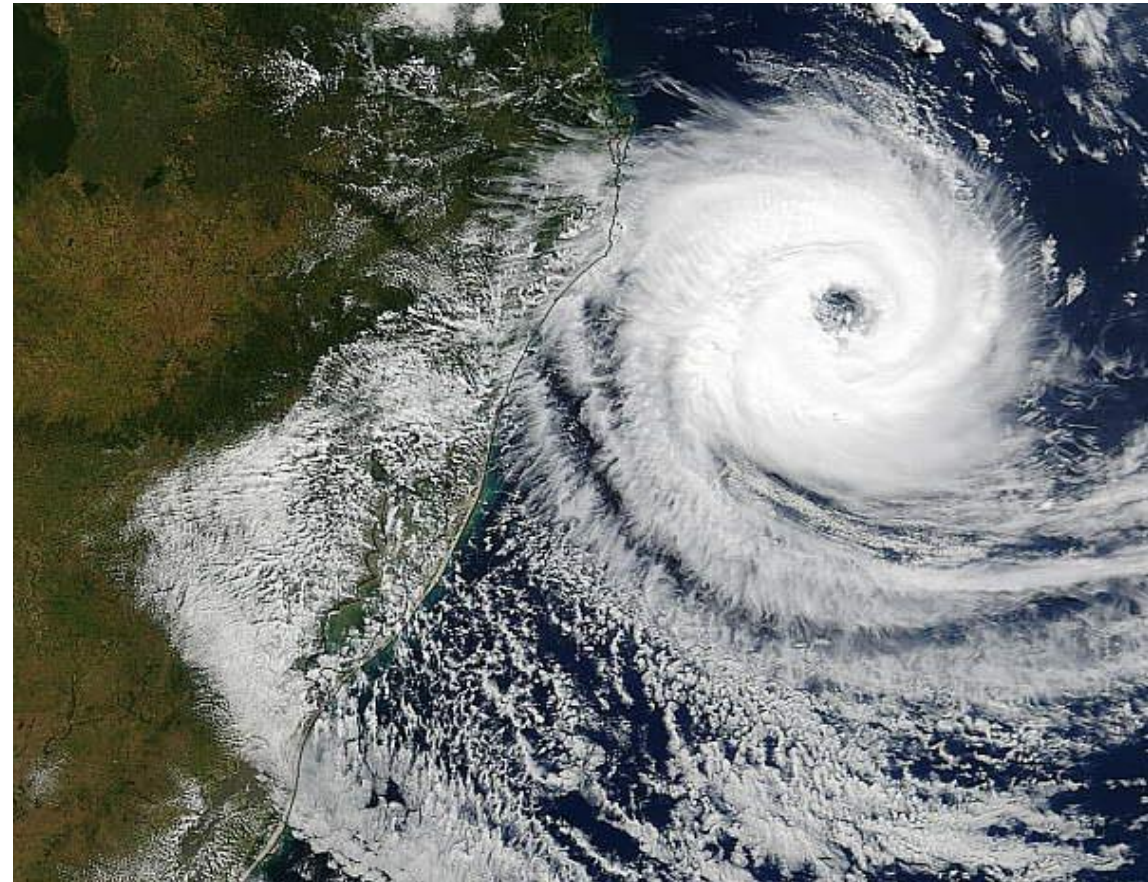


Nociones del aprendizaje de máquinas

Por: Erick Merino

Un poco de teoría de sistemas...

- Todo empieza por un **fenómeno** natural



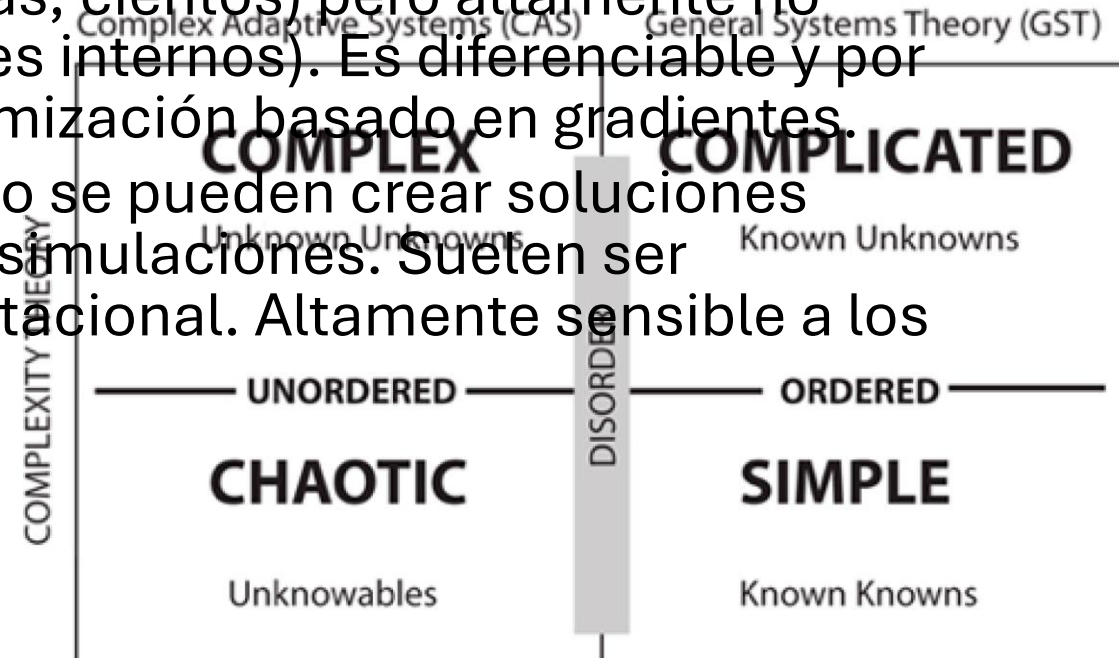
Formalización del fenómeno

- El fenómeno se debe abstraer a un **Sistema**
- Un sistema abstrae las características de un fenómeno natural
- Un primer enfoque es usar un sistema de diagramación
 - Existen sistemas de diagramación especializados para el tipo de sistema, por ejemplo UML
 - También se pueden usar diagramas de contexto
- Eventualmente hay que definir el sistema en función de sus entradas y salidas $y = f(x)$



