Machine Learning I

Francisco Javier Mercader Martínez

Índice

Tema 1: Introducción al Machine Learning

1.1) Tareas básicas del Machine Learning

¿Qué es el Machine Learning?

- Definición de Machine Learning

"Descubrir regularidades en datos mediante el uso de algoritmos, y mediante el uso de esas regularidades realizar alguna acción" (C. M. Bishop)

- Tareas básicas

Fundamentalmente cuatro:

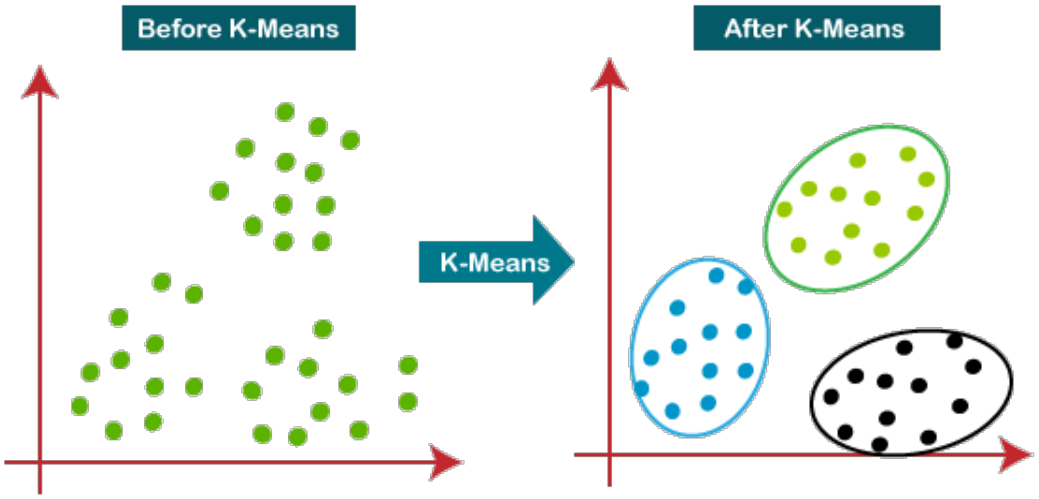
- Clasificación
  - → **Detección de spam:** Se trata de clasificar, mediante identificación de patrones, los correos electrónicos como spam o no spam.
  - → **Detección de fraudes:** Distinción entre transacciones legítimas y sospechosas basándose en patrones y características relevantes.
  - → **Análisis de sentimientos:** Los algoritmos de clasificación pueden utilizarse para determinar el sentimiento expresado en un texto, como positivo, negativo o neutro. Esto es útil para el análisis de opiniones en redes sociales, comentarios de clientes, revisiones de productos, etc.
  - → **Detección de objetos en imágenes:** Especialmente útil en la conducción de coches autónomos.
- Regresión
  - → **Estimación de la demanda de un producto:** Predicción de la demanda de un producto en función de variables como el precio, la publicidad, las tendencias del mercado, entre otras.
  - → **Predicción de la contaminación atmosférica:** Utilizando datos históricos de contaminantes, meteorología y otras variables relevantes, se puede aplicar la regresión para predecir los niveles de contaminación en una ubicación específica.
  - → **Análisis de la relación entre variables económicas:** La regresión puede utilizarse para explorar la relación entre variables económicas, como el crecimiento del PIB y el desempleo, con el fin de entender mejor su interdependencia y tomar decisiones políticas o empresariales informadas.
- Agrupamiento
- Asociación

- Tarea de agrupación en Machine Learning

El **agrupamiento** o **clustering** consiste en detectar agrupaciones en datos no etiquetados empleando alguna media de similitud entre ellas. El objetivo es descubrir patrones y estructuras dentro de los datos.

Algoritmos populares para clustering incluyen el K-Means, el DSCAN, el clustering jerárquico y Mapas Autoorganizados (SOM).

Ejemplo K-Means



- Tarea de asociación en Machine Learning

La tarea de **asociación** se centra en descubrir reglas de asociación entre eventos en un conjunto de datos, lo que significa identificar qué elementos tienden a aparecer juntos en dichos eventos. El objetivo es revelar después del afeitado, hay un 80% de posibilidades de que el cliente compre también crema de afeitado.

La asociación es una tarea no supervisada, los datos a menudo provienen de transacciones o eventos, y no se requieren etiquetas previas.

Algoritmos como Apriori se utilizan comúnmente para generar reglas de asociación en los datos, reglas como "Si A, entonces B". Estas reglas se utilizan en análisis de mercado y sistemas de recomendación.

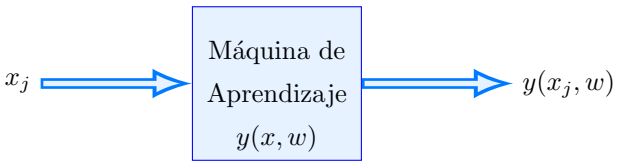
1.2) Generalización: subajuste y sobreajuste

1.2.1) Planteamiento del problema

En el contexto del Machine Learning, el **conjunto de hipótesis** se refiere a un conjunto de funciones o modelos matemáticos que se utilzian para aproximar una relación desconocida entre las **entradas** (x) y las **salidas deseadas o targets** (t) de un conjunto de datos.

Cada hipótesis representa una posible aproximación de la relación subyacente en los datos.

El objetivo del **aprendizaje supervisado** es encontrar la hipótesis que mejor se ajuste a los datos de entrenamiento manteniendo la capacidad de hacer predicciones precisas para datos nuevos **(capacidad de generalización)**.



- Necesario: Conjunto de entrenamiento

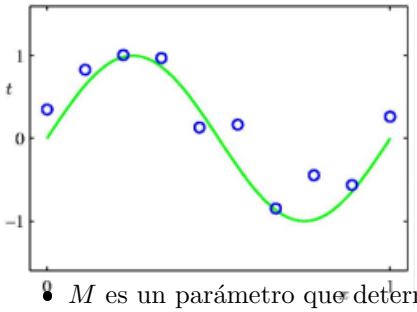
  Pares: $\{x_j, t_j\}$ con $j = 1, 2, \ldots, N$.
  $x_j = \{x_{j1}, x_{j2}, \ldots, x_{jD}\}$ entrada j-ésima; vector con D **componentes o características**.
  $t_j = \{x_{j1}, x_{j2}, \ldots, x_{jT}\}$ target j-ésimo; vector con T componentes.

- Objetivo: Aprendizaje supervisado

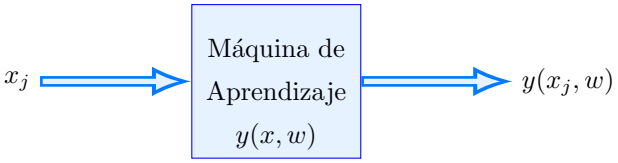
  Encontrar las **variables o pesos** del modelo $(\mathbf{w})$ que resuelvan el problema: $y(x_j, \mathbf{w}) \approx t_j$ para $j = 1, 2, \ldots, N$. A esta tarea se la denomina **entrenamiento**.

Ejemplo: Problema de **regresión**

$y = \sin(2\pi x) + n(x)$, donde $n(x)$ es un ruido gausiano pequeño.



Conjunto de entrenamiento: $\{x_j, t_j\}_{j=1}^{N=10}$



Aproximador polinómico: $y(x, \mathbf{w}) = w_0 + w_1 x + w_2 x^2 + \cdots + x_M x^M = \sum_{j=0}^{M} x_j x^j$

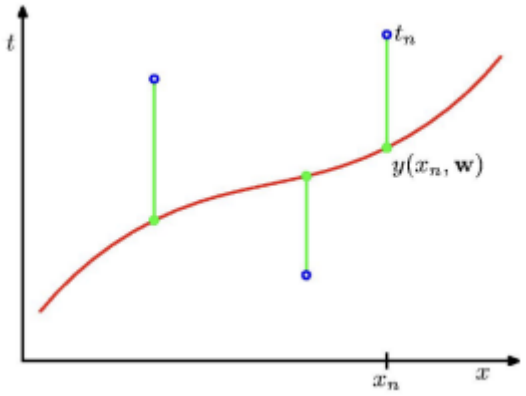
- M es un parámetro que determina la complejidad del modelo (orden del polinomio).
- Los parámetros no entrenables que determinan el modelo o el entrenamiento se denominan en Machine Learning **hiperparámetros**.

1.2.2) Planteamiento de la solución

Se quiere encontrar las variables del modelos (coeficientes del polinomio) para que éste minimice una función de coste o error, por ejemplo, la función de error SSE ("Sum of Square Error") dada por

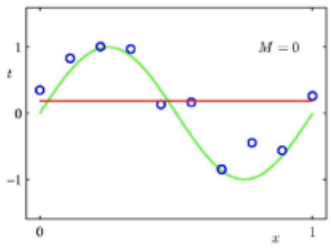
$$E(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^{N} \{y(x_n, \mathbf{w}) - t_n\}^2$$

Existe una solución analítica única mediante álgebra lineal.

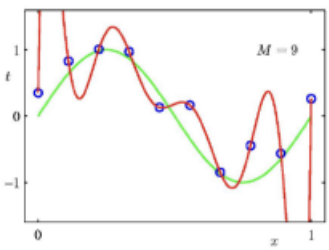


1.2.3) Peligros

1.2.4) Subajuste y sobreajuste



Ajuste pobre

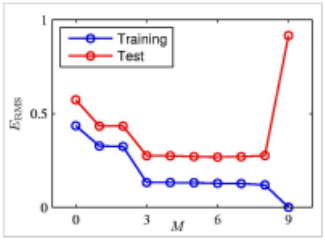


Sobreajuste ("overfitting")

1.2.5) Complejidad del modelo vs número de datos

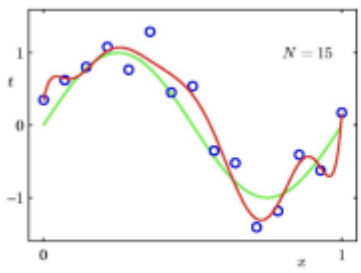
- Comoportamiento con M(N fijo)

Fijado N, la complejidad del modelo determina la generalización

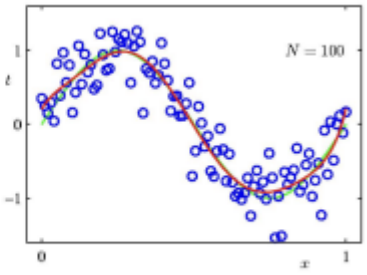


- Comportamiento con N (M fijo)

Fijado $N(M = 9)$, N condiciona la solución del problema: si es bajo, se puede sobreajustar, si es alto (con relación a la dimensión) se reduce el sobreajuste.



Sobreajuste



No sobreajuste

1.2.6) Conclusión

Hay que limitar la complejidad del modelo acorde con el número de datos disponibles.

En número de muestras (N) suele ser un parámetro fijo condicionado por el problema.

Hay que lidiar con el compromiso entre la complejidad y el error de generalización.

