

UNIVERSIDADE DA CORUÑA

APRENDIZAJE AUTOMÁTICO

TEMA 2:

APRENDIZAJE SUPERVISADO

2.1 INTRODUCCIÓN

- Dos fases del aprendizaje:
 - Entrenamiento
 - Se presentan los ejemplos al sistema (conjunto de entrenamiento)
 - El sistema “aprende” a partir de los ejemplos
 - El sistema modifica gradualmente sus parámetros ajustables hasta que la salida se ajuste a la salida deseada
 - Necesitamos una medida de rendimiento/precisión/error
 - Utilización
 - Nuevos ejemplos nunca vistos antes
 - Pedimos al sistema que **generalice**
 - Necesario un conjunto de test

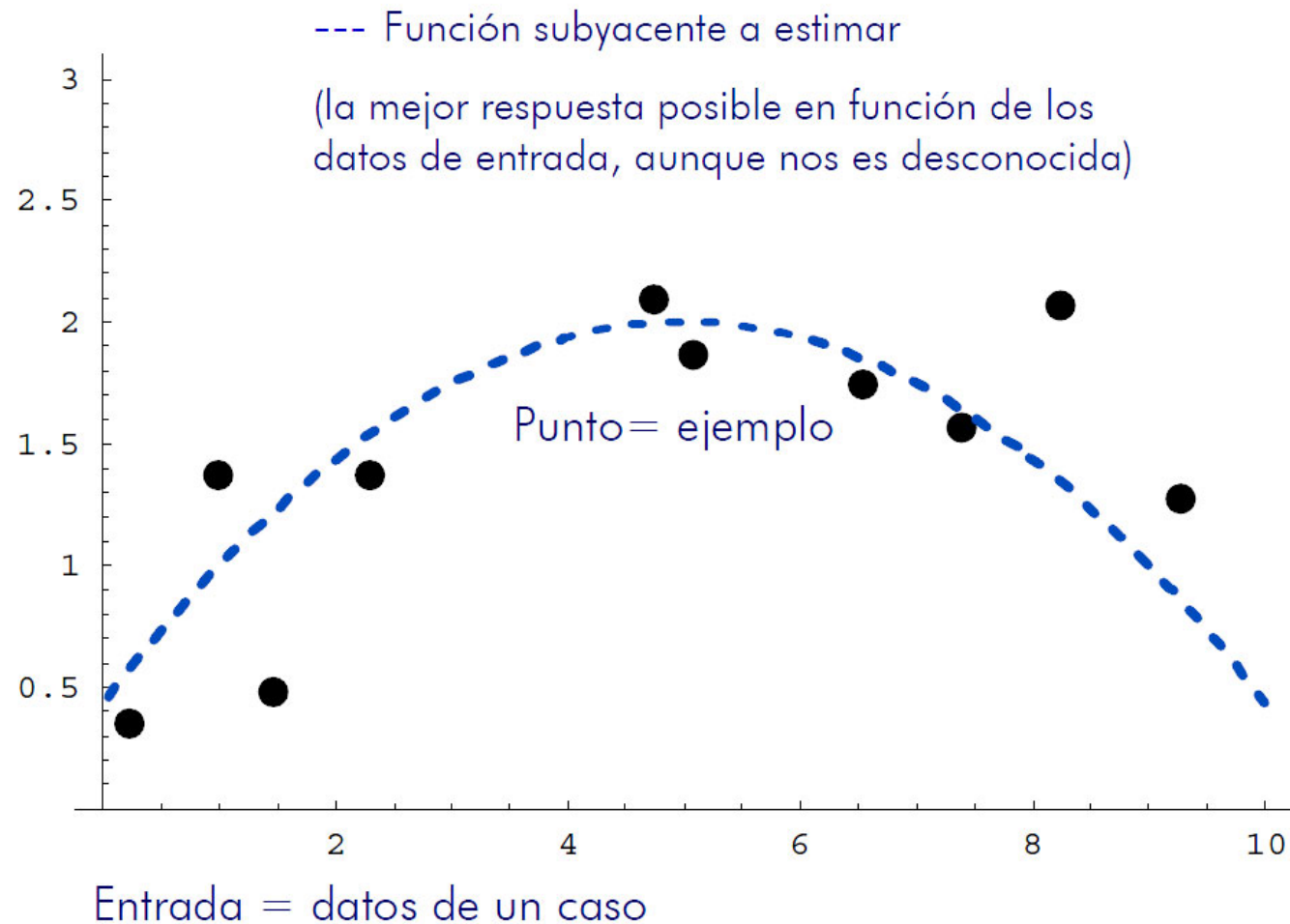
2.1 INTRODUCCIÓN

- Generalización:
 - No se quiere que el sistema memorice
 - Dar la respuesta correcta sólo a los ejemplos de entrenamiento
 - Difícil para los humanos, fácil para un ordenador
 - Se quiere aprender a generalizar
 - Más fácil para los humanos (conceptualizar) que para un ordenador (representar los conceptos)
 - La mente animal y humana busca patrones incluso donde no los hay
 - Pareidolias
 - La superstición de la paloma (Skinner, 1948)
 - Hay que extraer la esencia, la estructura de los datos, y no solamente aprender la respuesta correcta para algunos casos
 - Muy relacionado con la complejidad del modelo

2.1 INTRODUCCIÓN

- Ejemplo:

Salida
=
Valor a
predecir

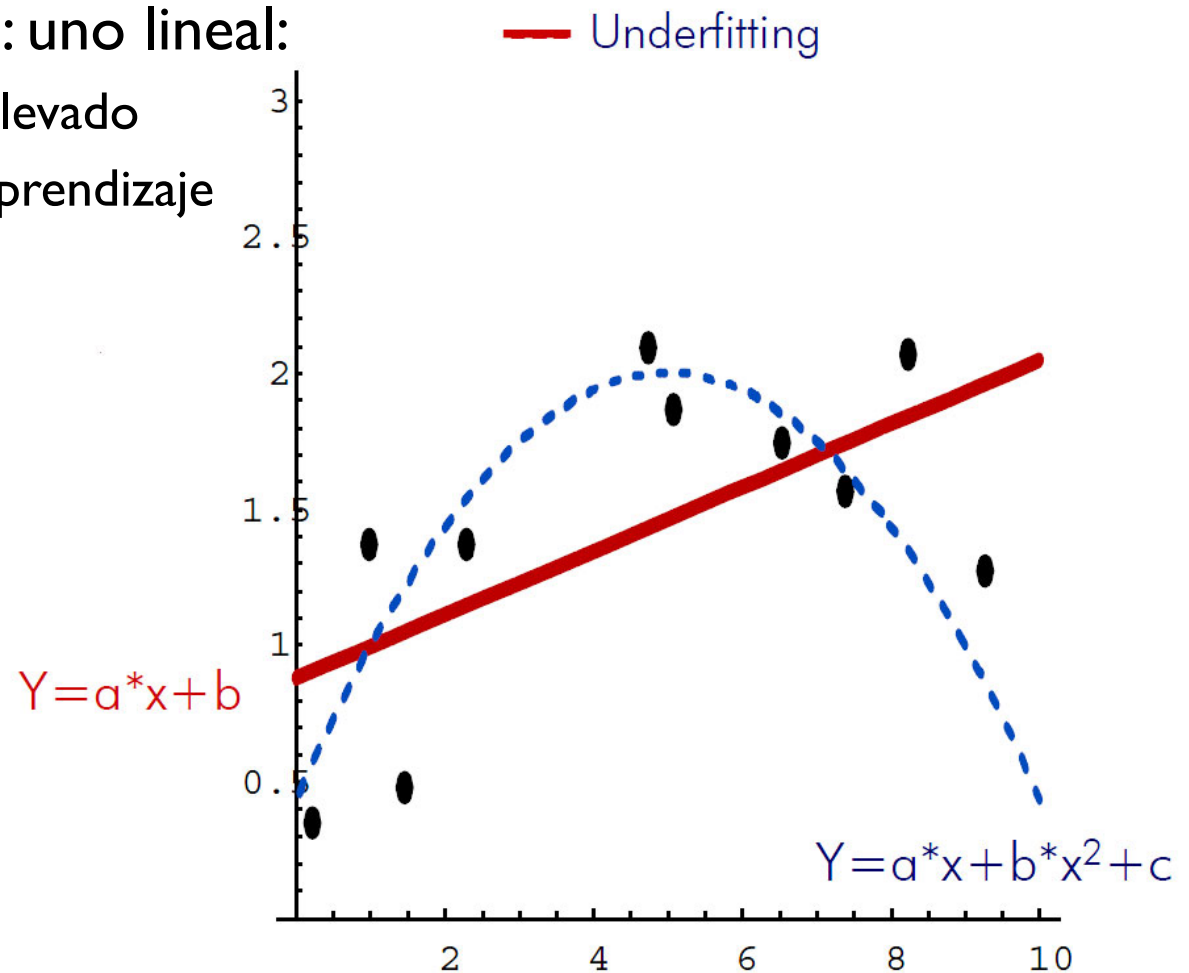


2.1 INTRODUCCIÓN

- Ejemplo:
 - Se escoge como modelo los polinomios
$$y = a_n x^n + a_{n-1} x^{n-1} + \dots + a_0$$
 - Parámetros: a_0, a_1, \dots, a_n
 - Se puede controlar explícitamente la complejidad del modelo dando distintos valores al hiperparámetro n
 - Valores pequeños: modelo con menos parámetros, más sencillo
 - Sólo puede hacer formas sencillas
 - Valores grandes: modelo con más parámetros, más complejo
 - Puede hacer formas complejas

2.1 INTRODUCCIÓN

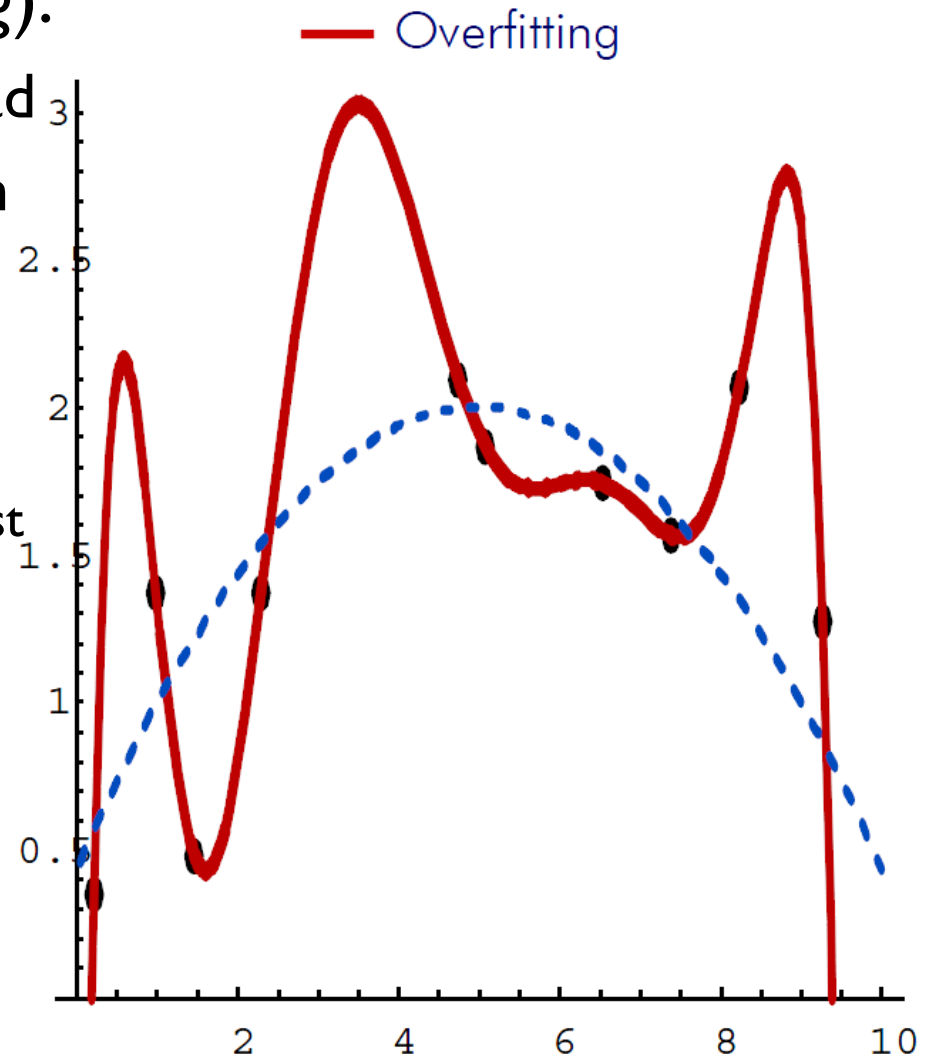
- Si se escoge un modelo demasiado simple
 - Por ejemplo: uno lineal:
 - Error muy elevado durante el aprendizaje y en el test



2.1 INTRODUCCIÓN

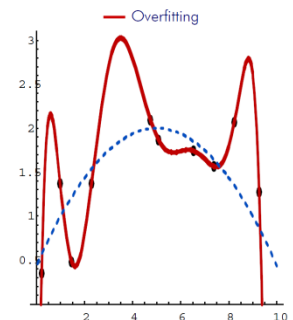
- **Sobreajuste (*overfitting*):**

- Aprende en profundidad pero no generaliza bien en un nuevo caso
- Error pequeño sobre los ejemplos de aprendizaje pero grande en los de test



2.1 INTRODUCCIÓN

- Sobreajuste (*overfitting*) vs. sobreentrenamiento (*overtraining*)
 - Sobreentrenamiento: se obliga al modelo a aprender bien los patrones de entrenamiento
 - Pierde capacidad de generalización
 - El modelo se sobreajusta
 - Sin embargo, el sobreajuste también puede aparecer por otros motivos, el más típico es que el modelo es demasiado complejo, por ejemplo:
 - Modelo polinómico de un grado muy alto, cuando uno lineal o de grado bajo sería suficiente
 - Ejemplo: dispositiva anterior
 - RNA con demasiadas neuronas ocultas



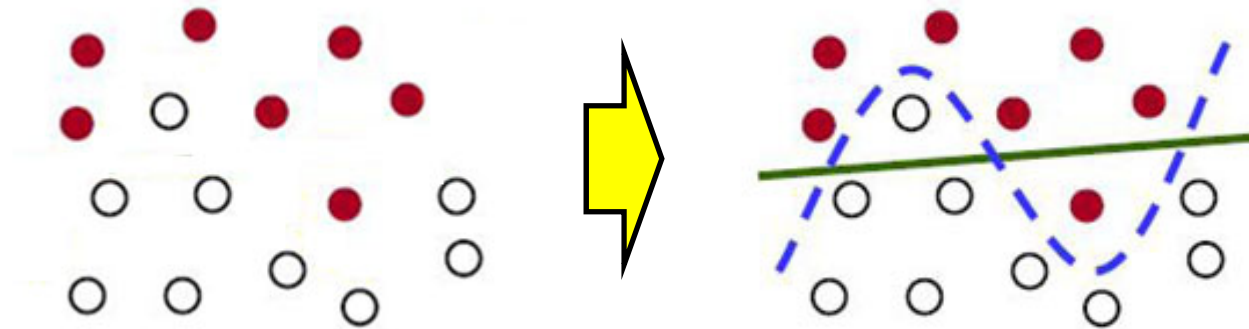
2.1 INTRODUCCIÓN

- **Parámetros vs. hiperparámetros**
 - **Parámetros:** Cada uno de los valores que definen el comportamiento de un modelo
 - Son internos del modelo
 - Su valor se fija mediante el proceso de entrenamiento
 - **Hiperparámetros:** Parámetros cuyo valor se usa para controlar el proceso de aprendizaje
 - En función de ellos se determinan los valores de los parámetros
 - Su valor lo fija el usuario

Modelo	Parámetros	Hiperparámetros
Polinomio	coeficientes a_i	orden
RNA	pesos, bias	Arquitectura, tasa de aprendizaje, funciones de activación, etc.
SVM	α	kernel, C, etc.

2.1 INTRODUCCIÓN

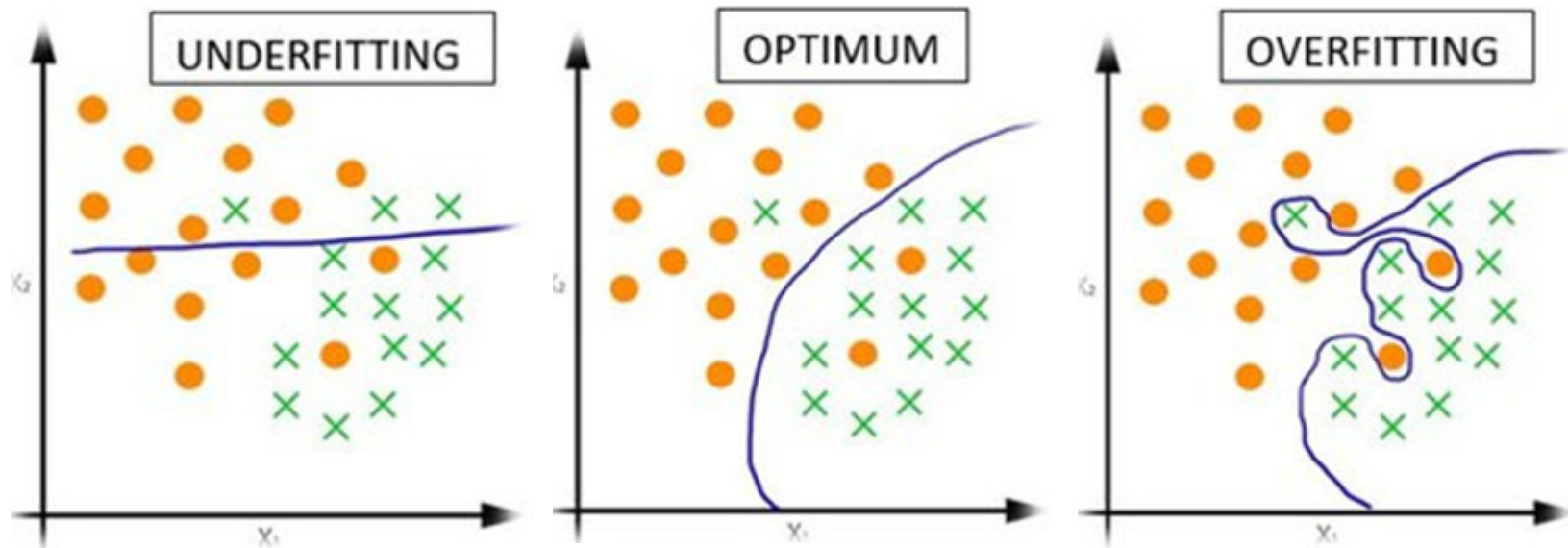
- ¿Cómo evitar el sobreajuste?
 - Una opción frecuente es controlar la complejidad del modelo:
 - Se prefiere el modelo más simple que sea capaz de explicar los datos
 - Ejemplo: ¿Qué modelo escoger?



- Modelo sencillo:
 - No explica bien todos los datos
- Modelo complejo:
 - Posiblemente subajustado

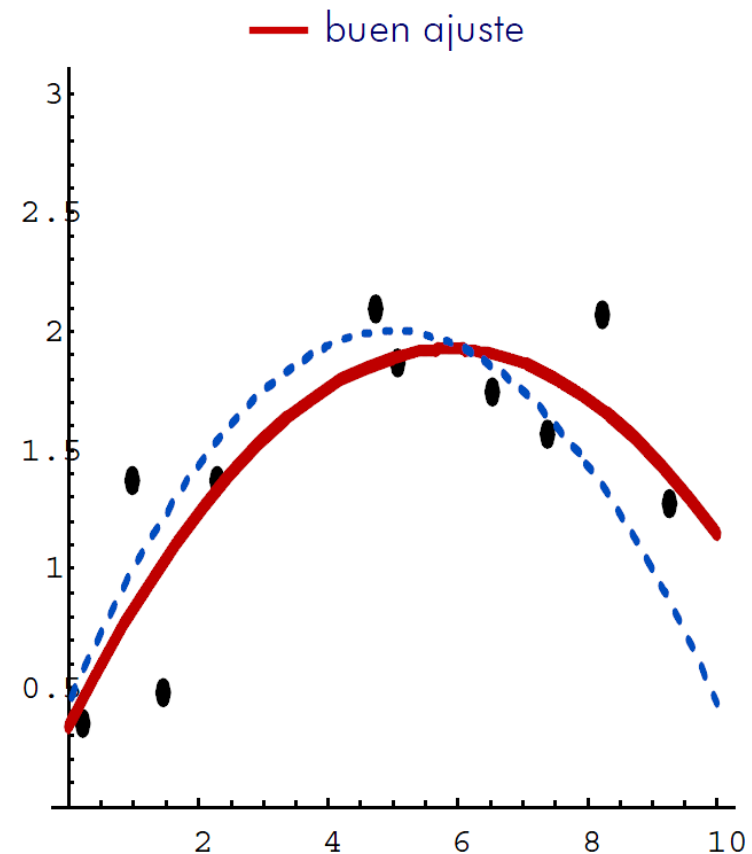
2.1 INTRODUCCIÓN

- *Overfitting vs. Underfitting*



2.1 INTRODUCCIÓN

- Un buen modelo
 - El modelo es suficientemente flexible como para capturar la forma de la curva, pero no tanto como para dar lugar a un sobreajuste



2.1 INTRODUCCIÓN

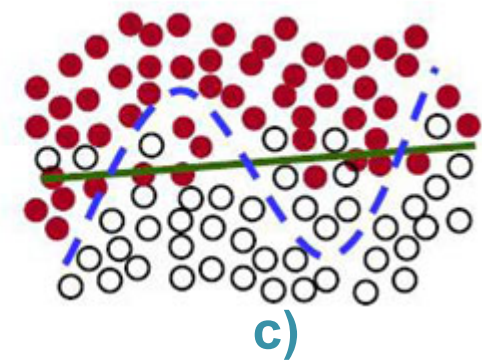
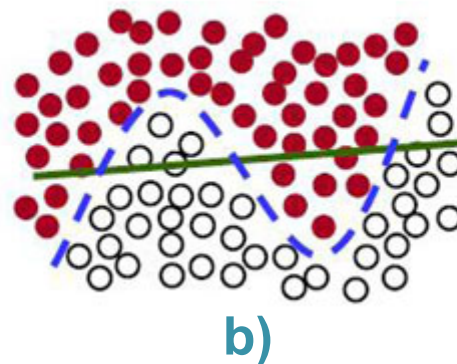
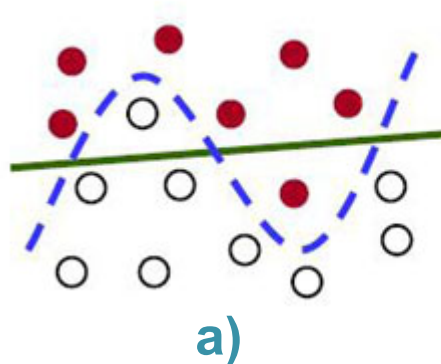
- Para cualquier algoritmo de aprendizaje, se debe encontrar el equilibrio entre (Dietterich):
 - La complejidad del modelo
 - El número de ejemplos disponibles
 - El error de generalización
- Ejemplo:

2.1 INTRODUCCIÓN

- Ejemplo:

- a) Pocos patrones

- El modelo complejo tiene una mayor precisión, aunque no mucho mayor.
 - Se podría pensar que el modelo sencillo es mejor (más generalización) porque los patrones mal clasificados son consecuencia del posible ruido
 - Problema: hay muy pocos patrones
 - Con tan pocos patrones, no se puede determinar la influencia del ruido en los mal clasificados



2.1 INTRODUCCIÓN

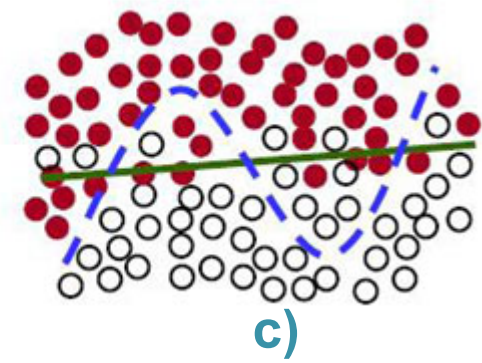
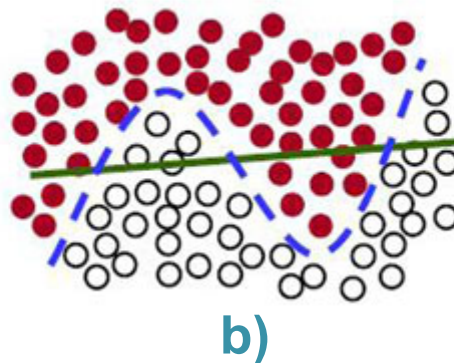
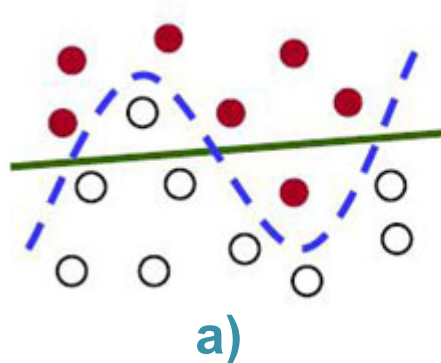
- Ejemplo:

- b) Muchos patrones, sin ruido

- El modelo complejo tiene mucha mejor precisión que el sencillo
 - Se escoge el modelo complejo
 - Escenario no realista: en el mundo real normalmente habría ruido

- c) Muchos patrones, con ruido

- Ambos modelos tienen un nivel de precisión similar
 - Se escoge el más sencillo



2.1 INTRODUCCIÓN

- La dimensión Vapnik-Chervonenkis (VC):
 - Mide la complejidad o capacidad de una determinada familia de funciones $f(x;\Theta)$
 - f representa la familia
 - Θ es el conjunto de parámetros
 - Se define como el máximo número de ejemplos (cardinalidad) que pueden ser explicados por la familia $f(x;\Theta)$
 - La complejidad se mide en número de ejemplos
 - Número de ejemplos que la familia f puede “separar” o “romper”
 - A mayor complejidad, mayor capacidad para ajustar los datos de entrenamiento y menor poder de generalización

