5.1 ¿Qué es el Aprendizaje Automático?

Un programa informático **aprende** de la **experiencia E** en relación a una **tarea T**, utilizando una **medida de rendimiento P**, si mejora sus prestaciones, medidas mediante **P**, en la realización de la tarea **T** a través de la experiencia **E**.

Esto suena complicado, pero es simple. Vamos a desglosarlo:

Ejemplo: Programa que juega a las damas

Componente	Significado	En el ejemplo de damas
T (Tarea)	Lo que queremos que haga el programa	Jugar a las damas
E (Experiencia)	Los datos o situaciones con las que aprende	Jugar muchas partidas
P (Performance/Rendimiento)	Cómo medimos si lo hace bien	Probabilidad de ganar la próxima partida

En resumen: El programa **aprende a jugar mejor a las damas** (T) jugando muchas partidas (E), y sabemos que aprende porque cada vez gana más veces (P).

5.2 ¿Por qué es importante el Aprendizaje Automático?

1. **Hacer viables ciertas aplicaciones** que serían imposibles de programar manualmente
   - Ejemplo: Reconocimiento facial con millones de variaciones
2. **Construir una IA de propósito general**
   - En lugar de programar reglas específicas, el sistema aprende de forma general
3. **Avances tecnológicos actuales**:
   - Mayor potencia de cálculo
   - Mayor capacidad de almacenamiento
   - Disponibilidad masiva de datos
   - Mejores algoritmos

5.3 Estrategias de Aprendizaje

Imagina que estás enseñando a un niño a identificar frutas:

5.3.1 Aprendizaje Supervisado

Definición: Durante el entrenamiento, le dices al sistema **exactamente qué respuesta es correcta** para cada ejemplo.

Analogía: Como un profesor que corrige un examen mostrando la respuesta correcta.

Ejemplo:

```
Entrada: [Imagen de manzana] → Etiqueta: "Manzana"
Entrada: [Imagen de naranja] → Etiqueta: "Naranja"
Entrada: [Imagen de plátano] → Etiqueta: "Plátano"
```

El sistema aprende: "Cuando vea esta forma y color → es una manzana"

5.3.2 Aprendizaje No Supervisado

Definición: Le das datos al sistema **SIN etiquetas**, y él debe encontrar patrones por sí mismo.

Analogía: Como darle a un niño una caja de botones y pedirle que los agrupe como quiera.

Ejemplo:

```
Le das: [ ,  ,  ,  ,  ,  , ]

El sistema agrupa:
Grupo 1 (rojos, redondos):
Grupo 2 (naranjas, redondos):
Grupo 3 (amarillos, alargados):
```

Aplicaciones reales: - Segmentación de clientes (agrupar clientes similares) - Detección de anomalías - Compresión de datos

5.3.3 Aprendizaje por Refuerzo

Definición: El sistema recibe **señales de recompensa o castigo** según sus acciones, pero no se le dice explícitamente qué hacer.

Analogía: Como entrenar a un perro con premios cuando hace algo bien.

Ejemplo: Robot aprendiendo a caminar

```
Acción: Da un paso hacia adelante
Resultado: No se cae
Señal: +10 puntos  (recompensa)

Acción: Se inclina demasiado
Resultado: Se cae
Señal: -50 puntos  (castigo)
```

Aplicaciones reales: - Videojuegos (AlphaGo) - Robots industriales - Vehículos autónomos

5.4 Tipos de Problemas en Aprendizaje Supervisado

5.4 Problemas de Regresión

Objetivo: Predecir un **valor numérico continuo**.

Definición simple: Encontrar la función (curva/línea) que mejor se ajuste a los datos.

Ejemplo visual:

```
Precio de casas según tamaño

Precio (€)
300k                       (140m², 280k€)
250k
200k
150k
100k                     Aprendemos la línea
 50k                      que mejor ajusta
               > Tamaño (m²)
    50  70  90  110  130
```

Lo que aprende el algoritmo: La línea que mejor predice el precio según el tamaño.

Pregunta típica: “¿Cuánto costará una casa de 120m²?” → Respuesta: ~250,000€

Otros ejemplos: - Predecir temperatura mañana - Estimar ventas del próximo mes - Predecir edad de una persona por su foto

5.4.2 Problemas de Clasificación

Objetivo: Asignar datos a **categorías discretas** (clases).