1.2.7) Descomposición sesgo-varianza. Coste cuadrático

- Solución óptima

En el contexto de machine learning y desde un punto de vista teórico, los datos de un problema se considera extraídos de una disposición $p(\mathbf{x}, t)$. Definamos:

- Salida del modelo regresor: $y(\mathbf{x})$, aporta la solución ($y(\mathbf{x}) \approx t$).
- Coste cuadrático para una entrada $\mathbf{x}$: $L = (y(\mathbf{x}) - t)^2$.
- Coste cuadrático promedio (MSE):

$$E[L] = \int_{\mathbf{x}} \int_t L(t, y(\mathbf{x}))p(\mathbf{x}, t)\, \mathrm{d}t\mathrm{d}\mathbf{x} \tag{1}$$

$$= \int_{\mathbf{x}} \int_t (y(\mathbf{x}) - t)^2 p(\mathbf{x}, t)\, \mathrm{d}t\mathrm{d}\mathbf{x} \tag{2}$$

El término cuadrático de la Ecuación (1), se puede escribir como

$$\begin{aligned}\{y(\mathbf{x}) - t\}^2 &= \{y(\mathbf{x}) - E_t[t|\mathbf{x}] + E_t[t|\mathbf{x}] - t\}^2 \\ &= \{y(\mathbf{x}) - E_t[t|\mathbf{x}]\}^2 + \{E_t[t|\mathbf{x}] - t\}^2 + 2\{y(\mathbf{x}) - E_t[t|\mathbf{x}]\}\{E_t[t|\mathbf{x}] - t\}\end{aligned}$$

que insertado en (1) produce

$$E[L] = \int \mathbf{x} \int_t \{y(\mathbf{x}) - E_t[t|\mathbf{x}]\}^2 p(\mathbf{x}, t)\, \mathrm{d}t\mathrm{d}\mathbf{x} + \int_{\mathbf{x}} \int_t \{E_t[t|\mathbf{x}] - t\}^2 P(\mathbf{x}, t)\, \mathrm{d}t\mathrm{d}\mathbf{x} + \int_{\mathbf{x}} \int_t 2\{y(x) - E_t[t|\mathbf{x}]\}\{E_t[t|\mathbf{x}] - t\}p(\mathbf{x}, t)\, \mathrm{d}t\mathrm{d}\mathbf{x}$$

y realizando la integral sobre t, se obtiene

$$E[L] = \int_{\mathbf{x}} \{y(\mathbf{x}) - E_t[t|\mathbf{x}]\}^2 p(\mathbf{x}) \mathrm{d}\mathbf{x} + \int_{\mathbf{x}} \mathrm{var}(t|\mathrm{x}) + 0 \tag{3}$$

- El primer término queda así debido a que el integrando no depende de t.
- El segundo término representa la variabilidad intrínseca del target t promediada sobre t y $\mathbf{x}$, y el valor mínimo posible del coste esperado ($E[L]$). Se considera, por tanto, un ruido irreducible del problema.
- El tercer término se anula ya que al realizar la integral sobre t se tiene

$$\begin{aligned}\int_t 2\{y(\mathbf{x}) - E_t[t|\mathbf{x}]\}\{E_t[t|\mathbf{x}] - t\} p(\mathbf{x}, t)\, \mathrm{d}t &= 2\{y(\mathbf{x}) - E_t[t|\mathbf{x}]\} \int_t \{E_t[t|\mathbf{x}] - t\} p(\mathbf{x}, t)\, \mathrm{d}t \\ &= E_t[t|\mathbf{x}] - E_t[t|\mathbf{x}] \\ &= 0\end{aligned}$$

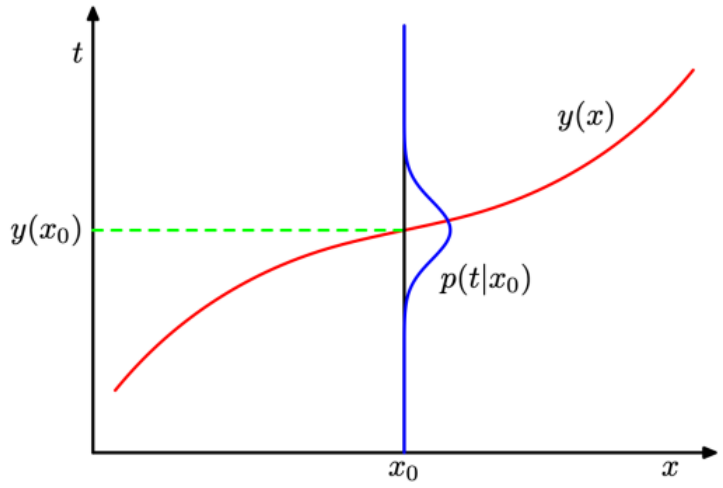
Como el segundo término de (3) no depende de nuestro regresor $y(\mathbf{x})$, la solución óptima que minimiza $E[L]$, y que llamaremos $h(\mathbf{x})$, es:

$$\boxed{h(\mathbf{x}) = E_t[t|\mathbf{x}]}$$

que anula el primer término de (3).

– Representación gráfica de la solución óptima (mínimo MSE). Ejemplo unidimensional.

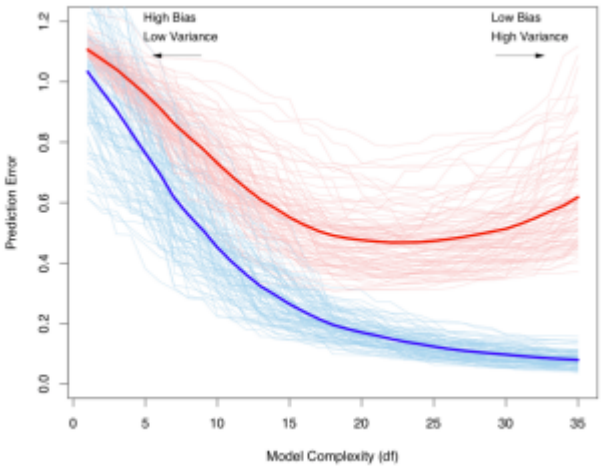
– Como acabamos de ver, la solución es $y(x) = E_t[t|x]$, es decir, $h(x)$.



En la práctica, **nunca tendremos infinitas muestras**, sino un conjuto finito D de N datos: $D = \{x_j, t_h\}_{j=1}^{N}$. Por esto, no podemos conocer $h(\mathbf{x})$ con exactitud.

Si modelamos $h(\mathbf{x})$ con una función paramétrica $y(\mathbf{x}, \mathbf{w})$ gobernada por el vector $\mathbf{w}$, entonces la incertidumbre del modelo se puede tratar de dos formas:

1) Considerando un único conjunto de datos D, de forma que la incertidumbre se expresa tratando $\mathbf{w}$ como una variable aleatoria (con una distribución a posteriori de $\mathbf{w}$, teoría bayesiana).
2) Considerando que se dispone de un gran número de conjuntos de datos diferentes, cada uno con N muestras extraídas independientemente de $p(\mathbf{x}, t)$, de forma que para cada uno de ellos, se obtiene –mediante algún algoritmo de entrenamiento– un predictor $y(\mathbf{x}, D)$ definido por un único vector $\mathbf{w}$ (estimación única para cada D).



Siguiendo la segunda opción, para cada conjunto D y muestra $\mathbf{x}$, se obtiene un error dado por:

$$\{y(\mathbf{x}, \mathbf{D}) - \mathbf{h}(\mathbf{x})\}^2$$

Introduciendo el promedio sobre D de nuestro regresor para $\mathbf{x}$, podemos re-escribir el error como

$$\{y(\mathbf{x}, D) - E_D[y(\mathbf{x}, D)] - h(\mathbf{x})\}^2 = \{y(\mathbf{x}, D) - E_D[y(\mathbf{x}, D)]\}^2 + \{E_D[y(\mathbf{x}, D)] - h(\mathbf{x})\}^2 + 2\{y(\mathbf{x}, D) - E_D[y(\mathbf{x}, D)]\}\{E_D[y(\mathbf{x}, D)] - h(\mathbf{x})\}$$

Promediando con respecto a D, se anula el tercer término y se obtiene el **error esperado para la muestra** x:

$$\boxed{E_D[\{y(\mathbf{x}, D) - h(\mathbf{x})\}^2] = \underbrace{E_D[\{y(\mathbf{x}, D) - E_D[y(\mathbf{x}, D)]\}^2]}_{\text{varianza}} + \underbrace{\{E_D[y(\mathbf{x}, D)] - h(\mathbf{x})\}^2}_{(\text{sesgo})^2}} \tag{4}$$

- Error esperado total: sesgo, varianza y ruido

Hasta ahora tenemos considerado el error producido por una única muestra $\mathbf{x}$. Incluyendo (4) en el primer término de (3), se obtiene el **error global esperado**

$$\boxed{\text{Error esperado} = \text{varianza} + (\text{sesgo})^2 + \text{ruido}}$$

donde

$$\text{varianza} = \int_{\mathbf{x}} E_D[\{y(\mathbf{x}, D) - E_D[y(\mathbf{x}, D)]\}^2] p(\mathbf{x})\, \mathrm{d}\mathbf{x}$$

es la diferencia cuadrática entre las predicciones de nuestro modelo y la media de dichas predicciones;

$$(\text{sesgo})^2 = \int_{\mathbf{x}} \{E_D[y(\mathbf{x}, D)] - h(\mathbf{x})\}^2 p(\mathbf{x})\, \mathrm{d}\mathbf{x}$$

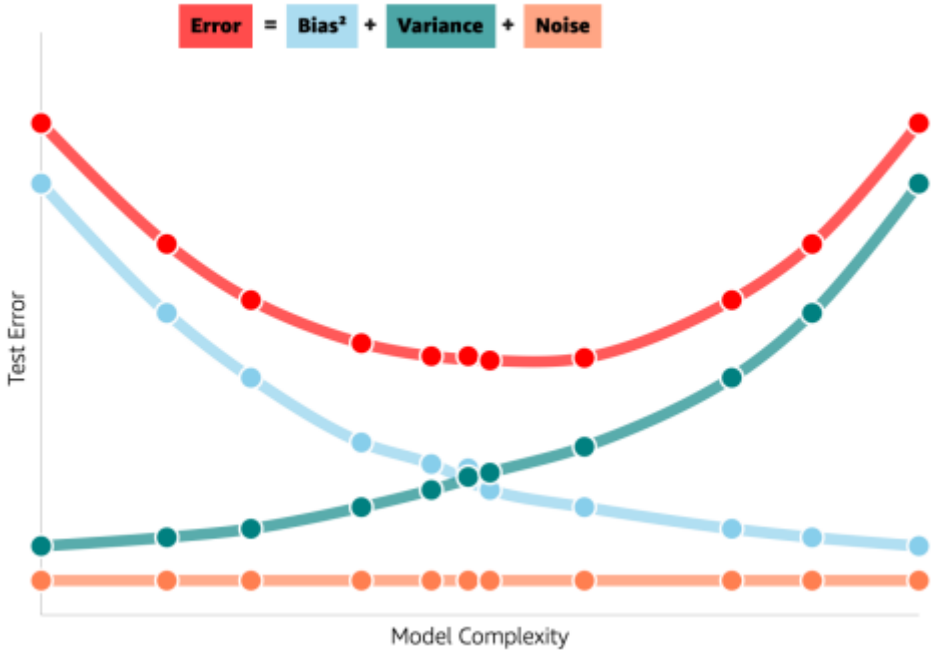
es la diferencia entre la predicción esperada de nuestro modelo y los valores verdaderos; y

$$\text{ruido} = \int_{\mathbf{x}} \int_{t} \{h(\mathbf{x}) - y\}^2 p(\mathbf{x}, t)\, \mathrm{d}t\, \mathrm{d}\mathbf{x}$$

es el error irreductible que siempre está presente. Es el segundo término de (3).

