Clase 5:
Métodos Supervisados II

1

SEBASTIÁN MALDONADO

DIPLOMADO *BUSINESS INTELLIGENCE*

03 DE ENERO, 2012

DIAPOSITIVAS: CRISTIÁN BRAVO, GASTÓN L'HUILLIER, SEBASTIÁN MALDONADO

Support Vector Machines

SEBASTIÁN MALDONADO.

Componentes Básicos SVMs

- Base de datos inicial

$$(x_1, y_1), (x_2, y_2), ..., (x_n, y_n), x \in \mathbb{R}^N, y_i \in \{+1, -1\}$$

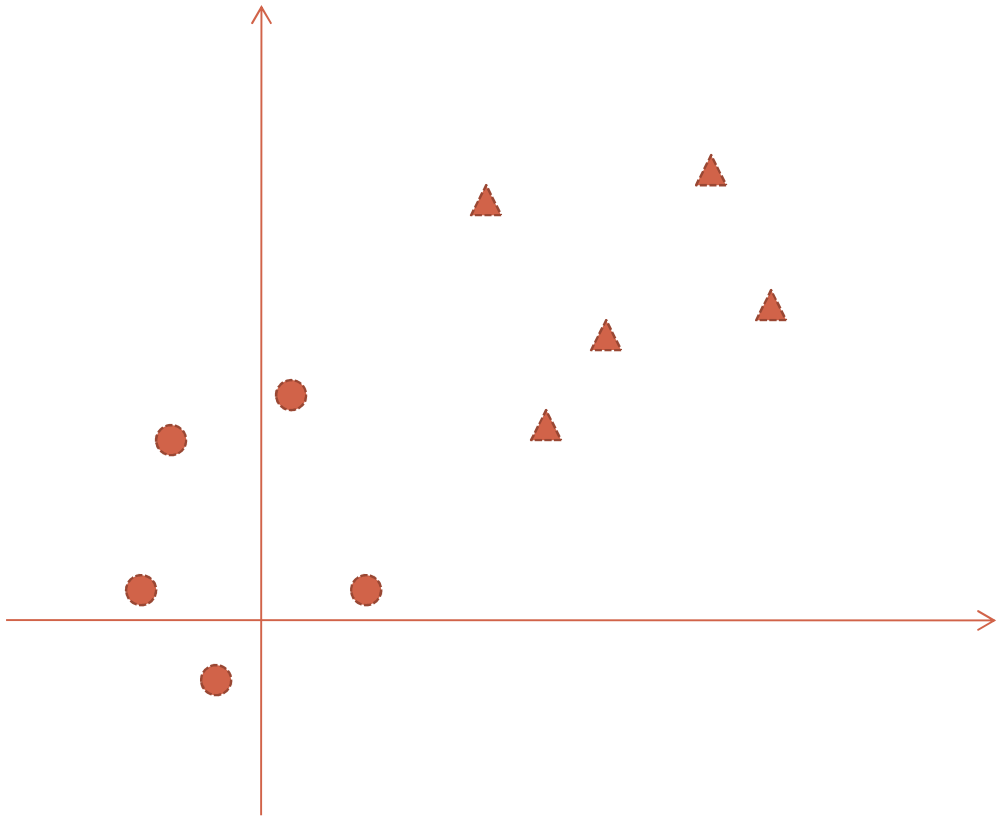
- Hiperplano separador

$$\langle w, x \rangle + b = 0$$
$$w \in \mathbb{R}^N, b \in \mathbb{R}$$

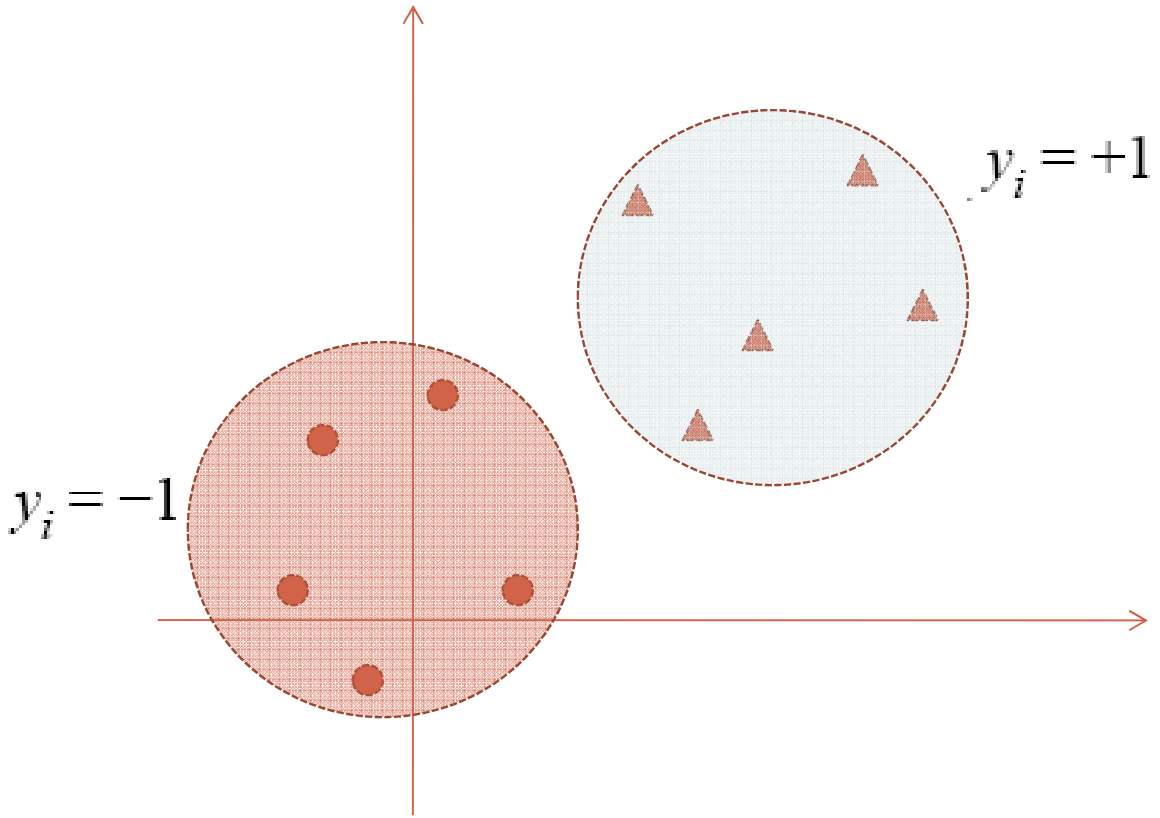
- Función de decisión

$$f(x_i) = sgn(\langle w, x_i \rangle + b)$$

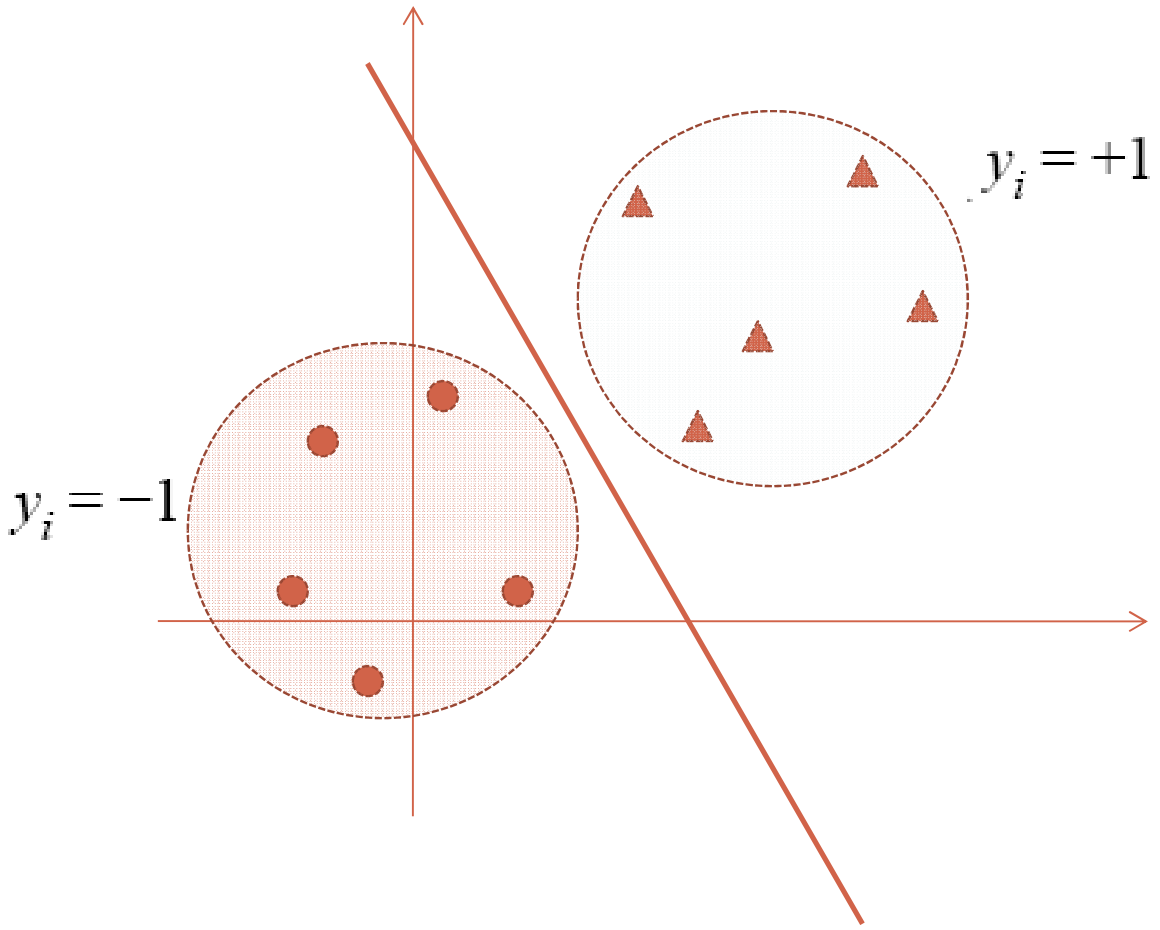
Descripción del problema



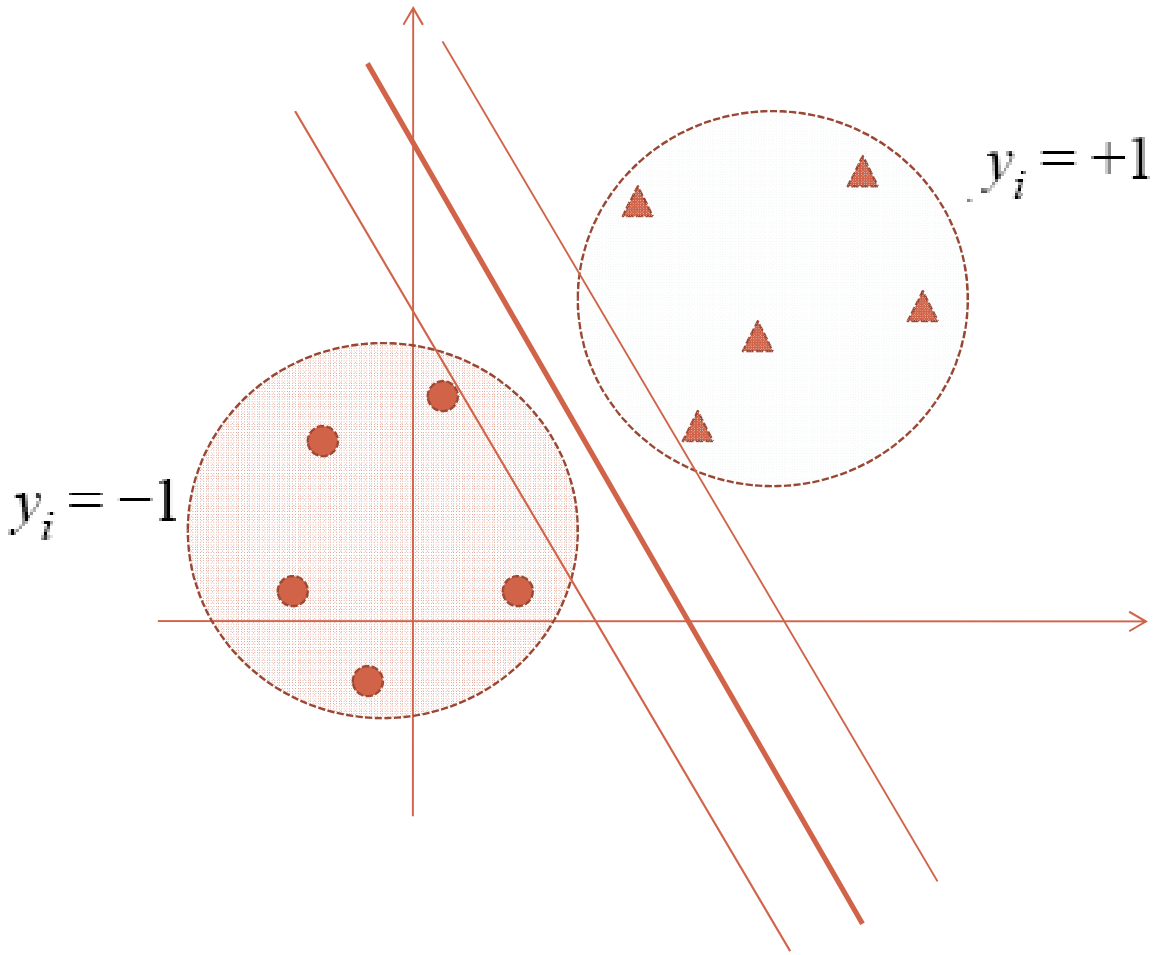
Descripción del problema (2)