2.1 INTRODUCCIÓN

- *Shatter* (rotura) de un conjunto de datos:
 - Sean n ejemplos $\{x_1, x_2, \dots, x_n\}$ y 2 clases
 - Cada ejemplo es un punto en \mathcal{R}^m (m atributos)
 - Existen 2^n posibles formas de asignar las clases (dicotomías)
 - Por ejemplo, para $n=3$ existen las 8 dicotomías:
 - $\{(0,0,0), (0,0,1), (0,1,0), (0,1,1), \dots, (1,1,1)\}$
 $\begin{matrix} x_1 & x_2 & x_3 & x_1 & x_2 & x_3 & x_1 & x_2 & x_3 & x_1 & x_2 & x_3 & x_1 & x_2 & x_3 \end{matrix}$
 - Se dice que la familia de funciones f hace un *shatter* (rompe) de los datos si para cualquier dicotomía existe un conjunto de parámetros Θ tal que $f(x;\Theta)$ la resuelve
 - Es decir, dados n patrones
 - (datos n puntos en \mathcal{R}^m)
 - f rompe esos patrones si es capaz de separarlos, se distribuyan como se distribuyan en las dos clases
 - Con una configuración de parámetros distinta para cada distribución de los patrones en clases

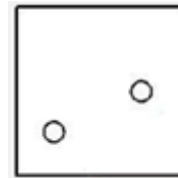
2.1 INTRODUCCIÓN

- La dimensión VC de la familia $f(x;\Theta)$ es el tamaño del mayor conjunto para el que existe un *shatter*
 - El decir, el mayor número de patrones que, distribuidos en el espacio de una manera concreta, se pueden distribuir de forma que se puedan romper con f
 - Si la dimensión VC de $f(x;\Theta)$ es h , existe al menos un conjunto de h ejemplos (una distribución de los ejemplos en el espacio) para el que hay un *shatter* con $f(x;\Theta)$
- Ejemplo con patrones con 2 atributos: en \mathcal{R}^2
 - Se toma como la familia f los hiperplanos
 - En \mathcal{R}^2 es una recta, deja unos ejemplos a un lado, y otros al otro
 - $f(x;\Theta) = w_0 + w_1x_1 + w_2x_2$
 - $\Theta = (w_0, w_1, w_2)$

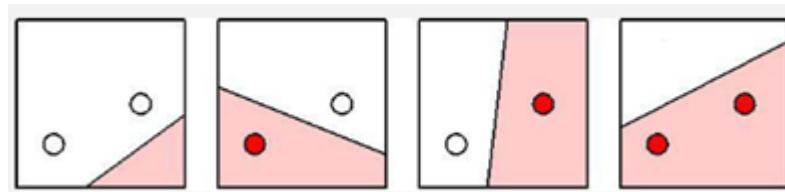
2.1 INTRODUCCIÓN

- Para $n=2$ se pueden resolver todas las dicotomías con f (hiperplanos)

- Conjunto de ejemplos:



- Para cada posible asignación a las clases (dicotomía), se puede encontrar un conjunto de parámetros (w_0, w_1, w_2) distinto (es decir, un plano distinto) que los separe:

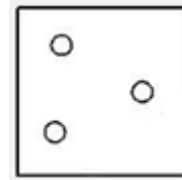


- En esta distribución de los ejemplos, se pueden separar todas las dicotomías
 - f tiene al menos dimensión 2

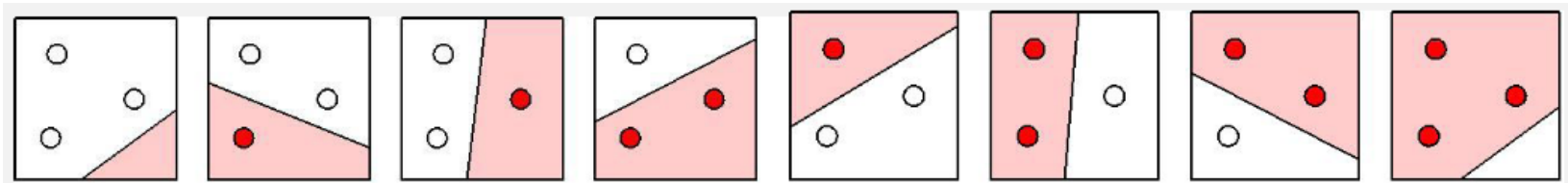
2.1 INTRODUCCIÓN

- Para $n=3$ se puede encontrar un conjunto de ejemplos tal que sea posible resolver todas las dicotomías con hiperplanos

- Conjunto de ejemplos:



- Se asignen como se asignen los 3 ejemplos a las dos clases, existe un conjunto de parámetros (w_0, w_1, w_2) tal que f separe los ejemplos

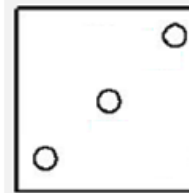


- En esta distribución de los ejemplos, se pueden separar todas las dicotomías
 - f tiene al menos dimensión 3

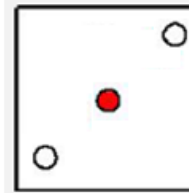
2.1 INTRODUCCIÓN

- Para $n=3$ se puede encontrar un conjunto de ejemplos tal que sea posible resolver todas las dicotomías con hiperplanos
- Aunque existen otros conjuntos de 3 elementos que no se puedan separar, basta con que haya uno que sí se pueda separar (anterior)

- Por ejemplo, esta disposición:



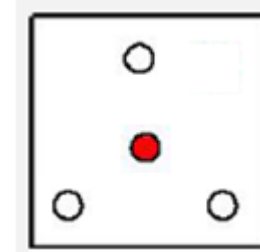
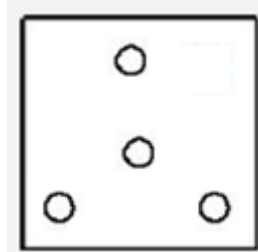
se puede asignar de tal forma que no sea separable con un hiperplano:



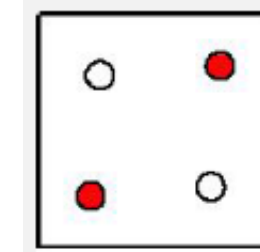
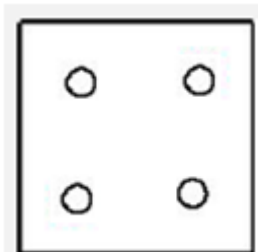
- Pero como la disposición anterior sí era separable, f tiene al menos dimensión 3

2.1 INTRODUCCIÓN

- Para $n = 4$ no es posible
 - No existe un conjunto de 4 ejemplos tal que f pueda separar todas las posibilidades de asignación de las clases
 - Se tome el conjunto que se tome, habrá una asignación no separable por f
 - Por ejemplo, para el conjunto se pueden asignar las clases de esta manera:



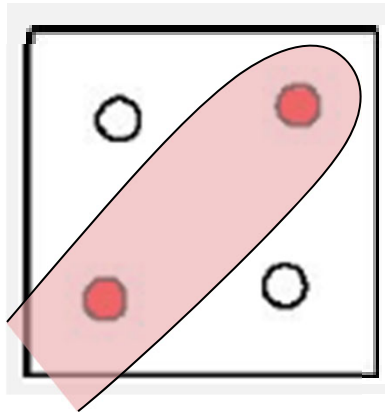
- Por ejemplo, para el conjunto se pueden asignar las clases de esta manera:



- Por tanto, los hiperplanos en \mathbb{R}^2 tienen dimensión 3
 - Dimensión baja ➡ “Poco complejos”

2.1 INTRODUCCIÓN

- Otra familia más compleja podría tener dimensión 4 o superior:
 - Podría separar un número más alto de patrones, se distribuyeran como se distribuyeran
 - Familia más compleja, con menos poder de generalización
 - Por ejemplo, polinomios de un grado superior



2.1 INTRODUCCIÓN

- La dimensión VC permite calcular el riesgo esperado (error en el test) a partir del riesgo empírico (error en el entrenamiento):
 - Cota superior, con probabilidad $1 - \eta$, para el riesgo esperado (error en el conjunto de test):

$$\text{cota}(e_{\text{test}}(f)) = \text{error}_{\text{train}} + \underbrace{\sqrt{\frac{h(\log \frac{2n}{h} + 1) - \log \frac{\eta}{4}}{n}}}_{\text{Confianza VC}}$$

- h es la dimensión VC de f
- n es el número de ejemplos de entrenamiento
- Se cumple $n > h$
- El segundo término se denomina confianza VC

2.1 INTRODUCCIÓN

$$\text{cota}(e_{\text{test}}(f)) = \text{error}_{\text{train}} + \sqrt{\frac{h(\log \frac{2n}{h} + 1) - \log \frac{\eta}{4}}{n}}$$

- Cuando n/h aumenta, la confianza VC **disminuye** y el riesgo en entrenamiento es una mejor aproximación al riesgo esperado
 - Disminuye la confianza VC: la cota del error de test se acerca al error de entrenamiento
 - Menos sobreajuste ➡ Mayor poder de generalización

2.1 INTRODUCCIÓN

$$\text{cota}(e_{\text{test}}(f)) = \text{error}_{\text{train}} + \sqrt{\frac{h(\log \frac{2n}{h} + 1) - \log \frac{\eta}{4}}{n}}$$

- Cuando n/h aumenta, la confianza VC **disminuye** y el riesgo en entrenamiento es una mejor aproximación al riesgo esperado
 - Más poder de generalización
 - n/h aumenta:
 - n aumenta:
 - El número de patrones de entrenamiento con respecto a la dimensión VC aumenta
 - Por tanto, cuantos más ejemplos de entrenamiento, mayor poder de generalización
 - En líneas generales
 - h disminuye:
 - El modelo es capaz de separar menos patrones
 - El modelo es menos complejo
 - Por tanto, cuanto menos complejo sea el modelo, mayor poder de generalización

2.1 INTRODUCCIÓN

$$\text{cota}(e_{\text{test}}(f)) = \text{error}_{\text{train}} + \sqrt{\frac{h(\log \frac{2n}{h} + 1) - \log \frac{\eta}{4}}{n}}$$

- Cuando n/h disminuye, la confianza VC **aumenta** y el riesgo esperado (test) se aleja del riesgo en entrenamiento
 - Se sobreajusta ➡ menos poder de generalización
 - n/h disminuye:
 - n disminuye:
 - El número de patrones de entrenamiento con respecto a la dimensión VC disminuye
 - Por tanto, cuantos menos ejemplos de entrenamiento, menor poder de generalización
 - En líneas generales
 - h aumenta:
 - El modelo es más complejo
 - Por tanto, cuanto más complejo sea el modelo, menor poder de generalización

2.1 INTRODUCCIÓN

- ¿Cómo conseguir un buen modelo?

- **Buen modelo: error en test bajo**



$$\text{cota}(e_{\text{test}}(f)) = \text{error}_{\text{train}} + \underbrace{\sqrt{\frac{h(\log \frac{2n}{h} + 1) - \log \frac{\eta}{4}}{n}}}_{\text{Confianza VC}}$$

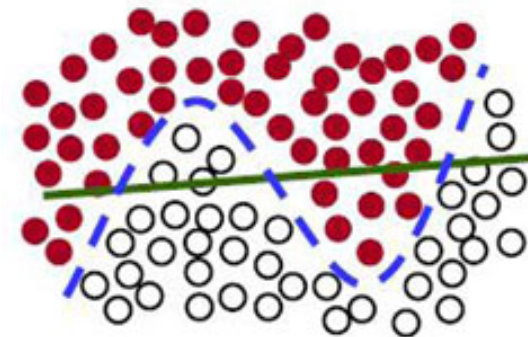
- Necesario:

- Error en entrenamiento bajo ➡ Evitar el *underfitting*
- Confianza VC baja ➡ Evitar el *overfitting* (sobreajuste)

- *Overfitting* (sobreajuste) vs. *Underfitting*

2.1 INTRODUCCIÓN

- ¿Cómo evitar el *underfitting*?
 - El modelo no es suficientemente complejo para el problema a resolver
 - El error de entrenamiento es demasiado alto
 - Ejemplo:
 - Modelo demasiado sencillo: 
 - Error de entrenamiento alto
 - Modelo con complejidad idónea: 
 - Error de entrenamiento bajo
- ¿Cómo saber cuál es la complejidad idónea a priori?



2.1 INTRODUCCIÓN

- ¿Cómo evitar el sobreajuste?

$$\text{cota}(e_{\text{test}}(f)) = \text{error}_{\text{train}} + \underbrace{\sqrt{\frac{h(\log \frac{2n}{h} + 1) - \log \frac{\eta}{4}}{n}}}_{\text{Confianza VC}}$$

- ¿Cómo aproximar el error de test al error de entrenamiento?
- Disminuir la confianza VC, 2 maneras:
 - Aumentar el número de patrones n
 - Disminuir la complejidad del modelo h

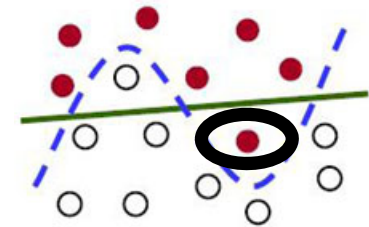
2.1 INTRODUCCIÓN

- ¿Cómo evitar el sobreajuste?

- Aumentar el número de patrones n

- Cuantos menos patrones se tiene, cada patrón se vuelve más y más importante

- Podría ser el único en representar una zona del espacio concreta, por ejemplo:



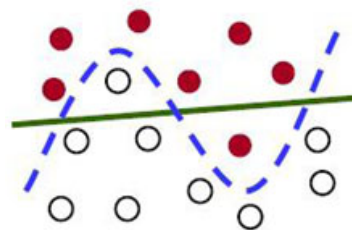
- Si estos patrones se escogiesen para formar parte del conjunto de test, estas zonas estarían sin representación en el conjunto de entrenamiento

- Comportamiento del modelo imprevisible en esta zona

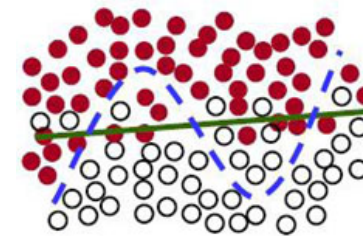
- **Entrenar con el número máximo de patrones posible, sobre todo si se tienen pocos patrones**

- Además, cuantos más patrones se tiene, más se disminuye el efecto del ruido:

- Redundancia en los patrones



Con pocos patrones, el ruido puede ser muy influyente

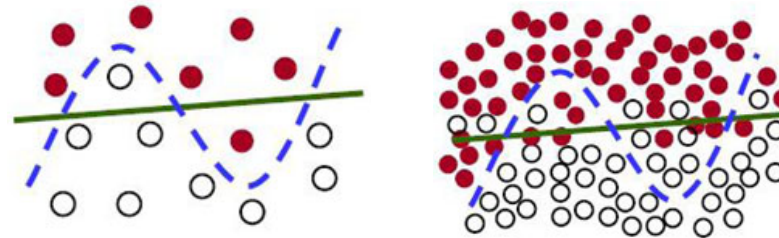






Con muchos patrones, se «diluye» el efecto del ruido

- En muchas ocasiones no es posible aumentar el número de patrones

2.1 INTRODUCCIÓN

- ¿Cómo evitar el sobreajuste?
 - Disminuir la complejidad del modelo h
 - Cuanto más sencillo es el modelo, menos capacidad tiene para aprender la forma compleja del ruido:

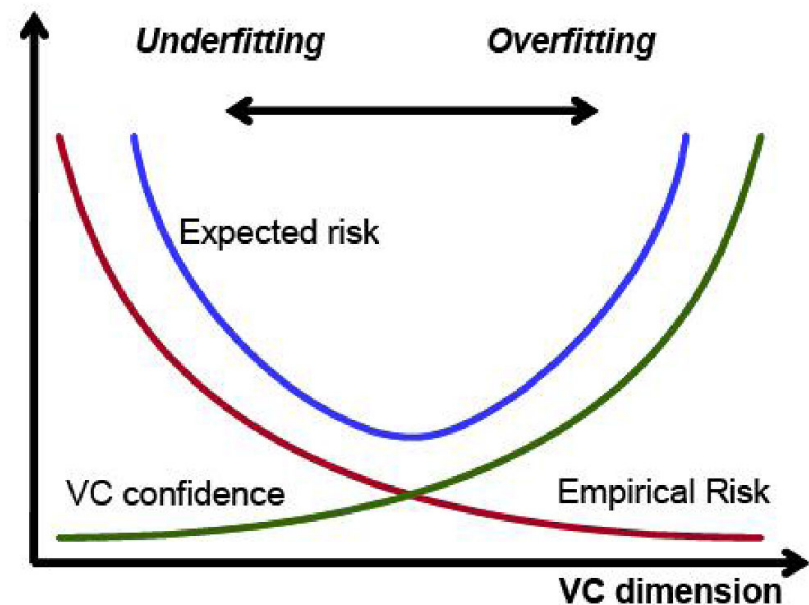


- Escoger un modelo sencillo en lugar de uno complejo
 - Por ejemplo:
 - Modelo complejo: 
 - Modelo sencillo: 
 - ¿Cómo saber cuál es la complejidad idónea a priori?
- o escoger un modelo complejo, pero parar el entrenamiento de forma prematura para no permitir «desarrollar» su complejidad
 - Por ejemplo, una RNA que utilice un conjunto de validación (parada temprana)
 - Por ejemplo:
 - RNA compleja que puede desarrollar una región de separación compleja: 
 - La misma RNA cuyo entrenamiento se ha parado de forma prematura: 

2.1 INTRODUCCIÓN

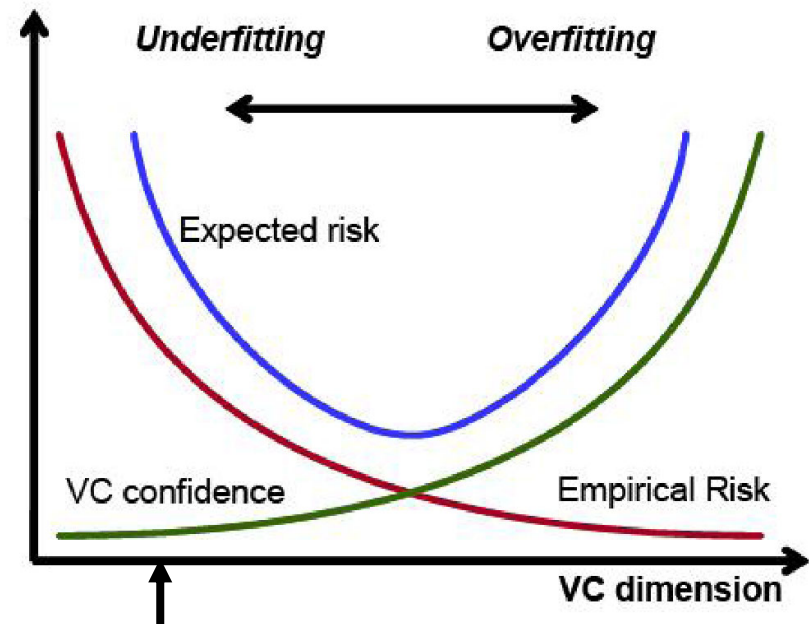
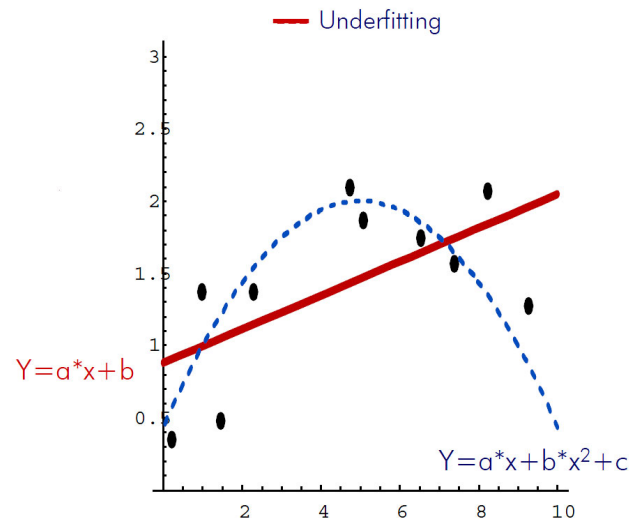
- Relación entre
 - Dimensión VC (complejidad)
 - Riesgo empírico (entrenamiento)
 - Riesgo esperado (test)
 - Confianza VC

$$\underbrace{\text{cota}(e_{\text{test}}(f))}_{\text{Riesgo esperado}} = \underbrace{error_{\text{train}}}_{\text{Riesgo empírico}} + \underbrace{\sqrt{\frac{h(\log \frac{2n}{h} + 1) - \log \frac{\eta}{4}}{n}}}_{\text{Confianza VC}}$$



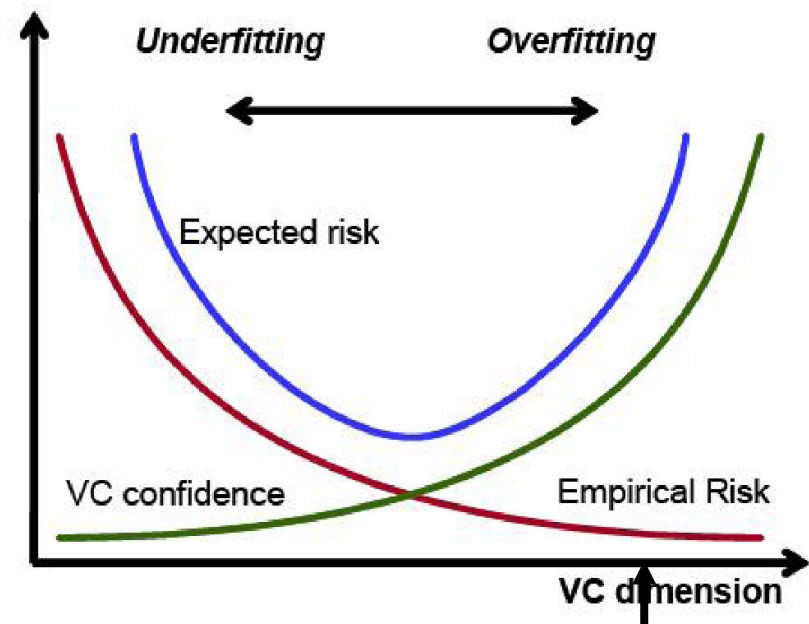
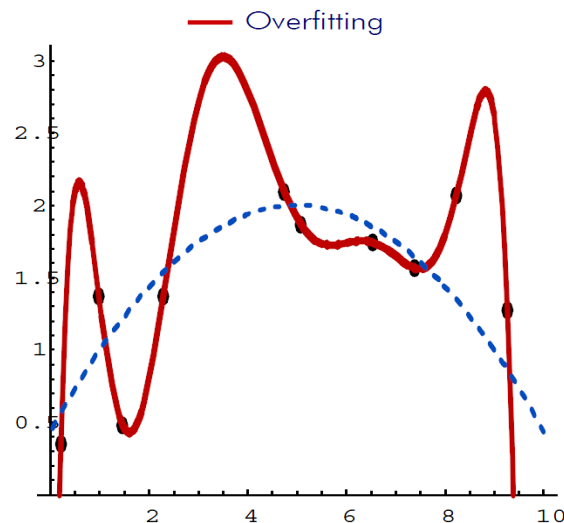
2.1 INTRODUCCIÓN

- El modelo óptimo se consigue buscando un equilibrio entre el riesgo sobre el conjunto de entrenamiento y la dimensión VC del modelo
 - Si se escoge un modelo demasiado sencillo (dimensión VC baja)
 - La confianza VC será muy baja (error de test similar al de entrenamiento)
 - Pero el error de entrenamiento (riesgo empírico) será muy alto
 - El riesgo esperado (suma) será muy alto
 - Ejemplo: polinomio grado 1



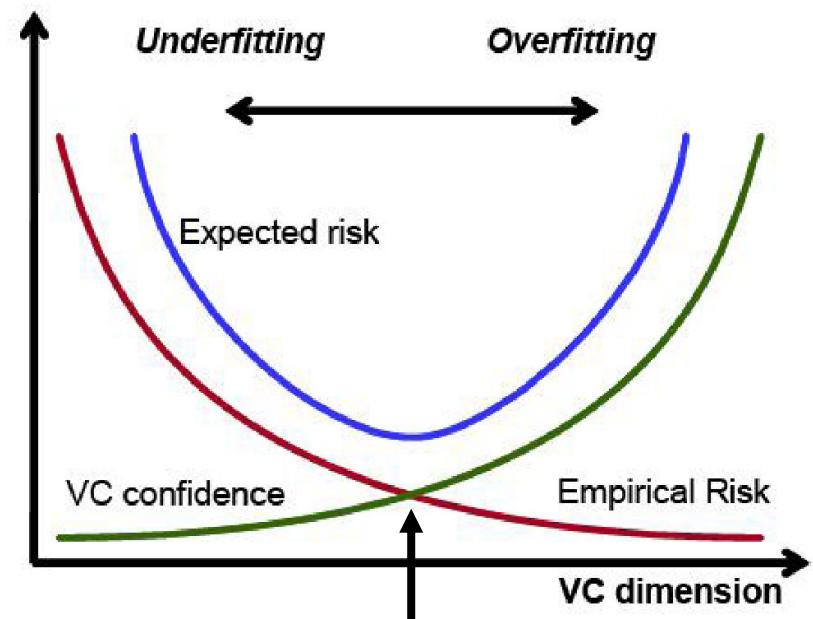
2.1 INTRODUCCIÓN

- El modelo óptimo se consigue buscando un equilibrio entre el riesgo sobre el conjunto de entrenamiento y la dimensión VC del modelo
 - Si se escoge un modelo demasiado complejo (dimensión VC alta)
 - El error de entrenamiento (riesgo empírico) será muy bajo
 - Pero la confianza VC será muy alta (error de test mucho mayor al de entr.)
 - El riesgo esperado (suma) será muy alto
 - Ejemplo: polinomio grado 8



2.1 INTRODUCCIÓN

- El modelo óptimo se consigue buscando un equilibrio entre el riesgo sobre el conjunto de entrenamiento y la dimensión VC del modelo
 - Buscar equilibrio entre un modelo demasiado sencillo y demasiado complejo



2.1 INTRODUCCIÓN

- ¿Cómo saber cuál es la complejidad idónea?
 - En teoría, se selecciona el modelo para el cual la cota anterior al riesgo esperado sea mínima
 - En la práctica, calcular la confianza VC no es viable en la mayoría de las situaciones
 - Existen expresiones explícitas que nos permiten establecer la dimensión VC para
 - Árboles de decisión
 - Redes de neuronas
 - Máquinas de Soporte Vectorial
 - etc.

