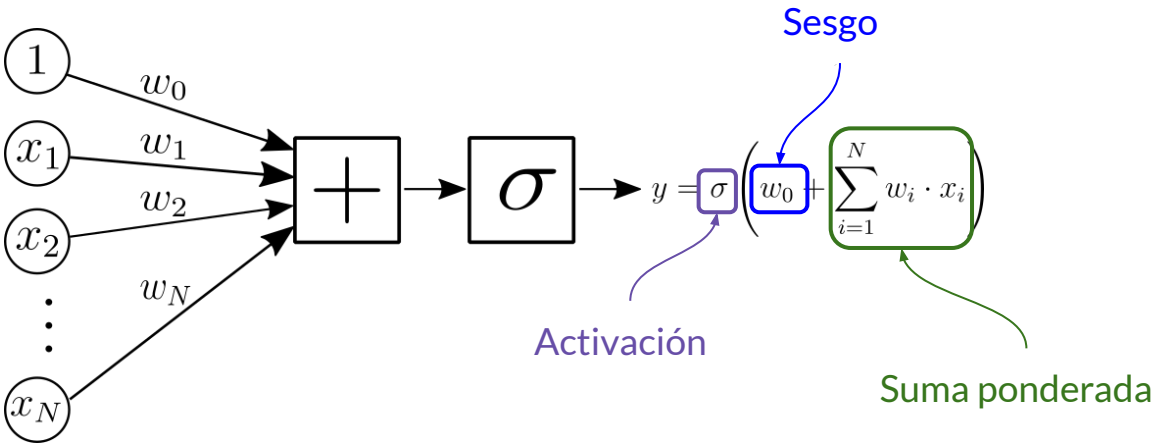
- Predicción (clase) final → moda de las predicciones de cada árbol





Modelos neuronales artificiales

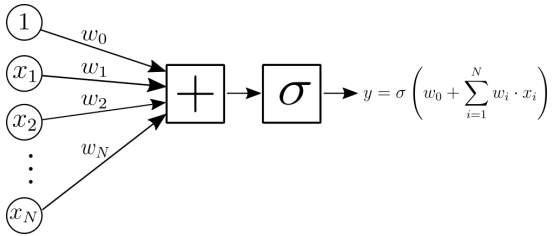
- Modelo que imita el comportamiento de las neuronas biológicas
- Unidad básica: **perceptrón**
  - Suma ponderada de las entradas con una función de activación

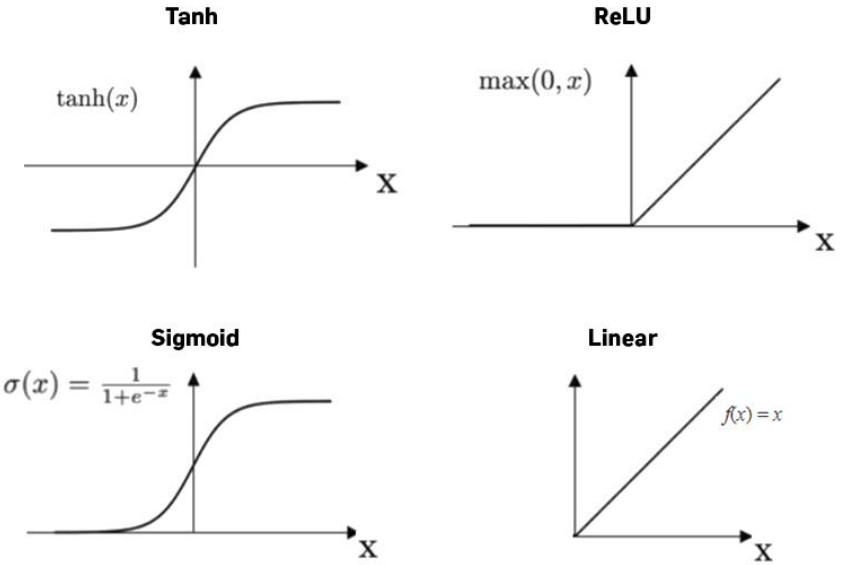


- Objetivo: aprender los pesos (w)