Representaciones gráficas

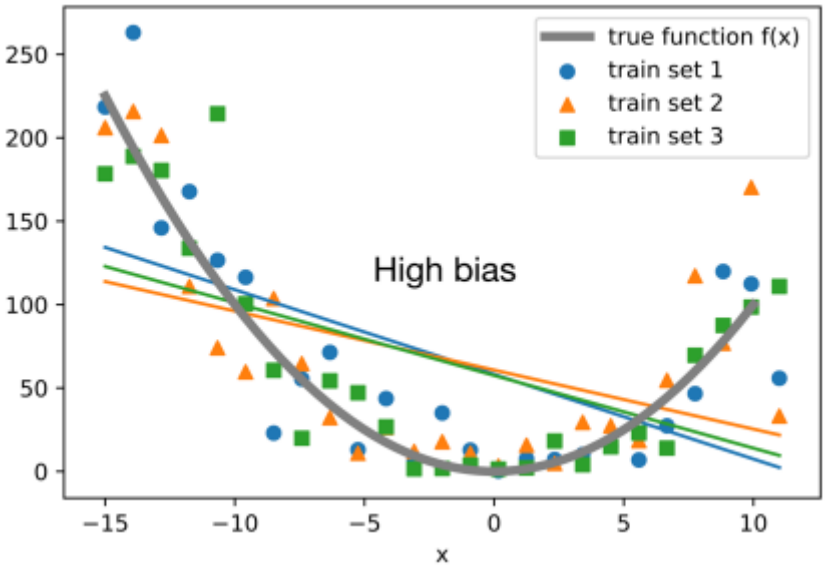
- Error esperado vs Complejidad



- Modelos complejos (alta capacidad, flexibles) producen varianzas elevadas y sesgos bajos
- Modelos simples (baja capacidad, rígidos) producen varianzas bajas y sesgos elevados

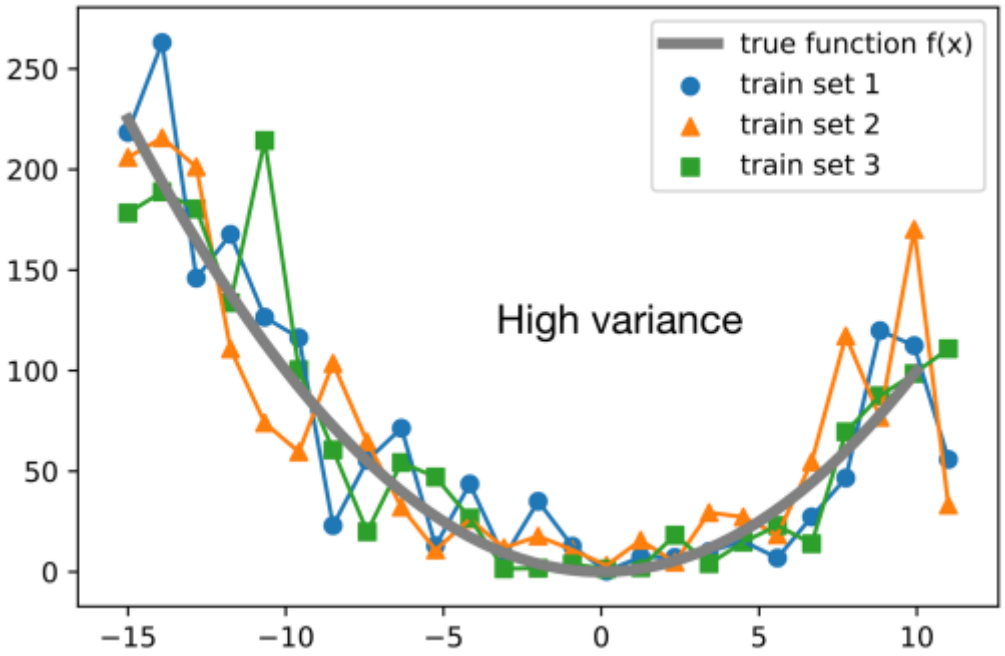
- Sesgo

Sea $f(\mathbf{x})$ una función desconocida que queremos aproximar, la llamaremos *"true function"*. Supongamos que tenemos diferentes conjuntos de entrenamiento extraídos de función de distribución definida como "$f(\mathbf{x})$ + noise". La siguiente figura muestra tres regresores lineales, uno para cada conjunto de entrenamiento.

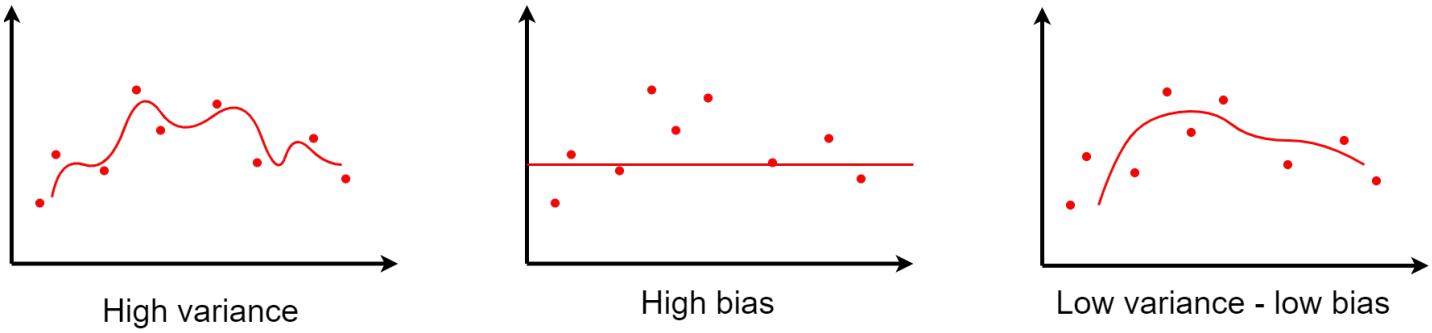


- Varianza

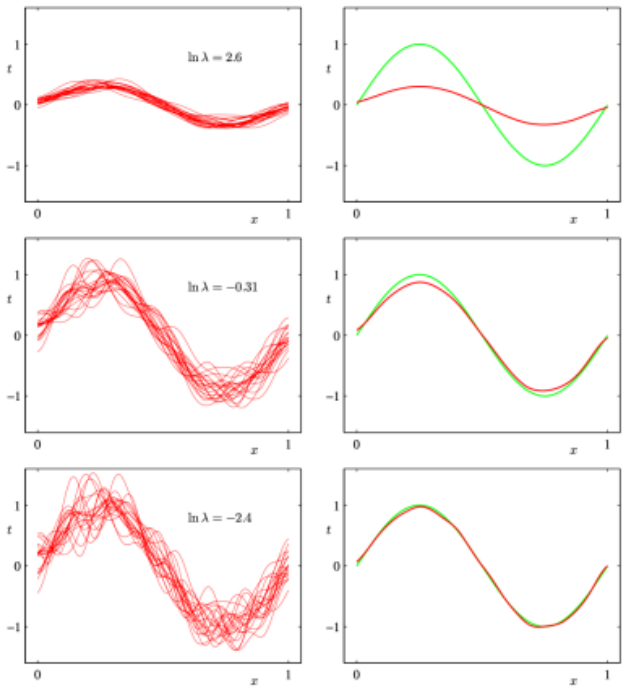
En este caso, las regresiones se realizan perfectamente mediante árboles de decisión sin podar. Cada uno, se ajusta perfectamente a los datos.



- Compromiso sesgo-varianza



- Mismo ejemplo de regresión considerado anteriormente.
- 100 conjuntos de datos, cada uno con 25 muestras. Se realizan 100 entrenamientos.
- Regresores: mezclas de 24 guasianas.
- λ parámetro que determina la complejidad del modelo. Aumenta hacia abajo.
- En la primer columna, se muestran 20 entrenamientos (modelos).
- Resultados:
  - Primera fila: complejidad baja, varianza pequeña y sesgo grande.
  - Última fila: complejidad alta, varianza grande y sesgo bajo.
  - Fila centro: solución intermedia. Mejor compromiso sesgo-varianza.



1.2.8) Algunas técnicas de generalización

¿Cómo evitar el sobre-ajuste?

- **"Early stopping":** Se detiene el entrenamiento del modelo antes de que alcance la convergencia total en los datos de entrenamiento. Se emplea un conjunto de parada.
- **Regularización L1 y L2** (Regresión Ridge y Lasso): Se agregan términos de penalización (normas L1 ó L2 de los coeficientes del modelo) a la función de pérdida durante el entrenamiento. Son técnicas **'Weight decay'**.
- **Dropout:** Es una técnica específica para redes neuronales. Durante el entrenamiento aleatoriamente se desactivan (ponen a cero) ciertas neuronas en cada paso. Esto evita que el modelo dependa demasiado de neuronas específicas y promueve una mejor generalización.
- **Aumento de datos:** Se aumenta el tamaño del conjunto de datos mediante técnicas como la rotación, la inversión y el recorte de imágenes. Esto ayuda a exponer al modelo a una mayor variedad de datos y reduce el riesgo de sobreajuste.
- **Feature Selection:** La selección adecuada de características es esencial para evitar el sobreajuste. Eliminar características irrelevantes o altamente correlacionadas puede reducir la complejidad del modelo y mejorar su generalización.
- **Ensemble learning:** Combina múltiples modelos más simples y menos propensos al sobreajuste. El Bagging (Bootstrap Aggregating) y el Boosting son ejemplos de ensemble learning.

¿Cómo evitar el sub-ajuste?

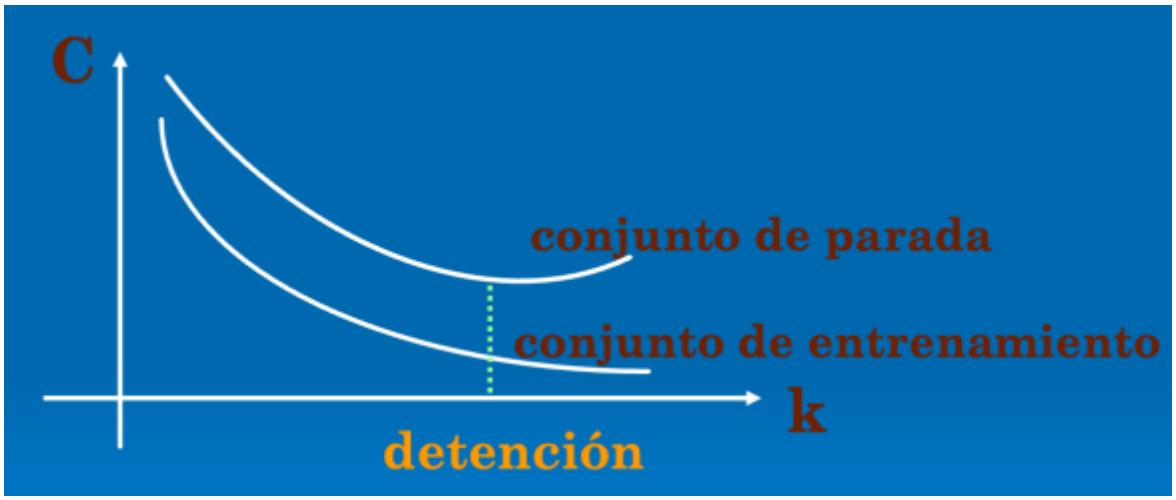
Evitar el sub-ajuste es tan importante como evitar el sobre-ajuste.

- Aumentar la complejidad del modelo.
- Aumentando el tiempo de entrenamiento.
- Añadiendo más características.
- Reducir la regularización.

1.2.9) Evitar el sobreajuste: "early stopping"

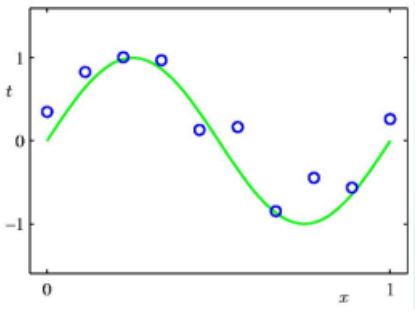
- **Sobreentrenamiento:** demasiados ciclos adaptan en exceso la red a las muestras de entrenamiento, no generalizando bien.