2.1 INTRODUCCIÓN

- *Bias vs. Variance*
 - **Precisión** (sesgo) vs. **consistencia** (varianza) de los modelos entrenados por su algoritmo
 - En estadística, el sesgo (*bias*) puede definirse como la diferencia entre la población real y el valor estimado esperado
 - La varianza, por su parte, mide la dispersión o incertidumbre de estas estimaciones
 - A medida que aumenta la complejidad del modelo
 - Disminuye el sesgo
 - Aumenta la precisión
 - Aumenta la varianza de las estimaciones
 - Disminuye la consistencia

2.1 INTRODUCCIÓN

- *Bias vs. Variance*
 - Se quiere aproximar una función $g(x)$ a partir de un conjunto de datos $D=\{x_i, y_i\}$ donde
 - x_i es un vector con las entradas de la muestra i
 - y_i es un escalar resultado de aplicar la función desconocida $y=g(x)+\varepsilon$
 - ε : fuente de error
 - Se busca encontrar la función $f_w(x)$ que mejor se ajuste a la verdadera función $g(x)$
 - Pero sin aprender el ruido ε
 - Es decir, se tiene $y=g(x)+\varepsilon$ pero se quiere aprender únicamente $f_w(x)=g(x)$
 - Se quiere medir el error tanto para el conjunto D como para cualquier muestra no contenida en D
 - Capacidad de generalización
 - Posible métrica de error: ECM $(y - f_w(x))^2$,

2.1 INTRODUCCIÓN

- *Bias vs. Variance*

- Se puede demostrar que, en promedio, el error que se comete para cualquier valor de $x \notin D$ (error de generalización) se descompone en:

- $$\text{error}(x) = \text{MSE} [f_w(x)] = E [(y - f_w(x))^2] =$$
$$= \text{Bias} [f_w(x)]^2 + \text{Var} [f_w(x)] + \sigma^2$$

- donde

- **Bias**

- Error asociado a la simplicidad del modelo

- **Var**

- Variabilidad del modelo frente a distintos conjuntos de entrenamiento

- Pequeños cambios en el conjunto de entrenamiento pueden producir grandes errores

- σ^2 es una cota al error mínimo que se puede alcanzar

2.1 INTRODUCCIÓN

- *Bias vs. Variance*
 - El algoritmo de AA tiene dos tipos de errores:
 - Error Irreducible
 - Estos errores son naturales, y no se pueden eliminar estas incertidumbres
 - Por ejemplo, ruido debido a un conjunto de características incompletas, o una aleatoriedad inherente
 - Error Reducible
 - Se puede controlar y minimizar este error para mejorar la precisión del resultado
 - Varianza
 - Sesgo (*bias*)

2.1 INTRODUCCIÓN

- *Bias vs. Variance*

- Sesgo (o *bias*):

- Diferencia entre el valor medio predicho por el modelo y el valor medio real
 - Si la diferencia entre estas dos magnitudes es elevada, se tiene un modelo demasiado simple que no ha aprendido las relaciones relevantes entre las variables disponibles y la variable a predecir
 - A este fenómeno se le llama **underfitting** o subajuste
 - Los errores de entrenamiento y test serán altos
 - En general, un modelo más simple tendrá más sesgo
 - Modelos con sesgo bajo: árboles de decisión, k-NN, SVM
 - Modelos con sesgo alto: regresión lineal, regresión logística
 - Realizan más suposiciones sobre la forma de la función objetivo
 - Paramétricos

2.1 INTRODUCCIÓN

- *Bias vs. Variance*

- Varianza:

- El modelo se entrena con subconjunto de todos los datos posibles
 - Para subconjuntos distintos el modelo resultante sea distinto
 - La varianza mide diferencia entre los modelos resultantes

- Varianza baja:

- Cambios en el conjunto de entrenamiento provocan pequeños cambios en la estimación de la función objetivo
 - Modelos de baja varianza: regresión logística, regresión lineal, análisis discriminante lineal

2.1 INTRODUCCIÓN

- *Bias vs. Variance*

- Varianza:

- Varianza alta:

- Cambios en el conjunto de entrenamiento provocan grandes cambios en la estimación de la función objetivo
 - El modelo es demasiado complejo y presta atención en exceso a las peculiaridades del subconjunto empleado
 - Particulariza
 - El modelo está fuertemente influenciados por los detalles de los datos de entrenamiento
 - Estos modelos sufren de **overfitting** o sobreajuste
 - Cuando son expuestos a datos no vistos, no proporcionan predicciones correctas (error de test alto)
 - Modelos con una alta varianza: k-NN, árboles de decisión y SVM
 - Generalmente, los algoritmos de *Machine Learning* no paramétrico que tienen mucha flexibilidad tienen una gran varianza

2.1 INTRODUCCIÓN

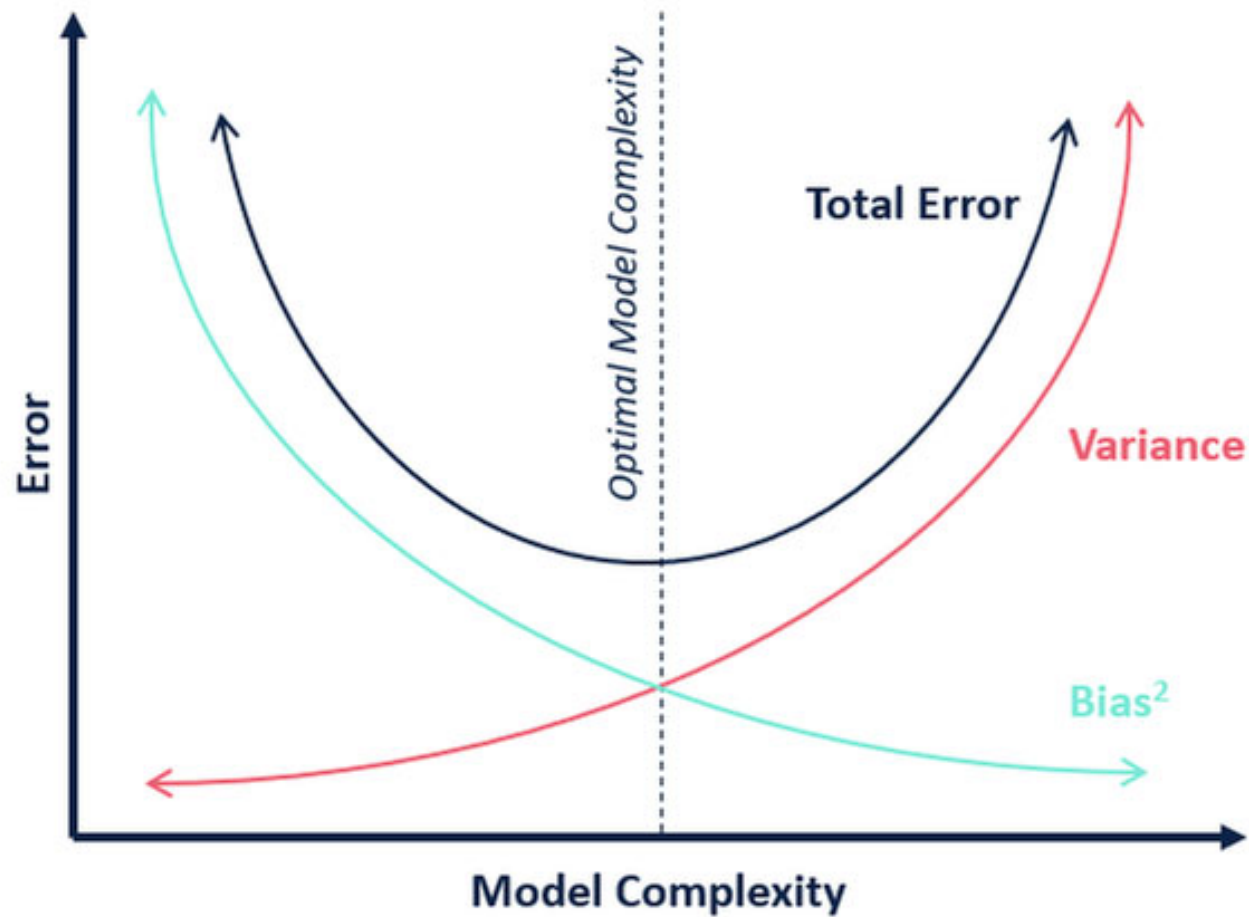
- *Bias vs. Variance*
 - Por tanto, el objetivo es buscar un algoritmo que proporcione mínimo sesgo y mínima varianza
 - Se puede reducir el sesgo y la varianza ya que son errores reducibles
 - Para ello, es necesario seleccionar un modelo que tenga la flexibilidad y complejidad adecuadas
 - Además, hay que utilizar datos adecuados para entrenar el modelo y reducir estos errores
 - En definitiva, lo que se busca es un modelo que:
 - que haya aprendido de los datos que se le han proporcionado (sesgo y error de entrenamiento bajos)
 - que sea capaz de generalizar ante nuevos datos (varianza y error de test bajos)
 - Es decir, que no presente *underfitting* ni *overfitting*

2.1 INTRODUCCIÓN

- *Bias vs. Variance*
 - Sin embargo, como regla general
 - Si se intenta disminuir el sesgo aumentando la complejidad del modelo, aumentará la varianza
 - Si se intenta minimizar la varianza disminuyendo la complejidad, aumentará el sesgo
 - Cuando se cambia un componente, afecta al otro
 - No se puede reducir ambos componentes a cero
 - Dicotomía del sesgo y la varianza
 - El reto de encontrar el punto adecuado de compromiso entre la complejidad del modelo y su generalidad
 - Necesario mantener el equilibrio entre sesgo y varianza

2.1 INTRODUCCIÓN

- *Bias vs. Variance*

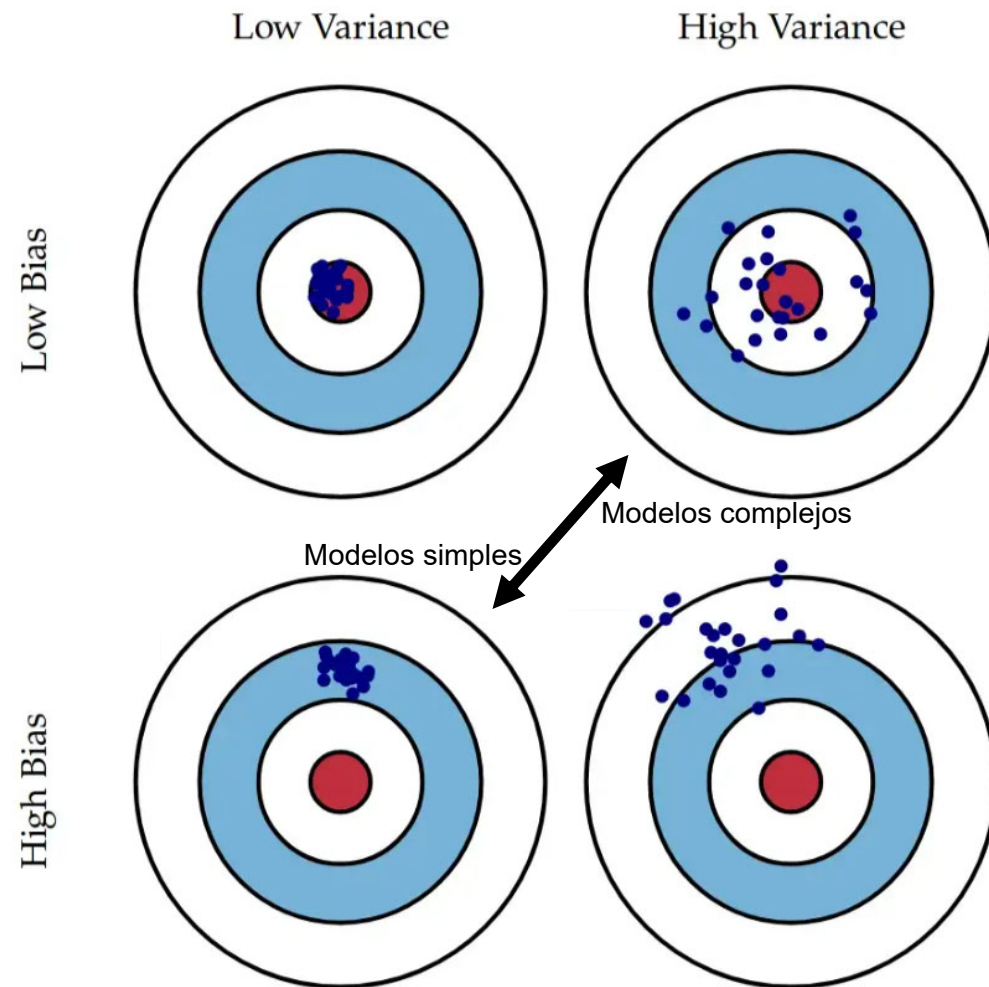


2.1 INTRODUCCIÓN

- *Bias vs. Variance*
 - Los algoritmos de baja varianza (alto sesgo) tienden a ser menos complejos, con una estructura subyacente simple o rígida.
 - Entrenan modelos que son consistentes, pero inexactos en promedio
 - Estos incluyen algoritmos paramétricos o lineales, como la regresión lineal o regresión logística
 - Los algoritmos de bajo bias (alta varianza) tienden a ser más complejos, con una estructura subyacente flexible.
 - Entrenan modelos que son precisos en promedio pero inconsistentes
 - Estos incluyen algoritmos no lineales o no paramétricos, como árboles de decisión y k-NN y SVM

2.1 INTRODUCCIÓN

- *Bias vs. Variance*
 - En general:

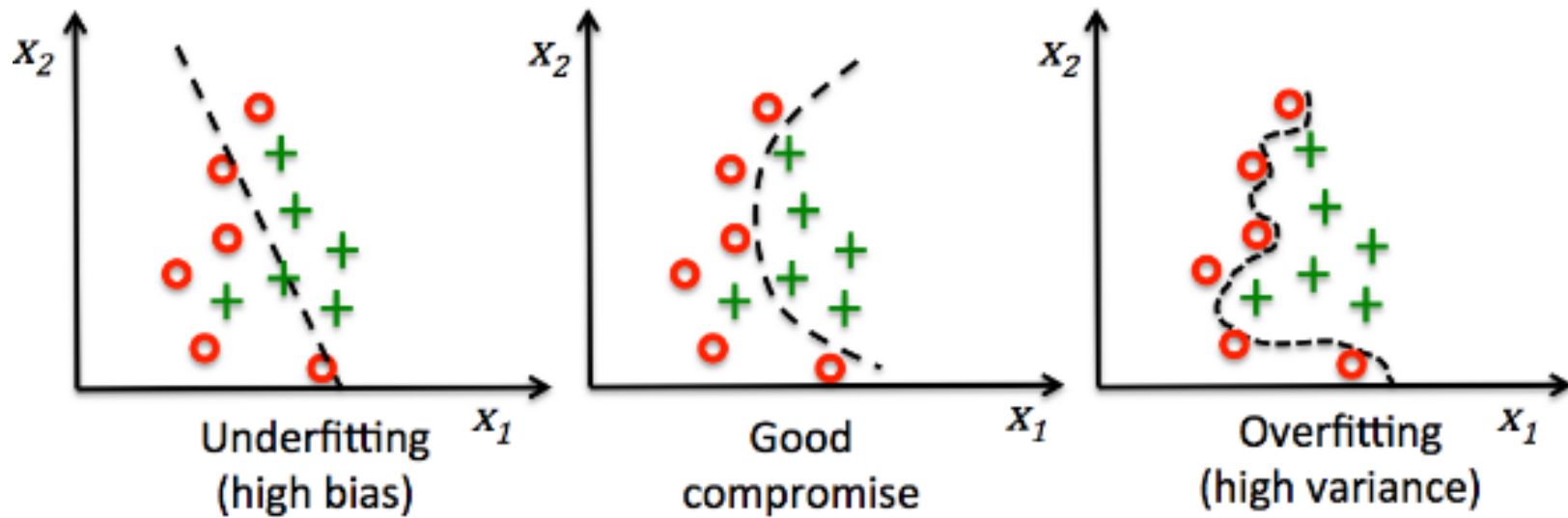


2.1 INTRODUCCIÓN

- *Bias vs. Variance*
 - Regularización:
 - Reducir la varianza a costa de cierto sesgo para un modelo escogido
 - Es decir, dado un modelo con una complejidad concreta, las técnicas de regularización impiden hacer uso de toda la complejidad
 - Por ejemplo, parando el entrenamiento antes de tiempo
 - Parada temprana
 - Por ejemplo, penalizando si sus parámetros toman valores muy elevados
 - Disminuir la varianza aumentando un poco el sesgo
 - En teoría, no es necesaria si la complejidad es la adecuada

2.1 INTRODUCCIÓN

- *Bias vs. Variance*



← Regularización

2.1 INTRODUCCIÓN

- Una vez elegido un modelo y dados unos datos para resolver un problema...
 - El conjunto de datos está formado por tuplas
 - $\langle \text{atributos}, \text{valor} \rangle$
 - Se desea un modelo o sistema que, ante nuevos valores de los atributos, sea capaz de devolver un valor consistente con el conjunto de datos
 - Los atributos pueden ser continuos (valores enteros o reales) o discretos (etiquetas)

...es necesario realizar el diseño experimental

2.1 INTRODUCCIÓN

- Objetivos del diseño experimental:
 - Realizar un aprendizaje (entrenamiento) eficaz del modelo elegido
 - Estimar el rendimiento real (error cometido) por el modelo obtenido
- Para lo cual se tiene: modelo + datos
- Pasos a seguir:
 1. Preprocesado de los datos
 2. Entrenamiento o ajuste del modelo
 3. Estimación del error real del modelo

2.1 INTRODUCCIÓN

- Preprocesado de datos:
 - ¿Cómo se pueden preparar los datos para que el proceso de aprendizaje sea más fácil y efectivo?
 - Varias tareas:
 - Transformación de datos (*data transformation*)
 - Limpieza de datos o *data cleaning*:
 - Proceso orientado a eliminar datos con ruido o incorrectos
 - Normalización
 - Construcción de nuevos atributos
 - Para facilitar el proceso de aprendizaje
 - Reducción de datos (*data reduction*)
 - Orientado a reducir el tamaño de los datos mediante la agregación y/o eliminación de características redundantes

2.1 INTRODUCCIÓN

- Preprocesado de datos: *Data Transformation*:
 - Se transforman unos datos en otros equivalentes para dejarlos listos para el proceso de aprendizaje:
 - Limpieza de datos
 - Normalización
 - Agregación
 - Generalización
 - Construcción de atributos

2.1 INTRODUCCIÓN

- Preprocesado de datos: Limpieza de datos:
 - Los datos del mundo real suelen presentarse en una forma incompleta, inconsistente y ruidosa
 - Habrá que limpiar los datos
 - Eliminando los datos que faltan
 - Suavizando el efecto del ruido
 - Eliminando datos fuera de rango
 - Corrigiendo inconsistencias

2.1 INTRODUCCIÓN

- Preprocesado de datos: Limpieza de datos:
 - Qué hacer ante los datos nulos
 - Antes de ponerse manos a la obra, es necesario analizar la razón de su existencia
 - ¿Es significativo que aparezcan datos nulos?
 - ¿Un dato nulo tiene algún significado?
 - Es posible que un dato nulo tenga significado, por ejemplo:
 - «Si la temperatura excede los X grados, es necesario conocer la presión»
 - En este caso, la base de datos tendrá las variables temperatura y presión
 - La variable presión tendrá datos nulos para aquellas instancias cuyos valores de temperatura sean inferiores a X

2.1 INTRODUCCIÓN

- Preprocesado de datos: Limpieza de datos:
 - Qué hacer ante los datos nulos
 - Existe una serie de maneras de enfrentarse a los datos nulos (1/2):
 - Ignorar la tupla
 - Si el dato que falta es el de la clase, o si existen nulos en varios de los atributos
 - Rellenar los datos de manera manual
 - Inabordable en la minería de datos
 - Usar una constante global para rellenar los datos
 - Cambiar cada dato ausente por *unknown*

2.1 INTRODUCCIÓN

- Preprocesado de datos: Limpieza de datos:
 - Qué hacer ante los datos nulos
 - Existe una serie de maneras de enfrentarse a los datos nulos (2/2):
 - Usar la media del atributo para sustitución
 - Usar la media del atributo obtenida con todos los ejemplares que pertenecen a la misma clase que el ejemplo a modificar
 - Utilizar técnicas de minería de datos para predecir el valor más probable en cada caso
 - Por ejemplo, un árbol de decisión, un SVM, etc.

2.1 INTRODUCCIÓN

- Preprocesado de datos: Limpieza de datos:
 - Qué hacer ante los datos nulos
 - Existe una serie de maneras de enfrentarse a los datos nulos (2/2):
 - Usar la media del atributo para sustitución
 - Usar la media del atributo obtenida con todos los ejemplares que pertenecen a la misma clase que el ejemplar a modificar
 - **Utilizar técnicas de minería de datos para predecir el valor más probable en cada caso**
 - Por ejemplo, un árbol de decisión, un SVM, etc.

2.1 INTRODUCCIÓN

- Preprocesado de datos: Limpieza de datos:
 - Qué hacer ante los datos nulos

- Ejemplo:

Instancia	Atrib. 1	Atrib. 2	Atrib. 3	Atrib. 4	Sal. Des.
1	X_{11}	X_{12}	X_{13}	X_{14}	Y_1
2	X_{21}	X_{22}	NULL	X_{24}	Y_2
3	X_{31}	X_{32}	X_{33}	X_{34}	Y_3
4	X_{41}	X_{42}	X_{43}	X_{44}	Y_4
5	X_{51}	X_{52}	X_{53}	X_{54}	Y_5

- Realizar nuevo modelo con:



Instancia	Atrib. 1	Atrib. 2	Atrib. 4		Sal. Des.
1	X_{11}	X_{12}	X_{14}	Y_1	X_{13}
3	X_{31}	X_{32}	X_{34}	Y_3	X_{33}
4	X_{41}	X_{42}	X_{44}	Y_4	X_{43}
5	X_{51}	X_{52}	X_{54}	Y_5	X_{53}

- Aplicar el modelo con los datos de la instancia 2 para calcular X_{23}

2.1 INTRODUCCIÓN

- Preprocesado de datos: Limpieza de datos:
 - Qué hacer ante los datos nulos
 - Utilizar técnicas de minería de datos para predecir el valor más probable en cada caso
 - Cuestiones (1/2)
 - ¿Introducir la salida deseada del problema original como nuevo atributo?
 - Si el original es problema de regresión
 - La salida deseada es un valor numérico
 - Si es problema de clasificación:
 - No introducir ese atributo
 - 2 opciones:
 - Realizar regresión sólo con las instancias de esa clase
 - Realizar regresión con todas las instancias

2.1 INTRODUCCIÓN

- Preprocesado de datos: Limpieza de datos:
 - Qué hacer ante los datos nulos
 - Utilizar técnicas de minería de datos para predecir el valor más probable en cada caso
 - Cuestiones (2/2)
 - ¿Regresión lineal, cuadrática...?
 - Complejidad del nuevo modelo

2.1 INTRODUCCIÓN

- Preprocesado de datos: Limpieza de datos:

- Datos con ruido:

- El ruido en los datos se ve, tradicionalmente, como un error en las variables que se están midiendo
 - Este error se modela como una variable aleatoria
 - Dada la variable X , se aproxima su valor con la expresión

$$\hat{X} = X + e$$

- ¿Cómo podemos suavizar los valores de \hat{X} de tal forma que minimicemos el error e ?
 - *Binning*
 - *Clustering*
 - Regresión

2.1 INTRODUCCIÓN

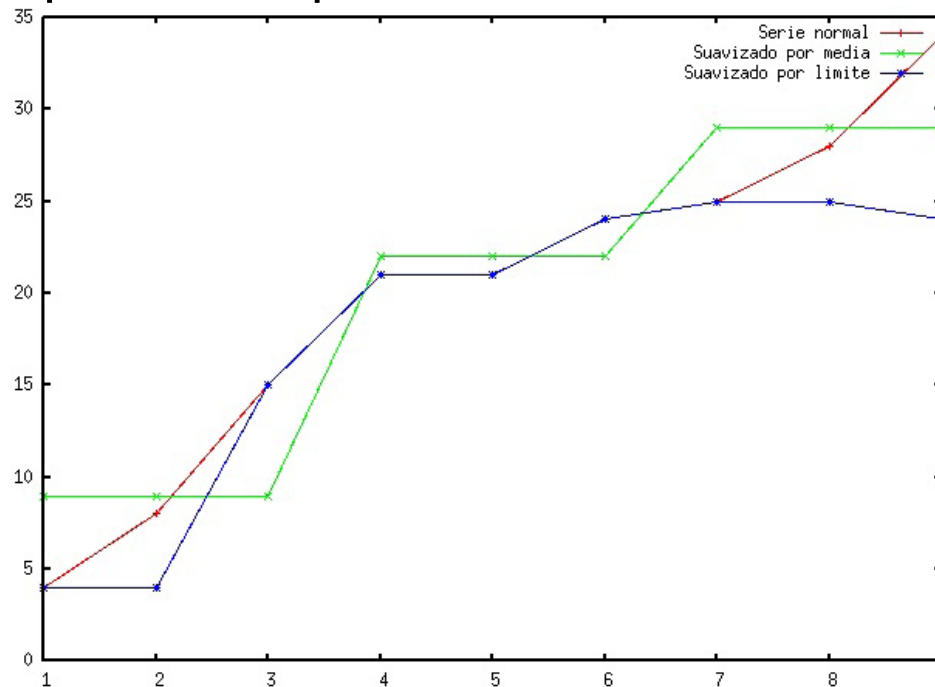
- Preprocesado de datos: Limpieza de datos:
 - Datos con ruido: *Binning*:
 - Una serie de valores ordenados se agrupan en porciones y luego se suaviza cada porción
 - De esta forma, lo que se hace es un tratamiento local del ruido ya que se actúa de manera individual en cada porción
 - Por ejemplo: 4, 8, 15, 21, 21, 24, 25, 28, 34
 - Se dividen en tres *bins* de la misma longitud:
 - 4, 8, 15
 - 21, 21, 24
 - 25, 28, 34

2.1 INTRODUCCIÓN

- Preprocesado de datos: Limpieza de datos:
 - Datos con ruido: *Binning*:
 - Por ejemplo: 4, 8, 15, 21, 21, 24, 25, 28, 34 (cont.)
 - Suavizado por media: Si se sustituye cada valor por la media, se está suavizando mediante medias, y queda
 - 9, 9, 9
 - 22, 22, 22
 - 29, 29, 29
 - Suavizado por límite: Si, en lugar de ello, se sustituye cada valor por el valor del extremo más cercano de su *bin*, queda
 - 4, 4, 15
 - 21, 21, 24
 - 25, 25, 34

2.1 INTRODUCCIÓN

- Preprocesado de datos: Limpieza de datos:
 - Datos con ruido: *Binning*:
 - Por ejemplo: 4, 8, 15, 21, 21, 24, 25, 28, 34 (cont.)
 - Si se representa, se puede ver cómo las curvas se suavizan:



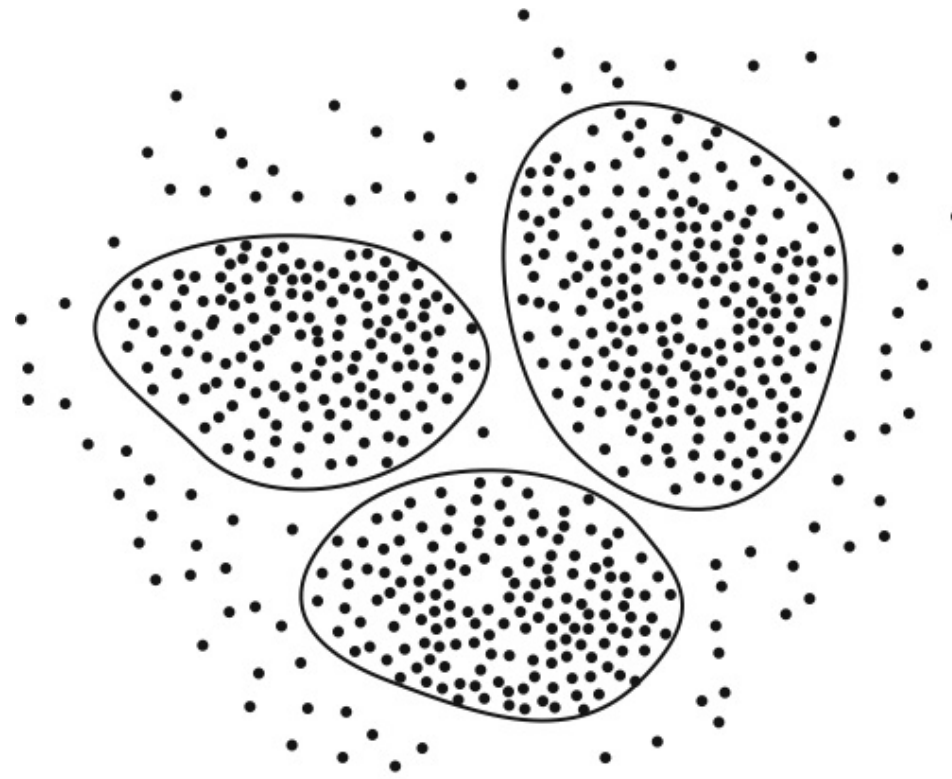
- El efecto del binning está en relación a la long. de las porciones

2.1 INTRODUCCIÓN

- Preprocesado de datos: Limpieza de datos:
 - Datos con ruido: *Binning*:
 - Por ejemplo: 4, 8, 15, 21, 21, 24, 25, 28, 34 (cont.)
 - Se puede hacer división de la misma anchura
 - Es decir, el mismo rango de valores por cada intervalo
 - Por ejemplo, dividir en bins de anchura 10:
 - 4-14, 15-24, 25-34
 - En este caso, los valores se repartirían:
 - 4, 8
 - 15, 21, 21, 24
 - 25, 28, 34
 - Se suaviza cada bin igual que antes

2.1 INTRODUCCIÓN

- Preprocesado de datos: Limpieza de datos:
 - Datos con ruido: *Clustering*:



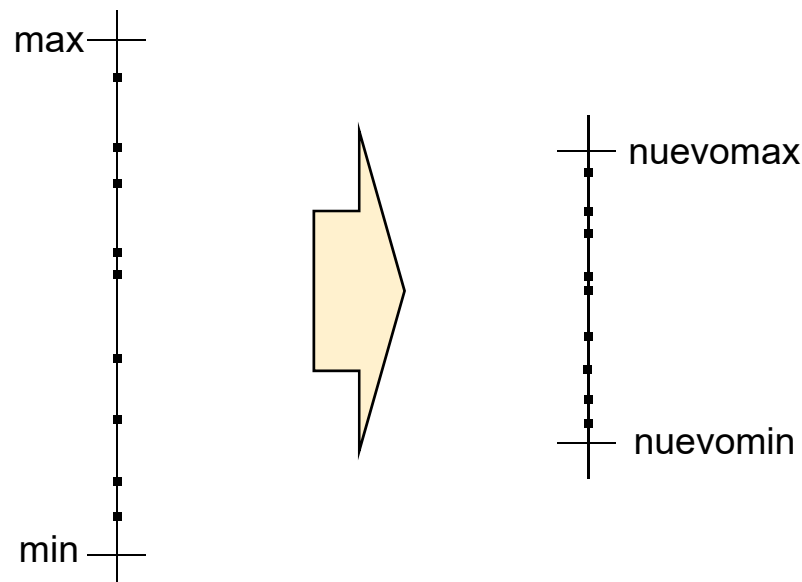
- Aquellos valores que caen fuera de los clusters pueden considerarse valores fuera de rango

2.1 INTRODUCCIÓN

- Preprocesado de datos: Limpieza de datos:
 - Datos con ruido: Regresión lineal:
 - Dado un conjunto de tuplas representado por dos variables, se halla la línea recta que mejor se ajusta a esos datos
 - Se minimiza un error basado en el cuadrado de diferencias cuadráticas entre puntos de la recta y reales
 - Si el número de variables es mayor que dos, se tiene **regresión lineal múltiple**
 - En realidad se busca una expresión matemática con la que reproducir los datos y, de esta manera, eliminar el error en la medida de lo posible
 - Sustituir los datos por el valor en esta expresión

2.1 INTRODUCCIÓN

- Preprocesado de datos: *Data Transformation*:
 - Normalización: Las formas más comunes son:
 - Normalización min/max (*normalization*):



2.1 INTRODUCCIÓN

- Preprocesado de datos: *Data Transformation*:
 - Normalización: Las formas más comunes son:

- Normalización min/max (*normalization*):

- Si los valores están acotados y se distribuyen en ese rango:

$$v' = \frac{v - \min}{\max - \min} (\text{nuevomax} - \text{nuevomin}) + \text{nuevomin}$$

- Generalmente entre 0 y 1, con lo que la ecuación quedaría

$$v' = \frac{v - \min}{\max - \min}$$

- también común entre -1 y 1

2.1 INTRODUCCIÓN

- Preprocesado de datos: *Data Transformation*:
 - Normalización: Las formas más comunes son:

- Normalización mediante escalado decimal
 - Se mueve el punto decimal en los valores del atributo
 - Las posiciones dependen del máximo en valor absoluto tal que

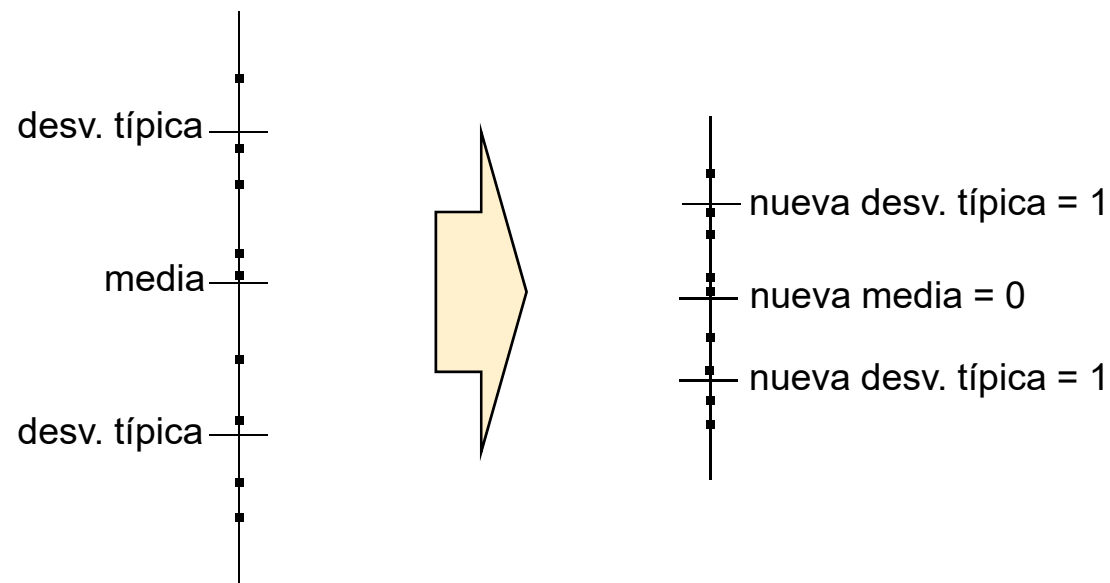
$$v' = \frac{v}{10^j}$$

en donde j es el entero más pequeño tal que $\text{Max}(|v'|) < 1$

- Por ejemplo: 15.3, 8.34, 732.83 ➡ 0.0153, 0.00834, 0.73283

2.1 INTRODUCCIÓN

- Preprocesado de datos: *Data Transformation*:
 - Normalización: Las formas más comunes son:
 - Normalización de media cero (*z normalization, standardization*)



2.1 INTRODUCCIÓN

- Preprocesado de datos: *Data Transformation*:

- Normalización: Las formas más comunes son:

- Normalización de media cero (*z normalisation, standardization*)

$$v' = \frac{v - \mu}{\sigma}$$

- Si los valores son el resultado de una medición y algún valor puede salirse y ser muy alto o muy bajo
 - Ejemplo: temperatura, altura, etc.
- En estos casos, si se normalizase entre máximo y mínimo, la existencia de un valor extremo fijaría ese valor como el máximo (o mínimo) y haría que, al normalizar, el resto de valores oscilase en un rango muy pequeño
 - Se podrían tomar como ruido

2.1 INTRODUCCIÓN

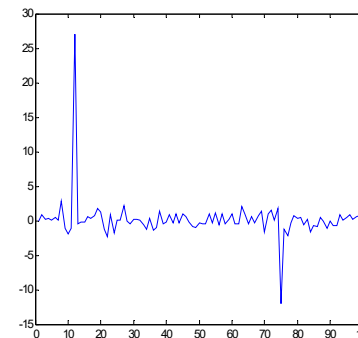
- Preprocesado de datos: *Data Transformation*:

- Normalización: Las formas más comunes son:

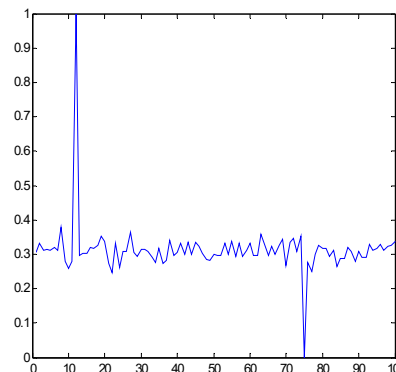
- Normalización de media cero (*z normalisation standardization*)

- Ejemplo:

- Valores de un atributo:



- Si se normalizase entre 0 y 1, los valores pasarían a ser los siguientes:



La mayoría de los valores oscilarían entre 0.28 y 0.34

2.1 INTRODUCCIÓN

- Preprocesado de datos: *Data Transformation*:
 - Agregación:
 - Agregar valores de atributos
 - Cuando es útil tener la suma de valores de otro/s atributo/s o instancias
 - Se crea un nuevo atributo con esta suma
 - Ídem con otras operaciones
 - Por ejemplo, agregar ventas diarias en semanales y/o mensuales
 - Si se necesita saber las ventas mensuales, tener sólo las diarias añade un nivel de complejidad innecesario

2.1 INTRODUCCIÓN

- Preprocesado de datos: *Data Transformation*:
 - Generalización:
 - Mediante jerarquías de conceptos, se sustituyen valores (categóricos o numéricos) por otros más abstractos
 - Por ejemplo: Calle por ciudad
 - ¿Se necesita el atributo calle o sólo se necesita saber la ciudad?
 - Si sólo se necesita saber la ciudad, calle añade un nivel de complejidad adicional innecesario
 - Por ejemplo: 40 por mediana edad

2.1 INTRODUCCIÓN

- Preprocesado de datos: *Data Transformation*:
 - Construcción de atributos:
 - Atributos nuevos, ¿para qué, y cómo?
 - Añadir atributos puede ayudar a poner a los algoritmos la tarea de análisis un poco más fácil
 - Si se puede combinar atributos mediante alguna expresión interesante, se consigue que el algoritmo no tenga que descubrir dicha expresión
 - Por ejemplo, si el atributo área es interesante para el algoritmo, se puede crear a partir de la altura y la anchura
 - Igual que en agregación y generalización, se puede reducir la complejidad del sistema

2.1 INTRODUCCIÓN

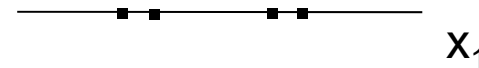
- Preprocesado de datos: Reducción de datos:
 - Cuando el conjunto de datos es realmente grande, es posible que las técnicas disponibles no sean adecuadas, hasta tal punto que sea totalmente imposible realizar el proceso de aprendizaje
 - El espacio de búsqueda crece demasiado

2.1 INTRODUCCIÓN

- Preprocesado de datos: Reducción de datos:
 - Formas de reducir datos:
 - Reducción de la numerosidad
 - Por ejemplo, del número de tuplas
 - Discretización de atributos y generación de jerarquías de conceptos
 - Agregación, generalización, construcción de atributos
 - Vistos anteriormente
 - Reducción de dimensiones
 - “La maldición de las dimensiones”

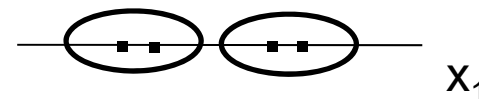
2.1 INTRODUCCIÓN

- Preprocesado de datos: Reducción de datos:
 - La maldición de las dimensiones:
 - Es un gran problema en minería de datos
 - Si los datos se ubican en un espacio de alta dimensionalidad (muchos atributos), la cantidad de valores crece exponencialmente
 - Además, es muy posible que muchos de estos atributos contengan información irrelevante, o poco significativa
 - Esto puede hacer que tuplas que deberían estar muy próximas puedan aparecer muy separadas, y viceversa
 - Ejemplo: Clasificación de calidad de un producto (alta/media/baja)
 - Un solo atributo (importante): Precio



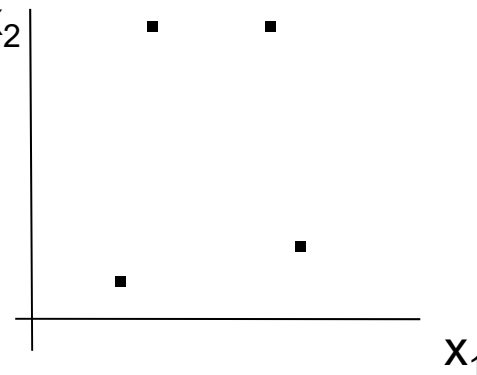
2.1 INTRODUCCIÓN

- Preprocesado de datos: Reducción de datos:
 - La maldición de las dimensiones:
 - Es un gran problema en minería de datos
 - Si los datos se ubican en un espacio de alta dimensionalidad (muchos atributos), la cantidad de valores crece exponencialmente
 - Además, es muy posible que muchos de estos atributos contengan información irrelevante, o poco significativa
 - Esto puede hacer que tuplas que deberían estar muy próximas puedan aparecer muy separadas, y viceversa
 - Ejemplo: Clasificación de calidad de un producto (alta/media/baja)
 - Un solo atributo (importante): Precio
 - Instancias próximas (similares)



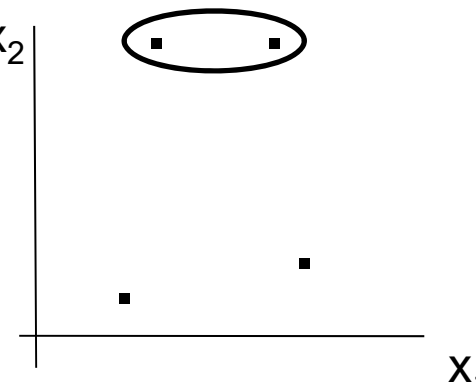
2.1 INTRODUCCIÓN

- Preprocesado de datos: Reducción de datos:
 - La maldición de las dimensiones:
 - Es un gran problema en minería de datos
 - Si los datos se ubican en un espacio de alta dimensionalidad (muchos atributos), la cantidad de valores crece exponencialmente
 - Además, es muy posible que muchos de estos atributos contengan información irrelevante, o poco significativa
 - Esto puede hacer que tuplas que deberían estar muy próximas puedan aparecer muy separadas, y viceversa
 - Ejemplo:
 - Un solo atributo (importante): Precio X_2
 - Segundo atributo (irrelevante):
 - Fecha de compra



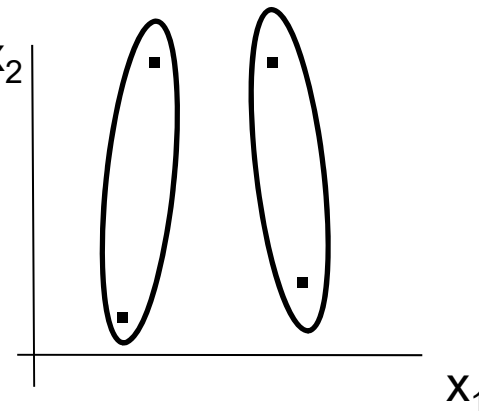
2.1 INTRODUCCIÓN

- Preprocesado de datos: Reducción de datos:
 - La maldición de las dimensiones:
 - Es un gran problema en minería de datos
 - Si los datos se ubican en un espacio de alta dimensionalidad (muchos atributos), la cantidad de valores crece exponencialmente
 - Además, es muy posible que muchos de estos atributos contengan información irrelevante, o poco significativa
 - Esto puede hacer que tuplas que deberían estar muy próximas puedan aparecer muy separadas, y viceversa
 - Ejemplo:
 - Un solo atributo (importante): Precio X_2
 - Segundo atributo (irrelevante):
 - Fecha de compra
 - Instancias próximas (similares)
 - (mismo día de compra)



2.1 INTRODUCCIÓN

- Preprocesado de datos: Reducción de datos:
 - La maldición de las dimensiones:
 - Es un gran problema en minería de datos
 - Si los datos se ubican en un espacio de alta dimensionalidad (muchos atributos), la cantidad de valores crece exponencialmente
 - Además, es muy posible que muchos de estos atributos contengan información irrelevante, o poco significativa
 - Esto puede hacer que tuplas que deberían estar muy próximas puedan aparecer muy separadas, y viceversa
 - Ejemplo:
 - Un solo atributo (importante): Precio X_2
 - Segundo atributo (irrelevante):
 - Fecha de compra
 - Instancias separadas (distintas)
 - (días con mucha separación)

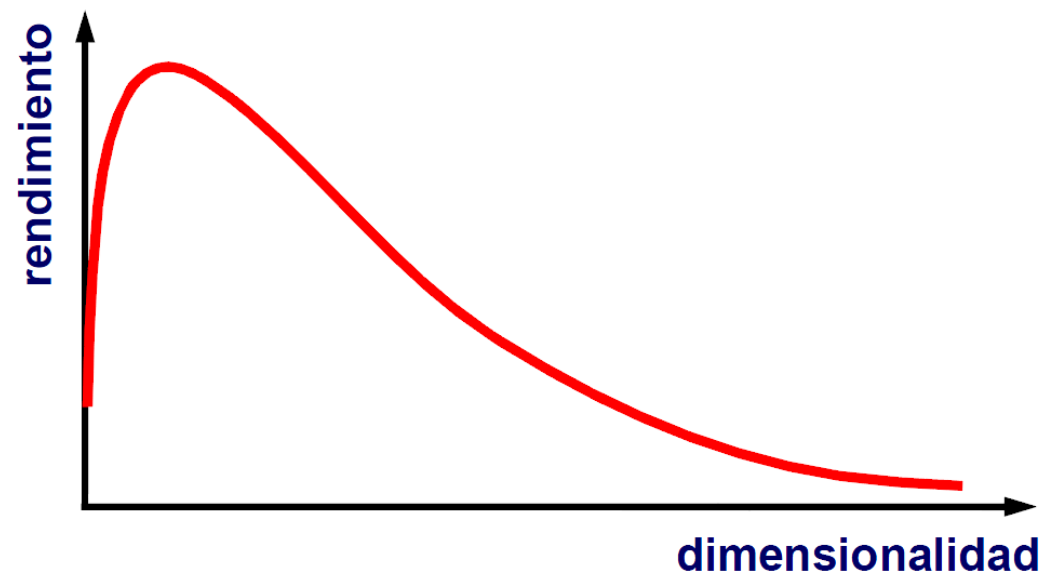


2.1 INTRODUCCIÓN

- Preprocesado de datos: Reducción de datos:
 - La maldición de las dimensiones:
 - Además, en general, cuanto mayor sea la dimensionalidad de entrada del sistema de aprendizaje, mayor es el número de ejemplos necesarios para obtener un buen modelo
 - La reducción de la dimensionalidad del espacio de entrada implica una disminución del número de atributos

2.1 INTRODUCCIÓN

- Preprocesado de datos: Reducción de datos:
 - La maldición de las dimensiones:
 - Ventajas de tener menos atributos:
 - Reduce la complejidad y el tiempo de ejecución
 - Mejora la capacidad de generalización del sistema
 - En general, un modelo que use menos atributos generaliza mejor
 - Un modelo que use muchos atributos particulariza más



2.1 INTRODUCCIÓN

- Preprocesado de datos: Reducción de datos:
 - La maldición de las dimensiones:
 - Necesario reducir el número de dimensiones:
 - Proyectar los datos en un espacio de menos dimensiones.
 - Técnicas:
 - *Principal Component Analysis (PCA)*
 - *Fisher's Linear Discriminant*
 - *Multi-dimensional Scaling*
 - *Independent Component Analysis (ICA)*

2.1 INTRODUCCIÓN

- Preprocesado de datos: Reducción de datos:
 - Análisis de Componentes Principales (PCA)
 - Es la técnica más utilizada en reducción de dimensionalidad
 - Las bases de datos suelen tener una gran redundancia
 - p variables o atributos altamente correlacionados
 - Tienen información común
 - Deseable eliminar redundancia
 - Para ello, se puede transformar el conjunto original de p variables en otro conjunto de nuevas variables **incorreladas** entre sí (que no tenga repetición o redundancia en la información) llamado conjunto de componentes principales.

2.1 INTRODUCCIÓN

- Preprocesado de datos: Reducción de datos:
 - Análisis de Componentes Principales (PCA)
 - Las nuevas variables son **combinaciones lineales** de las anteriores y se van construyendo según el orden de importancia en cuanto a la variabilidad total que recogen de la muestra.
 - El concepto de mayor información se relaciona con el de mayor variabilidad o varianza.
 - Cuanto mayor sea la variabilidad de los datos (varianza) se considera que existe mayor información, lo cual está relacionado con el concepto de entropía.
 - Por tanto, las nuevas variables están ordenadas por la información que contienen
 - Técnica muy utilizada, además de en AA, como análisis de datos

2.1 INTRODUCCIÓN

- Preprocesado de datos: Reducción de datos:
 - Análisis de Componentes Principales (PCA)
 - De modo ideal, se buscan $m < p$ variables que sean combinaciones lineales de las p originales y que estén **incorreladas**, recogiendo la mayor parte de la información o variabilidad de los datos.
 - Para hallarlas, se realiza el cálculo de la descomposición de autovalores de la matriz de covarianza
 - Si las variables originales están incorreladas de partida, entonces no tiene sentido realizar un análisis de componentes principales

2.1 INTRODUCCIÓN

- Preprocesado de datos: Reducción de datos:
 - Análisis de Componentes Principales (PCA)
 - Además, la suma de las varianzas de las variables originales y la suma de las varianzas de las componentes son iguales.
 - Esto permite hablar del porcentaje de varianza total que recoge un componente principal
 - De igual manera, se puede hablar de la varianza que recogen los primeros m componentes
 - Los que tienen más información
 - En la práctica, se tomarán sólo los primeros componentes
 - ¿Cuántos? Depende de cuánta información se quiera

2.1 INTRODUCCIÓN

- Preprocesado de datos: Reducción de datos:
 - Análisis de Componentes Principales (PCA)
 - Ejemplo: Base de datos de precios de casas en Boston:
 - Base de datos con 506 patrones
 - 13 atributos o variables:
 - 1. per capita crime rate by town
 - 2. proportion of residential land zoned for lots over 25,000 sq.ft.
 - 3. proportion of non-retail business acres per town
 - 4. 1 if tract bounds Charles river, 0 otherwise
 - 5. nitric oxides concentration (parts per 10 million)
 - 6. average number of rooms per dwelling
 - 7. proportion of owner-occupied units built prior to 1940
 - 8. weighted distances to five Boston employment centres
 - 9. index of accessibility to radial highways
 - 10. full-value property-tax rate per \$10,000

2.1 INTRODUCCIÓN

- Preprocesado de datos: Reducción de datos:
 - Análisis de Componentes Principales (PCA)
 - Ejemplo: Base de datos de precios de casas en Boston:
 - Base de datos con 506 patrones
 - 13 atributos o variables:
 - 11. pupil-teacher ratio by town
 - 12. $1000(B_k - 0.63)^2$, where B_k is the proportion of blacks by town
 - 13. % lower status of the population
 - This data is available from the UCI Machine Learning Repository:
<http://mllearn.ics.uci.edu/MLRepository.html>
 - Murphy, P.M., Aha, D.W. (1994). UCI Repository of machine learning databases [<http://www.ics.uci.edu/~mllearn/MLRepository.html>]. Irvine, CA: University of California, Department of Information and Computer Science.