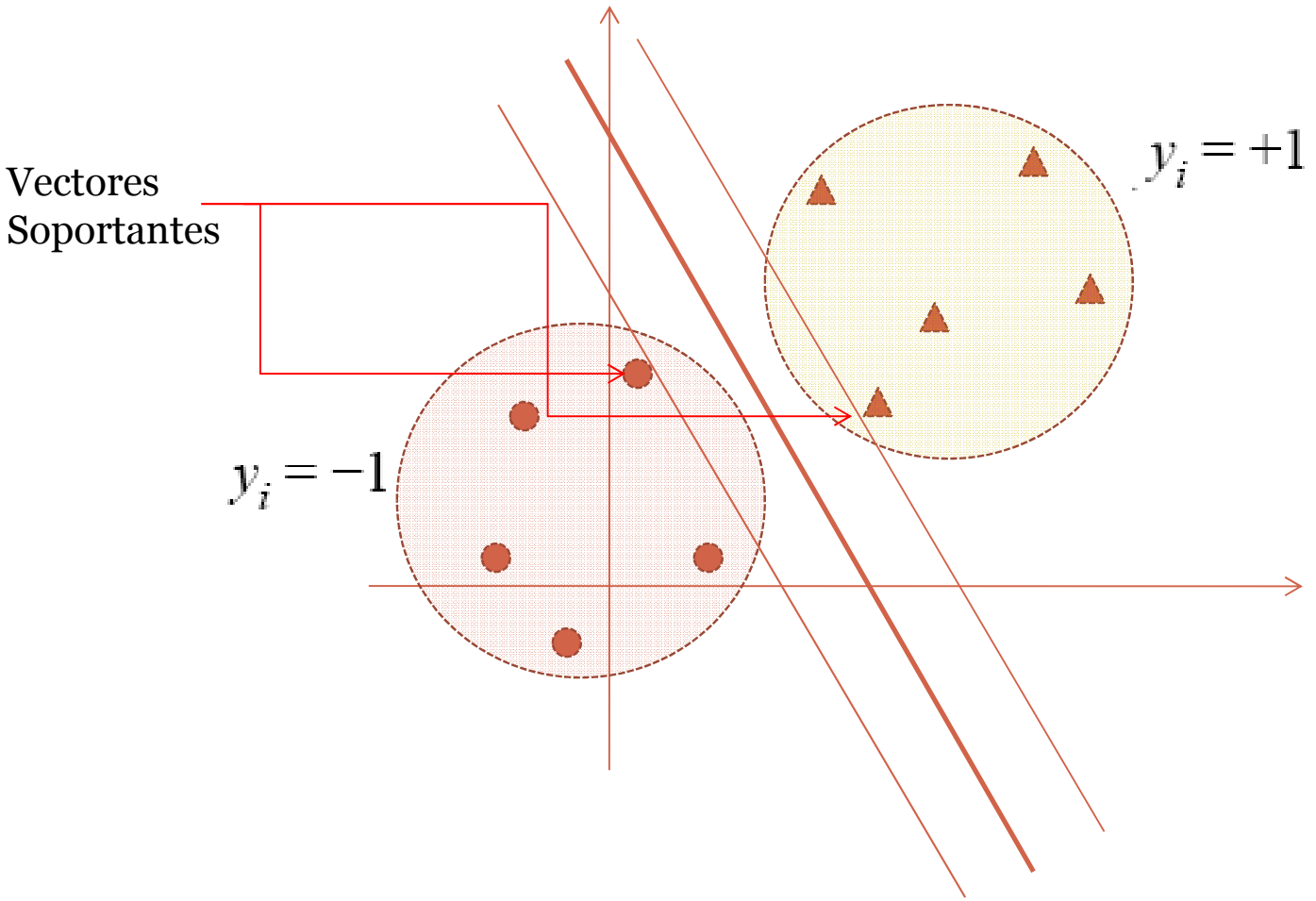
Descripción del problema (3)



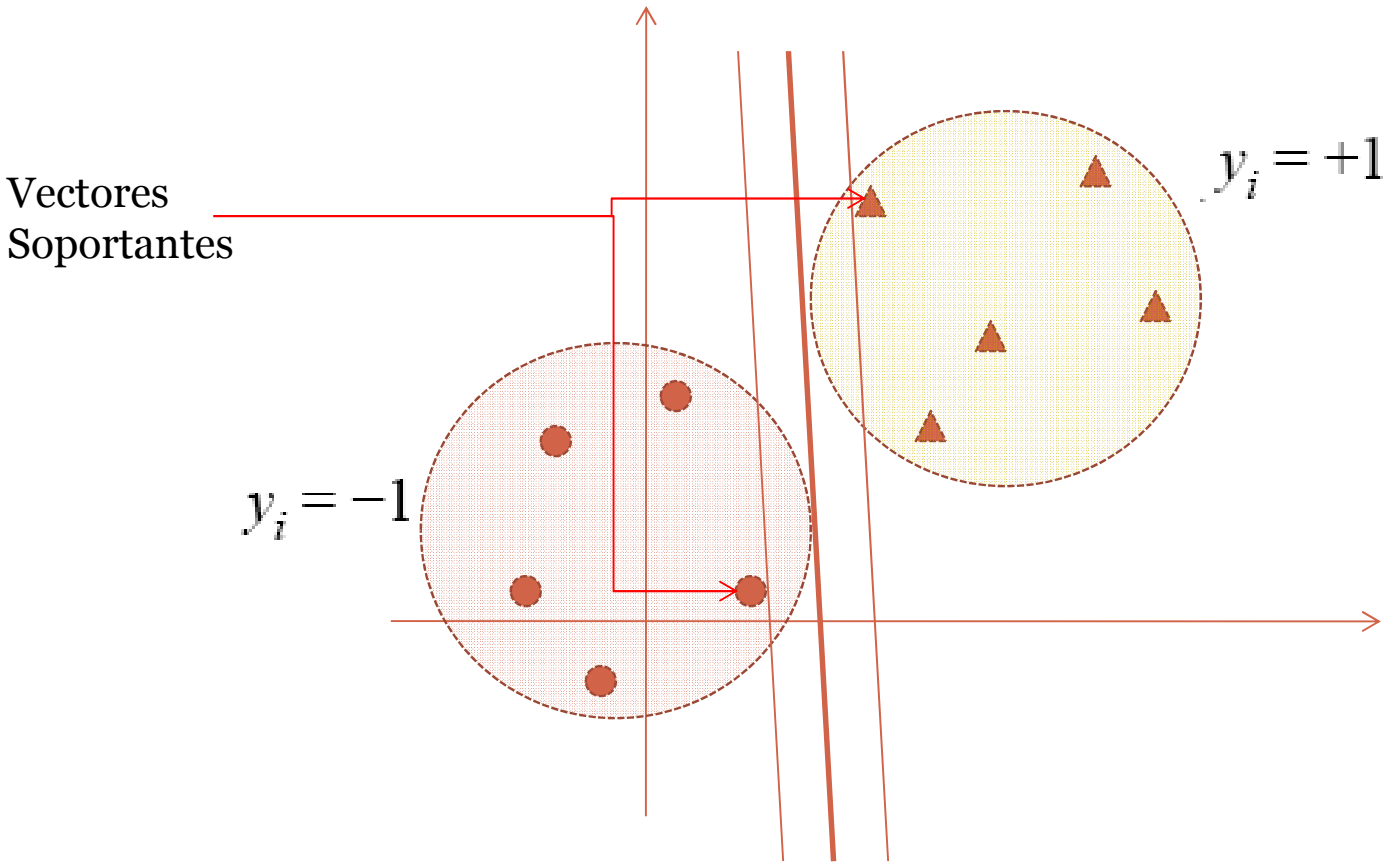
Descripción del problema (4)