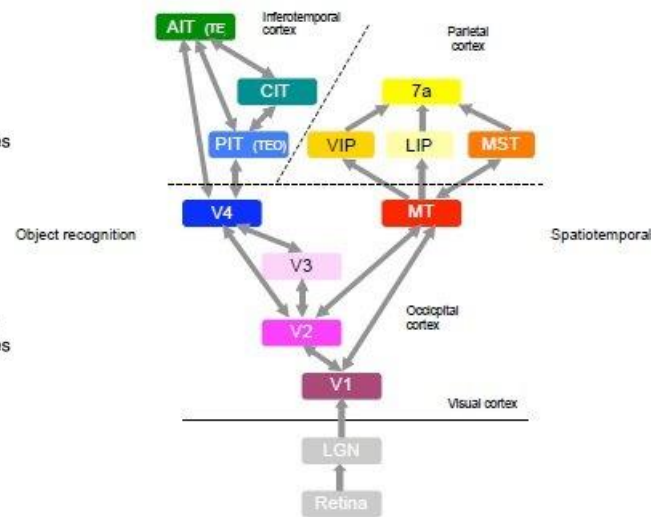
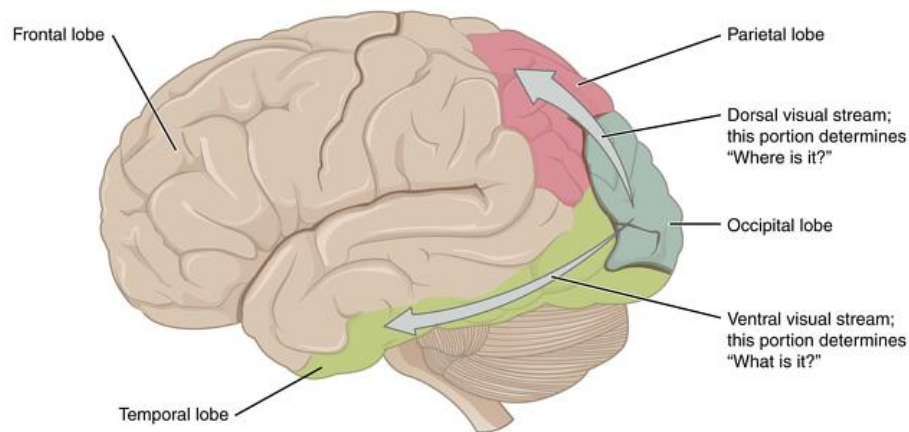
Sistemas complejos

- Un sistema puede ser:
 - Simple: Sistema lineal de pocas variables (contables). Se analiza con cálculo y álgebra.
 - Complicado: Muchas variables (incontables), pero en general es lineal. Se analiza estadísticamente.
 - Complejo: Variables contables (decenas, cientos) pero altamente no lineal (interacciones entre componentes internos). Es diferenciable y por lo tanto se puede usar el cálculo y optimización basado en gradientes.
 - Caótico: No diferenciable/integrable, no se pueden crear soluciones analíticas (cálculo), se deben ejecutar simulaciones. Suelen ser problemas de alta complejidad computacional. Altamente sensible a los datos de entrada, y no lineales.

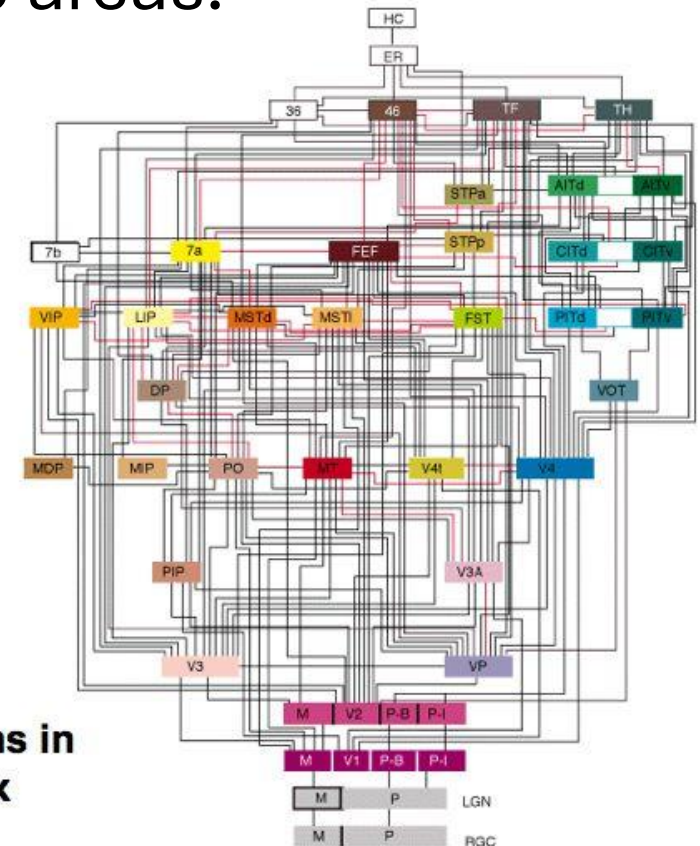


Un ejemplo de sistema complejo

- Las señales ópticas llegan a la parte trasera del cerebro a la corteza visual primaria, que se divide en 3 sub áreas.