2.1 INTRODUCCIÓN

- Preprocesado de datos: Reducción de datos:
 - Análisis de Componentes Principales (PCA)
 - Ejemplo: Base de datos de medidas corporales:
 - Base de datos con 252 patrones
 - 13 entradas:

1. Edad (años)	7. Cadera (cm)
2. Peso (libras)	8. Muslo (cm)
3. Altura (pulgadas)	9. Rodilla (cm)
4. Cuello (cm)	10. Tobillo (cm)
5. Pecho (cm)	11. Bíceps (cm)
6. Abdomen (cm)	12. Antebrazo (cm)
	13. Muñeca (cm)
 - Salida deseada:
 - Porcentaje de grasa corporal que tiene un individuo

2.1 INTRODUCCIÓN

- Preprocesado de datos: Reducción de datos:
 - Análisis de Componentes Principales (PCA)
 - Ejemplos:
 - Número de componentes para un determinado porcentaje de varianza original (información)

		<i>House</i>	<i>BodyFat</i>
Porcentaje de varianza deseada	50 %	1	1
	95 %	2	1
	99 %	2	1
	99.9 %	3	3
	99.99 %	5	5
	99.999 %	9	12
	100 %	13	13

2.1 INTRODUCCIÓN

- Preprocesado de datos: Reducción de datos:
 - Análisis de Componentes Independientes (ICA)
 - Técnica que puede identificar y extraer fuentes estadísticamente independientes que subyacen de los canales simples de captura de datos.
 - ICA permite la descomposición de la señal medida en distintos sitios como una mezcla lineal de fuentes de señales que son, según las estadísticas, máximamente **independientes**.
 - Se supone que la señal que se tiene es la mezcla lineal de distintas fuentes
 - Mezcla es el proceso que transforma unas señales, las fuentes, en otras, las observaciones.
 - ICA permite obtener las señales originales (fuentes)

2.1 INTRODUCCIÓN

- Preprocesado de datos: Reducción de datos:
 - Análisis de Componentes Independientes (ICA)
 - Pasos (similar a PCA):
 - Cálculo de los componentes independientes
 - Son combinaciones lineales de las variables originales
 - Son variables estadísticamente independientes (ejes no ortogonales)
 - PCA: ejes ortogonales
 - Eliminación de los componentes menos relevantes

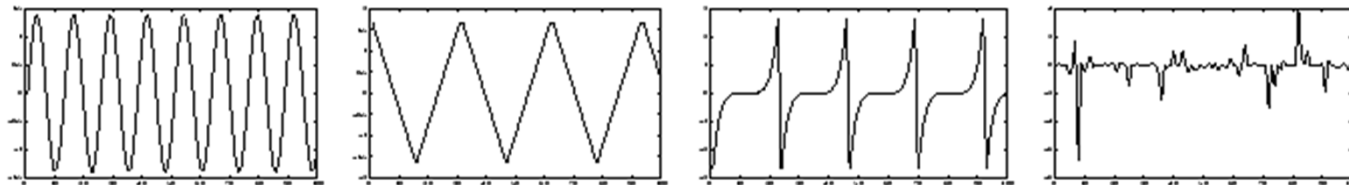
2.1 INTRODUCCIÓN

- Preprocesado de datos: Reducción de datos:
 - Análisis de Componentes Independientes (ICA)
 - Por ejemplo, en señales de electrocardiogramas (ECG)
 - Se puede esperar que no haya dependencia entre las fuentes de la señal ECG y ruidos contaminantes como el de línea de base (50Hz) o artefactos de movimiento (muscular)
 - ICA debería de separar todas estas fuentes: artefactos o ruidos de señales útiles
 - ICA es una técnica para la separación ciega de fuentes (*Blind Source Separation*, BSS)
 - BSS hace referencia al hecho de que las señales en el mundo físico aparecen contaminadas o “mezcladas” con otras de un modo desconocido, lo que dificulta la recuperación de las fuentes que en general son también desconocidas.
 - El objetivo del ICA es lograr invertir todo el proceso de la mezcla y recuperar las fuentes

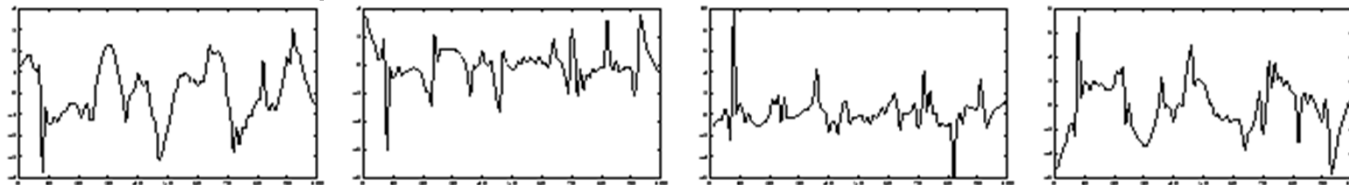
2.1 INTRODUCCIÓN

- Preprocesado de datos: Reducción de datos:
 - Análisis de Componentes Independientes (ICA)
 - Ejemplo:

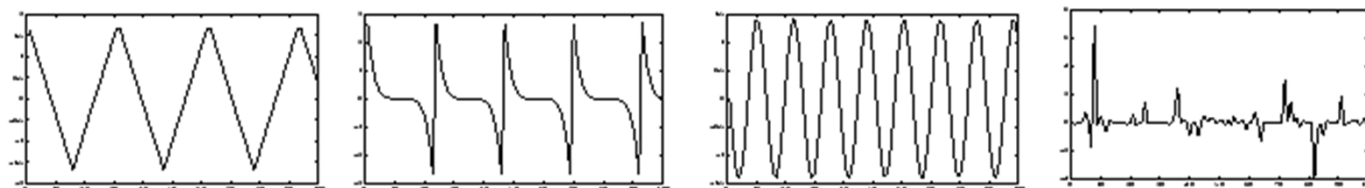
- 4 fuentes (independientes) (desconocidas):



- Las señales que se tienen son mezclas lineales de las anteriores:



- El análisis de esas señales permite estimar las fuentes, con un factor multiplicativo:



2.1 INTRODUCCIÓN

- Aprendizaje supervisado
 - El conjunto de datos está formado por tuplas
 - <atributos, valor>
 - **Conjunto de entrenamiento**
 - Se desea un modelo o sistema que, ante nuevos valores de los atributos, sea capaz de devolver un valor consistente con el conjunto de datos
 - **Entrenar un modelo**
 - **Capacidad de generalización**
 - Ser aplicado a patrones nuevos

2.1 INTRODUCCIÓN

- **Aprendizaje supervisado**
 - **Entrenamiento:**
 - La capacidad de un modelo para resolver un problema (conocimiento) está ligada a los ejemplos utilizados durante el aprendizaje.
 - El conjunto de entrenamiento debe:
 - Ser significativo:
 - Número suficiente de ejemplos
 - Si el conjunto de entrenamiento es reducido, la red no será capaz de resolver el problema de forma eficaz
 - Ser representativo:
 - Ejemplos diversos: todas las regiones del espacio de estados deben estar suficientemente cubiertas.
 - Si un conjunto de aprendizaje contiene muchos más ejemplos de un tipo que del resto, el modelo se especializará en dicho subconjunto y no será de aplicación general.

2.1 INTRODUCCIÓN

- Aprendizaje supervisado
 - Si un modelo ha sido bien escogido, con una complejidad adecuada, y entrenado, con un conjunto de patrones bien escogido, tendrá una buena capacidad de generalización
 - Funcionará bien con patrones nuevos
 - ¿Qué ocurre si se presentan patrones en alguna región donde en el entrenamiento no hubo patrones?
 - Conjunto de entrenamiento mal escogido: no era representativo
 - Esa zona donde se quiere utilizar no estaba representada

2.1 INTRODUCCIÓN

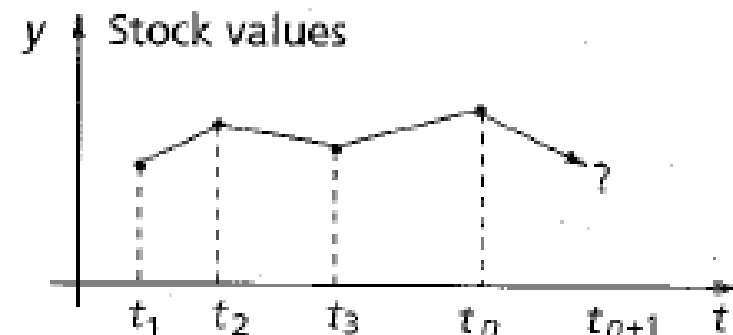
- Aprendizaje supervisado
 - Problemas que se pueden resolver:
 - Predicción
 - Ajuste de curvas
 - Regresión
 - Clasificación

2.1 INTRODUCCIÓN

- Aplicaciones:

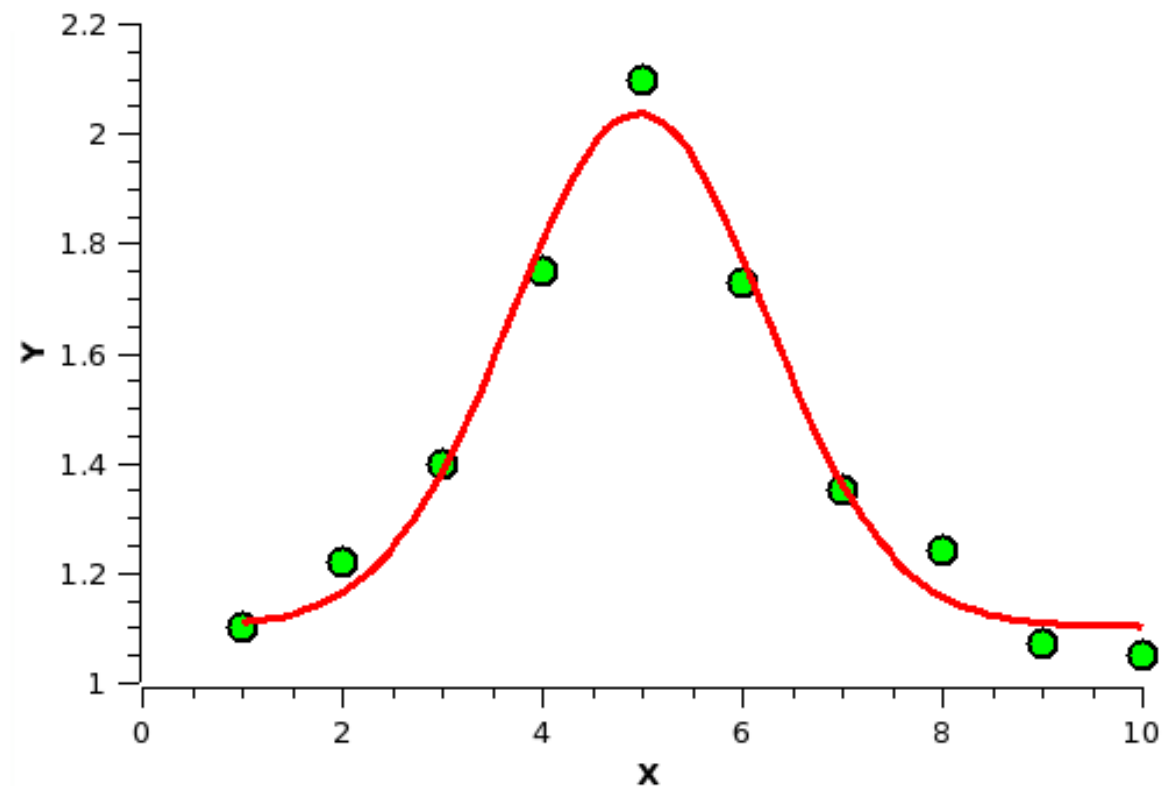
- Predicción

- Intenta determinar la función que mapea un conjunto de variables de entrada en una (o más) variables de salida.
 - Es básicamente numérica
 - Está basada en supuestos estadísticos
 - Ejemplos:
 - Monitoreo la reserva de plazas en empresas de aviación.
 - Predicciones financieras a corto plazo



2.1 INTRODUCCIÓN

- Aplicaciones:
 - Aproximación de curvas (*fitting*):
 - Reconocer la curva que subyace tras un conjunto de patrones (x_i, y)



2.1 INTRODUCCIÓN

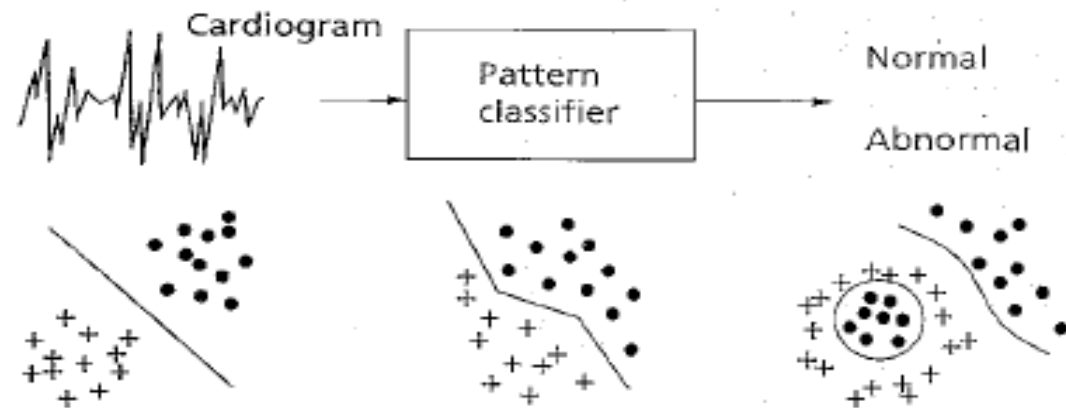
- Aplicaciones:
 - Regresión:
 - Genérico
 - A partir de una BD o conjunto de patrones, intentar encontrar un modelo de la relación entre dos conjuntos de ellos
 - Patrones: (entradas, salidas deseadas)
 - Relación desconocida
 - Modelo debe de ser generalizable, es decir, que ante nuevos patrones desconocidos devuelva unas salidas coherentes
 - Los otros problemas (predicción, ajuste de curvas) se pueden ver como casos particulares de este
 - Para evaluar la bondad de un sistema de regresión, se calcula algún tipo de error (error medio, ECM, etc.)

2.1 INTRODUCCIÓN

- Aplicaciones:

- Clasificación

- Clasifica objetos en un número finito de clases, dadas sus propiedades (entradas)
 - Busca una función de mapeo que permita separar la clase 1 de la clase 2 y esta de la clase 3...
 - El número de clases es finito y conocido
 - Se conoce la pertenencia a la clase de cada patrón



2.1 INTRODUCCIÓN

- Clasificación supervisada:
 - Para evaluar la bondad de un clasificador ya entrenado, no se mira el error cometido
 - No es significativo
 - Depende de cómo esté codificada la salida
 - En su lugar, existen unos criterios de evaluación de clasificadores:
 - Matriz de confusión
 - Curva ROC
 - Área bajo la curva
 - Índice kappa

2.1 INTRODUCCIÓN

- Clasificación supervisada:
 - Criterios de evaluación de clasificadores:
 - Matriz de confusión
 - Muestra la distribución de los errores cometidos por un clasificador a lo largo de las distintas categorías del problema

		Predicción	
		Negativo	Positivo
Real	Negativo	VN	FP
	Positivo	FN	VP

- MATLAB:

		Real	
		Negativo	Positivo
Predicción	Negativo	VN	FN
	Positivo	FP	VP

2.1 INTRODUCCIÓN

- Clasificación supervisada:
 - Criterios de evaluación de clasificadores:
 - Matriz de confusión

		Predicción		
		Negativo	Positivo	
Real	Negativo	VN	FP	Error Tipo I
	Positivo	FN	VP	

Error Tipo II

- VN: verdaderos negativos:
 - número de casos negativos correctamente clasificados como negativos
- FP: falsos positivos:
 - número de casos positivos incorrectamente clasificados como positivos
- FN: falsos negativos:
 - número de casos positivos incorrectamente clasificados como negativos
- VP: verdaderos positivos:
 - número de casos positivos correctamente clasificados como positivos

2.1 INTRODUCCIÓN

- Clasificación supervisada:
 - Criterios de evaluación de clasificadores:
 - Matriz de confusión:
 - Precisión (*accuracy*):

$$\frac{VN + VP}{VN + FN + VP + FP}$$

- Tasa de error:

$$\frac{FN + FP}{VN + FN + VP + FP}$$

		Predicción	
		Negativo	Positivo
Real	Negativo	VN	FP
	Positivo	FN	VP

2.1 INTRODUCCIÓN

- Clasificación supervisada:

- Criterios de evaluación de clasificadores:

- Matriz de confusión:

- Sensibilidad (*recall*):

- Tasa de verdaderos positivos

- Qué tasa de positivos acierta
- Probabilidad de que para un caso positivo se obtenga en el clasificador un resultado positivo

- Especificidad:

- Tasa de verdaderos negativos

- Qué tasa de negativos acierta
- Probabilidad de que para un caso negativo se obtenga en el clasificador un resultado negativo

$$\frac{VP}{FN + VP}$$

$$\frac{VN}{VN + FP}$$

		Predicción	
		Negativo	Positivo
Real	Negativo	VN	FP
	Positivo	FN	VP

2.1 INTRODUCCIÓN

- Clasificación supervisada:
 - Criterios de evaluación de clasificadores:
 - Matriz de confusión:
 - Valor predictivo positivo (*precision*):
 - Proporción de casos con valor positivo que han sido correctamente clasificados
 - Valor predictivo negativo:
 - Proporción de casos con valor negativo que han sido correctamente clasificados
 - Tasa de falsos positivos
 - 1 - especificidad
 - Tasa de falsos negativos
 - 1 - sensibilidad

$$VPP = \frac{VP}{VP + FP}$$

$$VPN = \frac{VN}{VN + FN}$$

		Predicción	
		Negativo	Positivo
Real	Negativo	VN	FP
	Positivo	FN	VP

2.1 INTRODUCCIÓN

- Clasificación supervisada:
 - Criterios de evaluación de clasificadores: Ejemplo:
 - Se tiene una base de datos de pacientes, 1000 casos:
 - 5 enfermos (positivos)
 - 995 sanos (negativos)
 - Si el clasificador usado los clasifica todos como sanos:

		Predicción	
		Negativo	Positivo
Real	Negativo	995	0
	Positivo	5	0

- Precisión (*accuracy*): 99.5% - Tasa de error: 0.5%
- Pero el clasificador no hace nada:
 - Ante cualquier entrada da una salida de “sano”

2.1 INTRODUCCIÓN

- Clasificación supervisada:
 - Criterios de evaluación de clasificadores: Ejemplo:
 - En lugar de mirar solamente la precisión:
 - Especificidad: 100%
 - Ha acertado todos los negativos
 - Sensibilidad (*recall*): 0%
 - Ha fallado todos los positivos
 - Valor predictivo positivo (*precision*): nan
 - No hay predicciones positivas

		Predicción	
		Negativo	Positivo
Real	Negativo	995	0
	Positivo	5	0

2.1 INTRODUCCIÓN

- Clasificación supervisada:
 - Criterios de evaluación de clasificadores:
 - Matriz de confusión: Sensibilidad vs. Valor predictivo positivo:
 - En muchos sistemas son más importantes los patrones positivos que los negativos. Por ejemplo: Sistemas de alarma, diagnóstico médico, detección de objetos importantes en imágenes y/o señales
 - Por ejemplo, detección de arritmias en señales de ECG
 - Por esta razón, muchas veces se usan estas métricas para saber cómo se está detectando o dejando de detectar los positivos:
 - Sensibilidad (*recall*):
 - De los positivos, cuántos acierta correctamente
 - Valor predictivo positivo (*precision*):
 - De los que clasifica como positivos, cuántos acierta
 - ¿Qué métrica es más valiosa?
 - ¡Depende del problema!

2.1 INTRODUCCIÓN

- Clasificación supervisada:
 - Criterios de evaluación de clasificadores:
 - Matriz de confusión: Sensibilidad vs. Valor predictivo positivo:
 - Ejemplo: Sistema de alarma:

		Predicción	
		Negativo	Positivo
Real	Negativo	837	3
	Positivo	6	48

- Precisión (*accuracy*): $(837+48)/(837+48+3+6) = 98.99\%$
- Sensibilidad (*recall*): $48/(48+6) = 88.89\%$
 - 6 situaciones de 54 en las que se debería haber dado alarma y no se hizo
 - Es decir, en un 11.11% de las situaciones de alarma no se dio alarma
- Valor predictivo positivo (*precision*): $48/(48+3) = 94.12\%$
 - Se han dado 3 alarmas falsas
 - Es decir, un 5.88% de las alarmas son falsas

2.1 INTRODUCCIÓN

- Clasificación supervisada:
 - Criterios de evaluación de clasificadores:
 - Matriz de confusión: Sensibilidad vs. Valor predictivo positivo:
 - Ejemplo: Sistema de alarma:
 - ¿Qué métrica es más valiosa? ¿Son los resultados aceptables?
 - ¿Se puede asumir que un 11.11% (6 de 54) de las alarmas no se detecten?
 - ¿Se puede aceptar que un 5.88% (3 de 51) de las alarmas sean alarmas falsas?
 - ¿Qué es más importante en este problema?
 - ¿Cuál el coste de una alarma no dada?
 - Coste alto: optimizar sensibilidad
 - ¿Cuál es el coste de una falsa alarma?
 - Coste alto: optimizar valor predictivo positivo
 - Sensibilidad:
 - Cuántas alarmas se dejan de dar
 - Valor predictivo positivo
 - Cuántas alarmas falsas se dan

2.1 INTRODUCCIÓN

- Clasificación supervisada:
 - Criterios de evaluación de clasificadores:
 - Matriz de confusión: Sensibilidad vs. Valor predictivo positivo:
 - ¿Qué métrica es más valiosa? ¿Son los resultados aceptables?
 - Ejemplo: Diagnóstico médico:
 - Más importante detectar correctamente los positivos aunque se den falsas alarmas:
 - Importante no clasificar como sano (negativo) a un enfermo (positivo):
 - Sensibilidad alta
 - No tan importante una falsa alarma (clasificar enfermo a un sano):
 - Se le harían más pruebas al paciente
 - Valor predictivo positivo no tan importante

2.1 INTRODUCCIÓN

- Clasificación supervisada:
 - Criterios de evaluación de clasificadores:

<u>Se usa...</u>	...cuando se quiere que el sistema...	...pero, a cambio, se corre el riesgo de que...
VPP	esté muy seguro cuando dé una alarma	haya más situaciones de alarma que se dejen de dar
VPN	esté muy seguro cuando no dé una alarma	haya más situaciones normales en las que se dé alarma
Sensibilidad	sea muy preciso en la detección de situaciones de alarma	haya más situaciones normales en las que se de falsas alarmas
Especificidad	sea muy preciso cuando en una situación normal no se da alarma	haya más situaciones de alarma que en las que no se da alarma

2.1 INTRODUCCIÓN

- Clasificación supervisada:

- Criterios de evaluación de clasificadores:

- F_1 -score:

- Media armónica entre sensibilidad (*recall*) y valor predictivo positivo (*precision*)

$$F_1 = \left(\frac{\text{recall}^{-1} + \text{precision}^{-1}}{2} \right)^{-1} = 2 \frac{\text{recall} \cdot \text{precision}}{\text{recall} + \text{precision}}$$

- Ofrece una medida de la clasificación correcta, pero no está tan influenciada por el posible desbalanceo entre clases
 - Si hay desbalanceo, la precisión (*accuracy*) puede dar una impresión errónea de los resultados obtenidos
 - Ejemplo anterior:

		Predicción	
		Negativo	Positivo
Real	Negativo	995	0
	Positivo	5	0

2.1 INTRODUCCIÓN

- Clasificación supervisada:
 - Criterios de evaluación de clasificadores:
 - ¿Precisión (*accuracy*) o F_1 -score?
 - 2 situaciones importantes:
 - 1. F_1 -score cuando se da una de las siguientes:
 - Es más importante la clasificación correcta de positivos que de negativos
 - Los errores de falso positivo y falso negativo tienen costos diferentes
 - Ejemplo: diagnóstico médico, alarmas, etc.
 - Precisión: todas las clases tienen importancia similar

2.1 INTRODUCCIÓN

- Clasificación supervisada:
 - Criterios de evaluación de clasificadores:
 - ¿Precisión (*accuracy*) o F_1 -score?
 - 2 situaciones importantes:
 - I. Es decir, se usa la precisión cuando los VP y VN son más importantes, mientras que F_1 -score se usa cuando los FN y FP son cruciales

Precisión (*accuracy*)

		Predicción	
		Negativo	Positivo
Real	Negativo		
	Positivo		

F_1 -score

		Predicción	
		Negativo	Positivo
Real	Negativo		
	Positivo		

2.1 INTRODUCCIÓN

- Clasificación supervisada:
 - Criterios de evaluación de clasificadores:
 - ¿Precisión (*accuracy*) o F_1 -score?
 - 2 situaciones importantes:
 2. La precisión se puede usar cuando la distribución de clases es parecida, mientras que F_1 -score es mejor métrica cuando hay clases desbalanceadas
 - Sensibilidad (*recall*) y VPP (*precision*) son métricas útiles en estos casos
 - En la mayoría de problemas de clasificación del mundo real, las clases están desbalanceadas, y por lo tanto F_1 -score suele ser una mejor métrica

2.1 INTRODUCCIÓN

- Clasificación supervisada:
 - Criterios de evaluación de clasificadores:
 - F_1 -score:
 - Se usa la media armónica porque penaliza valores “extremos”
 - El F_1 -score da un peso mayor a valores más bajos
 - Ejemplo:
 - Clases balanceadas (200 instancias de cada clase)
 - Clasificador que devuelve «Negativo» en la mayoría de las ocasiones

		Predicción	
		Negativo	Positivo
Real	Negativo	198	2
	Positivo	195	5

- VPP (*precision*) = 71.43%.
 - Sensibilidad (*recall*) = 2.50%
 - Precisión (*accuracy*) = 50.75%
- F_1 -score = 4.83%

2.1 INTRODUCCIÓN

- Clasificación supervisada:
 - Criterios de evaluación de clasificadores:
 - F_1 -score:
 - Se usa la media armónica porque penaliza valores “extremos”
 - El F_1 -score da un peso mayor a valores más bajos
 - Ejemplo:
 - Base de datos anterior (balanceada), 2 clasificadores:

		Predicción	
		Negativo	Positivo
Real	Negativo	160	40
	Positivo	40	160

Precisión (accuracy) = 80%
VPP (precision) = 80%
Sensibilidad (recall) = 80%
 F_1 -score = 80%

		Predicción	
		Negativo	Positivo
Real	Negativo	67	133
	Positivo	0	200

Precisión (accuracy) = 66.75%
VPP (precision) = 60.06%
Sensibilidad (recall) = 100%
 F_1 -score = 75%



Media aritmética:
80%

2.1 INTRODUCCIÓN

- Clasificación supervisada:
 - Matriz de confusión para problemas de múltiples clases:

Resultado del clasificador	Clase real			
	Clase 1	Clase 2	...	Clase N
Clase 1	Valor11	Valor12	...	Valor1N
Clase 2	Valor21	Valor22	...	Valor2N
...
Clase N	ValorN1	ValorN2	...	ValorNN

2.1 INTRODUCCIÓN

- Clasificación supervisada:
 - La matriz de confusión permite conocer cuál es el comportamiento de un clasificador
 - Tomar decisiones de cara a refinar el comportamiento del clasificador
 - Ejemplo: 3 clases:
 - 23 instancias de la clase A
 - 23 instancias de la clase B
 - 23 instancias de la clase C

2.1 INTRODUCCIÓN

- Clasificación supervisada:
 - La matriz de confusión permite conocer cuál es el comportamiento de un clasificador
 - Tomar decisiones de cara a refinar el comportamiento del clasificador
 - Ejemplo: 3 clases:

		Clasificador		
		A	B	C
Real	A	20	1	2
	B	3	11	9
	C	1	10	12

Existen 9 patrones que el clasificador ha clasificado incorrectamente como C y son de la clase B

Existen 10 patrones que el clasificador ha clasificado incorrectamente como B y son de la clase C