Definición simple: Encontrar la frontera que separa diferentes grupos.

Ejemplo visual:

```
Clasificar tumores: benigno vs maligno

Tamaño




              Frontera
             /
          /
        /
                 > Edad

    = Benigno (Clase 0)
    = Maligno (Clase 1)
```

Pregunta típica: "¿Este tumor es benigno o maligno?" → Respuesta: Clase (benigno/maligno)

Otros ejemplos: - Email: spam/no spam - Imagen: gato/perro/pájaro - Transacción: fraude/legítima

5.4.3 Importancia de las Características

Aumentar el número de características **relevantes** mejora el aprendizaje.

Ejemplo de clasificación de tumores:

Con 1 característica (solo tamaño):

```
Difícil separar benignos de malignos
```

Con 2 características (tamaño + edad):

```
Mejor separación
```

Con 3 características (tamaño + edad + densidad):

```
Separación aún mejor
```

5.5 Regresión Lineal

La regresión lineal es el algoritmo más básico de aprendizaje supervisado.

5.5.1 Componentes del Sistema

Notación estándar:

Símbolo	Significado
m	Número de ejemplos de entrenamiento
n	Número de características (features)
x	Variables de entrada / características
y	Variable de salida / respuesta
(x, y)	Un par entrada-salida genérico
$\mathbf{(x^{(i)}, y^{(i)})}$	El i-ésimo par entrada-salida
$\mathbf{x^{(i)}_j}$	Valor de la característica j en el ejemplo i

5.5.2 Modelo de Aprendizaje

Flujo del proceso:

```
1. Conjunto de Entrenamiento
   ↓
2. Algoritmo de Aprendizaje
   ↓
3. Hipótesis h (función aprendida)
```

```
   ↓
4. Para nueva entrada x → h predice y
```

5.3 La Hipótesis h

Forma de la hipótesis (regresión lineal simple):

$$y = h_\theta(x) = \theta_0 + \theta_1 \cdot x$$

Componentes: - **θ_0** (theta cero): Intercepto (valor cuando x=0) - **θ_1** (theta uno): Pendiente (inclinación de la recta)

Ejemplo numérico:

$$h_\theta(x) = 50,000 + 2,000 \cdot x$$

```
Interpretación:
- Casa de 0 m²: 50,000€ (base)
- Por cada m² adicional: +2,000€
- Casa de 100 m²: 50,000 + 2,000×100 = 250,000€
```

5.5.4 Función de Coste J(θ)

Objetivo: Medir qué tan bien se ajusta nuestra línea a los datos.

Concepto: Queremos que nuestra predicción $h_\theta(x)$ esté lo más cerca posible del valor real y.

Error Cuadrático Medio (MSE):

$$J(\theta_0, \theta_1) = \frac{1}{2m} \times \Sigma(h_\theta(x^{(i)}) - y^{(i)})^2$$

Desglosando la fórmula: 1. $h_\theta(x^{(i)})$: Predicción para el ejemplo i 2. $y^{(i)}$: Valor real del ejemplo i 3. $(h_\theta(x^{(i)}) - y^{(i)})$: Error en el ejemplo i 4. $(...)^2$: Elevamos al cuadrado (penaliza errores grandes) 5. Σ: Sumamos errores de todos los ejemplos 6. $\frac{1}{2m}$: Promediamos (el ½ simplifica derivadas)

Ejemplo numérico:

```
Datos reales:
x = [50, 100, 150]
y = [150k, 250k, 350k]

Nuestra línea: h_ (x) = 50k + 2k·x

Predicciones:
h_ (50) = 150k  → Error = (150k - 150k)² = 0
h_ (100) = 250k → Error = (250k - 250k)² = 0
```

```
h_ (150) = 350k → Error = (350k - 350k)² = 0

J( ) = 0   ¡Ajuste perfecto!
```

5.5.5 Descenso de Gradiente

Objetivo: Encontrar los valores de θ que **minimizan J(θ)**. **Concepto visual**: Imagina que estás en una montaña y quieres bajar al valle (mínimo).

```
          ← Valor alto de J( )
       /  \
      /    \
     /      \
    /        \
   /         \
  /   ↓ bajas  \
 /      ↓       \
/________ ______\ ← Mínimo (mejor  )
```