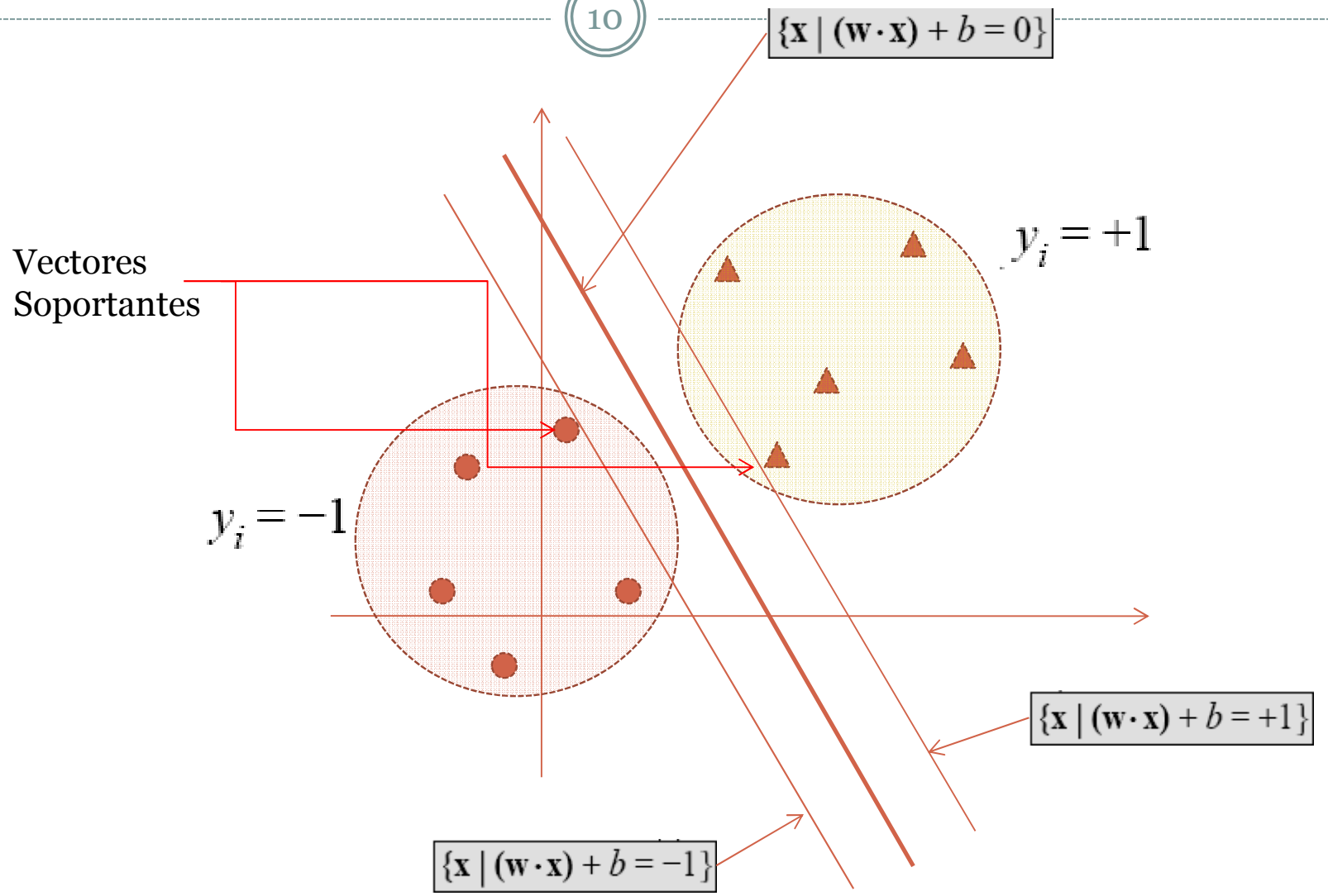
Descripción del problema (5)



Descripción del problema (6)



Descripción del problema (7)



Componentes Básicos SVMs (2)

- Incluir el "Penalización" de malas clasificaciones en el modelo. Consideramos variables de holgura para cada dato mal clasificado.
  - Datos no son perfectos, debemos considerar error para maximizar margen.
- Finalmente tenemos lo siguiente:
  - Minimizar el error en la separación de los objetos dados (del conjunto de entrenamiento)
  - Maximizar el margen de separación (mejora la generalización del clasificador en conjunto de test)
- SVMs: Algoritmo eficiente para encontrar tal hiperplano.

Formulación del problema (1)

- Formulación Primal del Problema:

$$\begin{aligned} \text{mín} \quad & \frac{1}{2}\sum_{i=1}^{n} w_i^2 + C\sum_{i=1}^{n}\xi_i \\ \text{subject to} \quad & y_i\left(\langle w^T, x_i\rangle + b\right) \geq 1 - \xi_i \\ & \xi_i \geq 0, \forall i = 1, ..., n \end{aligned}$$