El clasificador se confunde entre las clases B y C

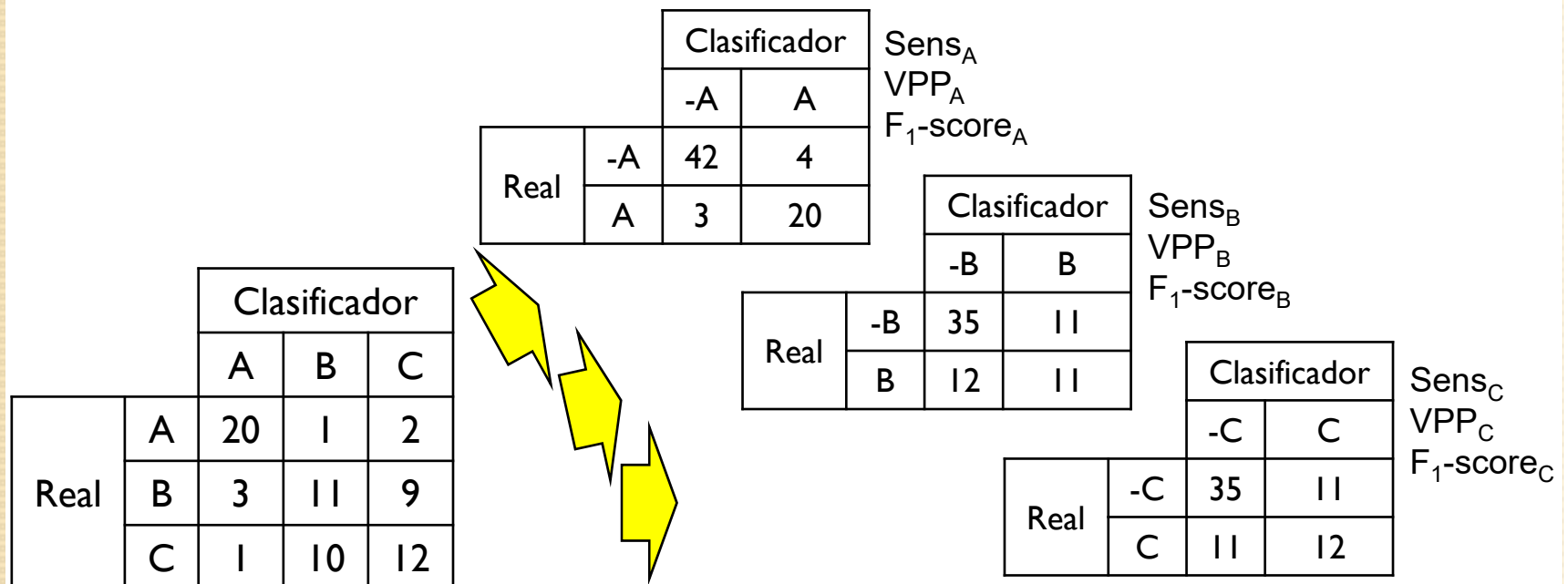
Intentar mejorar esto en el clasificador: nuevas características que distingan entre B y C

2.1 INTRODUCCIÓN

- Clasificación supervisada:
 - La mayoría de las métricas (sensibilidad, F_1 -score, VPP, etc.) son propias de clasificadores binarios
 - ¿Cómo usar estas métricas en un problema multiclase?
 - Calcular estas métricas para cada clase por separado
 - Si se tienen N clases ($N > 2$), se tratan como N problemas binarios
 - Cada problema consiste en separar una clase del resto
 - De esta forma, se puede calcular los valores de sensibilidad, VPP, F_1 -score, etc. para cada una de las clases
 - Para cada problema binario

2.1 INTRODUCCIÓN

- Clasificación supervisada:
 - La mayoría de las métricas (sensibilidad, F_1 -score, VPP, etc.) son propias de clasificadores binarios
 - ¿Cómo usar estas métricas en un problema multiclase?
 - Calcular estas métricas para cada clase por separado



2.1 INTRODUCCIÓN

- Clasificación supervisada:
 - La mayoría de las métricas (sensibilidad, F_1 -score, VPP, etc.) son propias de clasificadores binarios
 - ¿Cómo usar estas métricas en un problema multiclase?
 - Calcular estas métricas para cada clase por separado
 - ¿Cómo combinar los resultados en un único valor?
 - *Macro / Weighted / Micro*

2.1 INTRODUCCIÓN

- Clasificación supervisada:
 - *Macro*:
 - Se calcula la métrica para cada clase, y posteriormente su media aritmética sin ponderación
 - *Macro-averaged precision o macro-precision*
 - Promedio de los valores de VPP (*precision*)
 - *Macro-averaged recall o macro-recall*
 - Promedio de los valores de sensibilidad (*recall*)
 - *Macro-averaged F_1 -score o macro- F_1*
 - Promedio de los valores de F_1 -score

2.1 INTRODUCCIÓN

- Clasificación supervisada:
 - *Weighted*:
 - Se calcula la métrica para cada clase, y se calcula la media ponderada por el número de instancias de cada clase
 - El valor de F_1 -score podría no estar entre sensibilidad y VPP
 - **Útil cuando las clases están desbalanceadas**
 - *Weighted-averaged precision o weighted-precision*
 - Promedio ponderado de los valores de VPP (*precision*)
 - *Weighted-averaged recall o weighted-recall*
 - Promedio ponderado de los valores de sensibilidad (*recall*)
 - *Weighted-averaged F_1 -score o weighted- F_1*
 - Promedio ponderado de los valores de F_1 -score

2.1 INTRODUCCIÓN

- Clasificación supervisada:
 - *Micro:*
 - Se calculan las métricas de forma global contando el número total de VP, FN y FP
 - Útil en dos situaciones
 - Las clases **no son mutuamente excluyentes**
 - No importa el desbalanceo de las clases, sino el resultado global
 - *Micro-averaged precision o micro-precision*
 - *Micro-averaged recall o micro-recall*
 - *Micro-averaged F_1 -score o micro- F_1*
 - **Si las clases son mutuamente excluyentes, los tres valores son igual a la precisión (accuracy)**

2.1 INTRODUCCIÓN

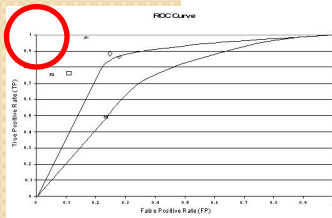
- Criterios de evaluación de clasificadores:
 - Curva ROC (*Receiver Operating Characteristic*)
 - Espacio bidimensional acotado en X e Y entre 0 y 1
 - Eje X: Tasa de falsos positivos (1-especificidad)
 - Cómo de mal clasifica los negativos
 - Eje Y: Tasa de verdaderos positivos (sensibilidad)
 - Cómo de bien clasifica los positivos
 - El clasificador se sitúa en un punto con coordenadas (x,y)

2.1 INTRODUCCIÓN

- Criterios de evaluación de clasificadores:

- Curva ROC (*Receiver Operating Characteristic*)

- Eje X: Tasa de falsos positivos (1-especificidad)
- Eje Y: Tasa de verdaderos positivos (sensibilidad)
- Punto (0,1):

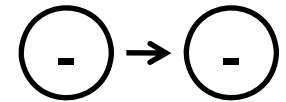


- Tasa de falsos positivos: 0%

- Ningún patrón negativo fue clasificado como positivo

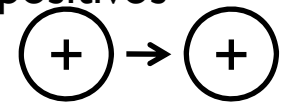
- Ningún negativo clasificado incorrectamente

- Todos los negativos clasificados correctamente como negativos



- Tasa de verdaderos positivos: 100%

- Todos los positivos clasificados correctamente como positivos



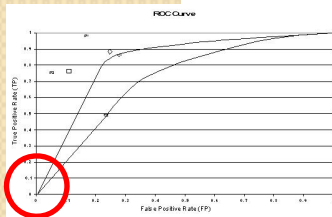
- Clasificador perfecto: Clasifica perfectamente todos los casos positivos y los negativos

2.1 INTRODUCCIÓN

- Criterios de evaluación de clasificadores:

- Curva ROC (*Receiver Operating Characteristic*)

- Eje X: Tasa de falsos positivos (1-especificidad)
- Eje Y: Tasa de verdaderos positivos (sensibilidad)
- Punto (0,0):

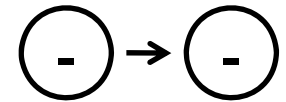


- Tasa de falsos positivos: 0%

- Ningún patrón negativo fue clasificado como positivo

- Ningún negativo clasificado incorrectamente

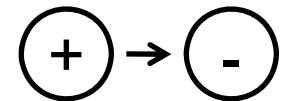
- Todos los negativos clasificados correctamente como negativos



- Tasa de verdaderos positivos: 0%

- Ningún positivo clasificado correctamente como positivo

- Todos los positivos clasificados incorrectamente como negativos



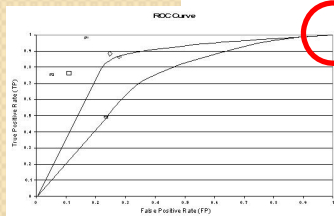
- Clasificador que predice todos los casos como negativos

2.1 INTRODUCCIÓN

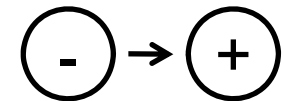
- Criterios de evaluación de clasificadores:

- Curva ROC (*Receiver Operating Characteristic*)

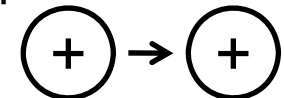
- Eje X: Tasa de falsos positivos (1-especificidad)
- Eje Y: Tasa de verdaderos positivos (sensibilidad)
- Punto (1,1):



- Tasa de falsos positivos: 100%
 - Todos los negativos incorrectamente clasificados como positivos



- Tasa de verdaderos positivos: 100%
 - Todos los positivos clasificados correctamente como positivos



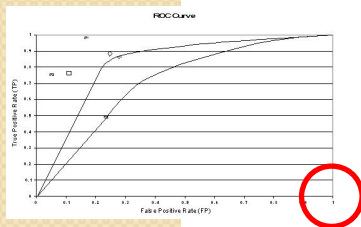
- Clasificador que predice todos los casos como positivos

2.1 INTRODUCCIÓN

- Criterios de evaluación de clasificadores:

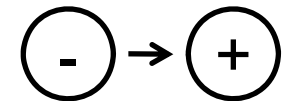
- Curva ROC (*Receiver Operating Characteristic*)

- Eje X: Tasa de falsos positivos (1-especificidad)
- Eje Y: Tasa de verdaderos positivos (sensibilidad)
- Punto (1,0):



- Tasa de falsos positivos: 100%

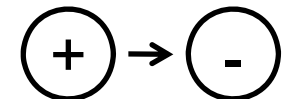
- Todos los negativos incorrectamente clasificados como positivos



- Tasa de verdaderos positivos: 0%

- Ningún positivo clasificado correctamente como positivo

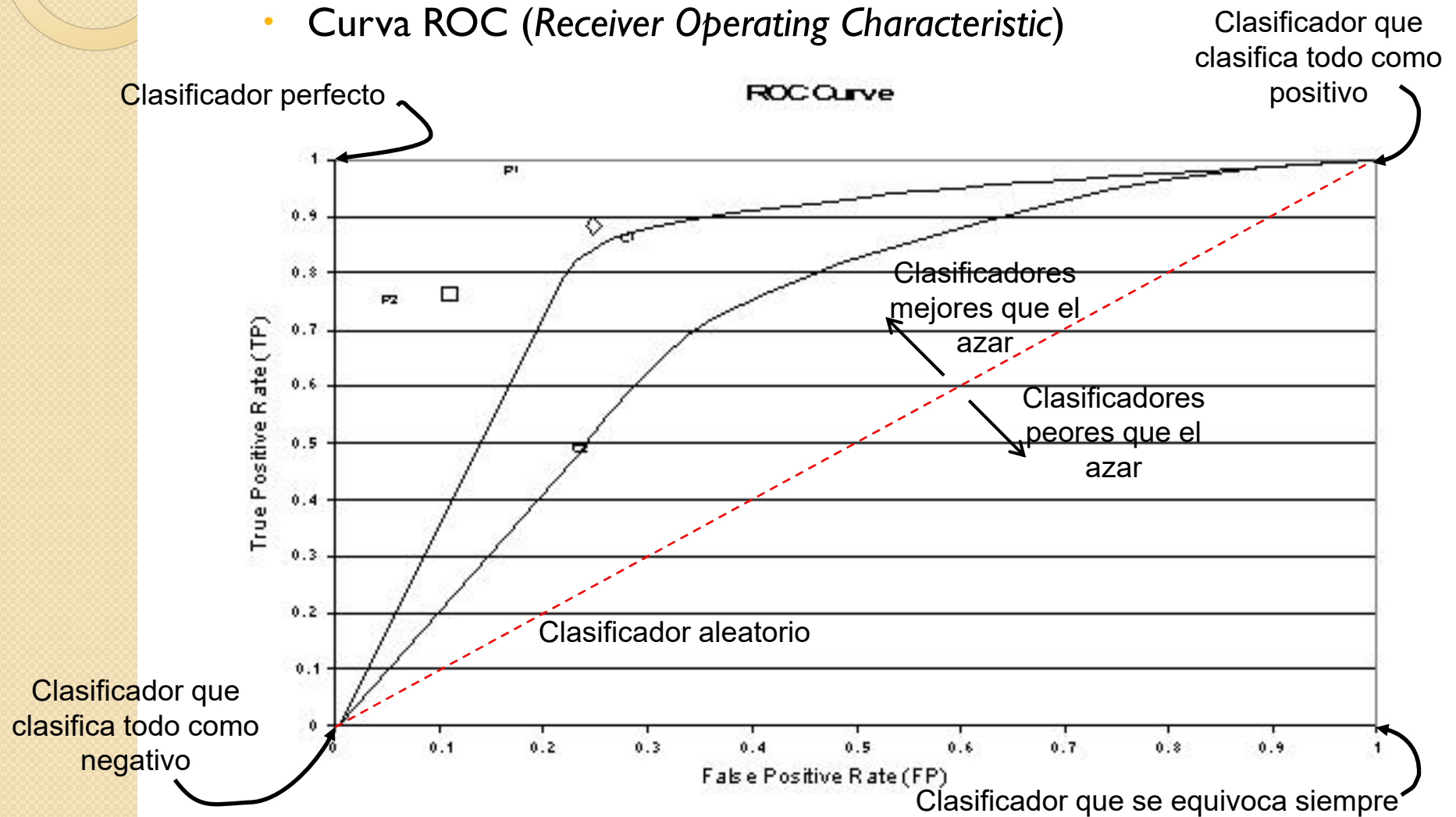
- Todos los positivos clasificados incorrectamente como negativos



- Todas las clasificaciones son incorrectas

2.1 INTRODUCCIÓN

- Criterios de evaluación de clasificadores:
 - Curva ROC (Receiver Operating Characteristic)

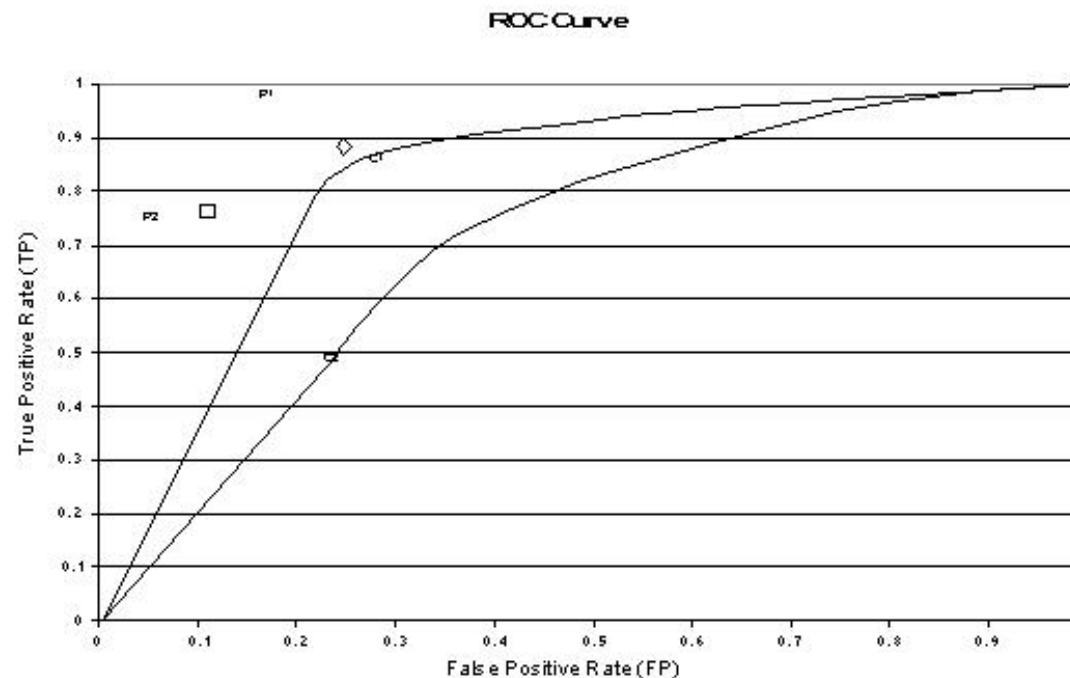


2.1 INTRODUCCIÓN

- Criterios de evaluación de clasificadores:
 - Curva ROC (*Receiver Operating Characteristic*)
 - El clasificador se sitúa en un punto concreto de coordenadas
 - (1-especificidad, sensibilidad)
 - Sin embargo, en muchas ocasiones el clasificador tiene algún parámetro que se puede variar para
 - aumentar VP a costa de incrementar FP
 - o decrementar FN a costa de decrementar VN
 - Por ejemplo, el umbral de las neuronas de salida de una RNA para clasificación en dos clases
 - Devuelven un valor continuo entre 0 y 1
 - Ese valor se pasa a un umbral:
 - Valores inferiores: salida 0 (se clasifican como negativos)
 - Valores mayores: salida 1 (se clasifican como positivos)
 - Si se aumenta este umbral, más patrones se clasifican como negativos, aumentan VN y FN
 - Si se disminuye, más patrones se clasifican como positivos, aumentan VP y FP

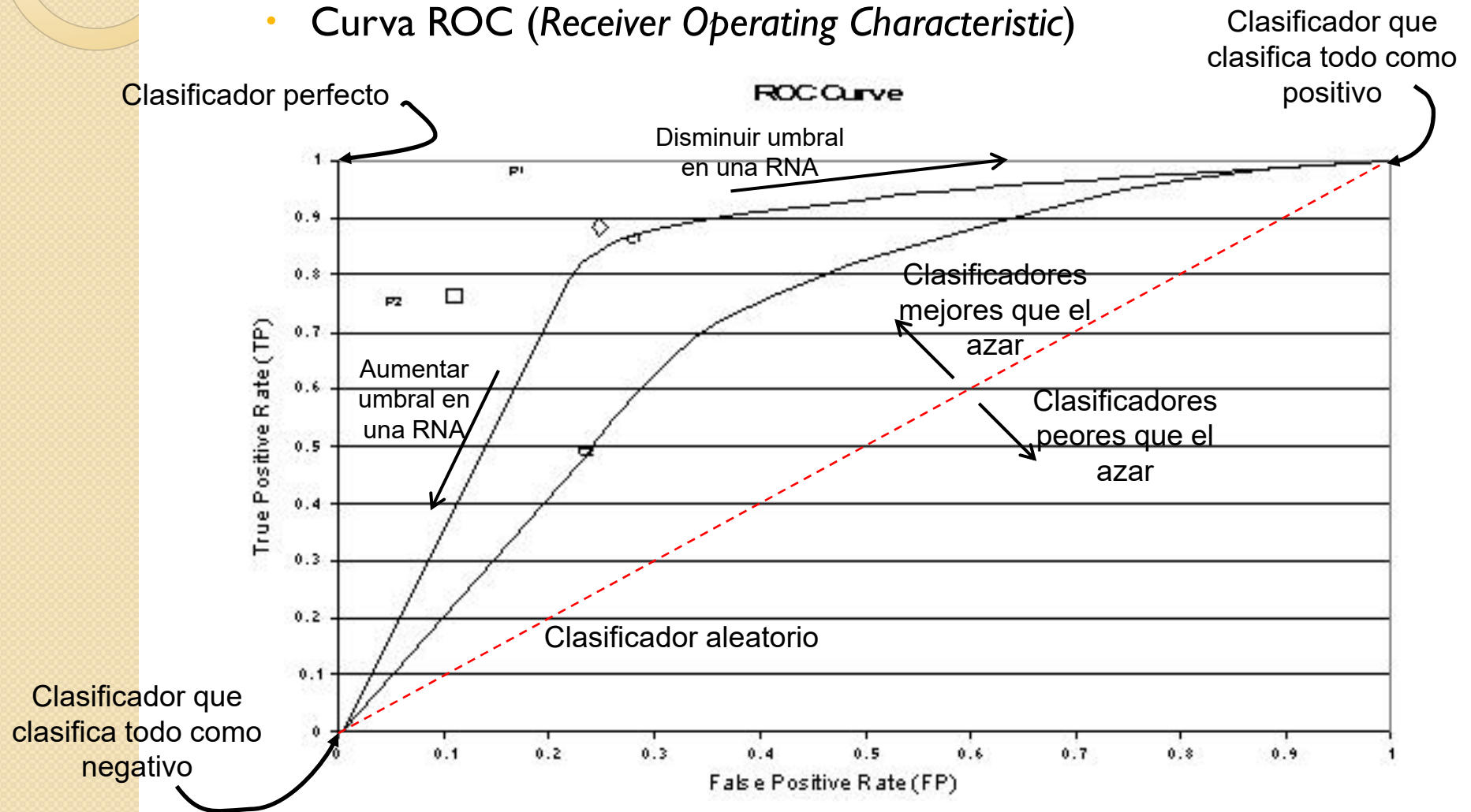
2.1 INTRODUCCIÓN

- Criterios de evaluación de clasificadores:
 - Curva ROC (*Receiver Operating Characteristic*)
 - Por tanto, cada valor de este parámetro colocaría al clasificador en una coordenada distinta
 - La curva ROC representa todas las posibilidades de este valor
 - Un clasificador no parametrizado se representaría con un solo punto



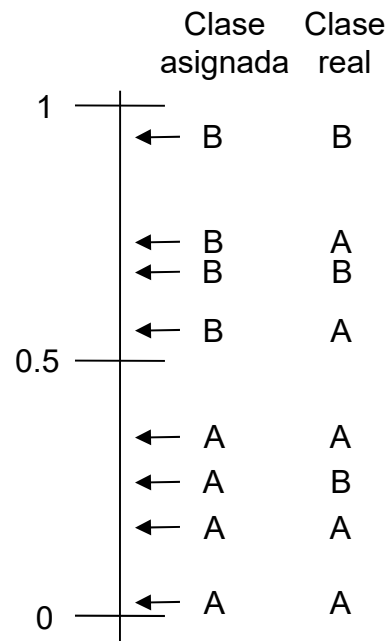
2.1 INTRODUCCIÓN

- Criterios de evaluación de clasificadores:
 - Curva ROC (Receiver Operating Characteristic)



2.1 INTRODUCCIÓN

- Criterios de evaluación de clasificadores:
 - Curva ROC (*Receiver Operating Characteristic*)
 - Ejemplo: RNA para clasificación en 2 clases A/B con salida continua entre 0 y 1
 - Se aplica un umbral
 - Valor: 0.5
 - Salidas superiores: "B" - Salidas inferiores: "A"



		RNA	
		A (-)	B (+)
Real	A (-)	3	2
	B (+)	1	2

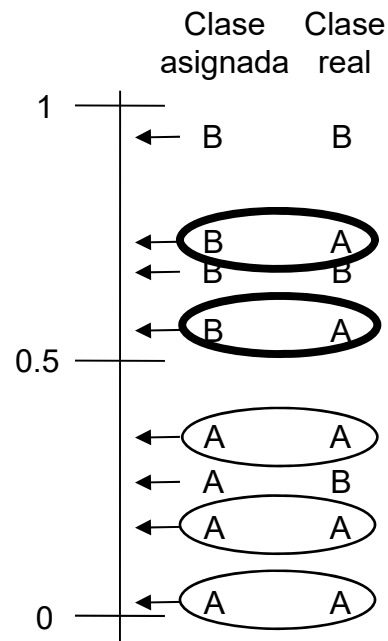
Tasa de falsos positivos: $2 / 5 = 0.4$

Tasa de verdaderos positivos: $2 / 3 = 0.66$

Con este umbral, la RNA se sitúa en el punto (0.4, 0.66)

2.1 INTRODUCCIÓN

- Criterios de evaluación de clasificadores:
 - Curva ROC (*Receiver Operating Characteristic*)
 - Ejemplo: RNA para clasificación en 2 clases A/B con salida continua entre 0 y 1
 - Se aplica un umbral
 - Valor: 0.5
 - Salidas superiores: "B" - Salidas inferiores: "A"



		RNA	
		A (-)	B (+)
Real	A (-)	3	2
	B (+)	1	2

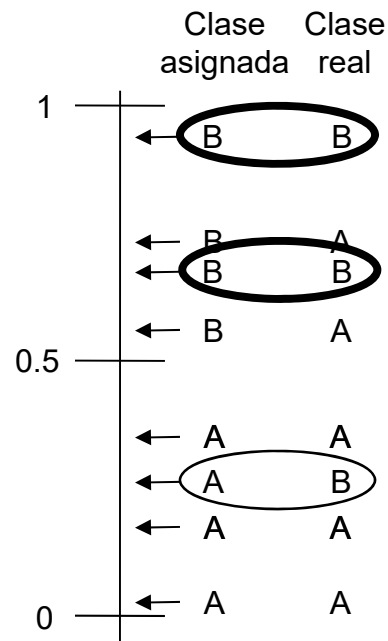
Tasa de falsos positivos: $2/5 = 0.4$

Tasa de verdaderos positivos: $2/3 = 0.66$

Con este umbral, la RNA se sitúa en el punto (0.4, 0.66)

2.1 INTRODUCCIÓN

- Criterios de evaluación de clasificadores:
 - Curva ROC (*Receiver Operating Characteristic*)
 - Ejemplo: RNA para clasificación en 2 clases A/B con salida continua entre 0 y 1
 - Se aplica un umbral
 - Valor: 0.5
 - Salidas superiores: "B" - Salidas inferiores: "A"