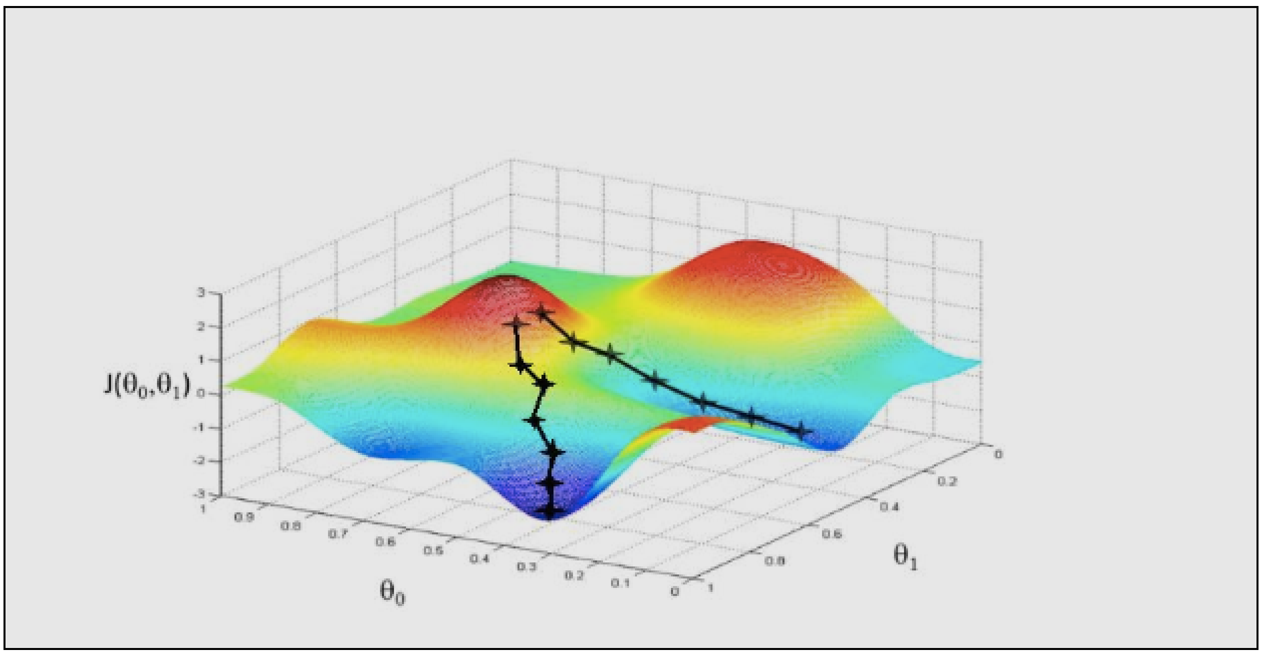

Figure 1: Pasted image 20251103114208.png

Algoritmo (repetir hasta convergencia):

$$\theta_j := \theta_j - \alpha \times \frac{\partial J}{\partial \theta_j}$$

Donde: - α (alpha): Tasa de aprendizaje (tamaño del paso) - $\frac{\partial J}{\partial \theta_j}$: Derivada parcial (dirección del descenso)

Forma específica para regresión lineal:

$$\theta_0 := \theta_0 - \alpha \times \frac{\partial}{\partial\theta_0} \times J(\theta_0, \theta_1)$$

$$\theta_1 := \theta_1 - \alpha \times \frac{\partial}{\partial\theta_1} \times J(\theta_0, \theta_1)$$

$$\frac{\partial}{\partial\theta_j} \times J(\theta_0, \theta_1) = J(\theta_0, \theta_1) = \frac{\partial}{\partial\theta_j} \times \frac{1}{2m} \times \Sigma(h_\theta(x^{(i)}) - y^{(i)})^2 = \frac{\partial}{\partial\theta_j} \times \frac{1}{2m} \times \Sigma(\theta_0 + \theta_1 x^{(i)} - y^{(i)})^2$$

$$j = 0 : \frac{\partial}{\partial\theta_0} \times J(\theta_0, \theta_1) = \frac{1}{m} \times \Sigma(h_\theta(x^{(i)}) - y^{(i)})$$

$$j = 1 : \frac{\partial}{\partial\theta_1} \times J(\theta_0, \theta_1) = \frac{1}{m} \times \Sigma(h_\theta(x^{(i)}) - y^{(i)}) \times x^{(i)}$$

5.5.6 Ejemplo Paso a Paso

Problema: Predecir precio de casas según tamaño.