Modelos neuronales artificiales

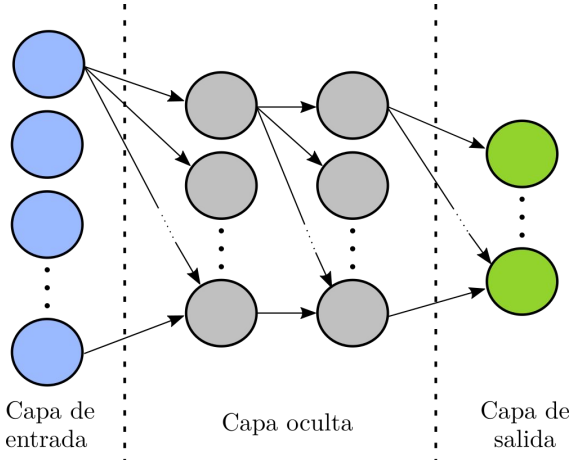
- Función de activación:



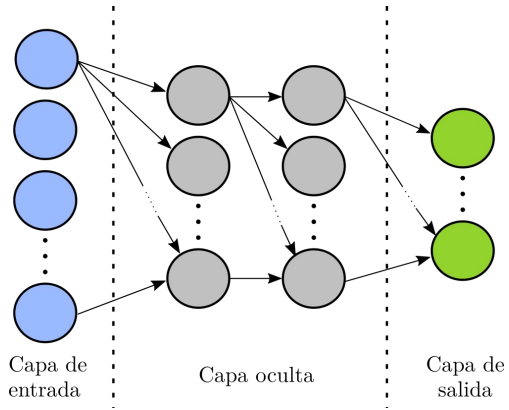


Modelos neuronales artificiales

- Se estructuran por capas:
  - Capa de entrada (*input*): Entrada de datos. No son perceptrones
  - Capa/s oculta/s (*hidden*): Normalmente no lineales
  - Capa de salida (*output*): Reporta el resultado