		RNA	
		A (-)	B (+)
Real	A (-)	3	2
	B (+)	1	2

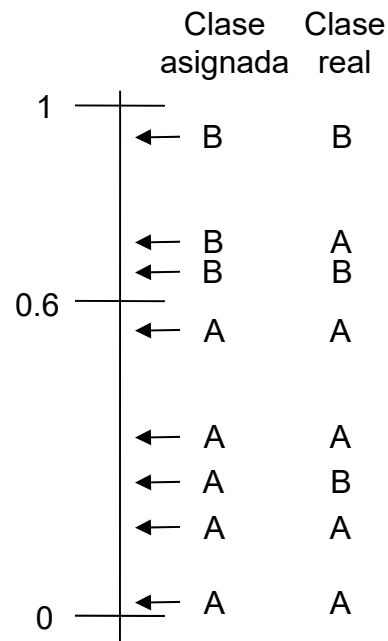
Tasa de falsos positivos: $2 / 5 = 0.4$

Tasa de verdaderos positivos: $2 / 3 = 0.66$

Con este umbral, la RNA se sitúa en el punto (0.4, 0.66)

2.1 INTRODUCCIÓN

- Criterios de evaluación de clasificadores:
 - Curva ROC (*Receiver Operating Characteristic*)
 - Ejemplo: RNA para clasificación en 2 clases A/B con salida continua entre 0 y 1
 - Se aumenta el umbral
 - Valor: 0.6
 - Salidas superiores: "B" - Salidas inferiores: "A"



		RNA	
		A (-)	B (+)
Real	A (-)	4	1
	B (+)	1	2

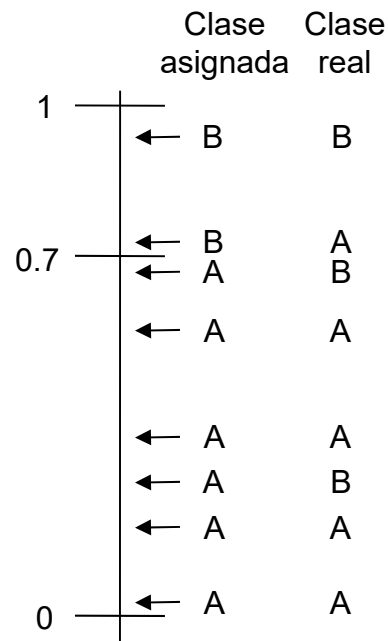
Tasa de falsos positivos: $1 / 5 = 0.2$

Tasa de verdaderos positivos: $2 / 3 = 0.66$

Con este umbral, la RNA se sitúa en el punto (0.2, 0.66)

2.1 INTRODUCCIÓN

- Criterios de evaluación de clasificadores:
 - Curva ROC (*Receiver Operating Characteristic*)
 - Ejemplo: RNA para clasificación en 2 clases A/B con salida continua entre 0 y 1
 - Se aumenta el umbral
 - Valor: 0.7
 - Salidas superiores: "B" - Salidas inferiores: "A"



		RNA	
		A (-)	B (+)
Real	A (-)	4	1
	B (+)	2	1

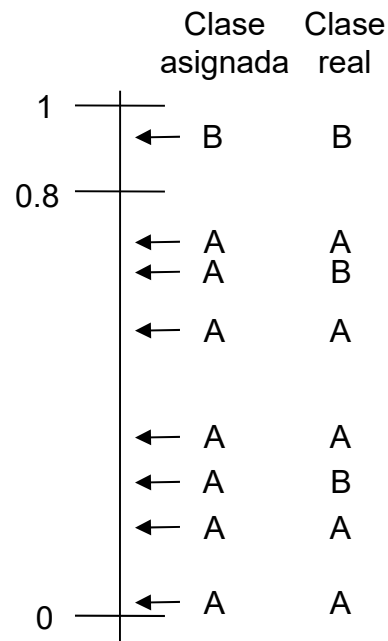
Tasa de falsos positivos: $1 / 5 = 0.2$

Tasa de verdaderos positivos: $1 / 3 = 0.33$

Con este umbral, la RNA se sitúa en el punto (0.2, 0.33)

2.1 INTRODUCCIÓN

- Criterios de evaluación de clasificadores:
 - Curva ROC (*Receiver Operating Characteristic*)
 - Ejemplo: RNA para clasificación en 2 clases A/B con salida continua entre 0 y 1
 - Se aumenta el umbral
 - Valor: 0.8
 - Salidas superiores: "B" - Salidas inferiores: "A"



		RNA	
		A (-)	B (+)
Real	A (-)	5	0
	B (+)	2	1

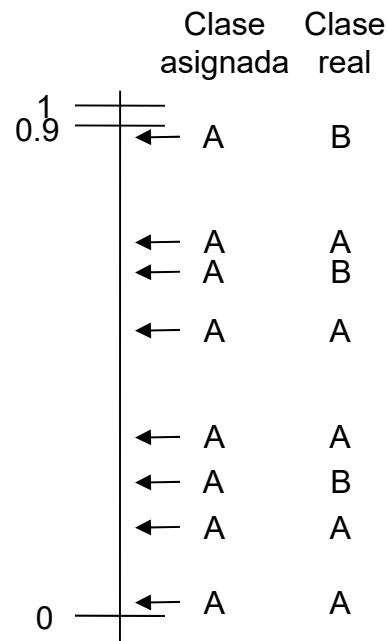
Tasa de falsos positivos: $0 / 5 = 0$

Tasa de verdaderos positivos: $1 / 3 = 0.33$

Con este umbral, la RNA se sitúa en el punto (0, 0.33)

2.1 INTRODUCCIÓN

- Criterios de evaluación de clasificadores:
 - Curva ROC (*Receiver Operating Characteristic*)
 - Ejemplo: RNA para clasificación en 2 clases A/B con salida continua entre 0 y 1
 - Se aumenta el umbral
 - Valor: 0.9
 - Salidas superiores: "B" - Salidas inferiores: "A"



		RNA	
		A (-)	B (+)
Real	A (-)	5	0
	B (+)	3	0

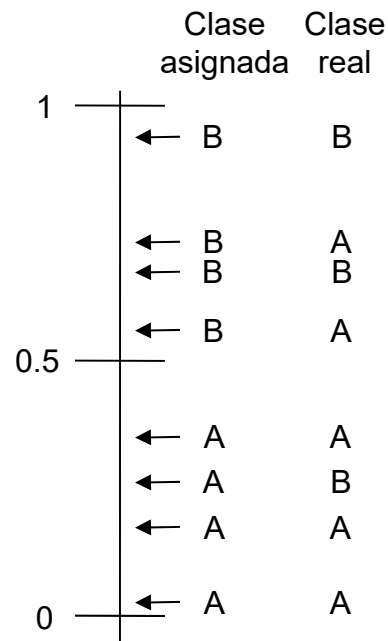
Tasa de falsos positivos: $0 / 5 = 0$

Tasa de verdaderos positivos: $0 / 3 = 0$

Con este umbral, la RNA se sitúa en el punto (0, 0)
Clasifica todo como A (negativo)

2.1 INTRODUCCIÓN

- Criterios de evaluación de clasificadores:
 - Curva ROC (*Receiver Operating Characteristic*)
 - Ejemplo: RNA para clasificación en 2 clases A/B con salida continua entre 0 y 1
 - Se aplica un umbral
 - Valor: 0.5
 - Salidas superiores: "B" - Salidas inferiores: "A"



		RNA	
		A (-)	B (+)
Real	A (-)	3	2
	B (+)	1	2

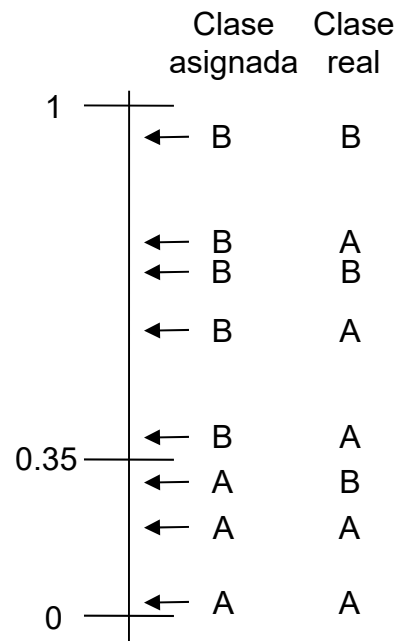
Tasa de falsos positivos: $2 / 5 = 0.4$

Tasa de verdaderos positivos: $2 / 3 = 0.66$

Con este umbral, la RNA se sitúa en el punto (0.4, 0.66)

2.1 INTRODUCCIÓN

- Criterios de evaluación de clasificadores:
 - Curva ROC (*Receiver Operating Characteristic*)
 - Ejemplo: RNA para clasificación en 2 clases A/B con salida continua entre 0 y 1
 - Se disminuye el umbral
 - Valor: 0.35
 - Salidas superiores: "B" - Salidas inferiores: "A"



		RNA	
		A (-)	B (+)
Real	A (-)	2	3
	B (+)	1	2

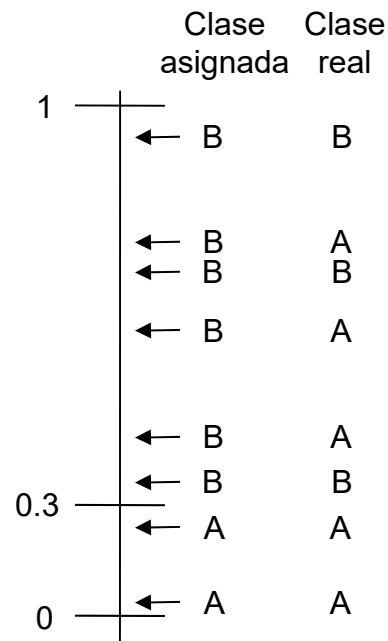
Tasa de falsos positivos: $3 / 5 = 0.6$

Tasa de verdaderos positivos: $2 / 3 = 0.66$

Con este umbral, la RNA se sitúa en el punto (0.6, 0.66)

2.1 INTRODUCCIÓN

- Criterios de evaluación de clasificadores:
 - Curva ROC (*Receiver Operating Characteristic*)
 - Ejemplo: RNA para clasificación en 2 clases A/B con salida continua entre 0 y 1
 - Se disminuye el umbral
 - Valor: 0.3
 - Salidas superiores: "B" - Salidas inferiores: "A"



		RNA	
		A (-)	B (+)
Real	A (-)	2	3
	B (+)	0	3

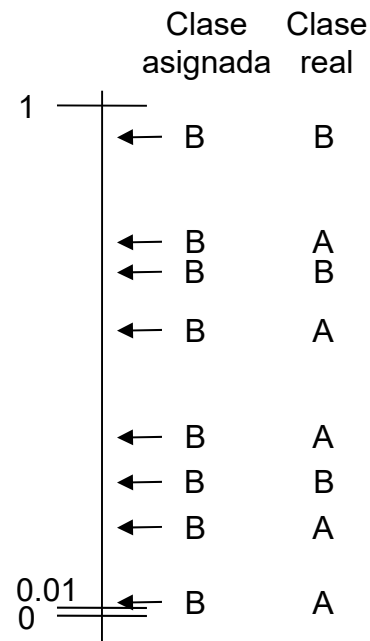
Tasa de falsos positivos: $3 / 5 = 0.6$

Tasa de verdaderos positivos: $3 / 3 = 1$

Con este umbral, la RNA se sitúa en el punto (0.6, 1)

2.1 INTRODUCCIÓN

- Criterios de evaluación de clasificadores:
 - Curva ROC (*Receiver Operating Characteristic*)
 - Ejemplo: RNA para clasificación en 2 clases A/B con salida continua entre 0 y 1
 - Se disminuye el umbral
 - Valor: 0.01
 - Salidas superiores: "B" - Salidas inferiores: "A"



		RNA	
		A (-)	B (+)
Real	A (-)	0	5
	B (+)	0	3

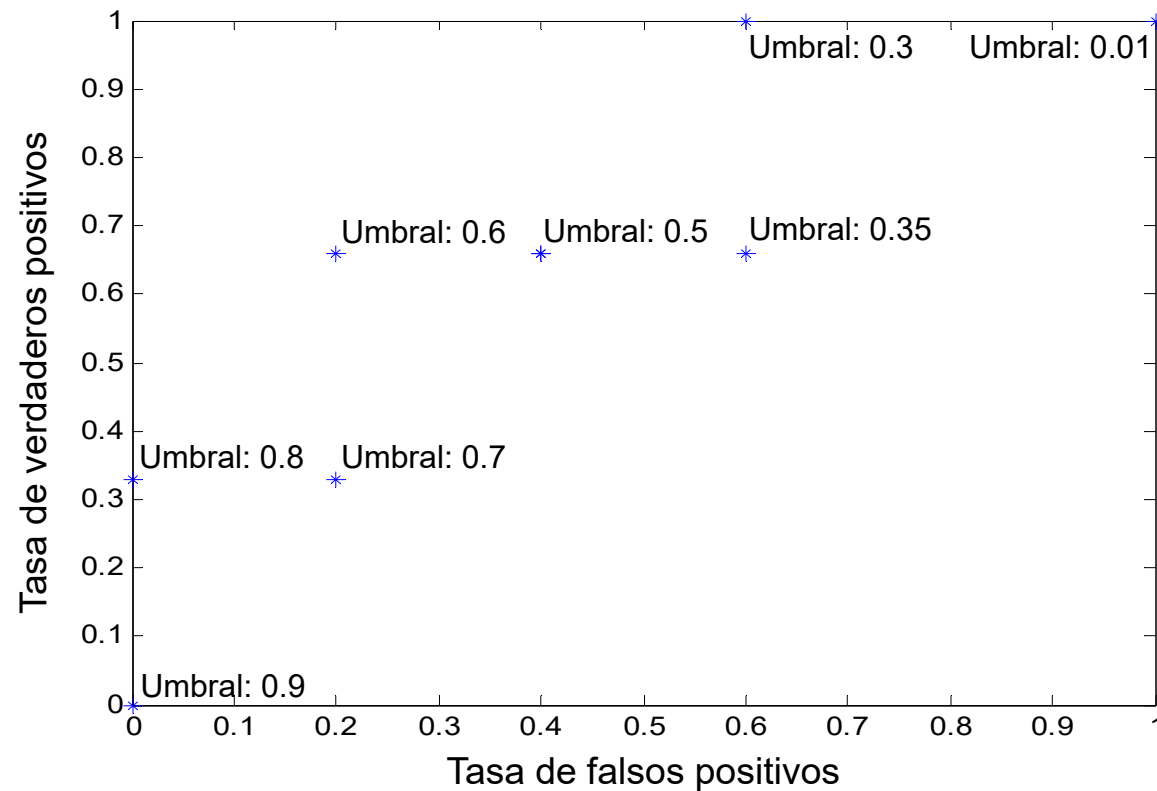
Tasa de falsos positivos: $5 / 5 = 1$

Tasa de verdaderos positivos: $3 / 3 = 1$

Con este umbral, la RNA se sitúa en el punto (1, 1)
Clasifica todo como B (positivo)

2.1 INTRODUCCIÓN

- Criterios de evaluación de clasificadores:
 - Curva ROC (*Receiver Operating Characteristic*)
 - Ejemplo: RNA para clasificación en 2 clases A/B con salida continua entre 0 y 1
 - Se aumenta/disminuye el umbral

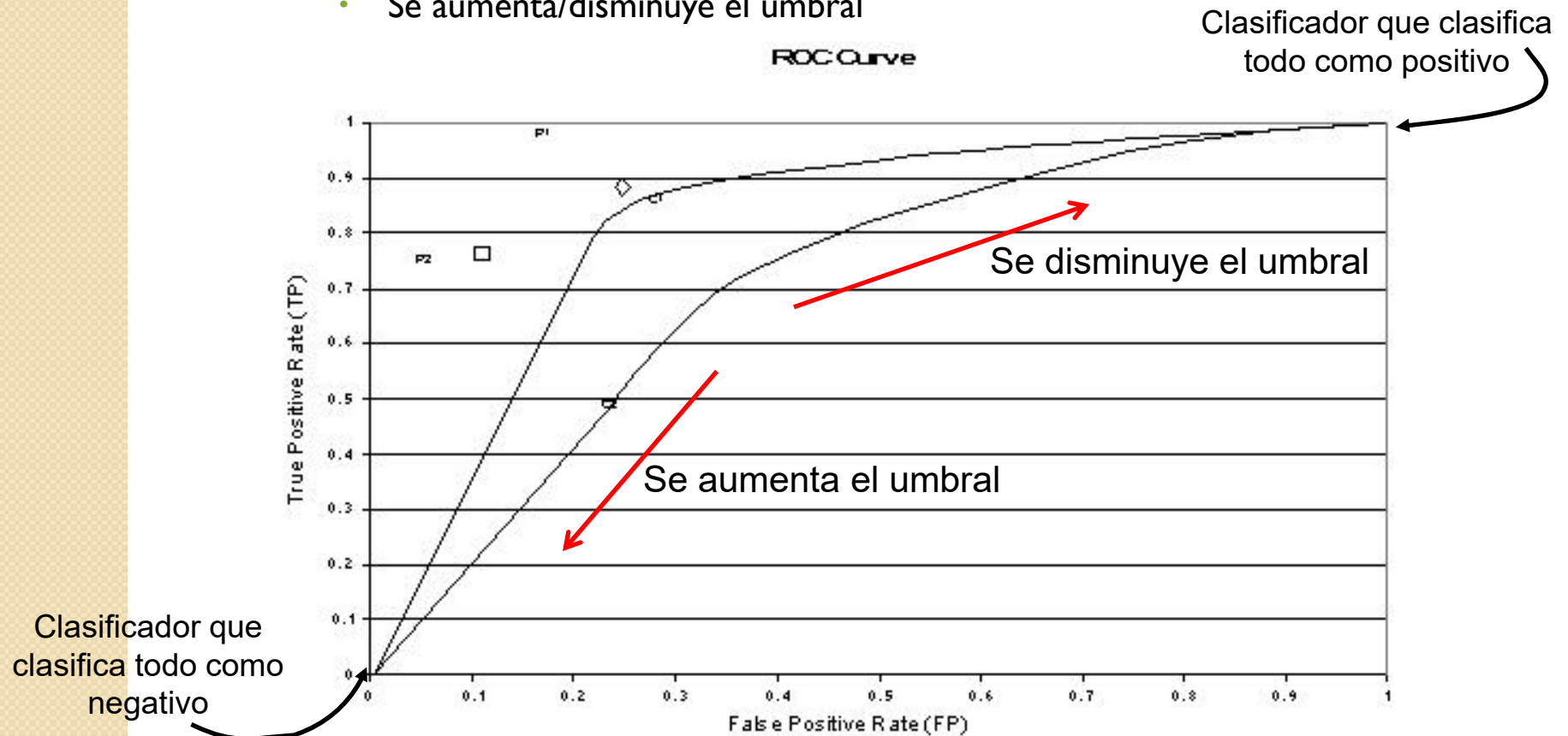


2.1 INTRODUCCIÓN

- Criterios de evaluación de clasificadores:

- Curva ROC (Receiver Operating Characteristic)

- Ejemplo: RNA para clasificación en 2 clases A/B con salida continua entre 0 y 1
 - Se aumenta/disminuye el umbral

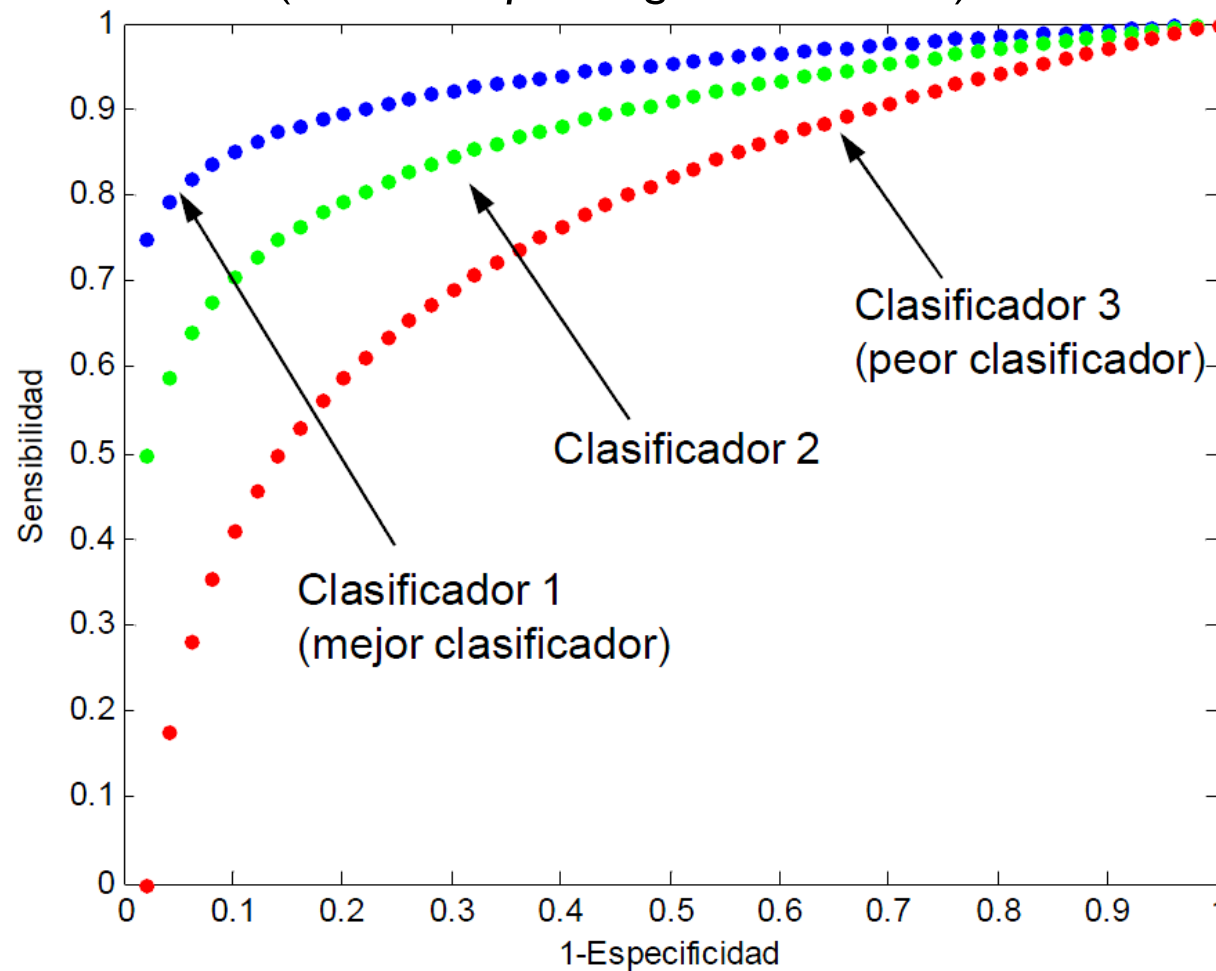


2.1 INTRODUCCIÓN

- Criterios de evaluación de clasificadores:
 - Curva ROC (*Receiver Operating Characteristic*)
 - En general, cuanto más se acerque la curva (o punto) al valor (0,1), mejor es el clasificador
 - Se puede utilizar para medir la bondad de un clasificador:
 - AUC (*Area Under the Curve*)
 - Porcentaje de espacio comprendido debajo de la curva
 - $AUC \leq 1$
 - Medida mejor que la precisión
 - Encapsula toda la información contenida en la matriz de confusión

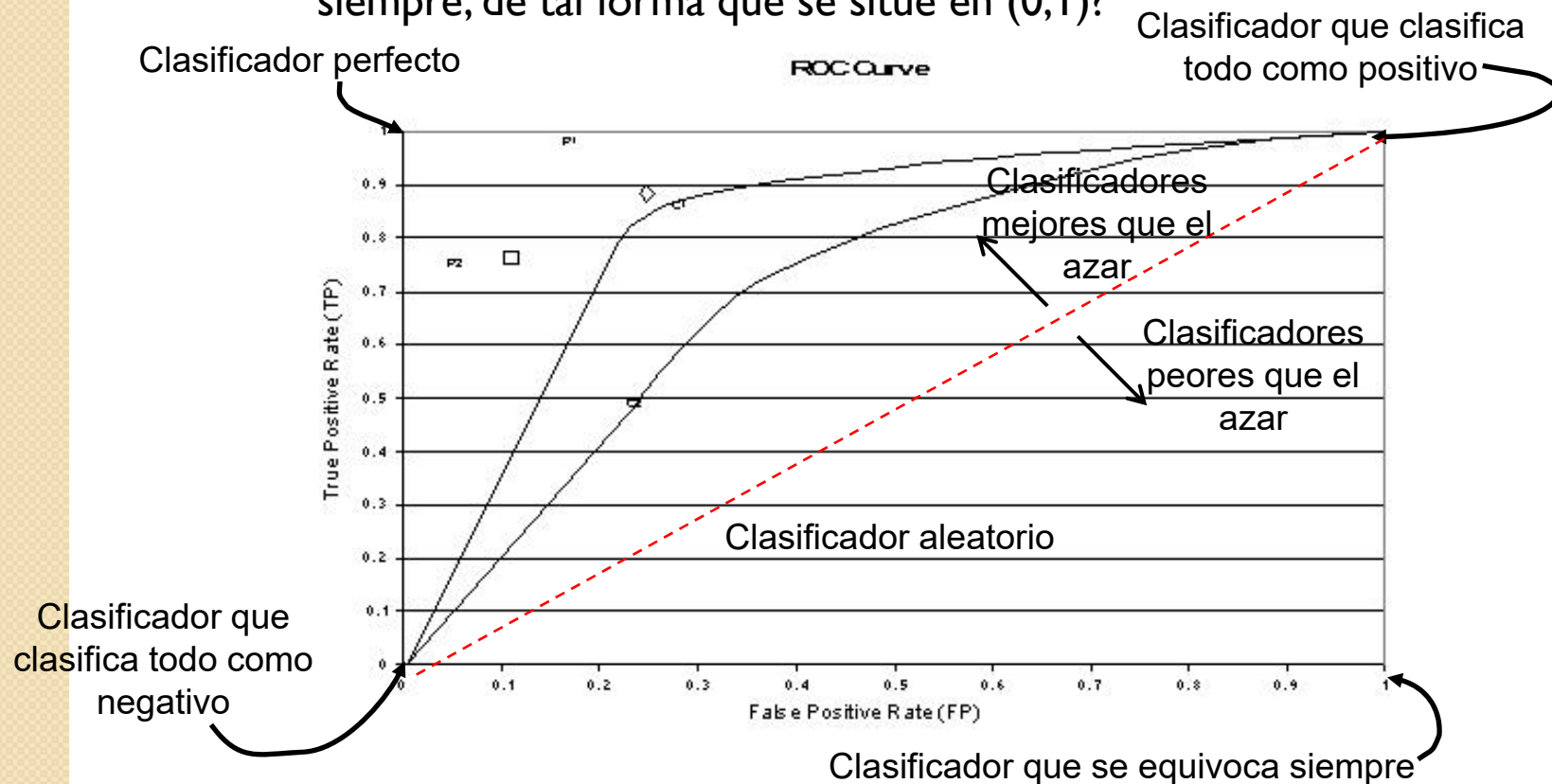
2.1 INTRODUCCIÓN

- Criterios de evaluación de clasificadores:
 - Curva ROC (*Receiver Operating Characteristic*)



2.1 INTRODUCCIÓN

- Criterios de evaluación de clasificadores:
 - Curva ROC (*Receiver Operating Characteristic*)
 - Si se tiene un clasificador situado en $(1,0)$ (se equivoca siempre), ¿se podría construir el clasificador perfecto devolviendo la salida opuesta siempre, de tal forma que se sitúe en $(0,1)$?