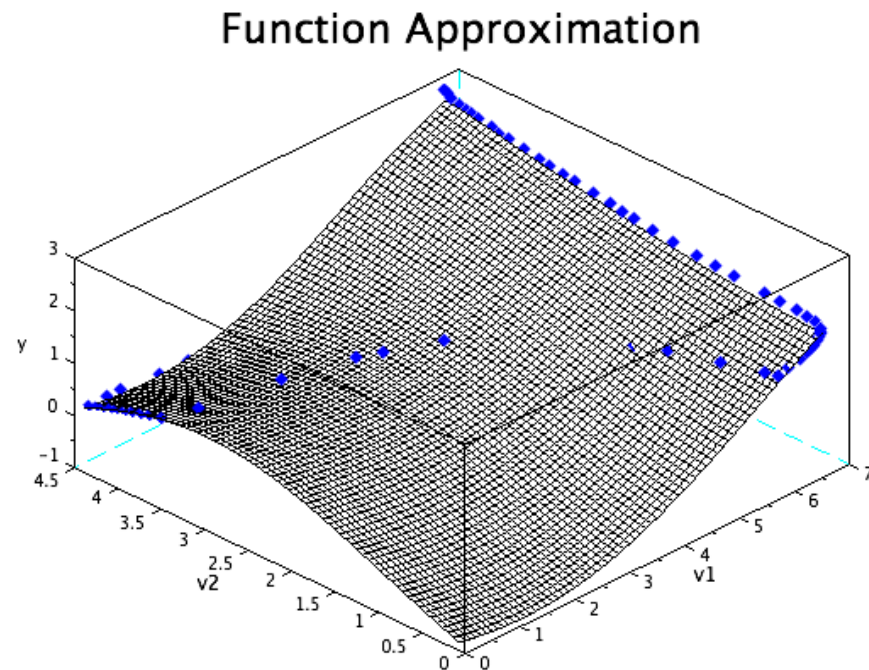


Example: connections in monkey visual cortex



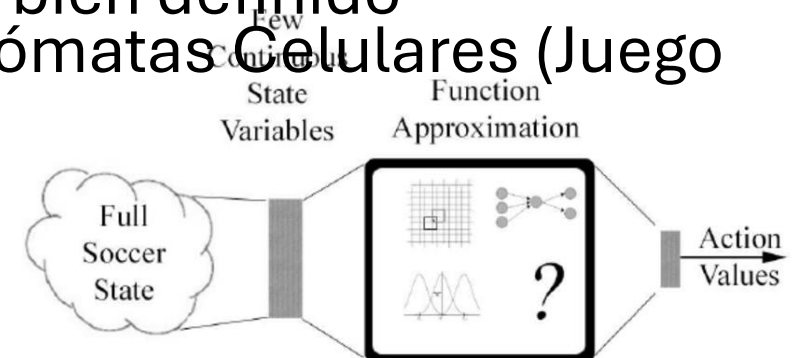
El desafío de aproximar una función

- En el caso que tenemos un sistema, pero no su función matemática definida, debemos encontrar una función matemática que aproxime con una determinada certeza el comportamiento del sistema



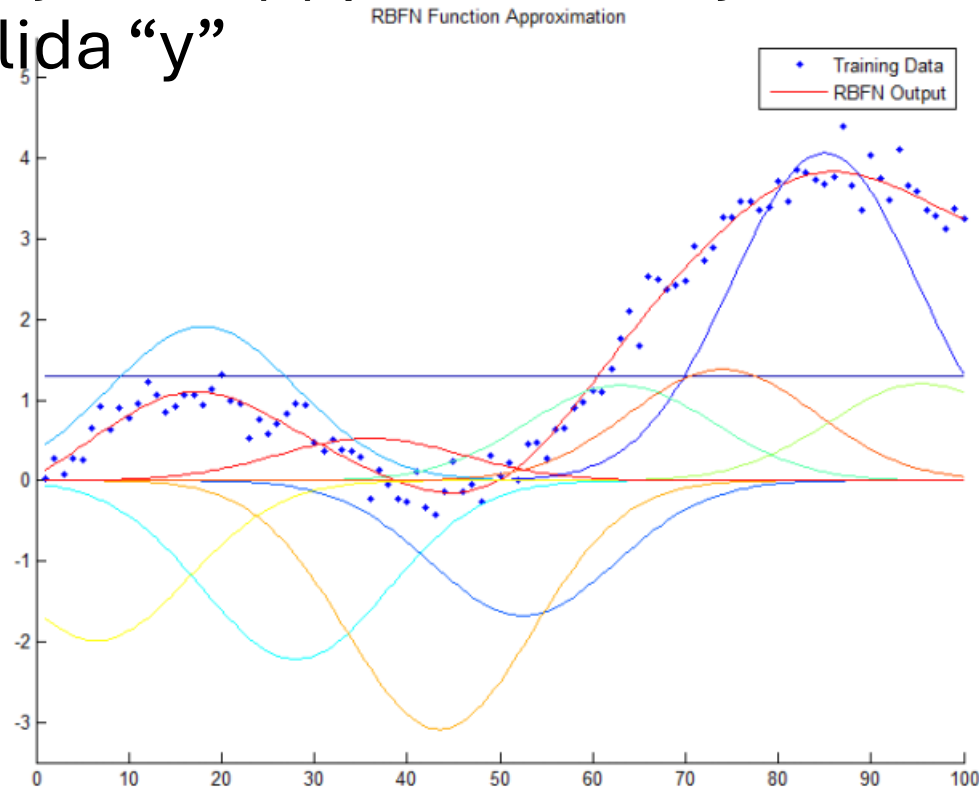
El desafío del aprendizaje de máquinas

- El desafío es entonces dada una función $y = F(x,t)$, llamada función de plantilla (template function), encontrar los parámetros “t” que devuelven una aproximación de $y=f(x)$
- Encontrar funciones de aproximación es un desafío
- Existen “aproximadores universales”, funciones plantilla que son capaces de aproximar cualquier función, demostradamente.
- También se usan funciones más simples, con propósitos de explicabilidad e interpretabilidad de la función resultante.
- También existen los “sistemas universales”, sistemas que pueden contener en su interior cualquier otro sistema bien definido matemáticamente. Ej: Máquina de Turing, Autómatas Celulares (Juego de pi)



Machine learning: Definición

- El desafío entonces es dada una función con parámetro “t” y entrada “x”, encontrar el conjunto de parámetros “t” que aproxime bien la función objetivo $f(x)$ para un conjunto de datos de entrada “x” y datos de salida “y”