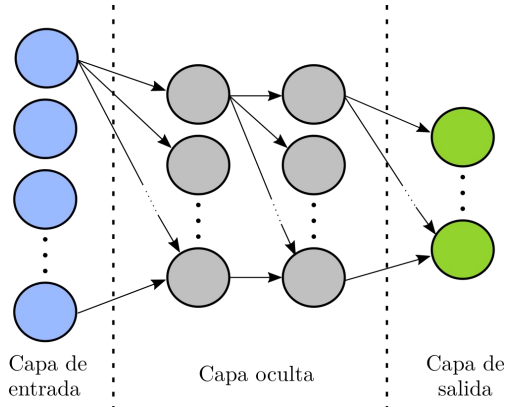


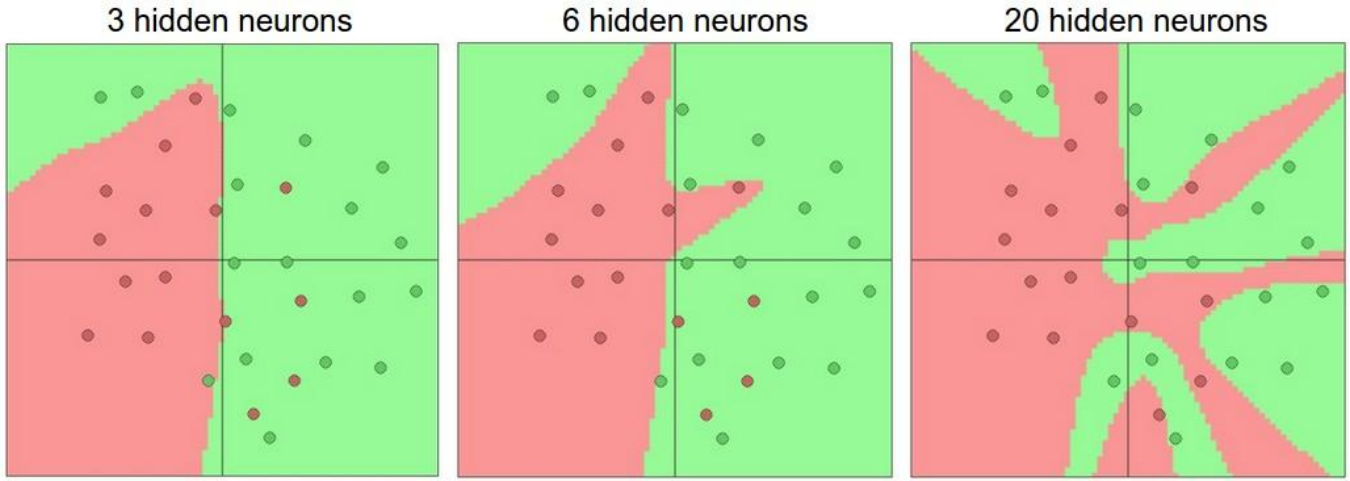
Modelos neuronales artificiales



- Capa de entrada: Tamaño igual a la cantidad de características
- Capa/s oculta/s: Según complejidad del problema y datos disponibles
- Capa salida:
  - Regresión: Un perceptrón por elemento a predecir
  - Clasificación:
    - Binaria: Un perceptrón por clase/dos perceptrones
    - Multiclase: Un perceptrón por clase

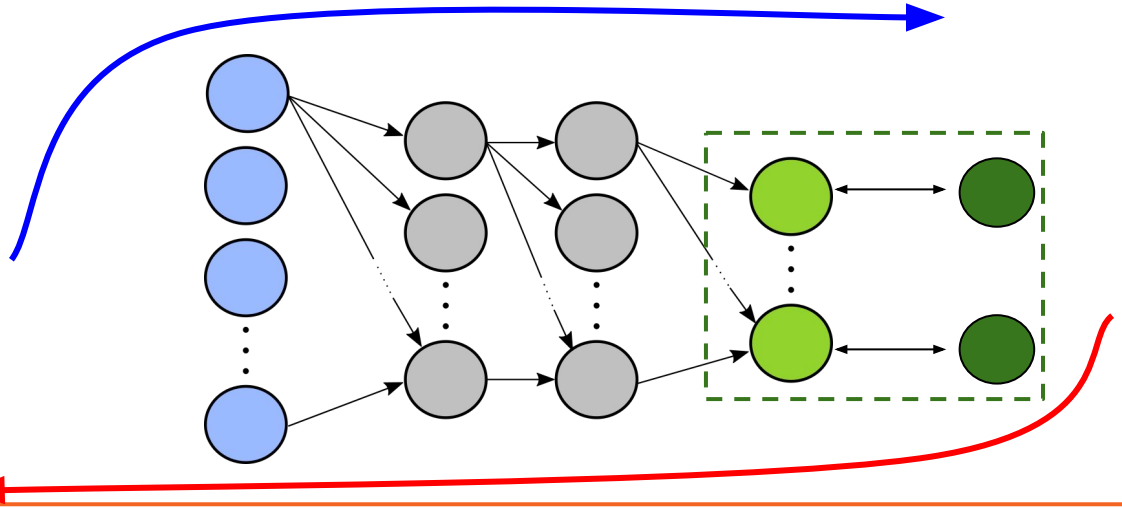
Modelos neuronales artificiales



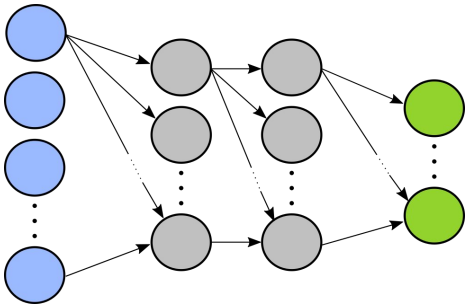


Modelos neuronales artificiales

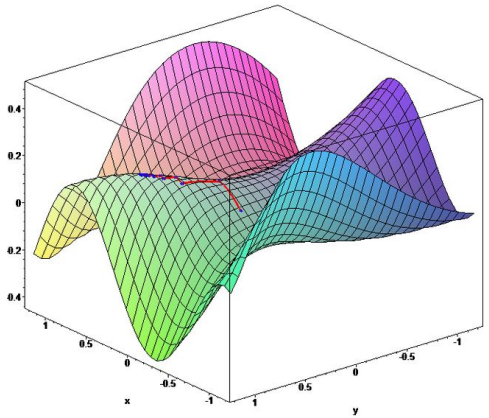
- Proceso de entrenamiento:
  1) Se introduce un dato en la red y se obtiene la salida (*forward*)
  2) Se compara el resultado obtenido con el esperado (pérdida o *loss*)
  3) Se propaga el error hacia atrás para adaptar los pesos (*backpropagation*)

