SVM

Data Mining

Ester Vidaña Vila

laSalle ENG
Universitat Ramon Llull

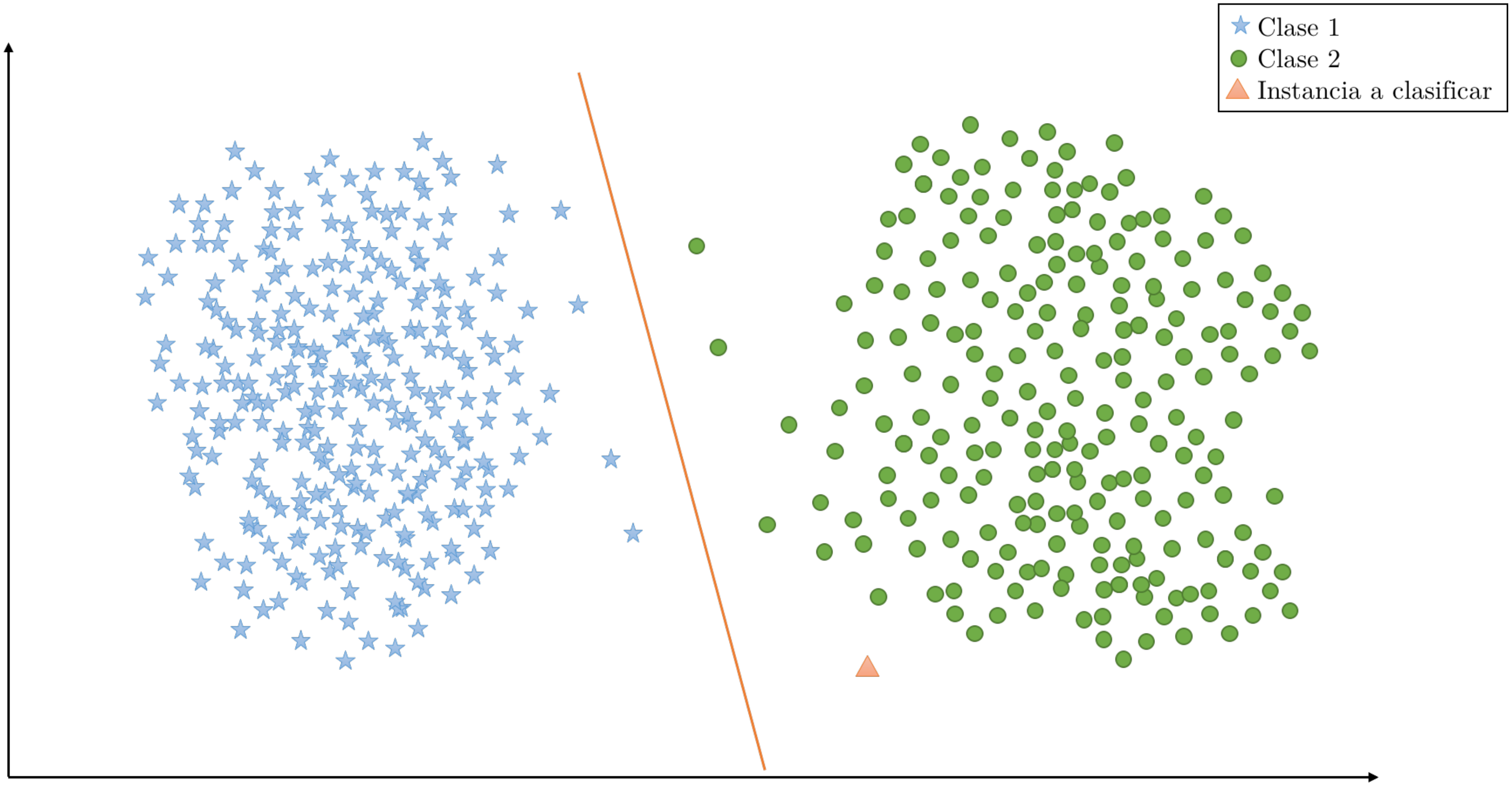
Support Vector Machines

- Es un método de clasificación que se basa en un conjunto de **hiperplanos** para caracterizar el dataset.

- Un hiperplano consiste en un sub-espacio de una dimensión menor al espacio en el que se encuentra:
  - Si el espacio fuera de 3 dimensiones, el hiperplano sería un plano 2D corriente (que dividiría en 2 el espacio tri-dimensional).
  - Si el espacio fuera de 2 dimensiones, un hiperplano se correspondería con una línea recta (que dividiría en 2 el espacio bi-dimensional).