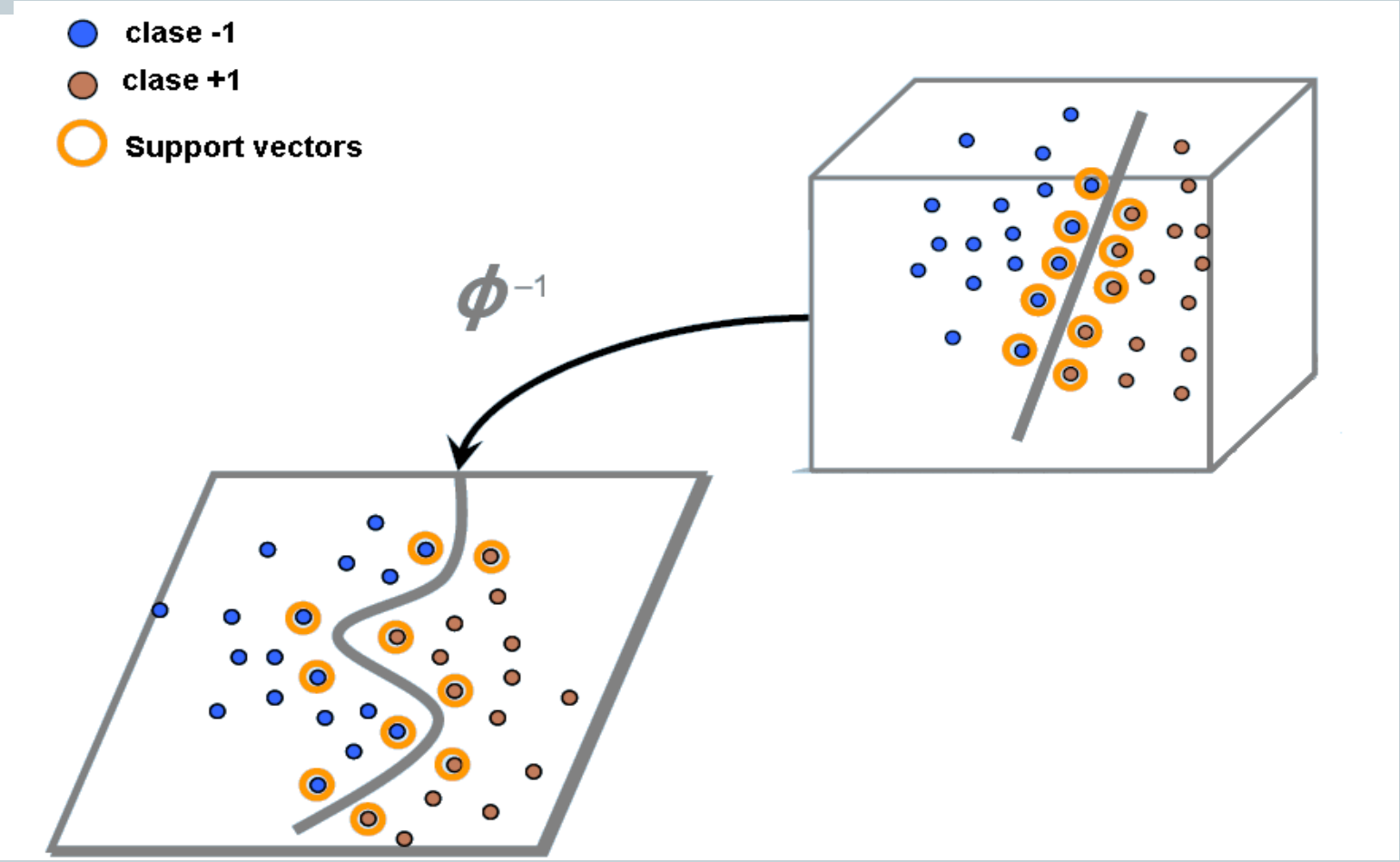
- Formulación Dual del Problema:

$$\begin{aligned} \text{máx} \quad & \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i,j=1}^{n}\alpha_i\alpha_j y_i y_j \langle x_i, x_j\rangle \\ \text{subject to} \quad & C \geq \alpha_i \geq 0, \forall i = 1, ..., n \\ & \sum_{i=1}^{n}\alpha_i y_i = 0 \end{aligned}$$

Separabilidad

- Resumiendo: Queremos utilizar hiperplanos.
  - ¡Problema!: ¿Y si los datos no son linealmente separables?
  - Teorema: Si el espacio es de dimensión infinita, todos los datos son linealmente separables.
- Solución: “Mapear” variables de un espacio a uno de mayor dimensión.
- Cuando buscamos un hiperplano en un conjunto de datos no linealmente separable, definimos una transformación que mapea los datos en otro espacio. (“Kernel Trick”)

Kernel Trick



Función Kernel (1)

- Definimos la función de “mapeo”:

$$\begin{aligned} \phi: \quad & \mathbb{R}^n \rightarrow F \\ & x \rightarrow \phi(x) \end{aligned}$$

- Definimos la función Kernel:

$$k(x, x') := (\phi(x) \cdot \phi(x'))$$

$$\begin{aligned} k: \quad & \mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathbb{R} \\ & (x, x') \rightarrow k(x, x') \end{aligned}$$

Función Kernel (2)

- Redefinimos el producto punto del problema dual por la función Kernel:

$$\begin{aligned} \text{máx} \quad & \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j k(x_i, x_j) \\ \text{subject to} \quad & C \geq \alpha_i \geq 0, \forall i = 1, ..., n \\ & \sum_{i=1}^{n} \alpha_i y_i = 0 \end{aligned}$$

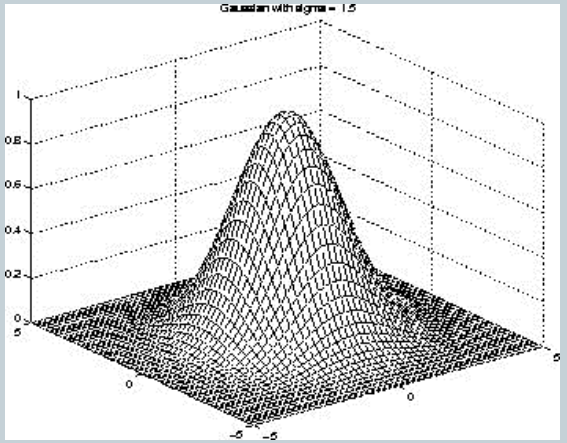
- Redefinimos la función de decisión:

$$f(x_j) = sgn(\sum_{i=1}^{n} \alpha_i y_i \cdot k(x_i, x_j) + b)$$

Ejemplos de Funciones Kernel

- "Radial Basis Function" o Kernel Gaussiano.

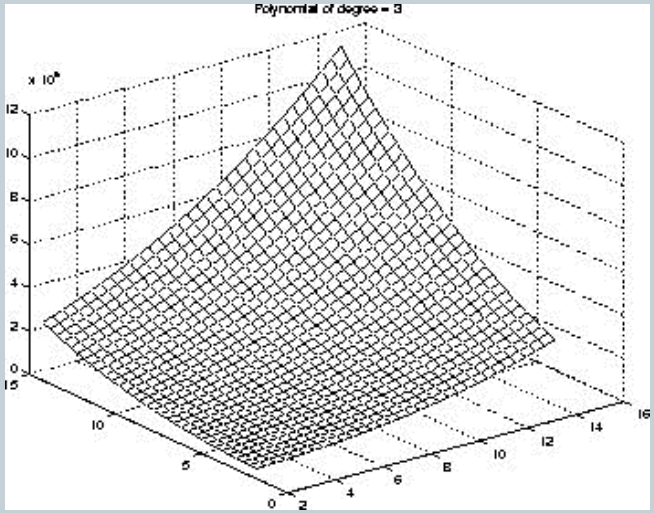
$$K(x_i, x_j) = \exp\left(-\frac{\|x_i - x_j\|^2}{\sigma^2}\right)$$

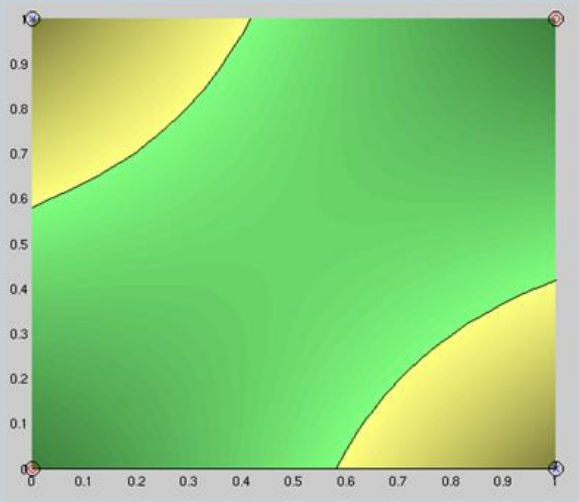


- ¡Ejemplo importante!
  - Dimensión infinita: Todo es separable por un hiperplano.
  - Parámetro permite ajustar curvatura de los datos.