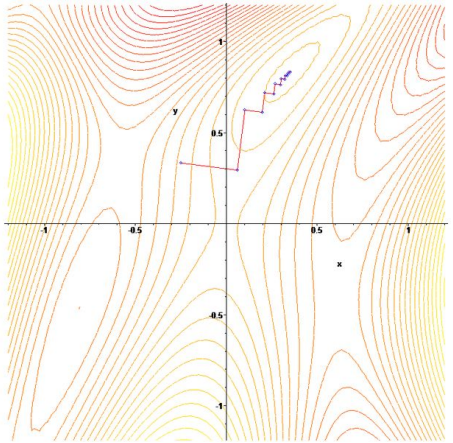


Modelos neuronales artificiales

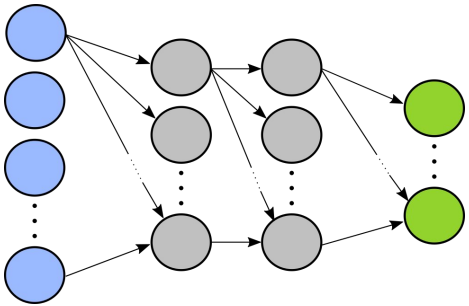


- Descenso por gradiente:
  - Objetivo del entrenamiento: minimizar la función de pérdida
  - Problema: función de gran dimensionalidad → ~~Métodos analíticos~~
  - Recorrer la función de pérdida para buscar un mínimo (idealmente, global)



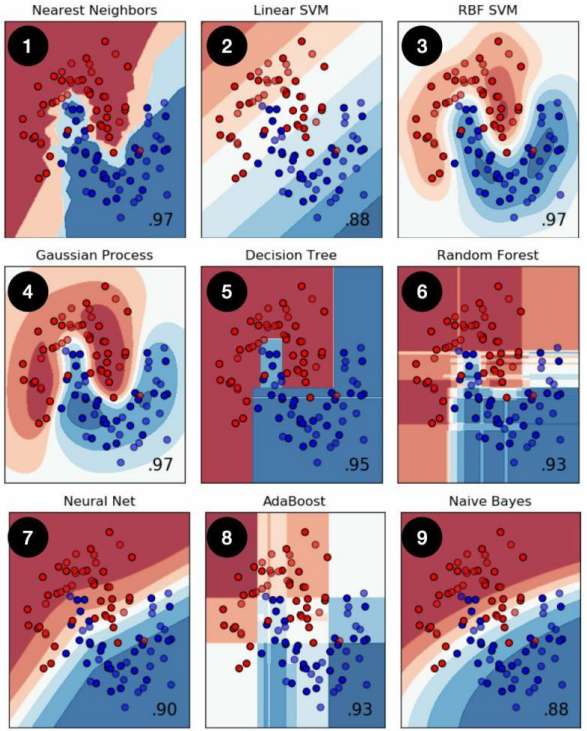


Modelos neuronales artificiales



- Detalles de entrenamiento:
  - Se utilizan grupos de datos (*batches*)
  - Época: cuando la red ha procesado todos los datos de entrenamiento
  - Se itera por varias épocas → Minimizar la pérdida
  - Optimizador → Método utilizado para recorrer la función de pérdida

Ejemplos de fronteras de decisión



Evaluación

Métricas de evaluación

- Hasta ahora hemos asumido que cualquier métrica de evaluación es válida para nuestro problema:
  - Por ejemplo, asumimos que la tasa de acierto es una buena métrica para cualquier problema de clasificación

- Sin embargo esto no es siempre así ya que ciertas métricas se ven *afectadas* por particularidades de nuestros datos
  - **Caso típico**: datos desbalanceados

- Estudiaremos esto con un ejemplo práctico

UA

cursos d'estiu | cursos de verano
rafael altamira

Introducción al *Machine Learning*

Aprendizaje supervisado

Profesor: José Javier Valero Mas