Datos de entrenamiento (m=3):

```
x (m²)  | y (precio €)
--------|-------------
50      | 150,000
100     | 250,000
150     | 350,000
```

Paso 1: Inicializar parámetros

```
  = 0
  = 0
 = 0.01 (tasa de aprendizaje)
```

Paso 2: Primera predicción

```
h_ (x) = 0 + 0·x = 0 (para todos los x)
```

Paso 3: Calcular coste inicial

```
J( ) = (1/6) × [(0-150k)² + (0-250k)² + (0-350k)²]
     = (1/6) × [22,500M + 62,500M + 122,500M]
     = 34,583M ← ¡Muy alto!
```

Paso 4: Calcular gradientes y actualizar θ

```
 J/   = (1/3) × [(0-150k) + (0-250k) + (0-350k)]
        = -250,000

 J/   = (1/3) × [(0-150k)×50 + (0-250k)×100 + (0-350k)×150]
```

```
       = -30,000,000

   := 0 - 0.01 × (-250,000) = 2,500
   := 0 - 0.01 × (-30,000,000) = 300,000
```

Paso 5: Nueva predicción

```
h_ (x) = 2,500 + 300,000·x
```

Repetir pasos 3-4 hasta que J(θ) deje de disminuir significativamente.

5.5.7 Tasa de Aprendizaje α

Concepto: Controla el tamaño de los pasos al descender.

Valor de α	Efecto	Problema
Muy pequeño (0.001)	Pasos muy pequeños	Convergencia MUY lenta □
Adecuado (0.01-0.1)	Pasos balanceados	Convergencia óptima □
Muy grande (10)	Pasos muy grandes	Puede no converger (oscila) □

Visualización del efecto de α:

```
 muy pequeño:
J( )



          ← Baja muy lentamente

       > Iteraciones

 muy grande:
J( )



          ← Oscila, no converge

       > Iteraciones

 adecuado:
J( )



          ← Converge rápidamente
          ---
       > Iteraciones
```

5.6 Regresión Lineal Multivariable

Hasta ahora: 1 característica (tamaño de casa) Ahora: Múltiples características (tamaño, habitaciones, edad, etc.)

5.6.1 Notación Extendida

Nueva forma de h:

```
h_ (x) =   +  x  +  x  + ... +  x

En forma vectorial:
h_ (x) =   × x

Donde:
  = [ ,  ,  , ...,  ]
x = [1, x , x , ..., x ]   (x  = 1 por convención)
```

5.6.2 Ejemplo con 3 características

Predicción de precio de casa:

```
Características:
x  = Tamaño (m²)
x  = Número de habitaciones
x  = Edad (años)

h_ (x) =   +  ·tamaño +  ·habitaciones +  ·edad

Ejemplo con valores aprendidos:
h_ (x) = 80,000 + 2,000·tamaño + 10,000·habitaciones - 1,000·edad

Para una casa de 100m², 3 habitaciones, 5 años:
h_ (x) = 80,000 + 2,000×100 + 10,000×3 - 1,000×5
       = 80,000 + 200,000 + 30,000 - 5,000
       = 305,000€
```

6.3 Normalización de Características

Si las características tienen rangos muy diferentes, el descenso de gradiente converge lentamente.

Ejemplo:

```
x  = Tamaño: rango [50 - 200]
x  = Habitaciones: rango [1 - 5]
```

Solución: Feature Scaling

```
x_j^(i) := (x_j^(i) -  ) / s
```

```
Donde:
  = media de la característica j
s  = desviación estándar de la característica j
```

Ejemplo numérico:

```
Original x  (tamaño): [50, 100, 150, 200]
  = (50+100+150+200)/4 = 125
s  = desviación = 55.9

Normalizado:
x  = [50-125]/55.9 = -1.34
x  = [100-125]/55.9 = -0.45
x  = [150-125]/55.9 = 0.45
x  = [200-125]/55.9 = 1.34

Ahora todos están aproximadamente en el rango [-2, 2]
```

5.6.4 Ecuación Normal (Método Analítico)

Alternativa al descenso de gradiente: Calcular θ directamente con álgebra lineal.