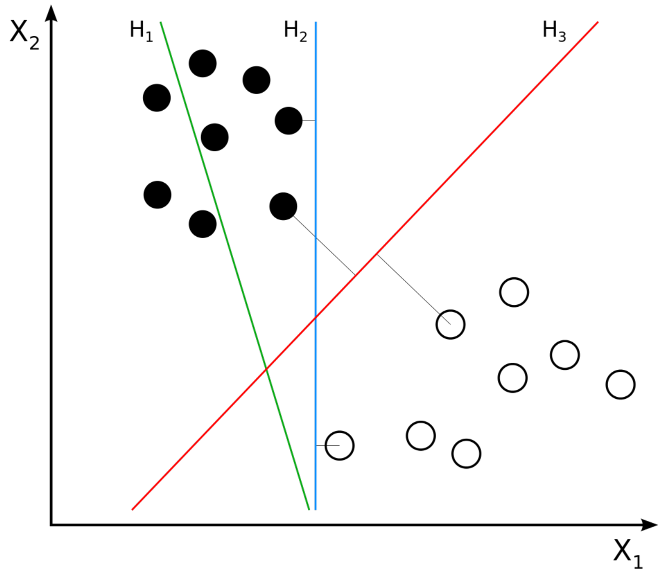
- Una vez caracterizado el espacio mediante hiperplanos, para clasificar una nueva instancia simplemente se deberá **proyectar** el dato a reconocer sobre espacio de hiperplanos creado para poder ver a qué clase pertenece.

Ejemplo de hiperplano



¿Cómo definir el **hiperplano**?

- El hiperplano podría tener muchas posiciones distintas y seguir separando completamente ambas clases, por lo que es necesario preguntarse qué hiperplano va a proporcionar **mejores resultados**.