Para evitarlo (mantener la generalización), se emplea un conjunto adicional, extraído del conjunto de entrenamiento llamado **conjunto de validación** (en este caso, actúa como conjunto de parada)

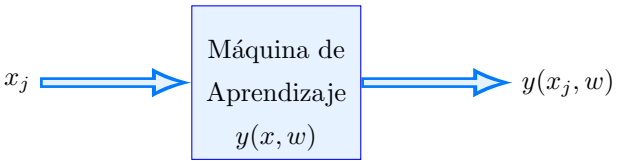


1.2.10) Evitar el sobreajuste: "Weight Decay"

Retomamos el problema de **regresión**: $y = \sin(2\pi \mathbf{x}) + n(\mathbf{x})$, donde $n(\mathbf{x})$ es un ruido gausiano pequeño.



Conjunto de entrenamiento: $\{x_j, t_j\}_{j=1}^{N=10}$



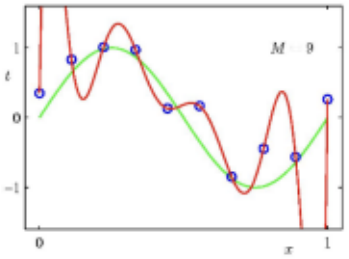
Aproximador polinómico: $y(x, \mathbf{w}) = w_0 + w_1 x + w_2 x^2 + \cdots + x_M x^M = \sum_{j=0}^{M} x_j x^j$

- Técnica de regularización

Valores de los pesos para varios M

	$M=0$	$M=1$	$M=6$	$M=9$
$w_0^\star$	0.19	0.82	0.31	0.35
$w_1^\star$		-1.27	7.99	232.37
$w_2^\star$			-25.43	-5321.83
$w_3^\star$			17.37	48568.31
$w_4^\star$				-231639.30
$w_5^\star$				640042.26
$w_6^\star$				-1061800.52
$w_7^\star$				1042400.18
$w_8^\star$				-557682.99
$w_9^\star$				125201.43

Los pesos aumentan con el orden del modelo para ajustarse al ruido

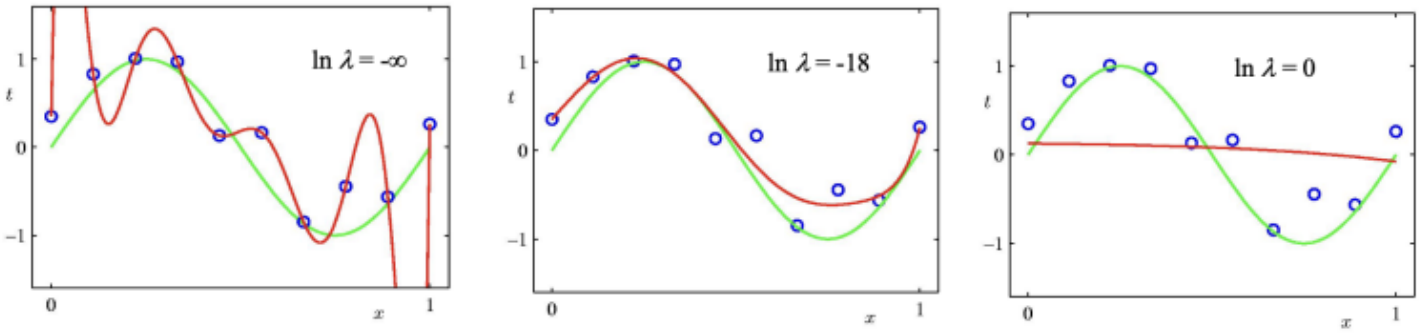


"Weight Decay": Se regulariza la solución mediante la minimización adicional (cuadrática) de los pesos

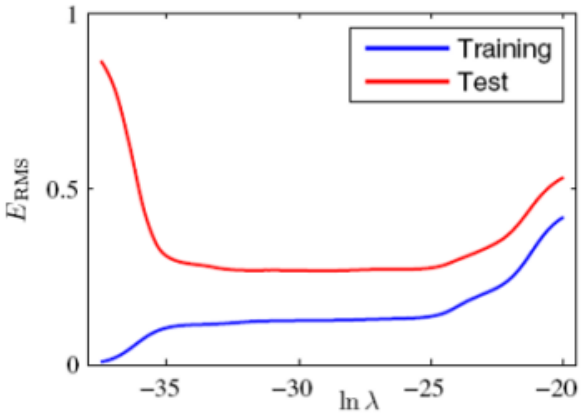
$$\tilde{E}(\mathbf{w}) = \frac{1}{2}\sum_{i=1}^{N}\{y(x_n, \mathbf{w}) - t_n\}^2 + \underbrace{\frac{\lambda}{2}\|\mathbf{w}\|^2}_{\text{Regularización}}$$

Regularizando para $M = 9$

	$\ln \lambda = -\infty$	$\ln \lambda = -18$	$\ln \lambda = 0$
$w_0^\star$	0.35	0.35	0.13
$w_1^\star$	232.37	4.74	-0.05
$w_2^\star$	-5321.83	-0.77	-0.06
$w_3^\star$	48568.31	-31.97	-0.05
$w_4^\star$	-231639.30	-3.89	-0.03
$w_5^\star$	640042.26	55.28	-0.02
$w_6^\star$	-1061800.52	41.32	-0.01
$w_7^\star$	1042400.18	-45.95	-0.00
$w_8^\star$	-557682.99	-91.53	0.00
$w_9^\star$	125201.43	72.68	0.01



En la figura, se observa como λ determina la generalización de la solución



Una forma sencilla de encontrar el valor óptimo de λ (la complejidad del modelo) es mediante la evaluación de las prestaciones del **Conjunto de Validación** (el mismo conjunto empleado para early-stopping).

1.3) Evaluación de prestaciones

Para evaluar las prestaciones (**rendimiento**) de un modelo de machine learning ya entrenado, se utilizan diversas métricas de evaluación dependiendo del tipo de problema (clasificación, regresión, etc.) y las características del conjunto de datos.

Métricas más comunes:

- Problemas de regresión.
  - Error cuadrático medio (Mean Square Error, MSE)
  - Error absoluto medio (Mean Absolute Error, MAE)
  - Coeficiente de Determinación R^2.
- Problemas de clasificación.
  - Matriz de confusión.
  - Exactitud (Accuracy)

- Precisión (Precision)
- Sensibilidad (Recall) y Especificidad (Specificty)
- F1-score
- ROC

1.3.1) Regresión

- **Error Cuadrático Medio (MSE):**

$$MSE(\mathbf{w}) = \frac{1}{N}\sum_{i=1}^{N}\{y(x_n, \mathbf{w}) - t_n\}^2$$

- **Error Absoluto Medio (MAE):**

$$MAE(\mathbf{x}) = \frac{1}{N}\sum_{i=1}^{N}|y(x_n, \mathbf{w}) - t_n|$$

En estas fórmulas, N el número total de instancias, $y(x_n, w)$ la salida obtenida del modelo w y t_n la salida real de la instancia n. Interpretación de MSE y MAE: A menor error mejor siempre será el modelo.

- **Coeficiente de Determinación R^2**

  Métrica a utilizar en tareas de regresión. Indica la cantidad proporcional de variación en la variable de respuesta y, explicada según las variables independientes X. Es una medida adimensional. Toma valores en el intervalo $[0, 1]$. Cuanto mayor es el valor, mejor es el ajuste del modelo.

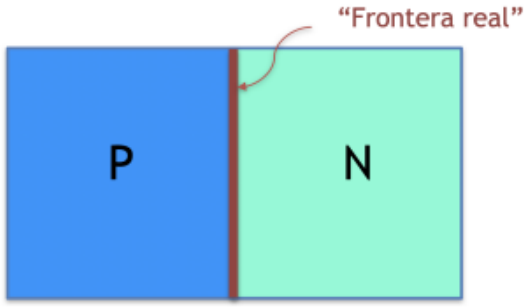
La fórmula es:

$$R^2 = 1 - \frac{\sum(t_i - y_i)^2}{\sum(y_i - \mu_t)^2}.$$

Siendo t_i la salida esperada de la instancia i, y_i la salida del modelo para la instancia i y μ_t la media de los valores de salida de todas las instancias. Interpretación de R^2.

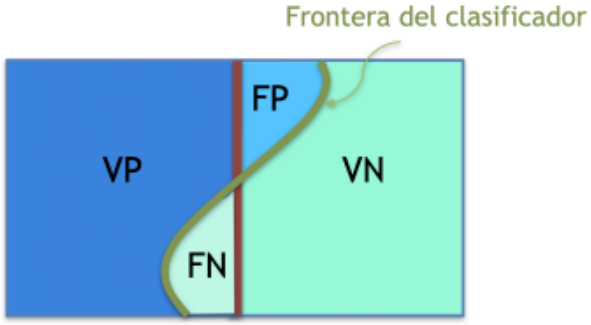
- Valor entre 0 y 1.
- Más cercano a 0 $\longrightarrow$ modelo con poco ajuste.
- Más cercano a 1 $\longrightarrow$ modelo con mayor ajuste.
- Umbral entre el nivel de ajuste se encuentra superior a 0.5, para algunas áreas superior a 0.75.

1.3.2) Clasificación



Supongamos un problema de clasificación binario con datos bidimensionales positivos (clase P) y negativos (clase N). En la figura anexa, se muestra el espacio de los datos y la frontera real (ideal) del problema.

Cuando se entrena un modelo con los datos disponibles, se obtendrá una frontera diferente a la ideal, como se muestra en la figura.



El objetivo es conseguir un clasificador que produzca una frontera lo más parecida a la real, es decir,

$$VP = P \longrightarrow FN = 0$$

y

$$VN = N \longrightarrow FP = 0$$

¿Cómo medir este parecido?

Con **métricas** como:

- Matriz de Confusión
- Exactitud (Accuracy)
- Precisión (Precision)
- Sensibilidad (Recall) y Especificidad (Specificty)
- F1-score
- ROC

- Matriz de confusión

Matriz de Confusión		Etiquetas reales		
		Positivo (P)	Negativo (N)	
Resultado del clasificador	Positivo (P)	VP	FP	P'
	Negativo (N)	FN	VN	N'
		P	N	

- Exactitud (Accuracy)

Es el porcentaje de acierto (con independencia de la clase)

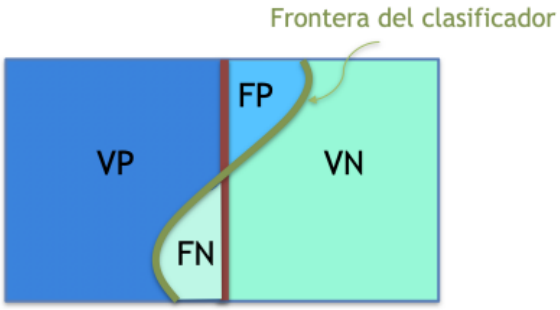
$$\text{Exactitud} = \frac{\text{VP} + \text{VN}}{\text{P} + \text{N}}$$

Desventaja: mala métrica para problemas desbalanceados

- Precisión (Precision)

Es el porcentaje de acierto de las predicciones positivas.

$$\text{Precisión} = \frac{\text{VP}}{\text{VP} + \text{FP}} = \frac{\text{VP}}{P'}$$



- Sensibilidad (Recall)

Es el porcentaje de casos positivos detectados

$$\text{Sensibilidad} = \frac{\text{VP}}{\text{VP} + \text{FN}} = \frac{\text{VP}}{\text{P}}$$

Ejemplo en diagnóstico médico: probabilidad de detectar a un enfermo.

También se denomina TPR (True Positive Ratio) o Probabilidad de detección (P_D).

- Especificidad (Specificity)

Es el porcentaje de casos negativos detectados.

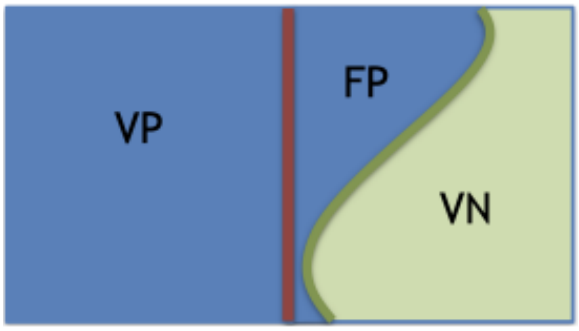
$$\text{Especificidad} = \frac{\text{VN}}{\text{VN} + \text{FP}} = \frac{\text{VN}}{\text{N}}$$

Ejemplo en diagnóstico médico: probabilidad de detectar a un sano.