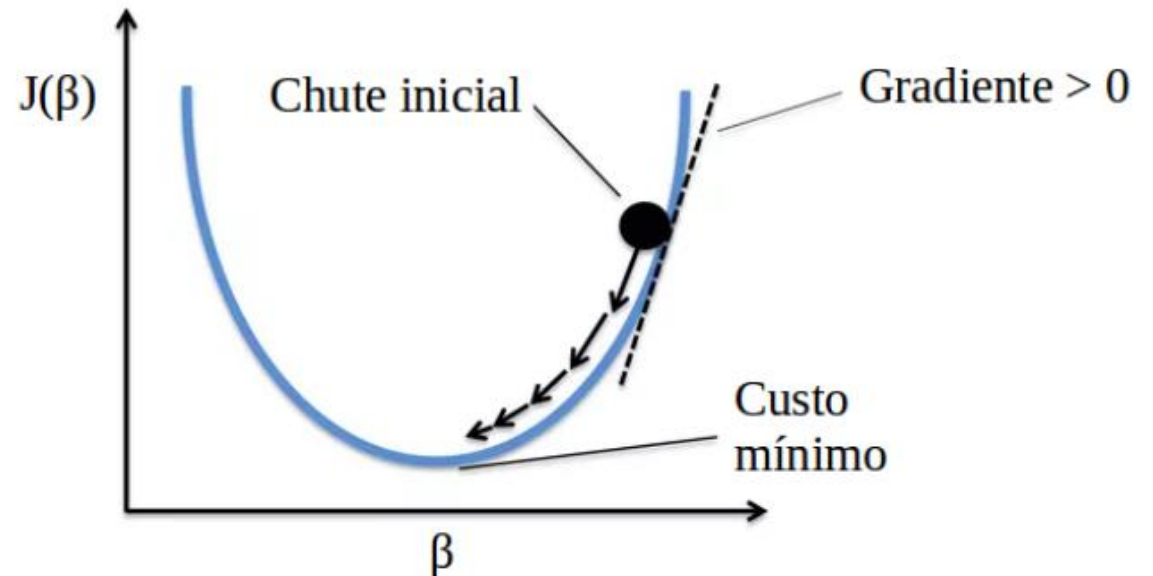


2 tipos de tareas fundamentales

- Existen 2 tipos de tareas fundamentales:
 - Aprendizaje supervisado:
 - Encontrar una función de aproximación para definir $f(x)$ dado una lista de datos pares coordenados x e y
 - Ejemplos: Categorización (variables “ y ” categóricas)
 - Ejemplos: Regresión (variables “ y ” continuas)
 - Aprendizaje no supervisado
 - Encontrar la función de generación de los datos $f(x)$ sin tener un objetivo “ y ”
 - Ejemplos: Clusterización, reducción de la dimensionalidad
 - Ejemplos: Sistemas generativos (GAN, GPT)

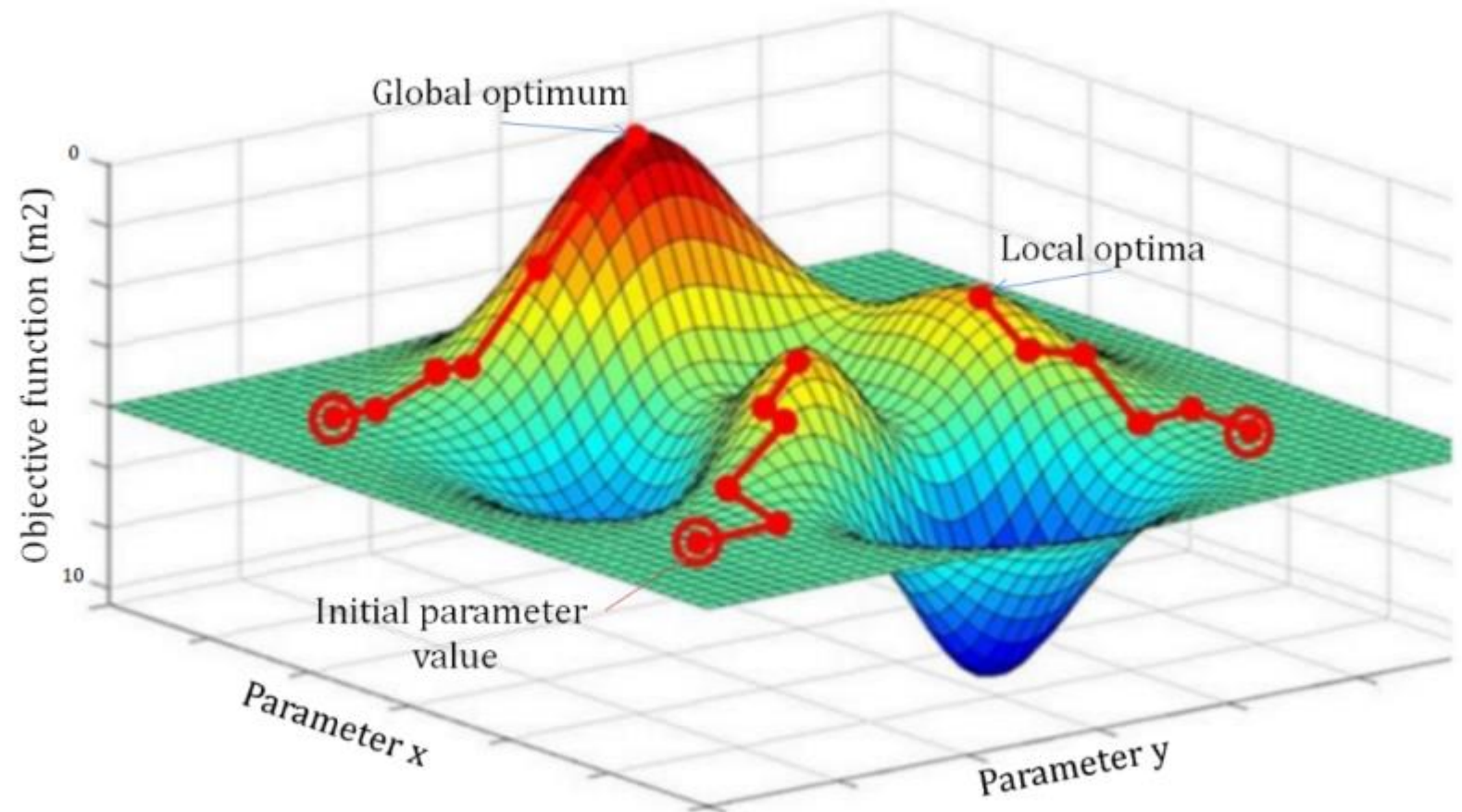
Optimización de parámetros “t”