Ejemplos de Funciones Kernel (2)

- Kernel Polinomial $K(x_i, x_j) = (x_i \cdot x_j + 1)^b$





- Generalización de hiperplano lineal.
- Puede servir bien para algunos dataset puntuales.

Ventajas y Desventajas

- Ventajas:
  - Potente algoritmo clasificador. Teoremas de aprendizaje son los más potentes actualmente.
  - Se puede aplicar a tipos de datos “raros”: Imágenes, texto, etc. (Filtros de Spam). Aplicables a cualquier espacio de Hilbert.
  - Flexibilidad total.
- Desventajas:
  - Gran cantidad de elementos a determinar (kernels).
  - Clasificador binario.
  - No permite clases múltiples de forma nativa.

Validación de Modelos y Diseño de Experimentos

SEBASTIÁN MALDONADO.

Validación de Modelos

- Se deben considerar varias puntos al evaluar el desempeño de un determinado modelo:
  - Estrategia de entrenamiento y prueba:
    - Holdout
    - Validación Cruzada (*cross validation*)
  - Tipos de erros y medidas de evaluación
  - Evaluación de costo de clasificación
  - Métodos de evaluación con Gráficos
    - curvas ROC

Conceptos básicos

- Datos de Entrenamiento:
  - Datos utilizados para **entrenar** el modelo
- Datos de Prueba:
  - Datos utilizados para **probar** el modelo
- Datos Objetivo (Predicción):
  - Datos sobre los cuales se ejecuta posteriormente el modelo
- Error del Modelo (Prueba):
  - Observaciones mal clasificadas sobre observaciones totales.
  - Tasa de éxito = 1 – error de testeo

Holdout

- Contando con una sola base de datos, se debe diseñar un método de medición independiente.
  - Idea intuitiva: Dividir en dos partes. Ambas **representativas**.
- Problemas:
  - Todas las clases deben estar bien representadas en ambos conjuntos.
  - Existe un *trade-off* entre la cantidad de datos considerados para el conjunto de entrenamiento y de prueba.
    - Es necesario un conjunto de entrenamiento mayor para estimar un buen modelo.
    - Es necesario un conjunto de prueba mayor para tener una buena estimación del error.

Holdout (II)

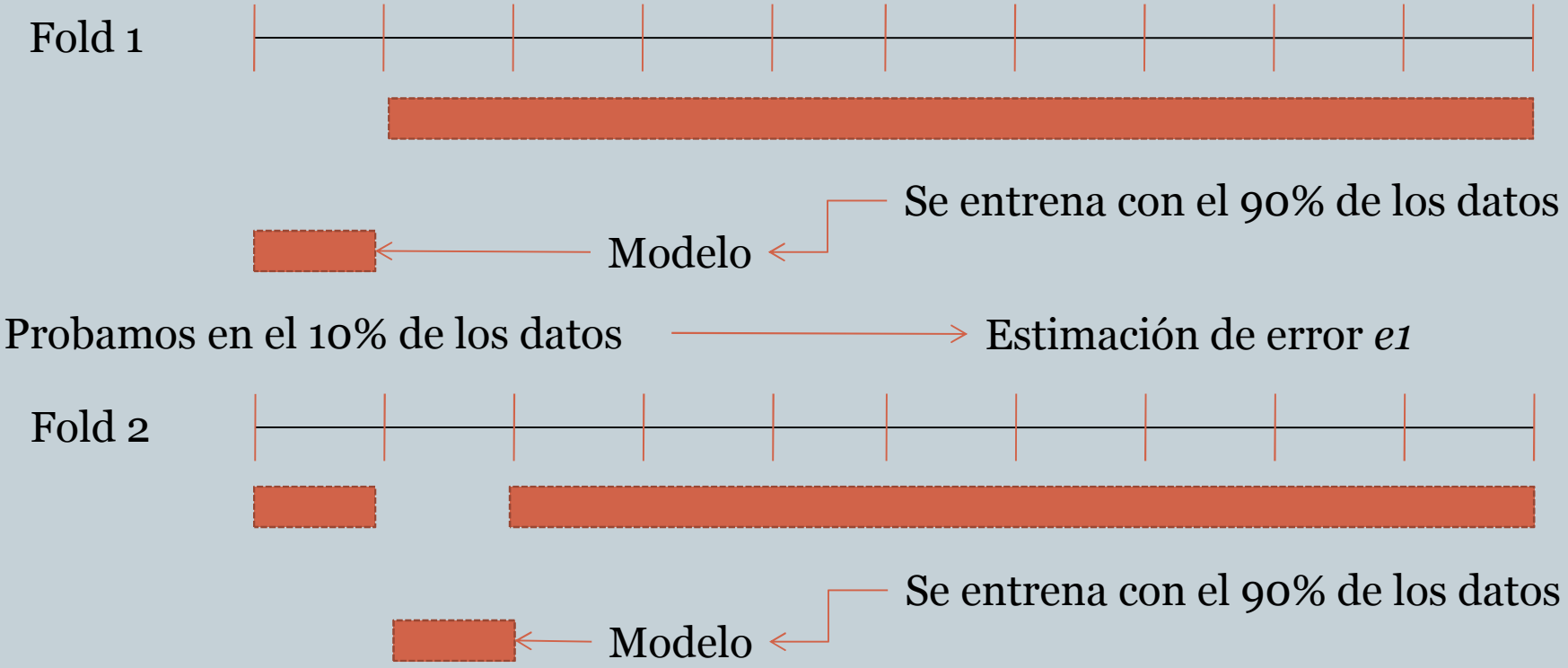
- Opciones:
  - "Holdout estratificado": Igual frecuencia de clases en cada partición Entrenamiento / Prueba.
- Se considera generalmente la regla 75% entrenamiento y 25% prueba para la división de la base de datos.
  - Esto dependerá de la cantidad de casos.
  - Intentar no dejar menos de 100 o 200 casos para validar.

Cross Validation

- Validación Cruzada de "*n-folds*"
  - Se subdividen los datos en **n** subconjuntos disjuntos.
  - Se considera la evaluación de **n-1** subconjuntos para el entrenamiento del modelo y **1** subconjunto para la prueba.
  - Repetir hasta que los **n** subconjuntos fueron evaluados como prueba.
    - Estimación del error: Promedio de los errores considerados para las **n** evaluaciones de la prueba.
    - El caso más usado es una validación cruzada de *10-folds*.