También se denomina TNR (True Negative Ratio).

1.3.2.1) Interpretación de la Precisión y la Sensibilidad

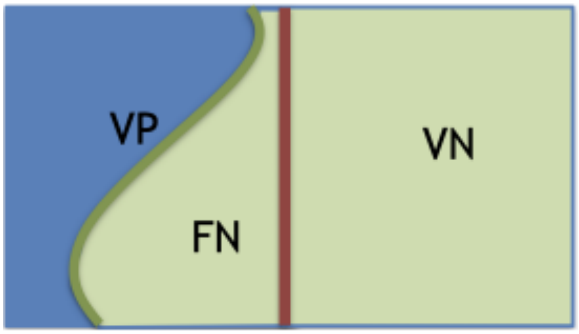
Conjuntamente indican si existe un sesgo hacia valores positivos o negativos



Precisión baja + Sensibilidad alta = Sesgo hacia la clase de negativos (frontera desplazada hacia la clase de negativos). Se puede tener un 100% de sensibilidad con una muestra negativa acertada.

Precisión alta + Sensibilidad baja = Sesgo hacia la clase de positivos (frontera desplaza hacia la clase de positivos). Se puede tener un 100% de precisión con una sola muestra positiva acertada.

Típico de conjuntos **desbalanceados**.



- **Puntuación F1** (F1 score): La puntuación F1 es la media armónica de la precisión y sensibilidad, donde la puntuación

de la F1 alcanza su mejor valor en 1 (precisión y sensibilidad perfectas) y el peor en 0.

- Esta es una métrica muy utilizada en problemas desbalanceados.

Se define como:

$$\text{F1} = 2 \times \frac{\text{sensibilidad} \times \text{precisión}}{\text{sensibilidad} + \text{precisión}}$$

- ROC (Receiver Operating Characteristic)

Es la representación gráfica de la **Sensibilidad** vs **Probabilidad de Falsa Alarma** para un sistema clasificador binario según se varía el umbral de discriminación (valor a partir del cual decidimos que un caso es un positivo).

- Probabilidad de Falsa Alarma, FPR

Es la probabilidad de detección incorrecta de un negativo, es decir, 1 − (la probabilidad de detección correcta de un negativo). Así

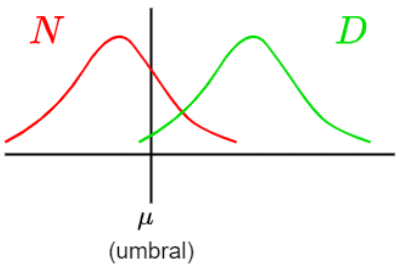
$$\text{Falsa alarma} = 1 - \text{Especificidad} = 1 - \frac{\text{VN}}{\text{N}} = \frac{\text{FP}}{N}$$

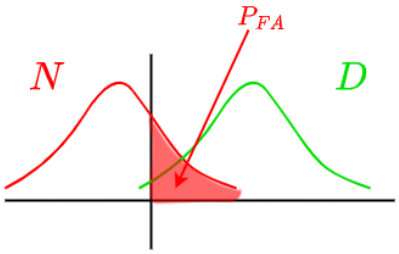
RECORDAR:

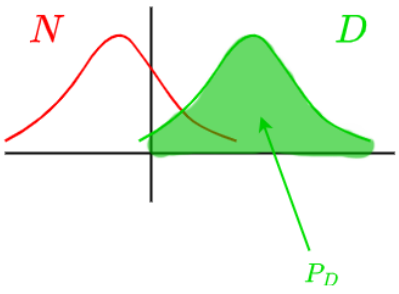
Sensibilidad = TPR ó P_D

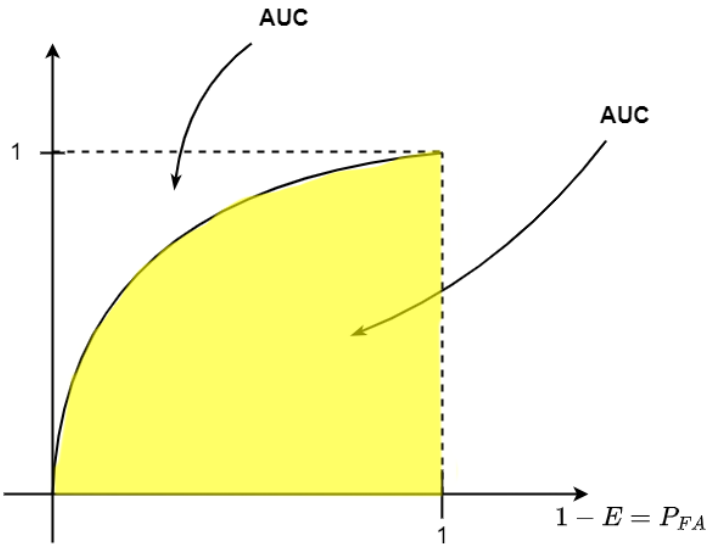
Falsa Alarma = FPR (False Positive Ratio) ó P_{FA}

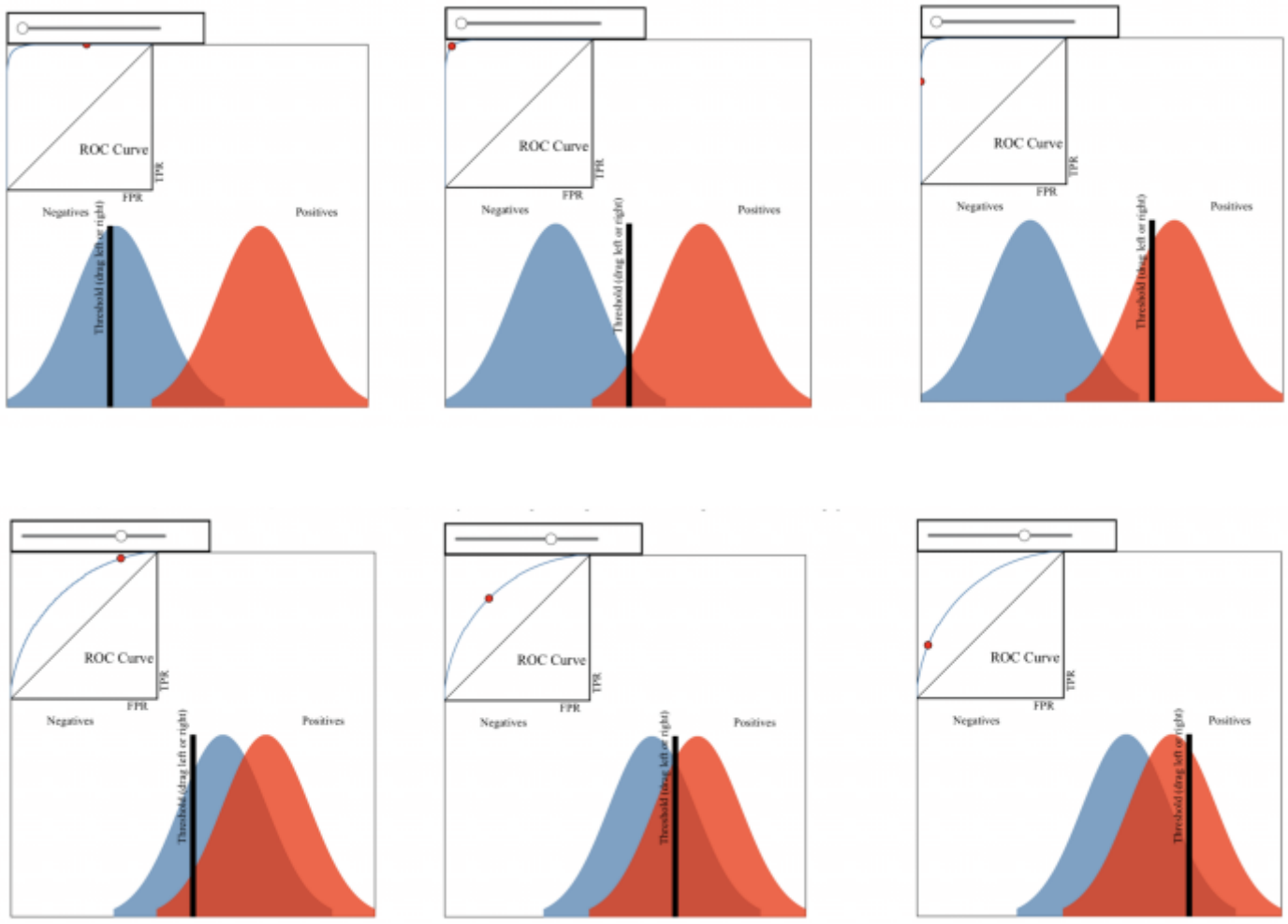
- AUC (Área Bajo la Curva)











1.3.2.2) Puntuación Kappa

La métrica de puntuación kappa k tiene sus orígenes se remontan al campo de la psicología: se utiliza para medir la concordancia entre dos evaluadores o clasificadores humanos a la hora de calificar sujetos. En el área del machine learning se utiliza para medir el rendimiento de la clasificación. Se trata de un coeficiente para medir la concordancia existente en la observación de un fenómeno por dos observaciones distintos

$$k = \frac{P_o - P_e}{1 - P_e}$$

donde:

- P_o representa la probabilidad observada de que coincida el modelo con la clase real y se calcula simplemente como las instancias correctamente clasificadas frente al total: $P_o = \frac{VP + VN}{N}$
- P_e representa la probabilidad de que ambos coincidan por mera casualidad (probabilidad esperada): $P_e = \left(\frac{VP + FN}{N} \cdot \frac{VP + FP}{N}\right) + \left(\frac{FP + VN}{N} \cdot \frac{FN + VN}{N}\right)$

Interpretación de k:

- Si el modelo coincide con la realidad será perfecto $k = 1$
- A partir de 0.6 y/o 0.8 según autor se considera un buen modelo de predicción.

1.3.3) Hold-out

Las métricas deben evaluarse tras el entrenamiento con **datos nuevos** para medir la capacidad de generalización del modelo.

- El método Hold-out

Técnica del machine learning para evaluar el rendimiento y la precisión de un modelo de aprendizaje supervisado.

- Se divide el conjunto de datos disponible en dos subconjuntos mutuamente excluyentes: el conjunto de entrenamiento y el conjunto de test (o de prueba).
- Porcentaje típicos 75% para entrenamiento y 25% para test.
- En general, depende del tamaño del conjunto de datos y la complejidad del problema.

En la práctica, el conjunto de entrenamiento se divide, a sus vez, en dos: entrenamiento y validación. Así, tenemos:

- Conjunto de **Entrenamiento:** para entrenar modelo (cálculo de pesos).
- Conjunto de **Validación:** para tareas de diseño (por ejemplo, cálculo de λ) y otras tareas (por ejemplo, parada del entrenamiento).
- Conjunto de **Test:** para evaluar la capacidad de generalización del modelo final seleccionado. Este conjunto se retiene hasta el final (Método de Retención o Hold Out).

Evaluación de un modelo concreto

Una vez que el conjunto de datos ha sido dividido, se procede de la siguiente manera:

1) Se entrena el modelo empleando el conjunto de entrenamiento y el de validación.
2) Se evalúa su rendimiento del modelo utilizando el conjunto de test para medir la capacidad de generalización. Se calculan las métricas oportunas.