- Existen diferentes métodos de optimización numérica
- El más popular es el método del descenso del gradiente
- Consiste en que dado un punto “x1” en el espacio de búsqueda de los parámetros “t” a optimizar, calcula su gradiente de la función de error, y “avanza” una tasa de aprendizaje “épsilon”



Ejemplo visual del descenso del gradiente

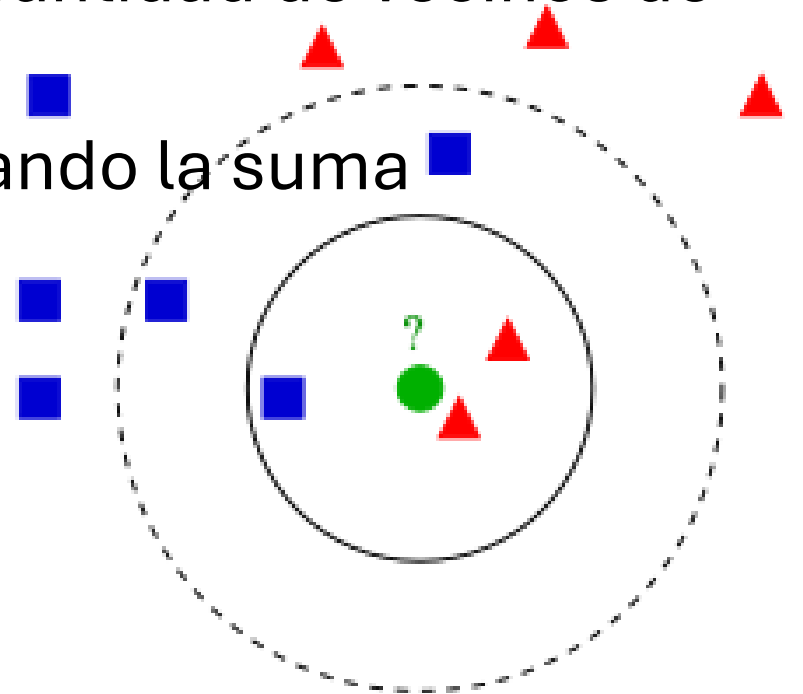
- Se usa un parámetro de aleatoriedad en la búsqueda para evitar óptimos locales



Modelos más populares

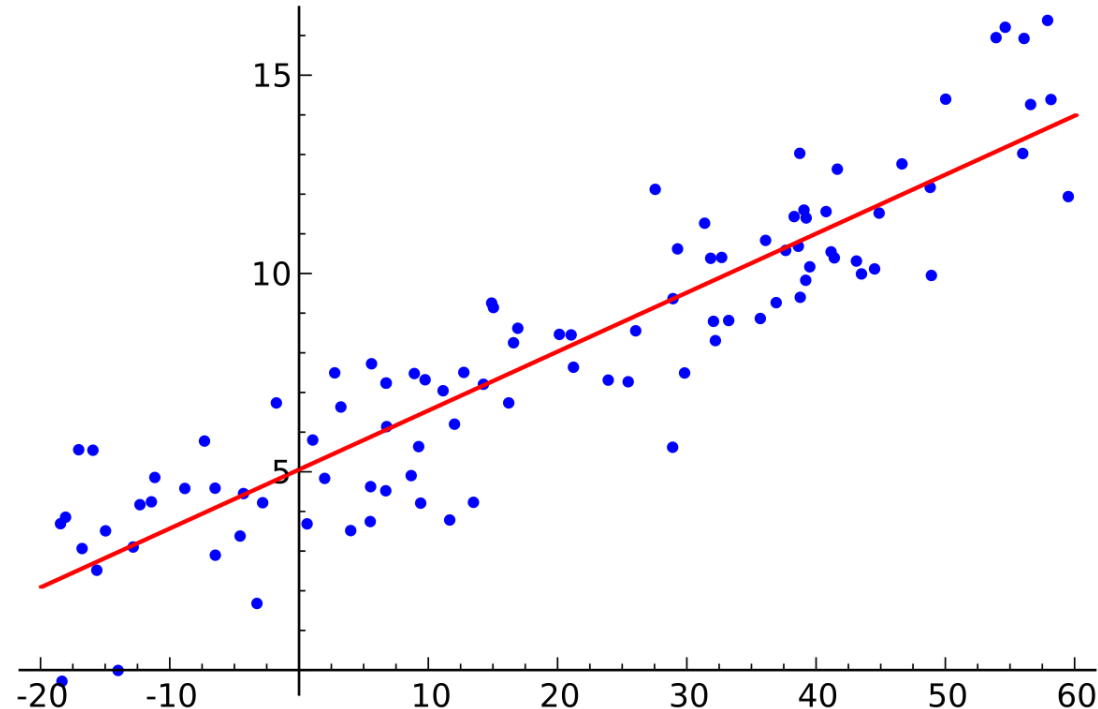
K Nearest Neighbors

- No tiene una función matemática explícita
- Guarda todos los datos de entrenamiento
- Dado un nuevo dato, lo clasifica según la cantidad de vecinos de cada clase o categoría
- También se puede usar para regresión, usando la suma ponderada de los k-vecinos más cercanos




Regresión Lineal

- Usa una función lineal como función de aproximación
- Los parámetros “t” a optimizar son la pendiente y el intercepto