Cross Validation (II)

- Ejemplo validación Cruzada de "*10-folds*"



Cross Validation (III)

- K Validación Cruzada de "*n-folds*"
  - Se evalúa *K* veces una validación cruzada de *n-folds*.
  - Su intención es minimizar (promediando) el ruido incorporado.
- Este método es muy costoso, pero es el método estándar en los experimentos en papers.
  - Usualmente k = 100 o 1000 y n = 10.
  - La idea es que si los experimentos tienden a infinito, el promedio del error tiende al valor sistemático.

Medidas para Clasificación

- Veamos la matriz de confusión:

VP: verdadero positivo
VN: verdadero negativo
FP: falso positivo
FN: falso negativo

		Valor Observado	
		Clase 1	Clase 2
Predicción	Clase 2	VP	FP
	Clase 1	FN	VN

Medidas para Clasificación (II)

N = VP + VN + FP + FN

Tasa de éxito = (VP + VN) / N

Error de predicción = 1 – Tasa de éxito

- Otras medidas:
  - **Precision:** Porcentaje de las observaciones correctamente clasificadas como "clase 1".

$$\textbf{Precision} = VP / (VP + FP)$$

  - **Recall:** Porcentaje de todas las observaciones que deban ser clasificadas como "clase 1", sean clasificadas correctamente.

$$\textbf{Recall} = VP / (VP + FN)$$

Medidas Adicionales

- Tiempo de entrenamiento del modelo
- Tiempo de evaluación de nuevas observaciones en el modelo
- Interpretabilidad de los resultados obtenidos por el modelo
- El campo de aplicación del modelo obtenido
- Costo de obtener los datos adecuados para el modelo
- Costo de actualización del modelo
  - Estas medidas apuntan a “performance” y aplicabilidad.

Costo de Clasificación

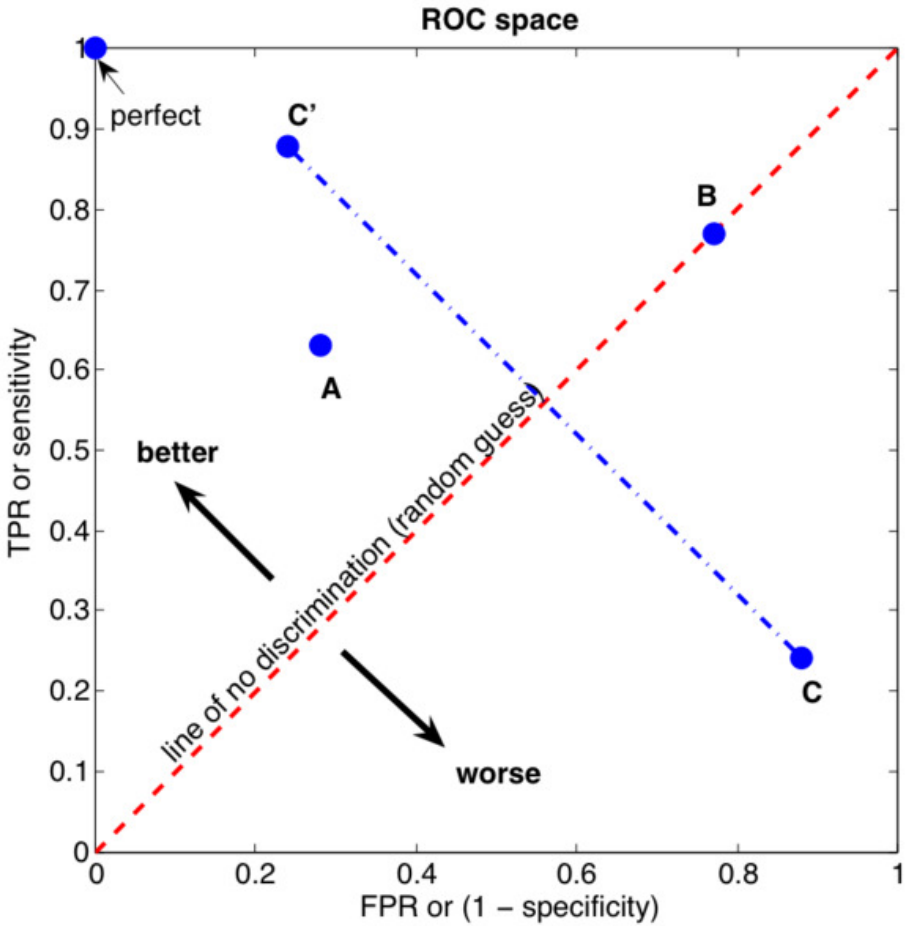
- En base a la matriz de confusión se puede generar la siguiente función de costo:

$$\text{Costo Modelo} = \text{Costo}(FP)^* FP + \text{Costo}(FN) * FN$$

- Se puede extender el concepto directamente para problemas que necesiten la estimación de múltiples clases.
- La idea es encontrar un modelo que permita **minimizar** el costo.

Curvas ROC

- Curvas ROC (Receiver Operating Characteristic)



Curvas ROC (II)

- Sensibilidad (Sensitivity): Positivos correctamente determinados.

$$Sensibility = \frac{\#true\ positives}{\#true\ positives + \#false\ negatives}$$

- Especificidad: Negativos correctamente determinados.

$$Specificity = \frac{\#true\ positives}{\#true\ positives + \#false\ negatives}$$

Multiclasificadores

SEBASTIÁN MALDONADO.