Selección del mejor modelo (entre varios posibles) y de la red final:

1) Se entrena un número de modelos (por ejemplo, para valores distintos de λ) y se escoge aquel que produce menor error de validación.
2) Después, el modelo seleccionado se entrena con todas las muestras disponibles (entrenamiento y validación) y se evalúa con el conjunto test.

Tema 2: Aprendizaje Supervisado

2.1) Árboles de Decisión

- Definición

Los árboles de decisión son máquinas de aprendizaje supervisado que sirven para clasificar o aproximar.

Supongamos el siguiente problema

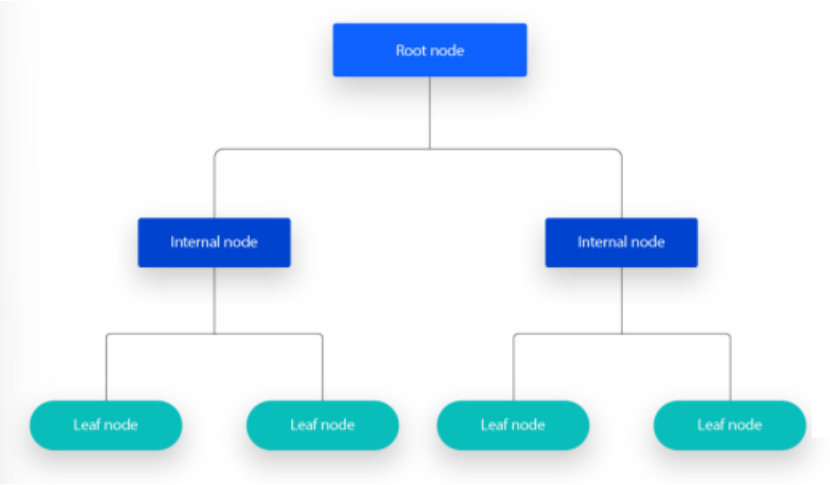
Paciente	Presión Arterial	Urea en sangre	Gota	Hipotiroidismo	Administrar Tratamiento
1	Alta	Alta	Sí	No	No
2	Alta	Alta	Sí	Sí	No
3	Normal	Alta	Sí	No	Sí
4	Baja	Normal	Sí	No	Sí
5	Baja	Baja	No	No	Sí
6	Baja	Baja	No	Sí	No
7	Normal	Baja	No	Sí	Sí
8	Alta	Normal	Sí	No	No
9	Alta	Baja	No	No	Sí
10	Baja	Normal	No	No	Sí
11	Alta	Normal	No	Sí	Sí
12	Normal	Normal	Sí	Sí	Sí
13	Normal	Alta	No	No	Sí
14	Baja	Normal	Si	Sí	No

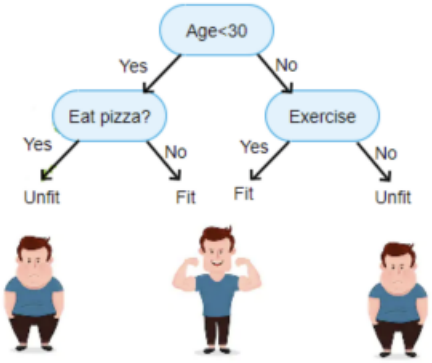
- Planteamiento del problema: ¿Cuál es la **mejor secuencia de preguntas** para saber la clase a la que pertenece un objeto descrito por sus atributos?
- Evidentemente, la "mejor respuesta" es aquella que con el **menor número de preguntas**, devuelve una respuesta suficientemente buena.
- ¿Qué es mejor preguntar primero si tiene gota o cómo tiene la presión arterial?

2.1.0.1) Arquitectura

Un árbol de decisión es una estructura jerárquica que consta de un nodo raíz, ramas, nodos internos y nodos hoja.

- Comienzo con un **nodo raíz** sin ramas entrantes. Las ramas salientes del nodo raíz alimentan los nodos internos.
- Los **nodos internos** evalúan características disponibles para formar subconjuntos homogéneos, indicados por nodos hoja o nodos terminales.
- Los **nodos hoja** representan todos los resultados posibles dentro del conjunto de datos.





2.1.0.2) Ventajas y desventajas

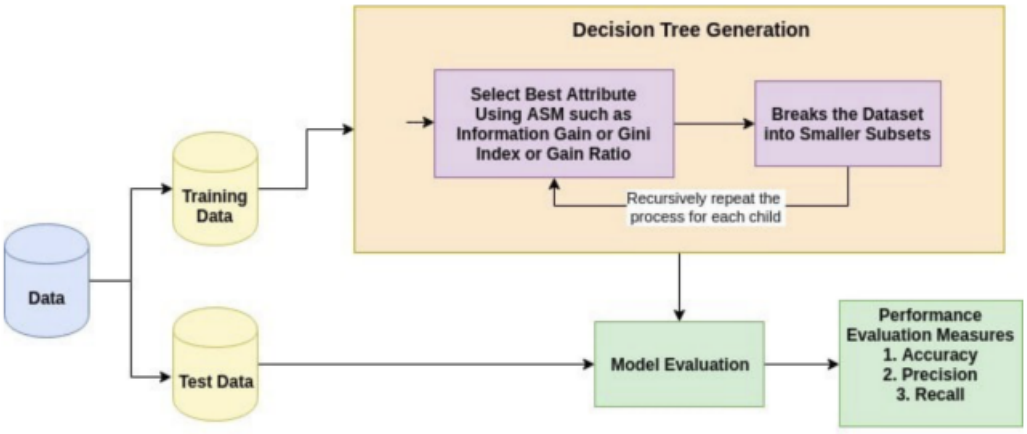
- Pros
  - Fáciles de entender e interpretar.
  - Sirven también para establecer reglas
  - No lineales
  - Menos pre-procesado de los datos: son robustos ante presencia de datos erróneos (outlier), valores faltantes o tipo de datos.
  - Es un método no paramétrico (por ejemplo, no hay suposición acerca del espacio de distribución y la estructura del clasificador).
- Contras
  - **Sobreajuste:** Los árboles más pequeños son más fáciles de interpretar, pero los más grandes pueden resultar en sobreajuste.
  - Perdida de información al categorizar variables continuas.
  - **Precisión:** Otros métodos (por ejemplo, SVM) a menudo tienen tasas de error 30% más bajas que los árboles básico (ID.3 y CART).
  - **Inestabilidad:** un pequeño cambio en los datos puede modificar ampliamente la estructura del árbol (distintos conjuntos, distintos árboles). Varianza elevada.

- Definición alternativa: **recursividad**

Un árbol de decisión es una estructura recursiva formada por nodos, en el que existe:

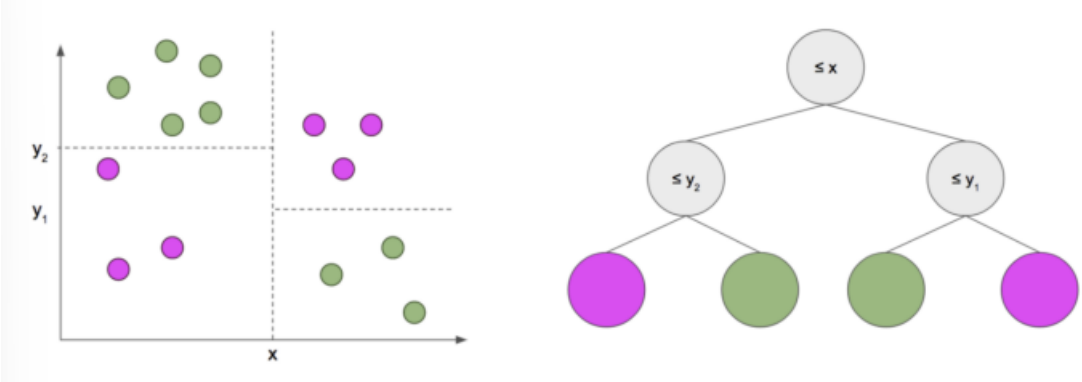
- Un nodo raíz
- El nodo raíz tiene uno o más subnodos.
- Cada uno de los subnodos puede ser, a su vez, raíz de un árbol

Esta característica recursiva hace que muchos de los algoritmos para crearlos se comporten también de manera recursiva.

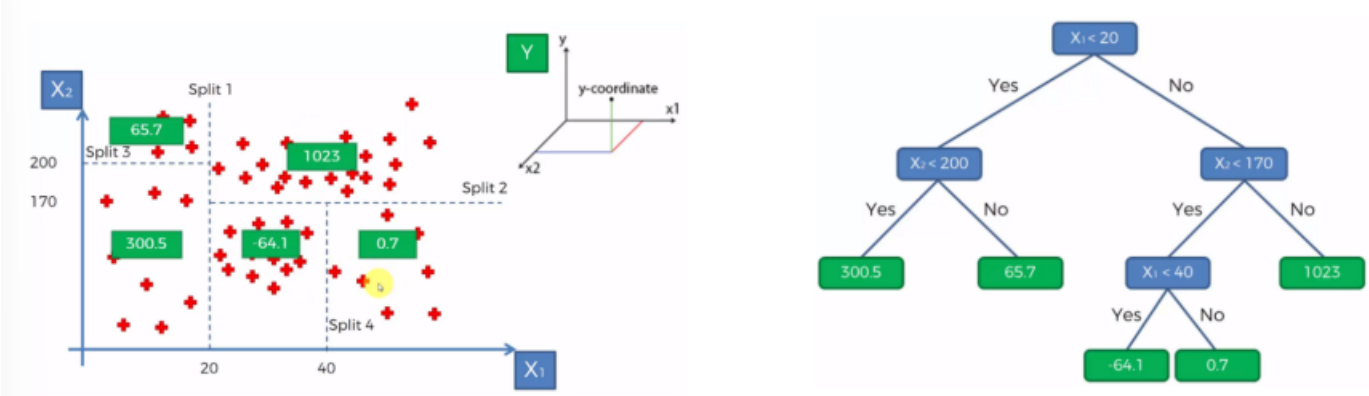


2.1.0.3) Clasificación vs Regresión

- Clasificación
  - La variable dependiente es categórica.
  - Los valores de los nodos hoja son la **moda** de las observaciones de la región



- Regresión
  - La variable dependiente es continua.
  - Los valores de los nodos hoja son la **media** de las observaciones de la región.



2.1.1) Construcción de árboles de decisión

2.1.1.1) Particiones

Cada nodo define una **partición** del conjunto de entrenamiento en función de los datos que representa.
Las particiones producen subconjuntos que son **exhaustivos** y **excluyentes**.
Cuestiones clave:

- **Tipos de particiones:** cuantos más, más posibilidad de encontrar patrones y, por tanto, los árboles más precisos y expresivos.
- **Número de particiones:** A más particiones mayor complejidad. Equilibrio entre complejidad y precisión.
- Selección del **mejor atributo** en cada paso.
- Selección del **mejor valor** de umbral de los valores.

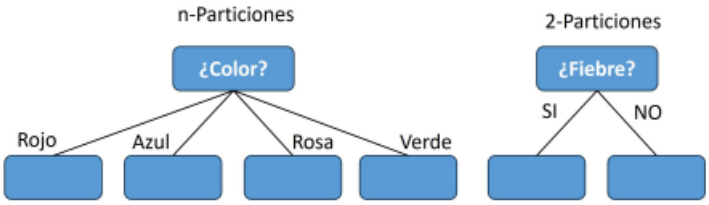
2.1.1.2) Particiones posibles

Los algoritmos más populares sólo proponen un tipo de partición para valores nominales y otro para valores numéricos:

- **Particiones nominales:** En el caso que tengamos un atributo x_i que tenga como posibles valores $\{v_1, v_2, \ldots, v_n\}$ sólo es posible la partición

$$(x_1 = v_1, x_2 = v_2, \cdots, x_n = v_n)$$

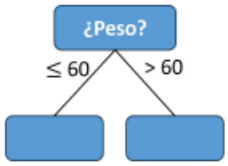
que da lugar a árboles con nodos con más de dos nodos hijos.