Regresión Logística

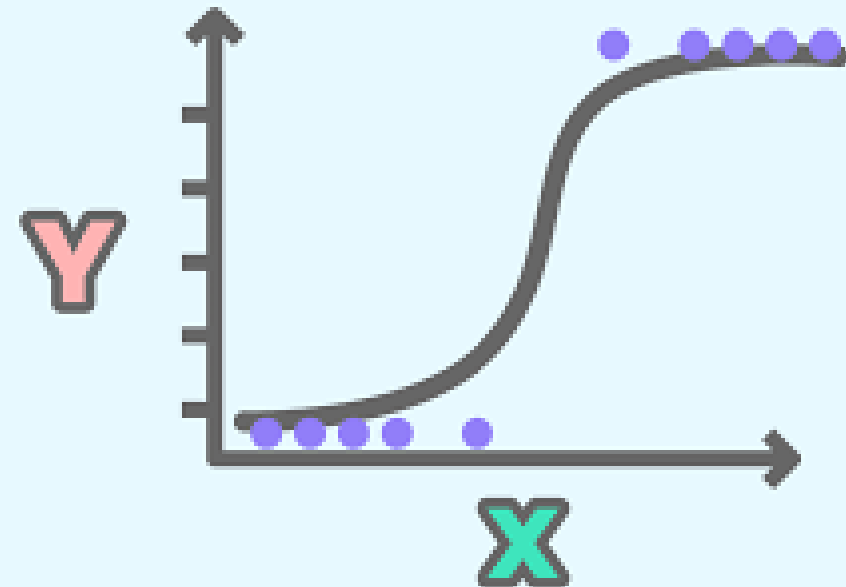
- Usa una función logística (o sigmoide) como función de aproximación
- Los parámetros “t” a optimizar son los parámetros Beta
- Se usa para clasificar entre 2 categorías binarias

**REGRESIÓN LOGÍSTICA BINARIA**

ECUACIÓN:
$$y = \frac{1}{1 + e^{-f(x)}}$$

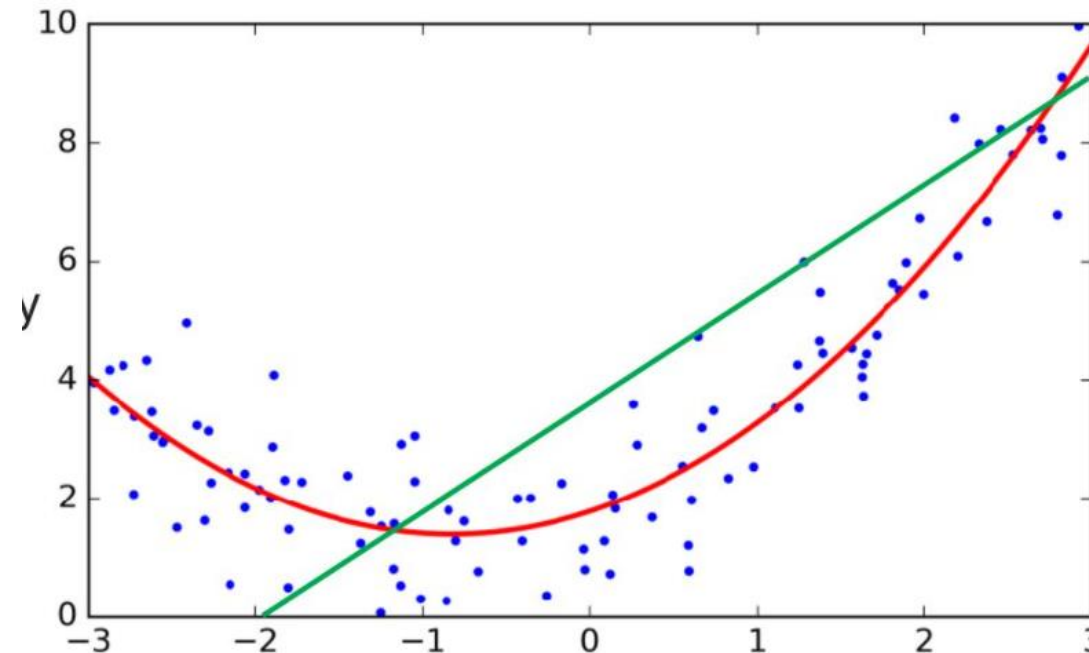
DONDE:
$$f(x) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 \dots + \beta_n x_n$$

y : Variable a predecir
 x : Variable predictora
 β : Coeficiente
 e : Constante épsilon