Fórmula:

```
 = (X X) ¹ X y

Donde:
X = matriz de características (m × n+1)
y = vector de salidas (m × 1)
```

Comparación con Descenso de Gradiente:

Descenso de Gradiente	Ecuación Normal
Necesita elegir α	No necesita α □
Necesita muchas iteraciones	Sin iteraciones □
Funciona bien con n grande (millones) □	Lento con n > 10,000 □
Necesita normalizar features	No necesita normalizar □
Complejidad $O(kn^2)$	Complejidad $O(n^3)$

5.7 Regresión Logística

Objetivo: Resolver problemas de **clasificación** (no regresión, a pesar del nombre).

5.7.1 ¿Por qué no usar regresión lineal para clasificación?

Problema visual:

```
Clasificar tumor: 0 (benigno) o 1 (maligno)

y
1                   ← Malignos
                      ← Benignos
0
           > Tamaño

Si usamos regresión lineal:
1

0
           >

Problemas:
- h(x) puede ser > 1 o < 0
- Un tumor MUY grande altera toda la línea
```

5.7.2 Función Sigmoide (Logística)

Solución: Usar una función que siempre devuelva valores entre 0 y 1.

Fórmula:

```
g(z) = 1 / (1 + e )

h_ (x) = g( x) = 1 / (1 + e^(- x))
```

Gráfica:

```
g(z)
  1
          /
 0.5      / ← Umbral
        /
  0
          > z
  -∞   0   +∞

z >> 0 → g(z)   1
z = 0  → g(z) = 0.5
z << 0 → g(z)   0
```

Interpretación:

```
h_ (x) = 0.7

Significa: "Hay un 70% de probabilidad de que y=1"
```

5.7.3 Frontera de Decisión

Regla de decisión:

```
Si h_ (x)   0.5 → predecir y=1
Si h_ (x) < 0.5 → predecir y=0

Como g(z)   0.5 cuando z   0:
Si   x   0 → predecir y=1
Si   x < 0 → predecir y=0
```

Ejemplo 1: Frontera lineal

```
h_ (x) = g(-3 + x  + x )

Predecir y=1 cuando: -3 + x  + x    0
Es decir: x  + x    3

Gráfica:
x
         y=1 (  )
 3      /
       / ← Frontera: x  + x  = 3
      /
     / y=0 (  )
         > x
    3
```

Ejemplo 2: Frontera circular

```
h_ (x) = g(-1 + x ² + x ²)

Predecir y=1 cuando: x ² + x ²   1

Gráfica:
x



        0      ← Círculo de radio 1



         > x

Interior ( ): y=0
Exterior ( ): y=1
```

5.7.4 Función de Coste para Regresión Logística

Problema: El error cuadrático hace que J(θ) sea no convexa (múltiples mínimos).

Solución: Nueva función de coste:

```
Cost(h_ (x), y) = -log(h_ (x))        si y=1
Cost(h_ (x), y) = -log(1 - h_ (x))   si y=0

Forma compacta:
Cost(h_ (x), y) = -y·log(h_ (x)) - (1-y)·log(1-h_ (x))

Función de coste total:
J( ) = -(1/m) Σ [y  ·log(h_ (x  )) + (1-y  )·log(1-h_ (x  ))]
```

Intuición:

```
Si y=1:
- h_ (x) = 1 → Cost = -log(1) = 0   Sin penalización
- h_ (x) = 0.5 → Cost = -log(0.5) = 0.69
- h_ (x) = 0.1 → Cost = -log(0.1) = 2.3   Penalización alta

Si y=0:
- h_ (x) = 0 → Cost = -log(1) = 0   Sin penalización
- h_ (x) = 0.5 → Cost = -log(0.5) = 0.69
- h_ (x) = 0.9 → Cost = -log(0.1) = 2.3   Penalización alta
```