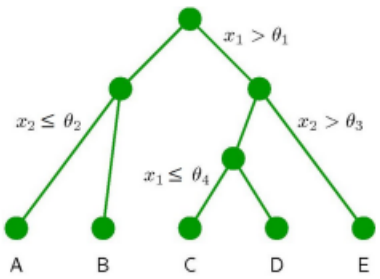
En el caso de árboles binarios se tienen que evaluar n particiones (una por cada posible valor), definidas por $(x_i = v_i, x_i \neq v_i)$.

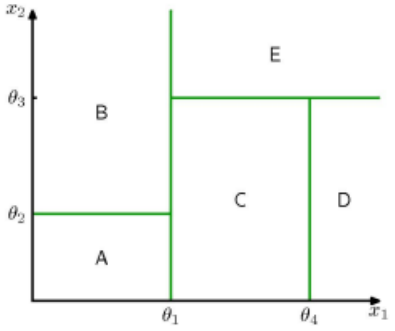
- **Particiones numéricas:** Si el atributo x_i es numérico y continuo, se intenta definir particiones que separe las instancias en intervalos de la forma

$$(x_i \leq a, x_i > a)$$



eligiendo diferentes valores de a tenemos diferentes particiones. La expresividad resultante se conoce como *expresividad cuadrangular* y que no relacionan atributos (sólo un atributo cada vez).





2.1.2) ID3: Algoritmo básico de aprendizaje

> El algoritmo básico de aprendizaje es el **ID3 (Iterative Dichotomiser 3)**, J. Ross Quinlan, investigador australiano que propuso el método en 1983

El método ID3 trata de encontrar una partición que asegure la **máxima capacidad predictiva y la máxima homogeneidad** de las clases
Medida de homogeneidad: la **entropía**
Repetición de **"cortes en dos"** hasta que se cumpla una determinada condición

2.1.2.1) Entropía

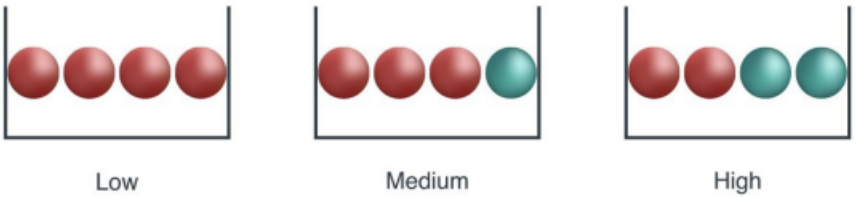
Para determinar el mejor atributo, el ID3 utiliza la **entropía**.

Sea S un conjunto de entrenamiento. Sea $p_\oplus$ la proporción de instancias positivas en S y $p_\ominus$ la proporción de instancias negativas en S. La **entropía de** S es:

$$H(S) = p_\oplus \log_2 \frac{1}{p_\oplus} + p_\ominus \log_2 \frac{1}{p_\ominus} = -p_\oplus \log_2 p_\oplus - p_\ominus \log_2 p_\ominus$$

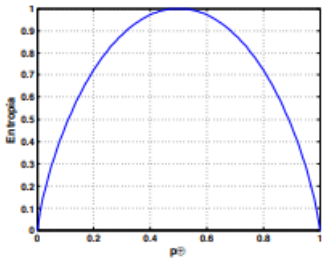
(Relación de la entropía con los conceptos de desorden, equiprobabilidad y homogeneidad).

La entropía nos mide la homogeneidad de los datos (relación inversa).



Para el caso binario:

- Entropía igual a 1 → mínima homogeneidad (equiprobabilidad: $p_\ominus = p_\oplus$).
- Entropía igual a 0 → máxima homogeneidad (todas las instancias de una clase)



A la hora de construir un árbol es preferible crear nodos con nodos hoja homogéneos, es decir, de **baja entropía**.

2.1.2.2) Ganancia de Información

Sea un conjunto de datos $\mathcal{X}$ con entropía $H(\mathcal{X})$.

Si elegimos un atributo A para crear un nodo del árbol, la entropía esperada es:

$$H(\mathcal{X}, A) = \sum_{v \in \text{valores}(A)} \frac{|\mathcal{X}_v|}{\mathcal{X}} H(\mathcal{X}_v)$$

siendo $\mathcal{X}_v$ el subconjunto de $\mathcal{X}$ con todas las instancias con $A = v$.

Por lo tanto, la reducción esperada de la entropía, o lo que es lo mismo la **Ganancia de Información**, al elegir el atributo A como nodo de decisión del árbol es

$$\text{Ganacia}(\mathcal{X}, A))H(\mathcal{X}) - H(\mathcal{X}, A)$$

Por tanto, se elige el atributo que produzca hojas homogéneas, es decir, la **máxima** ganancia de información.

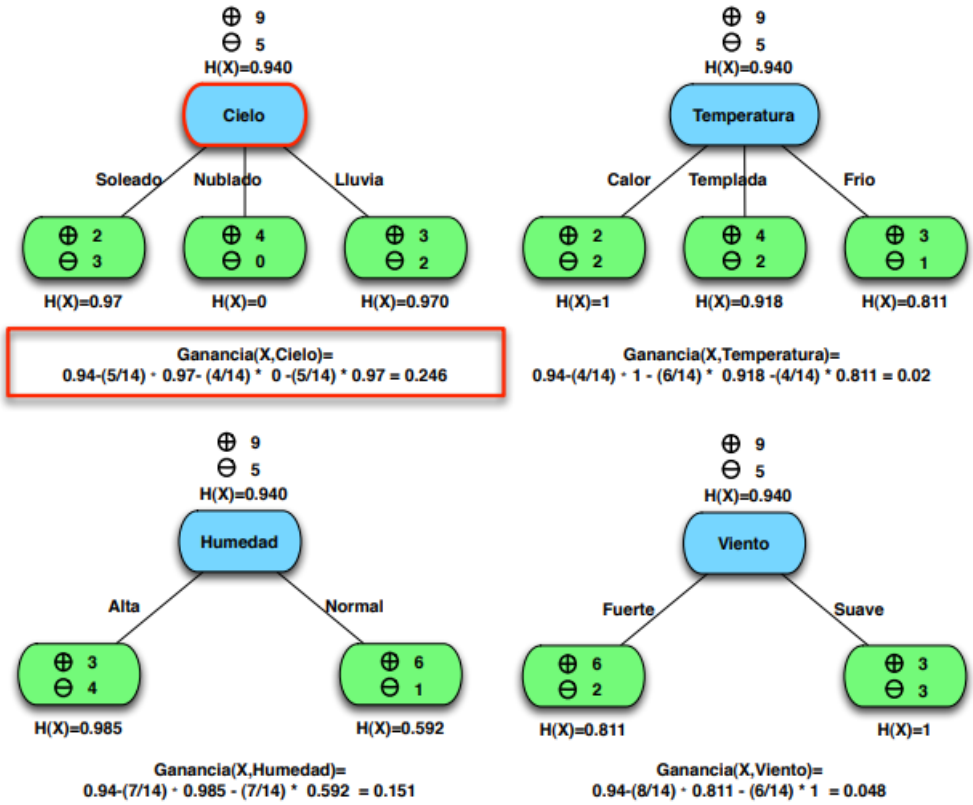
Ejemplo

Atributos nominales (no numéricos)

Día	Cielo	Temperatura	Humedad	Viento	Jugar
D1	Soleado	Calor	Alta	Flojo	No
D2	Soleado	Calor	Alta	Fuerte	No
D3	Nublado	Calor	Alta	Flojo	Si
D4	Lluvia	Templado	Alta	Flojo	Si
D5	Lluvia	Frío	Normal	Flojo	Si
D6	Lluvia	Ario	Normal	Fuerte	No
D7	Nublado	Ario	Normal	Fuerte	Si
D8	Soleado	Templado	Alta	Flojo	No
D9	Soleado	Ario	Normal	Flojo	Si
D10	Lluvia	Templado	Normal	Flojo	Si
D11	Soleado	Templado	Normal	Fuerte	Si
D12	Nublado	Templado	Alta	Fuerte	Si
D13	Nublado	Calor	Normal	Flojo	Si
D14	Lluvia	Templado	Alta	Fuerte	No

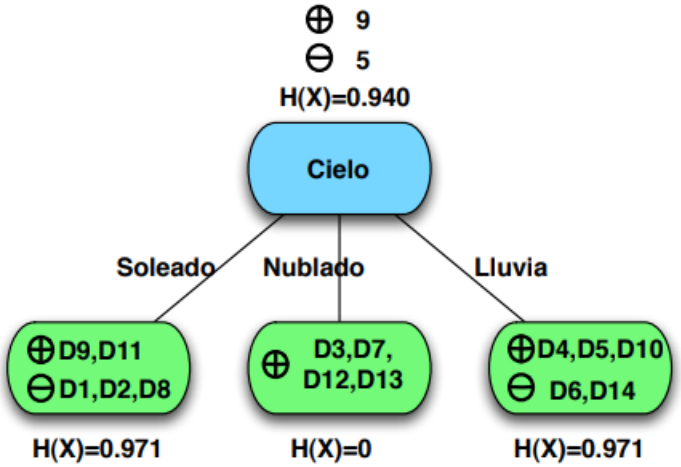
En el ejemplo del tenis tenemos 9 objetos clasificados como $\oplus$ y 5 como $\ominus$, con lo que

$$H([9\oplus, 5\ominus]) = -0.642 \cdot \log_2 0.642 - 0.58 \cdot \log_2 0.358 = 0.94$$



Por lo tanto, el atributo que ofrece una mayor ganancia de información es el atributo **Cielo**.

Utilizando **Cielo** como nodo raíz el árbol inicial quedaría:



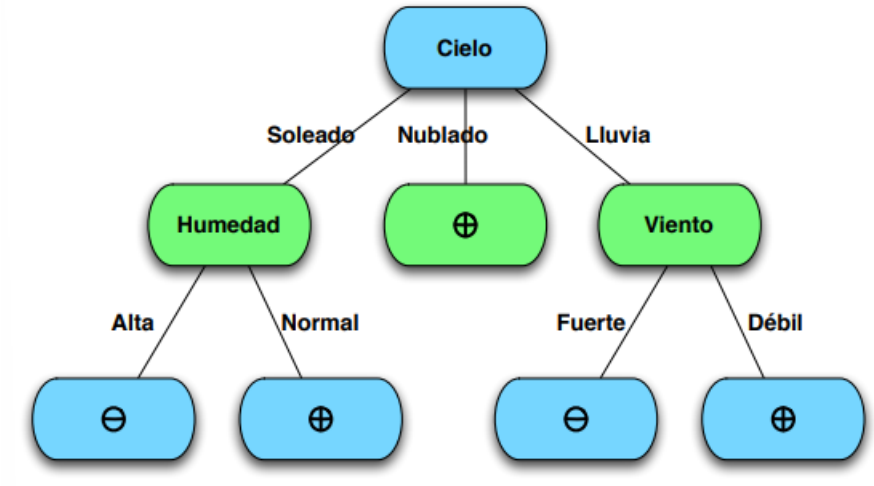
$$Ganancia(\mathcal{X}, A) = H(\mathcal{X}) - \sum_{v \in \text{valores}(A)} \frac{|\mathcal{X}_v|}{|\mathcal{X}|} H(\mathcal{X}_v)$$

Día	Cielo	Temperatura	Humedad	Viento	Jugar
D1	Soleado	Calor	Alta	Flojo	No
D2	Soleado	Calor	Alta	Fuerte	No
D3	Nublado	Calor	Alta	Flojo	Si
D4	Lluvia	Templado	Alta	Flojo	Si
D5	Lluvia	Frio	Normal	Flojo	Si
D6	Lluvia	Frio	Normal	Fuerte	No
D7	Nublado	Frio	Normal	Fuerte	Si
D8	Soleado	Templado	Alta	Flojo	No
D9	Soleado	Frio	Normal	Flojo	Si
D10	Lluvia	Templado	Normal	Flojo	Si
D11	Soleado	Templado	Normal	Fuerte	Si
D12	Nublado	Templado	Alta	Fuerte	Si
D13	Nublado	Calor	Normal	Flojo	Si
D14	Lluvia	Templado	Alta	Fuerte	No

Ahora habría que repetir el proceso con los nodos correspondientes a los valores **soleado** y **lluvia** (el nodo **nublado** sólo contiene una clase).