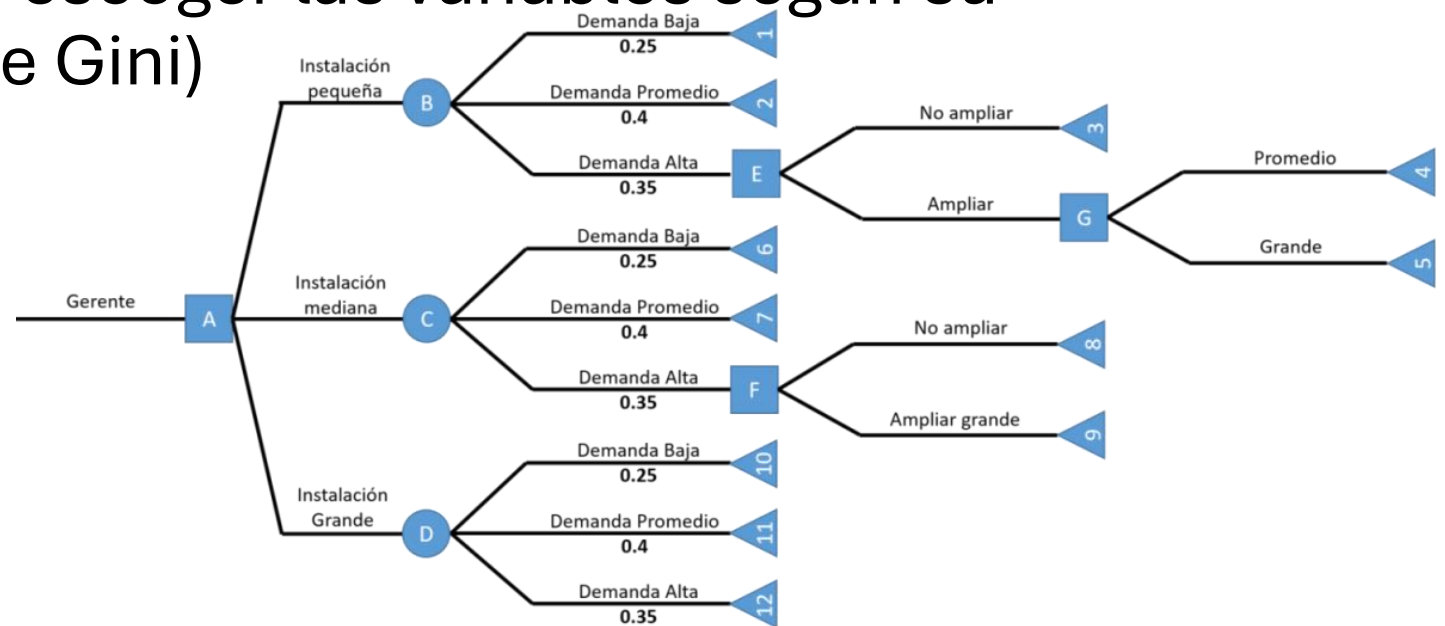
Regresión polinomial

- Usa un polinomio de grado dado por el programador (hiperparámetro)
- Se usa principalmente en regresión de patrones simples causales o semicausales donde la función lineal es insuficiente



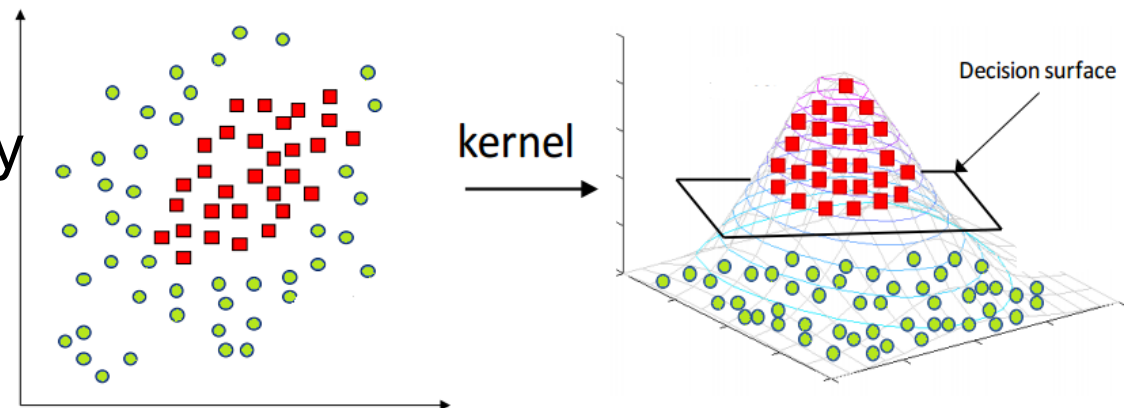
Árboles de decisión

- El modelo consiste en un árbol, donde cada nodo intermedio es una decisión (una variable elegida y un rango de esa variable), y cada nodo final (hoja) es una clasificación en relación con la proporción de los datos resultantes.
- Una estrategia común es escoger las variables según su dispersión (coeficiente de Gini)



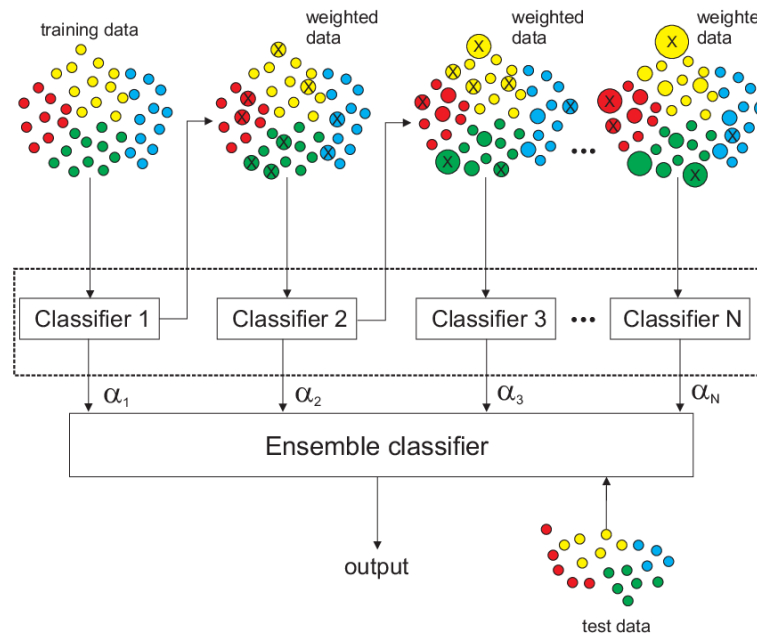
SVM: Máquinas de soporte de vectores

- Es una función de aproximación universal (demostrado)
- Matemáticamente es equivalente a una red neuronal de 2 capas
- Usando el “truco del núcleo” construye variables “extra” o artificiales para mapear un espacio de variables $f(x,t)$ e R^n a una o varias dimensiones extra
- En este espacio multidimensional con variables extra, usa un clasificador lineal para separar 2 categorías (sólo es un clasificador binario)
- En caso de usar para clasificación múltiple, se usan varios SVM binarios y un sistema de votación