El Problema

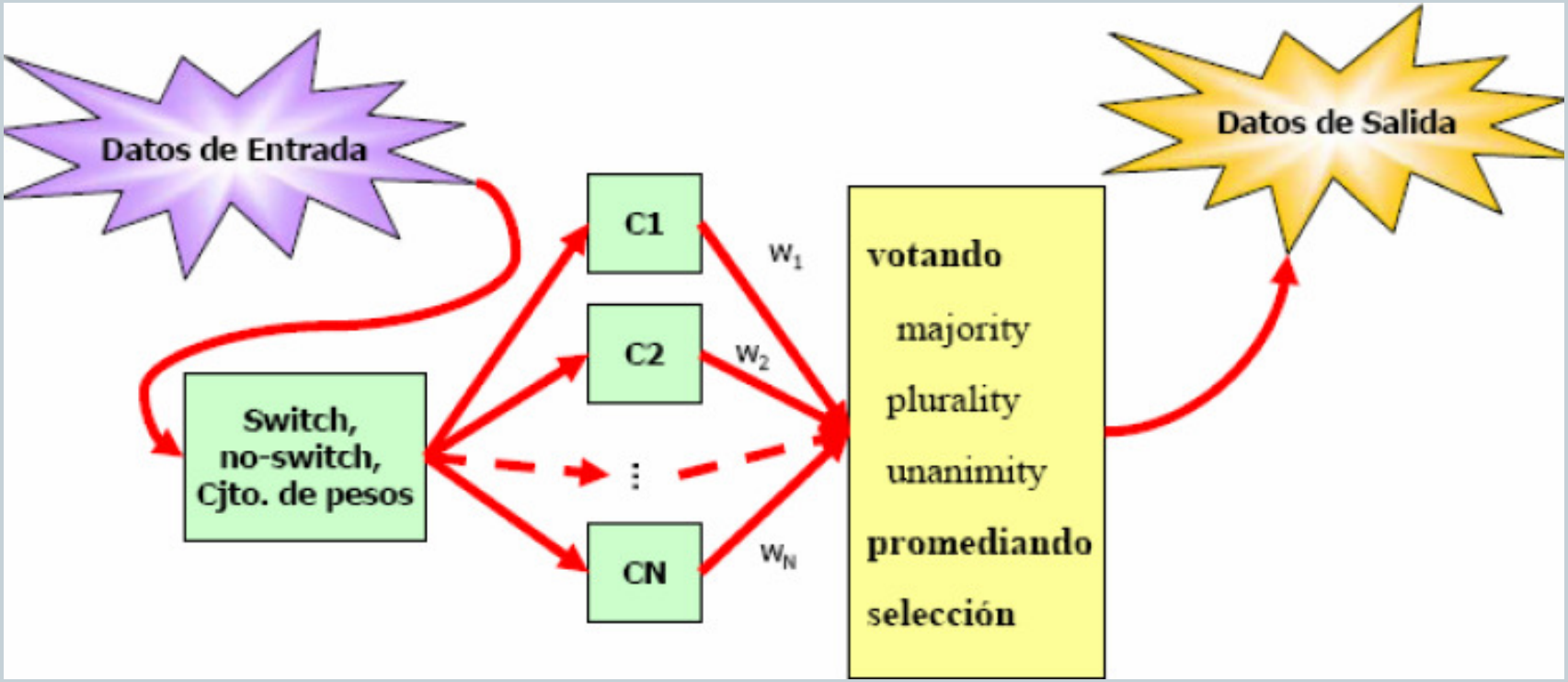
- Tenemos muchas formas de hacer Clasificaciones y Regresiones.
  - En general la idea es no quedarnos con una sola técnica, aunque sea la **mejor técnica** (que nos entregue los mejores resultados de acuerdo a una instancia del problema), o la **más efectiva** (presentando la oportunidad de tener un modelo robusto ante nuevos elementos a clasificar).

- **Idea 1:** Explotar las características de cada método para obtener mejores resultados.

- **Idea 2:** Combinarlos resultados de cada método de la manera adecuada.

¡Idea!

- "Mezclar" de alguna manera los clasificadores para lograr resultados mejorados.



Modelos para construcción de MC

1. **Subsampling** de las observaciones de entrenamiento mediante técnicas de resampling (boosting, bagging).

2. Manipulación de la **selección de atributos** para entrenar distintos modelos con conjuntos de atributos distintos.

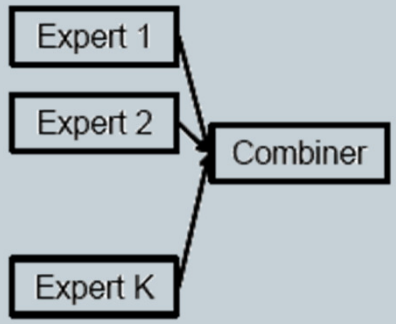
3. **Manipulación de la variable dependiente** de manera de buscar solución a problemas representados por distintos valores objetivos.

4. **Modificación de los parámetros** del clasificador de manera de obtener distintos modelos asociados a un conjunto de entrenamiento dado.

5. **Diversificación de modelos** a utilizar para determinar los valores asociados a la solución del problema objetivo.

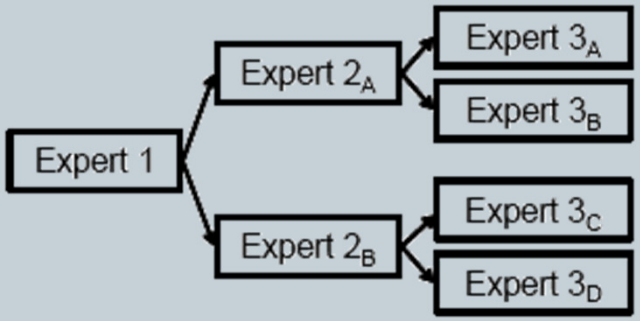
Estructuras de Múltiples Clasificadores

- **Paralelos**



- **En cascada**

- **Jerarquicos**



Estructuras de Múltiples Clasificadores (2)

- **Paralelos:** Todos los clasificadores son invocados independientemente y sus resultados son combinados a través de algún criterio adecuado.
- **En cascada:** Clasificadores son llamados de manera secuencial.
- **Jerárquicos:** Clasificadores son llamados a través de una estructura "de árbol" definida por la jerarquía.

Estrategias de Combinación

- **Estáticos**
  - **No entrenables:** La votación se realiza de manera independiente y no paramétrica el desempeño de cada clasificador. (Majority voting, promedio de resultados, etc).
  - **Entrenables:** El "combinador" inicia una nueva fase de entrenamiento de manera de mejorar el desempeño general de los modelos generados. (nuevos modelos, MIP, etc)
- **Adaptativos**
  - La función que define al "combinador" depende de los atributos iniciales considerados para los distintos modelos.

Subsampling el conjunto de entrenamiento: Bagging.

- **Bagging**
  - **(Bootstrap aggregation)** crea un multiclasificador entrenando modelos individuales en muestras derivadas de la técnica de resampling "**bootstrap**" como conjunto de entrenamiento.
  - Uno de los resultados de sampling-con-reemplazo, cada clasificador debe ser entrenado con el promedio del 63.2% de los datos de entrenamiento.
    - N observaciones tienen la probabilidad 1-(1-1/N)^N de ser seleccionadas al menos una vez en N muestras. Si N-> ∞, converge a (1-1/e) = 0.632
  - Bagging usualmente utiliza componentes de clasificación de la misma clase (e.g. árboles de decisión), y el combinador usual es de tipo "majority voting".

Subsampling el conjunto de entrenamiento: Boosting

- **Boosting**
  - Utiliza una técnica de resampling distinta a bagging, la cual mantiene una probabilidad constante de 1/N para la selección de cada observación.
  - Se va actualizando la probabilidad utilizada en el resampling en el tiempo, basado en el desempeño del modelo.
  - Basado en el concepto de "clasificador débil", donde basta un modelo que entregue resultados un poco mejor que al azar. (mayor a una clasificación de un 50%)
  - Existe una gran variante de técnicas de boosting, uno de los ejemplos más utilizados es el algoritmo Adaboost.

Boosting: Adaboost

- AdaBoost: (Adaptive boosting) Permite al experimentador ir agregando nuevas componentes de clasificación al modelo a medida que se va logrando un error más pequeño.
- Algoritmo:
  1. En la iteración n-ésima se provee al "clasificador débil" con una distribución Dn(i).
     1. Dn(i) representa la probabilidad de seleccionar la observación i-ésima.
  2. Entrenar en base formada por la distribución Dn(i).
     1. Medir tasa de error en base a un test de hipótesis Hn con respecto a Dn(i).
  3. Se genera Dn+1(i) , bajando la probabilidad de seleccionar aquellas observaciones que fueron bien clasificadas en la iteración n e incrementando la probabilidad de aquellas mal clasificadas.