2.1 INTRODUCCIÓN

- Clasificación supervisada:
 - Criterios de evaluación de clasificadores:
 - Índice Kappa
 - Propuesto por Cohen en 1960 para dos evaluadores, generalizado posteriormente por Fleiss
 - Mide la concordancia entre dos clasificadores
 - Hasta qué punto coinciden en su medición
 - En lugar de tomar las salidas de uno de los clasificadores, se puede utilizar las salidas deseadas
 - En este caso, da una medida de concordancia de un clasificador con las salidas deseadas

2.1 INTRODUCCIÓN

- Clasificación supervisada:
 - Criterios de evaluación de clasificadores:
 - Índice Kappa
 - Si se tiene dos métodos de clasificación, se puede elaborar una matriz similar a la matriz de confusión:

		Método B	
		Negativo	Positivo
Método A	Negativo	a	b
	Positivo	c	d

- La forma más sencilla de medir la concordancia es la proporción de coincidencias frente al total de individuos: $(a+d)/n$

2.1 INTRODUCCIÓN

- Criterios de evaluación de clasificadores:
 - Índice Kappa
 - Sin embargo, aunque no exista ninguna relación entre los dos métodos de clasificación, se encontraría algún grado de concordancia entre ellos debido al azar
 - Por ejemplo, si ambos métodos consistieran en tirar una moneda al aire, coincidirían un 50% de las veces en promedio
 - Si un método fuera un clasificador y el otro fuera el azar, también encontraría concordancia
 - Si un método fueran las salidas deseadas, y el otro el azar, se encontraría concordancia
 - El **índice de concordancia kappa** permite determinar hasta qué punto la concordancia observada es superior a la que se espera obtener por puro azar

2.1 INTRODUCCIÓN

- Criterios de evaluación de clasificadores:
 - Índice Kappa
 - P_e : proporción de concordancia esperada por azar:
 - Probabilidad de acuerdo debido al azar:
 - De acuerdo con la tabla, la probabilidad de que un método clasifique a un individuo como negativo es:

		Método B		
		Negativo	Positivo	
Método A	Negativo	a	b	f_1
	Positivo	c	d	f_2
		c_1	c_2	n



Método A	Negativo	$a + b = f_1$
	Positivo	$c + d = f_2$

n

Para el método A: f_1/n

Método B	
Negativo	Positivo
$a + c = c_1$	$b + d = c_2$

n

Para el método B: c_1/n

2.1 INTRODUCCIÓN

- Criterios de evaluación de clasificadores:
 - Índice Kappa
 - P_e : proporción de concordancia esperada por azar:
 - Probabilidad de acuerdo debido al azar:
 - Si existe independencia entre ambos métodos de clasificación, la probabilidad de que coincidan clasificando al mismo individuo como **negativo** será el producto de las dos probabilidades (sucesos independientes): $\frac{f_1}{n} \cdot \frac{c_1}{n} = \frac{f_1 \cdot c_1}{n^2}$
 - De igual manera, la probabilidad de que coincidan clasificando al mismo individuo como **positivo** es $\frac{f_2}{n} \cdot \frac{c_2}{n} = \frac{f_2 \cdot c_2}{n^2}$
 - En consecuencia, la probabilidad de acuerdo en cualquiera de las dos clasificaciones será la suma de ambos valores, esto es:

$$P_e = \frac{f_1 \cdot c_1 + f_2 \cdot c_2}{n^2}$$

2.1 INTRODUCCIÓN

- Criterios de evaluación de clasificadores:

- Índice Kappa

- P_e : proporción de concordancia esperada por azar:

- **$1 - P_e$: margen de acuerdo posible no atribuible al azar**

- P_0 : proporción de concordancia observada (en tanto por 1)

- $(a+d)/n$

		Método B	
		Negativo	Positivo
Método A	Negativo	a	b
	Positivo	c	d

- En caso de acuerdo perfecto, toma el valor de 1

- Si uno de los métodos son las salidas deseadas, es la precisión

- **$P_0 - P_e$: concordancia observada no atribuible al azar**

- A la concordancia observada se le resta la esperada por azar

2.1 INTRODUCCIÓN

- Criterios de evaluación de clasificadores:
 - Índice Kappa

$$K = \frac{P_0 - P_e}{1 - P_e}$$

- $1 - P_e$: margen de acuerdo posible no atribuible al azar
 - De ese margen, se observa sólo una parte $P_0 - P_e$
- $P_0 - P_e$: concordancia observada no atribuible al azar

2.1 INTRODUCCIÓN

- Criterios de evaluación de clasificadores:

- Índice Kappa

$$K = \frac{P_0 - P_e}{1 - P_e}$$

- $1 - P_e$ representa el margen de acuerdo posible no atribuible al azar
 - De ese margen, se observa sólo una parte $P_0 - P_e$
- Por tanto,
 - En caso de concordancia perfecta el valor de kappa es 1
 - $P_0 = 1$
 - Si la concordancia observada P_0 es igual a la esperada por el azar P_e , el índice kappa vale 0
 - $P_0 = P_e$
 - En el caso de que el acuerdo observado sea inferior al esperado por el azar, el índice kappa es menor que cero (no hay concordancia)
 - $P_0 < P_e$

2.1 INTRODUCCIÓN

- Criterios de evaluación de clasificadores:
 - Índice Kappa
 - Landis y Koch propusieron unos márgenes para valorar el grado de acuerdo en función del índice Kappa:

kappa	Grado de acuerdo
<0	Sin acuerdo
0 – 0.2	Insignificante
0.2 – 0.4	Bajo
0.4 – 0.6	Moderado
0.6 – 0.8	Bueno
0.8 – 1	Muy bueno

2.1 INTRODUCCIÓN

- Criterios de evaluación de clasificadores:
 - Índice Kappa
 - Generalizable fácilmente para clasificaciones multinomiales
 - Más de dos categorías

	A	B	C	
A	a	b	c	f_1
B	d	e	f	f_2
C	g	h	i	f_3
	c_1	c_2	c_3	n

$$P_e = \sum_i \frac{f_i \cdot c_i}{n^2}$$

- Generalizable para más de dos evaluadores

2.1 INTRODUCCIÓN

- Clasificación supervisada:
 - Criterios de evaluación de clasificadores:
 - La bondad de un clasificador (precisión, sensibilidad, especificidad, AUC, etc.) nunca se evalúa en el conjunto de entrenamiento
 - El conjunto de entrenamiento ha sido utilizado para ajustar los parámetros del clasificador (ej: pesos de una RNA)
 - Por tanto, va a ofrecer buenos resultados
 - Necesario otro conjunto: test
 - No participa para nada en el entrenamiento
 - Los resultados en este conjunto (precisión, AUC, etc.) son los que dictaminan la bondad de un clasificador
 - Se suele dividir el conjunto de patrones en dos conjuntos entrenamiento/test **disjuntos**

2.1 INTRODUCCIÓN

- Clasificación supervisada:
 - Técnicas de evaluación de clasificadores (1/2):
 - *Holdout* (partición simple del conjunto de datos):
 - Se divide el conjunto de casos en dos grupos: conjunto de entrenamiento (2/3) y conjunto de test (1/3).
 - El conjunto de entrenamiento se usa para generar el clasificador y el de test para evaluarlo.
 - Validación cruzada (*cross-validation*):
 - Se divide el conjunto de casos en K subconjuntos del mismo tamaño. Se utilizan K-1 subconjuntos como datos de entrenamiento y 1 subconjunto como datos de test.
 - Se repite para los K subconjuntos y se calcula la media de la evaluación.
 - Suele utilizarse K=10 (*10-fold crossvalidation*)

2.1 INTRODUCCIÓN

- Clasificación supervisada:
 - Técnicas de evaluación de clasificadores (2/2):
 - Dejar uno fuera (*leave one out*):
 - Validación cruzada con K igual al número de casos.
 - *Bootstrapping*:
 - El conjunto de entrenamiento se escoge como una muestra aleatoria con reemplazamiento.

2.1 INTRODUCCIÓN

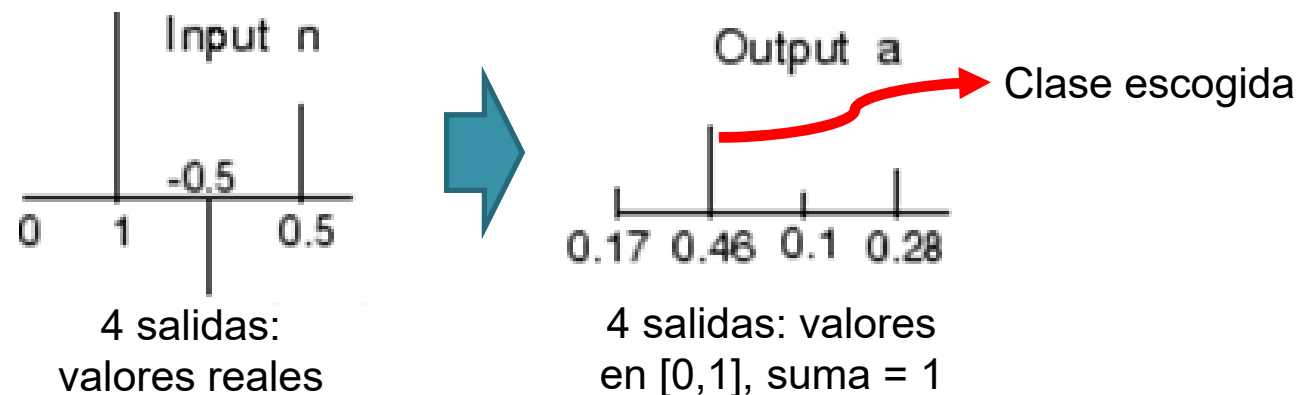
- Clasificación supervisada:
 - Codificación de la salida si el modelo no devuelve salidas categóricas (ej: RR.NN.AA.), 2 posibilidades:
 - Como números reales
 - Únicamente si en el mundo real existe un orden entre las clases equivalente
 - Ejemplo: clasificar la calidad de un producto baja/media/alta
 - Salidas 0/0.5/1 o 0/1/2 ...
 - Problema clasificación se convierte en problema de regresión
 - Como valores booleanos (mucho más común)
 - 2 clases (A/-A, -/+ , A/B, etc.)
 - Más de 2 clases (**one-hot encoding**)

2.1 INTRODUCCIÓN

- Clasificación supervisada:
 - Codificación de la salida: Como valores booleanos
 - Más de 2 clases (**one-hot encoding**):
 - Se usa una salida por clase
 - Salida deseada: cada salida deseada toma el valor 1 si esa instancia pertenece a esa clase, o 0 en caso contrario
 - Salida obtenida: Valores reales:
 - RNA: Se aplica la función *softmax* a las salidas de la última capa (salida y_i):
$$\hat{y}_i = \frac{e^{y_i}}{\sum_{j=1}^C e^{y_j}}$$
 - Pasa valores reales a valores comprendidos entre 0 y 1
 - La suma de todos es igual a 1
 - Cada valor: **probabilidad** de pertenencia a esa clase
 - Salida final: clase con mayor certidumbre/probabilidad

2.1 INTRODUCCIÓN

- Clasificación supervisada:
 - Codificación de la salida: Como valores booleanos
 - Más de 2 clases (**one-hot encoding**):
 - Se usa una salida por clase
 - Salida deseada: cada salida deseada toma el valor 1 si esa instancia pertenece a esa clase, o 0 en caso contrario
 - Salida obtenida: Valores reales:
 - RNA: Ejemplo función *softmax*:



2.1 INTRODUCCIÓN

- Clasificación supervisada:
 - Codificación de la salida: Como valores booleanos
 - Más de 2 clases (**one-hot encoding**):
 - RNA: cálculo del error:
 - **cross-entropy**:
 - Dadas dos distribuciones $p(x)$, con los valores verdaderos, y $q(x)$ con los estimados, se calcula la entropía para un conjunto de instancias x en como

$$H(p, q) = - \sum_x p(x) \cdot \log(q(x))$$

- En la salida de una RNA para un patrón n , las salidas deseadas $y = p(x)$, y las salidas obtenidas $\hat{y} = q(x)$:

$$H(y, \hat{y}) = - \sum_{n=1}^N y_n \cdot \log(\hat{y}_n)$$

2.1 INTRODUCCIÓN

- Clasificación supervisada:
 - Codificación de la salida: Como valores booleanos
 - Más de 2 clases (**one-hot encoding**):
 - RNA: cálculo del error:

- **cross-entropy**:

$$H(y, \hat{y}) = - \sum_{n=1}^N y_n \cdot \log(\hat{y}_n)$$

- $y_n \cdot \log(\hat{y}_n)$ es el producto escalar de un vector de c componentes (c clases): $y_n \cdot \log(\hat{y}_n) = \sum_{c=1}^c y_{nc} \log(\hat{y}_{nc})$
 - Dado que sólo hay una salida deseada igual a 1, el resto son 0, sólo se evalúa la salida \hat{y}_n correspondiente a la salida deseada positiva:

- Promediando en todos los patrones: $J = -\frac{1}{N} \sum_{n=1}^N \log(\hat{y}_n)$

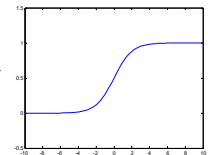
2.1 INTRODUCCIÓN

- Clasificación supervisada:
 - Codificación de la salida: Como valores booleanos
 - 2 clases (A/-A, -/+ , A/B, etc.):
 - Se codifica la salida deseada como valores 0/1 o -1/1
 - Depende del clasificador
 - Por ejemplo:
 - RNA: 0/1
 - SVM: -1/1

2.1 INTRODUCCIÓN

- Clasificación supervisada:
 - Codificación de la salida: Como valores booleanos
 - 2 clases (A/-A, -/+ , A/B, etc.):
 - El modelo devuelve una salida real
 - Además del valor de clasificación, da información de la seguridad o certeza de la misma
 - RR.NN.AA.
 - Salidas deseadas: 0/1
 - Se suele usar la función de transferencia sigmoideal
 - Valores altos: salida tiende a 1
 - Mayor seguridad de clasificación como positivo
 - Valores bajos: salida tiende a 0
 - Mayor seguridad de clasificación como negativo
 - Clasificación: Umbral en 0.5

$$f(n) = \frac{1}{1 + e^{-n}}$$



2.1 INTRODUCCIÓN

- Clasificación supervisada:
 - Codificación de la salida: Como valores booleanos
 - 2 clases (A/-A, -/+ , A/B, etc.):
 - El modelo devuelve una salida real
 - Además del valor de clasificación, da información de la seguridad o certeza de la misma
 - SVM
 - Salidas deseadas: -1/1
 - Salida tiende a ∞ : mayor seguridad de clas. como positivo
 - Salida tiende a $-\infty$: mayor seguridad de clas. como negativo
 - Clasificación:
 - Umbral en 0
 - Se mira el signo

2.1 INTRODUCCIÓN

- Clasificación supervisada:
 - Codificación de la salida: Como valores booleanos
 - 2 clases (A/-A, -/+ , A/B, etc.):
 - RNA: Cálculo del error:
(\widehat{y}_n : salida deseada, y_n : salida obtenida)
 - Diferencia entre salida y salida deseada
 - EM, ECM, etc.
$$\sum_{n=1}^N |\widehat{y}_n - y_n|$$
 - **Cross-entropy** binaria:

$$-\frac{1}{N} \sum_{n=1}^N [\widehat{y}_n \log(y_n) + (1 - \widehat{y}_n) \log(1 - y_n)]$$

2.1 INTRODUCCIÓN

- Clasificación supervisada:
 - Muchos modelos solamente permiten realizar clasificación binaria
 - Dos clases
 - Por ejemplo Máquinas de Soporte Vectorial
 - Para poder aplicarlos a problemas multiclase:
 - Votación basada en la combinación de clasificadores binarios
 - “Uno contra el resto” o “Uno contra todos”
 - “Uno contra a uno”

2.1 INTRODUCCIÓN

- Clasificación supervisada:
 - “Uno contra el resto” o “Uno contra todos”:
 - Considera el problema como una colección de problemas binarios (2 clases)
 - Requiere el uso de k clasificadores binarios
 - Uno por clase
 - En el i -ésimo problema (de los k totales) los datos pertenecientes a la clase i se les asigna la etiqueta $+1$ y al resto de datos -1 (o 0)
 - Similar a **one-hot-encoding**
 - Para seleccionar la clase a la que pertenece cada dato se suele emplear un esquema de votación entre los k clasificadores

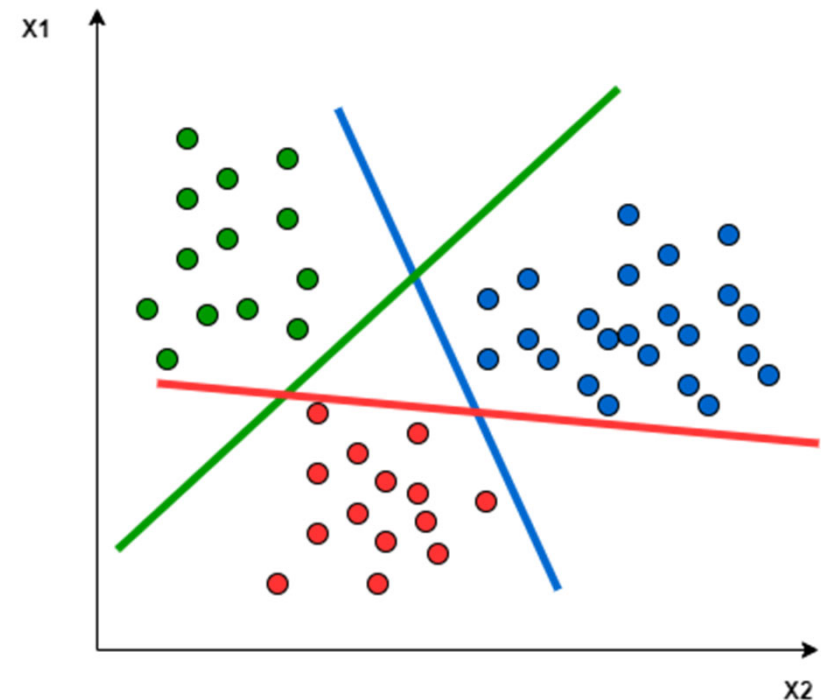
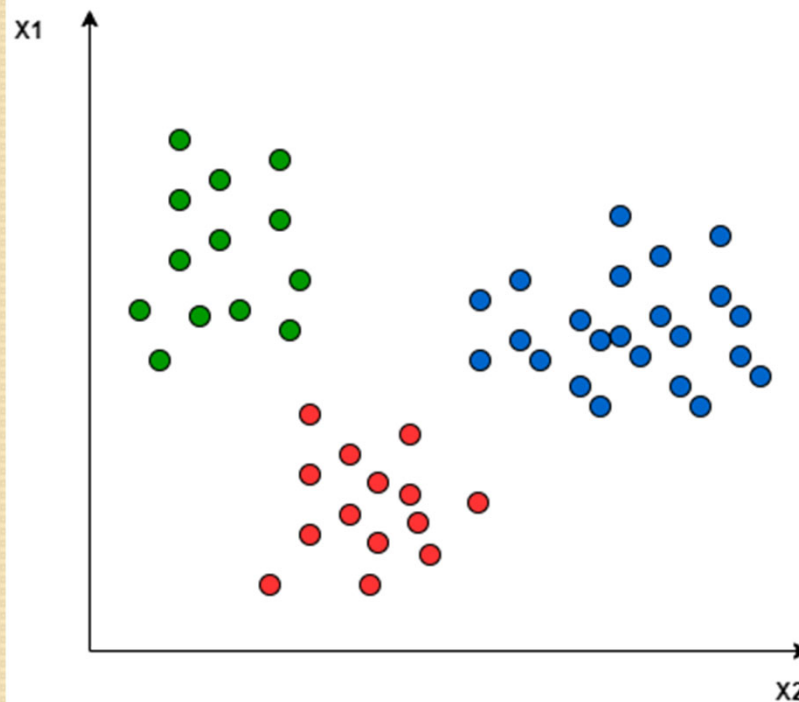
2.1 INTRODUCCIÓN

- Clasificación supervisada:
 - “Uno contra el resto” o “Uno contra todos”:
 - Esquema de votación habitual: el método ganador de todos (*winner-takes-all*)
 - Sea $f_i(x, w)$ la salida del clasificador i
 - Salida numérica
 - Se asigna la etiqueta del clasificador que tiene una mayor certeza de que el dato (x) está en la clase $+1$, es decir

$$f(x, w) = \arg \max_i (f_i(x, w))$$

2.1 INTRODUCCIÓN

- Clasificación supervisada:
 - “Uno contra el resto” o “Uno contra todos”:
 - Ejemplo:



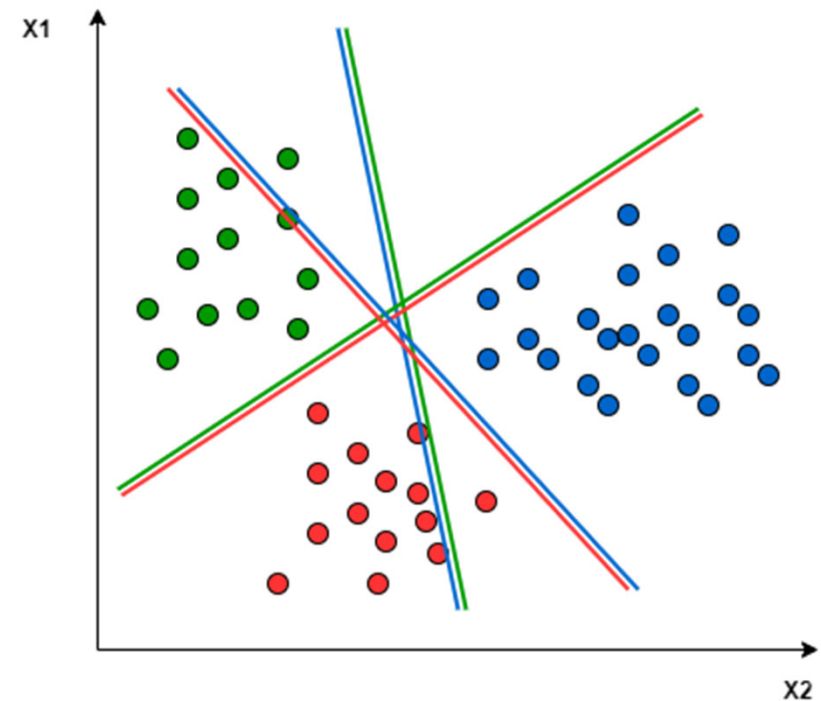
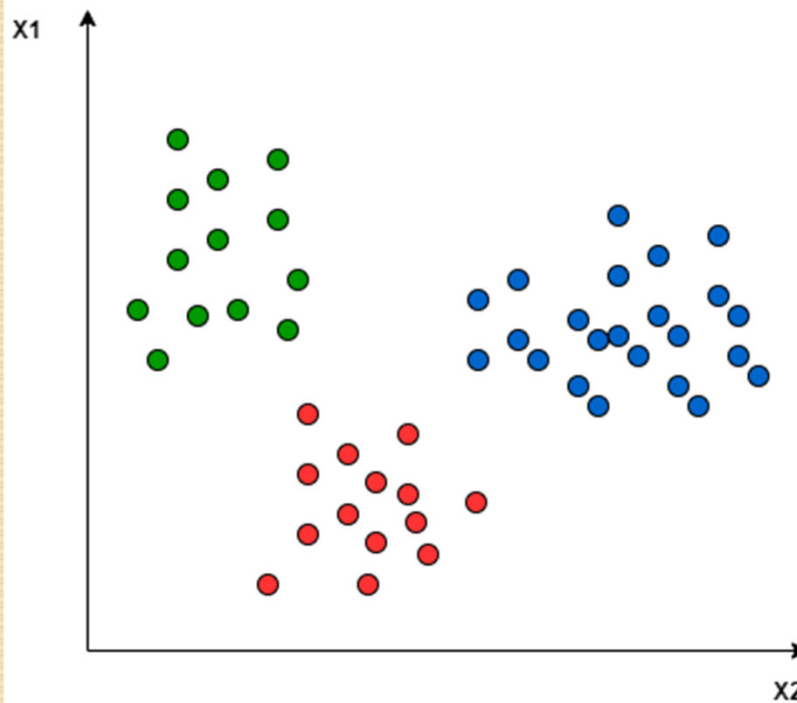
2.1 INTRODUCCIÓN

- Clasificación supervisada:
 - “Uno contra uno”:
 - En este método se construyen $k(k-1)/2$ modelos separando cada clase de cada una de las otras
 - Es decir, se crean todas las posibles parejas de una clase contra otra, y se resuelven esos problemas, en total son $k(k-1)/2$
 - Para cada par (i,j) , $i < j$, de etiquetas de la clase se construye un clasificador binario, clasificando los puntos de la clase i con la etiqueta $+1$ y los de la clase j con la etiqueta -1
 - Posteriormente se emplea un método de votación para seleccionar la clase de cada dato

$$\hat{k} = \underset{k}{\operatorname{argmin}} \frac{\sum_{l=1}^L |m_{kl}| g(m_{kl}, s_l)}{\sum_{l=1}^L |m_{kl}|}$$

2.1 INTRODUCCIÓN

- Clasificación supervisada:
 - “Uno contra uno”:
 - Ejemplo:



2.1 INTRODUCCIÓN

- Clasificación supervisada vs. Clasificación no supervisada (*clustering*):
 - *Clustering* (Clasificación no supervisada):
 - Intenta agrupar las instancias en grupos
 - Los objetos que forman cada grupo deben ser similares
 - Se examinan según algún nivel de similitud
 - Por ejemplo, la cercanía de los unos con los otros
 - No se conoce a priori el número de clases o *clusters*
 - No se conoce a qué *cluster* pertenece cada patrón
 - Objetivos:
 - Clasificación supervisada: desarrollar un modelo que, ante un nuevo patrón de clase desconocida, lo clasifique en una de las clases conocidas de forma coherente con los patrones de entrenamiento
 - Clasificación no supervisada: analizar los datos, mediante algún tipo de agrupamiento