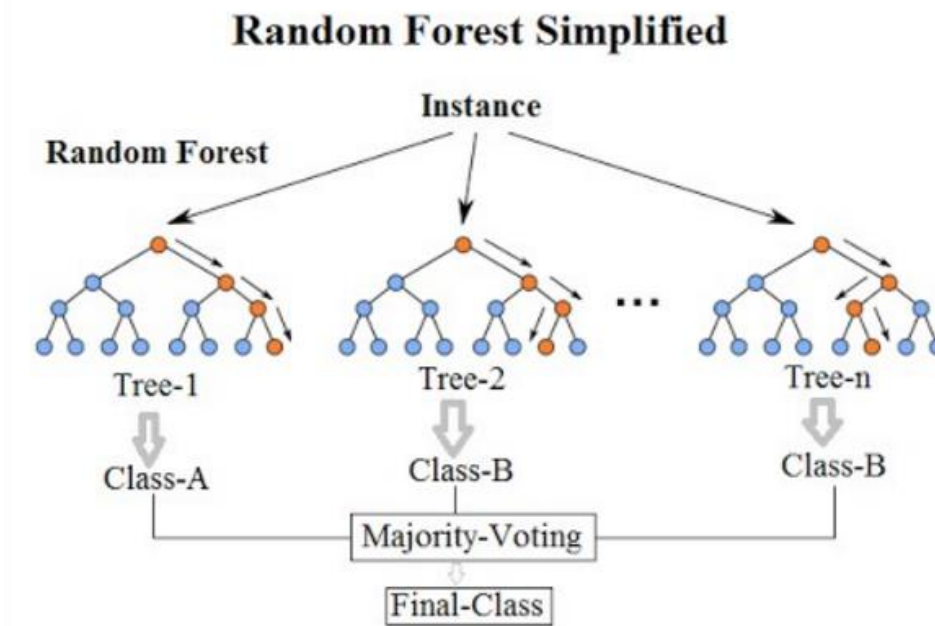
Métodos ensamblados: Adaboost

- Usa n funciones de aproximación
- Por cada función de aproximación le asigna un “peso” en función de su capacidad de predecir la función $f(x)$
- El modelo final es una combinación lineal de varios otros modelos más simples



Bosques aleatorios (Bagging)

- Usa n árboles de decisión como modelos base
- El modelo resultante es el resultado de aplicar una votación de cada modelo base

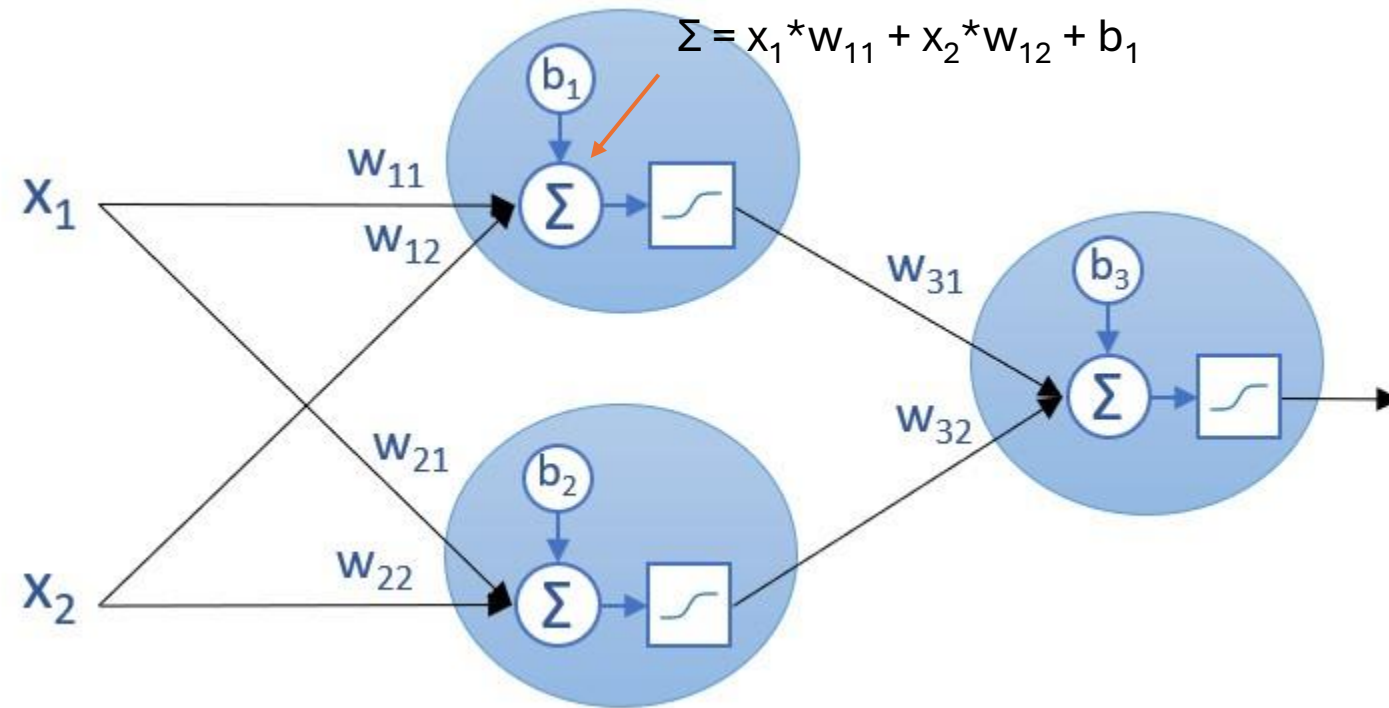


Redes neuronales

Redes neuronales

- Son una familia de modelos o funciones de aproximación
- Se basan en nodos interconectados (neurona), donde cada nodo es una función lineal + función no lineal de activación
- Según el esquema de conexiones se definen diversas arquitecturas: Feed Forward, Convolutacional, Recurrente, Transformer, etc.
- Los parámetros “t” de optimización son los pesos de la función lineal de cada nodo
- Se puede optimizar usando la técnica de Backpropagation, que es usando derivadas parciales de orden creciente según la capa (para la capa 5, usa derivadas parciales de orden 5) para calcular el gradiente.

Esquema de una red neuronal básica (los parámetros w_{ij} y b_i son los optimizables)



Feed Forward

- Todas las neuronas de una capa se conectan con todas las neuronas de la siguiente
- En inferencia, siempre se mueven hacia adelante. En entrenamiento, se usa backpropagation hacia atrás