5.7.5 Algoritmo de Descenso de Gradiente

Actualización (¡idéntica en forma a regresión lineal!):

```
  :=   -   × (1/m) × Σ[h_ (x  ) - y  ] × x

Pero recuerda:
h_ (x) = 1/(1 + e^(- x))  ← Diferente de regresión lineal
```

7.6 Clasificación Multiclase

Estrategia: One-vs-All (uno contra todos)

Proceso:

```
Problema: Clasificar emails en 3 categorías
- Clase 1: Personal
- Clase 2: Trabajo
- Clase 3: Spam

Entrenamos 3 clasificadores:

Clasificador 1: ¿Es Personal? (sí vs no)
Clasificador 2: ¿Es Trabajo? (sí vs no)
Clasificador 3: ¿Es Spam? (sí vs no)
```

```
Para un nuevo email:
h (x) = 0.2 (20% probabilidad Personal)
h (x) = 0.7 (70% probabilidad Trabajo) ← ¡Máximo!
h (x) = 0.1 (10% probabilidad Spam)

Predicción final: Trabajo
```

5.8 Regresión Polinómica

Objetivo: Ajustar curvas (no solo líneas rectas).

Idea: Crear nuevas características elevando las originales a potencias.

```
Original:
h_ (x) =   +  x

Cuadrática:
h_ (x) =   +  x +  x²

Cúbica:
h_ (x) =   +  x +  x² +  x³
```

Ejemplo visual:

```
Precio de casa según tamaño

Lineal:        Cuadrática:       Cúbica:






     >            >                  >

Simple         Mejor ajuste        Puede sobreajustar
```

5.9 Overfitting (Sobreajuste)

5.9.1 Los Tres Escenarios

Ejemplo con regresión:

```
UNDERFITTING         GOOD FIT        OVERFITTING
(Subajuste)         (Buen ajuste)     (Sobreajuste)
```

```
h(x)= + x      h(x)= + x+ x²   h(x)= +...+ x

Sesgo alto          Balance          Varianza alta
No captura          Captura            Captura el ruido
el patrón           el patrón          también
```

Ejemplo con clasificación:

```
UNDERFITTING         GOOD FIT         OVERFITTING




Línea recta         Curva suave       Frontera errática
muy simple          apropiada        se ajusta al ruido
```

5.9.2 Soluciones al Overfitting

Solución 1: Reducir número de características

```
Manualmente:
- Eliminar características poco relevantes

Automáticamente:
- Algoritmos de selección de características
```

Solución 2: Regularización

Concepto: Penalizar parámetros θ muy grandes.

Nueva función de coste:

```
J( ) = [coste original] +  /(2m) × Σ  ²
                              Término de regularización

Donde:
 (lambda) = parámetro de regularización
```

Efecto:

```
  = 0:        Sin regularización → Posible overfitting
  pequeño:    Poca regularización → Balance
  grande:     Mucha regularización → Posible underfitting

Ejemplo:
Si   es muy grande, forzamos todos los     0
Resultado: h(x)     (función constante) → Underfitting
```

Regresión lineal regularizada:

```
   :=   -   × [(1/m)Σ(h_ (x  )-y  )x   + ( /m) ]
                 término original      regularización
```

5.10 Flujo Completo de un Proyecto de ML

```
1. DEFINIR EL PROBLEMA
   ¿Regresión o Clasificación?
        ↓
2. RECOPILAR DATOS
   Conjunto de entrenamiento (x, y)
        ↓
3. PREPROCESAR DATOS
   - Limpiar datos
   - Normalizar características
   - Dividir en entrenamiento/test
        ↓
4. ELEGIR MODELO
   - Regresión lineal
   - Regresión logística
   - Otro algoritmo
        ↓
5. ENTRENAR MODELO
   - Inicializar
   - Minimizar J( ) con descenso de gradiente
        ↓
6. EVALUAR MODELO
   - Probar en datos de test
   - Verificar overfitting/underfitting
        ↓
7. AJUSTAR Y MEJORAR
   - Cambiar
   - Añadir/quitar características
   - Aplicar regularización
        ↓
8. DESPLEGAR
   Usar el modelo en producción
```