- Para el nodo Cielo = Soleado:
  - $\mathcal{X}_{\text{soleado}} = \{D_1, D_2, D_8, D_9, D_1 1\}$ con $H(\mathcal{X}_{\text{soelado}}) = 0.971$
  - Ganancia$(\mathcal{X}_{\text{soleado}}, \text{Temperatura}) = 0.971 - \frac{2}{5} \cdot 0 - \frac{2}{5} \cdot 1 - \frac{1}{5} \cdot 0 = 0.570$
  - Ganancia$(\mathcal{X}_{\text{soelado}}, \text{Humedad}) = 0.971 - \frac{3}{5} \cdot 0 - \frac{2}{5} \cdot 0 = 0.971$
  - Ganancia$(\mathcal{X}_{\text{soleado}}, \text{Viento}) = 0.971 - \frac{2}{5} \cdot 1 - \frac{3}{5} \cdot 0.918 = 0.019$
- Para el nodo Cielo = Lluvia:
  - $\mathcal{X}_{\text{lluvia}} = \{D_4, D_5, D_6, D_{10}, D_{14}\}$ con $H(\mathcal{X}_{\text{lluvia}}) = 0.971$
  - Ganancia$(\mathcal{X}_{\text{lluvia}}, \text{Temperatura}) = 0.971 - \frac{3}{5} \cdot 0.918 - \frac{2}{5} \cdot 1 - \frac{1}{5} \cdot 0 = 0.820$
  - Ganancia$(\mathcal{X}_{\text{lluvia}}, \text{Humedad}) = 0.971 - \frac{2}{5} \cdot 1 - \frac{3}{5} \cdot 0.918 = 0.820$
  - Ganancia$(\mathcal{X}_{\text{lluvia}}, \text{Viento}) = 0.971 - \frac{3}{5} \cdot 0 - \frac{2}{5} \cdot 0 = 0.971$

Por lo tanto, el árbol resultante sería



Día	Cielo	Temperatura	Humedad	Viento	Jugar
D1	Soleado	Calor	Alta	Flojo	No
D2	Soleado	Calor	Alta	Fuerte	No
D3	Nublado	Calor	Alta	Flojo	Si
D4	Lluvia	Templado	Alta	Flojo	Si
D5	Lluvia	Frio	Normal	Flojo	Si
D6	Lluvia	Frio	Normal	Fuerte	No
D7	Nublado	Frio	Normal	Fuerte	Si
D8	Soleado	Templado	Alta	Flojo	No
D9	Soleado	Frio	Normal	Flojo	Si
D10	Lluvia	Templado	Normal	Flojo	Si
D11	Soleado	Templado	Normal	Fuerte	Si
D12	Nublado	Templado	Alta	Fuerte	Si
D13	Nublado	Calor	Normal	Flojo	Si
D14	Lluvia	Templado	Alta	Fuerte	No

Todos los nodos hoja tienen una entropía nula (solo instancias de una clase).

2.1.2.3) Error global

Es la probabilidad de error, es decir, suma ponderada de los errores de todas las hojas del árbol.

$$E = \sum_{i=1}^{n_h} w_i e_i$$

donde

- n_h es el numero de hojas del árbol.
- w_i es el peso o probabilidad de la hoja i, es decir, la probabilidad de que una instancia sea clasificada por la partición representada por la rama que acaba en la hoja i.
- e_i es el error correspondiente a la rama que acaba en la hoja i (número de instancia erróneas que caen en la hoja i entre el número de instancias que caen en la hoja i).

2.1.2.4) Algoritmo

El algoritmo básico de aprendizaje es el ID3 (Iterative Dichotomiser 3).

Algoritmo árbol ← aprenderArbol(*datos*)

```
1: si todos los ejemplos en datos tienen la misma etiqueta entonces
2:    devolver un nodo hoja con dicha etiqueta
3: sino
4:    Sea A el atributo que clasifica mejor a los objetos en datos
5:    para todo posible valor v de A hacer
6:       data(v) ← todos los objetos con A = v
7:       Añadir nueva rama ← aprenderArbol(data(v))
8:    fin para
9:    devolver árbol
10: fin si
```

ID3(Instancias, Etiquetas, Atributos)

Entrada: `Instancias`: el conjunto de datos

Entrada: `Etiquetas`: el conjunto de posibles clases.

Entrada: `Atributos`: el conjunto de atributos en el conjunto `Instancias`.

si todas las instancias son positivas **entonces**

devolver el nodo raíz con etiqueta +.

sino si todas las instncias son negativas **entonces**

devolver el nodo raíz con la etiqueta −.

sino si `Atributos`=∅ **entonces**

devolver el nodo raíz con el valor de `Etiquetas` más probable en `Instancias`.

fin si

Sea A el atributo que clasifica mejor las instancias en *datos*.

Crear un árbol con un nodo etiquetado con A

para todo posible valor v_i del atributo A **hacer**

añadir un arco bajo la raíz con la comprobación $A = v_i$.

sea `Instancias` v_i el subconjunto de `Instancias` con $A = v_i$.

si $\texttt{Instancias}_{v_i} = \varnothing$ **entonces**

añadir un nodo hoja al arco añadido con el valor de `Etiquetas` más probable en `Ejemplos`.

sino

añadir al nuevo árbol el subárbol generado por **ID3** ($\texttt{Instancias}_{v_i}$,`Etiquetas`, `Atributos`$-\{A\}$).

fin si

fin para

devolver nodo raíz

2.1.3) Sobre-ajuste

2.1.3.1) Espacio de hipótesis y sobre-ajuste

El **espacio de hipótesis** H (no confundir con la entropía) en árboles de decisión abarca todas las posibles combinaciones de atributos y valores que pueden formar árboles de decisión, y nuestra tarea es encontrar la hipótesis más adecuada para clasificar correctamente las instancias de entrada.

En general, se prefieren hipótesis cortas para evitar el sobre-ajuste (**overfitting**).

- Sobre-ajuste

Dado un espacio de hipótesis H, se dice que una hipótesis particular $h \in H$ sobreajusta los datos de entrenamiento si existe un hipótesis alternativa $h' \in H$, tal que h presenta un error menor que h' sobre los ejemplos de entrenamiento, pero h' presenta un error menor que h sobre el conjunto total de observaciones.

2.1.3.2) Proceso de poda

- ¿Cómo podemos evitar el sobre-aprendizaje o sobre-ajuste?

Poda: Eliminar condiciones de las ramas del árbol encontrar más pequeños. Existe dos tipos de poda:

- **Prepoda:** El proceso se realiza durante la construcción del árbol, estableciendo un criterio de parada.
  - El número de instancias por nodo.
  - El error esperado.
  - MDL (Minimun Description Lenght).

- **Postpoda:** El proceso se realiza después de la construcción del árbol
  - Consiste en ir eliminando nodos de abajo a arriba mientras se vaya cumpliendo un criterio determinado.

Se pueden combinar ambas aproximaciones.

2.1.3.3) Otras medidas

Medidas alternativas: En algunos casos se suele utilizar otras medidas como:

- El *Ratio* de la ganancia de información, para evitar el hecho de que se favorece la selección de los atributos con más valores.
- MSE para regresión
- Índice Gini empleado por el algoritmo CART
- DKM, basados en AUC, MDL

2.1.4) Algoritmo CART y Otros

2.1.4.1) CART

Cart (**C**lasificación **A**nd **R**egression **T**rees). Similar al ID3 pero:

- Permite que la variable que define la clase sea continua y no construye un conjunto de reglas.
- Utiliza el índice de Gini en vez de la ganancia de información para seleccionar el mejor atributo (la mejor partición).
- Utiliza también el esquema de partición recursiva utilizando una estrategia voraz.
- También permite resolver problemas de regresión.

Se elige la partición que produce el menor valor de la función de coste.

- Para regresión: RSME.
- Para clasificación: GINI

$$\text{Gini}(p) = \sum_{i=1}^{n} p_i(1 - p_i)$$

  con p_i las proposiciones de instancias de la clase i en la partición.

Prepoda: Utiliza como criterio de parada el número mínimo de instancias asignadas al nodo.

Postpoda: Utiliza un criterio que controla la improtancia relativa del error frente la complejidad (tamaño del árbol).

2.1.4.2) C4.5, C5.0

C4.5: Permite, a diferenciaque ID3, que las características puedan ser continuas, definiendo de forma dinámica un atributo discreto particionando los atributos continuos en un conjunto discreto de intervalos.

- C4.5 transforma los árboles obtenidos en un conjunto de reglas del tipo *if-then*. La precisión de cada regla es evaluada de forma independiente para determinar el orden en el que deben ser aplicadas.
- Un proceso de poda elimina antecedentes de las reglas si con esto se mejora la precisión de la misma.
- C5.0, utiliza menos memoria y obtiene un conjunto de reglas menor y más preciso.
- J48 es la implementación en código abierto de C4.5

2.1.5) Random Forests

Random Forest es una técnica que construye un grán número de árboles de decisión no correlacionados.

Se basa en la técnica de Agregación de Bootstrap (Bagging), técnica de agregación de clasificadores o regresores que:

- Aumenta la precisión y estabilidad reduciendo los efectos del ruido.
- Reduce la varianza en las predicciones.
- Ayuda a evitar el sobre ajuste.

La predicción del modelo se elige analizando las predicciones de cada uno de los árboles considerados en el modelo.

2.1.5.1) Algoritmo

Algoritmo RF $\leftarrow$ RandomForest(*datos*)

1: **para** $i \leftarrow 1 \rightarrow$ *n_arboles* **hacer**
2: Extraer una muestra de tamaño *size(data)* de *datos* por bootstrapping
3: Construir un árbol T_i repitiendo recurivamente para cada nodo hoja
4:
1. Selecionar aleatoriamente m atributos.
2. Seleccionar la mejor partición de las inducidas por los m atributos.
3. Dividir el nodo en dos nodos hijos.
4. Si el tamaño de nodo alcanza $n_{\min}$ no continuar dividiendo.

5: **fin para**
6: **devolver** El conjunto de árboles $\{T_i\}_1^{n\_arboles}$.

Si tenemos p atributos las recomendaciones para el valor de m son:

- Clasificación: $\lfloor \sqrt{m} \rfloor$
- Regresión: $\lfloor p/3 \rfloor$
- Siendo el valor mínimo 1.

Una vez se han construido los árboles para hacer predicciones:

- Clasificación: La clase más votada por los árboles.
- Regresión: La media de todas las predicciones

2.1.5.2) OBB e importancia de las variables

OBB (out of the bag) error estime: Para cada muestra se predice el error utilizando sólo los árboles en los que no ha sido utilizada (no ha sido elegida en bootstrapping)

- Los valores son parecidos a los que se obtienen mediante una validación cruzada de N-pliegues.

Importancia de los atributos:

- En cada partición del árbol se registra la mejora del criterio de división.
- Este valor se considera la importancia del atributo.
- Para cada variable se agregan los valores generados en cada árbol.

2.1.6) Conclusiones

La utilización de árboles de decisión es muy popular en muchas disciplinas.

Experimentalmente han demostrado una buena capacidad de clasificación.

La clave reside en la función a optimizar a la hora de elegir el atributo para disgregar el árbol.

Los ensambles nos permiten mejorar los resultados combinando información.