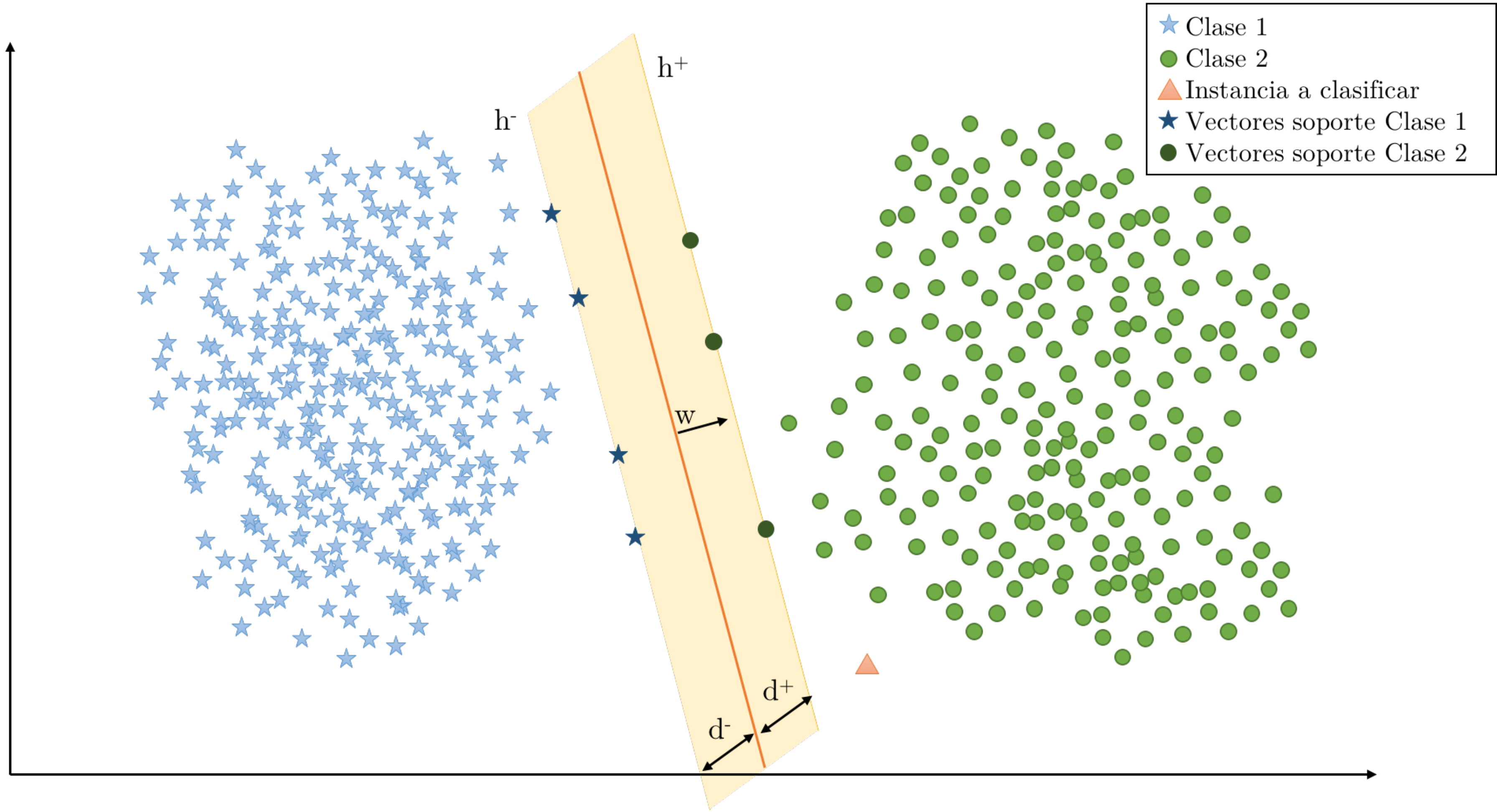
Vectores de soporte

- Vamos a necesitar los conceptos de **margen máximo** y **vectores de soporte** (o ***support vectors***).

- En este tipo de clasificador no se van a utilizar todas las muestras para calcular el hiperplano que las separa:
  - Se van a utilizar únicamente los **vectores de soporte** (que coinciden con un número limitado de muestras de entrenamiento).
  - Cada clase deberá tener un número limitado de vectores de soporte con los que calcular el hiperplano que separa ambas clases.

- La condición para que un vector se pueda considerar vector de soporte es que **la distancia entre el plano que contiene los vectores de soporte de una clase y el plano que contiene los vectores de soporte de otra clase sea máxima** y **esté vacía de muestras:**
  - El área entre los dos planos tiene que ser la mayor posible sin que dentro se encuentren vectores pertenecientes a ninguna clase.
  - Ésta distancia se denomina **margen**, por lo que es imprescindible que el margen sea máximo para encontrar el hiperplano solución.

Vectores de soporte



¿Cómo encontramos el mejor hiperplano?

- Para calcular el hiperplano, se debe enfocar el problema como un problema de **optimización** en el que la función a maximizar es, precisamente, el **margen**.
- Se va a expresar el hiperplano como:

$$\mathbf{W}^{\mathsf{T}}\mathbf{X}_i + b = 0$$

  - Siendo W el vector ortogonal al hiperplano solución y b el coeficiente de intersección.
- Las muestras que se van a clasificar correctamente van a ser aquellas que se encuentren **fuera** del margen.
  - Supongamos un margen de 1:

$$h^{+} \rightarrow \mathbf{W}^{\mathsf{T}}\mathbf{X}_i + b = +1$$

$$h^{-} \rightarrow \mathbf{W}^{\mathsf{T}}\mathbf{X}_i + b = -1$$

- Si nombramos a nuestras etiquetas:

$$\text{estrella azul} \rightarrow y_1 = +1$$

$$\text{círculo verde} \rightarrow y_2 = -1$$

- Asumiendo que las clases son linealmente separables:
  - $\vec{x}_i \cdot \vec{w} + b \geq +1$ para $y_1 = +1$
  - $\vec{x}_i \cdot \vec{w} + b \leq -1$ para $y_2 = -1$

$$\boldsymbol{y_i \cdot (\vec{x}_i \cdot \vec{w} + b) - 1 \geq 0}, \forall_i$$

¿Cómo encontramos el mejor hiperplano?

- Con un poco de álgebra, encontramos que el margen (función a maximizar), es:

$$\text{margen} = \mathrm{d}^{+} + \mathrm{d}^{-} = 2\frac{1}{\|\mathbf{W}\|}$$

- Con un poco más de álgebra, y aplicando el Lagranjiano, vemos que tenemos que minimizar:

$$\text{minimizar } \mathcal{L}_P = \frac{1}{2}\|\vec{w}\|^2 - \sum_{i=1}^{l} \alpha_i \cdot y_i \cdot (\vec{x}_i \cdot \vec{w} + b) + \sum_{i=1}^{l} \alpha_i$$

$$\textit{sujeto a las condiciones } \alpha_i \geq 0, \forall_i$$

- También podemos emplear su denominada forma dual:

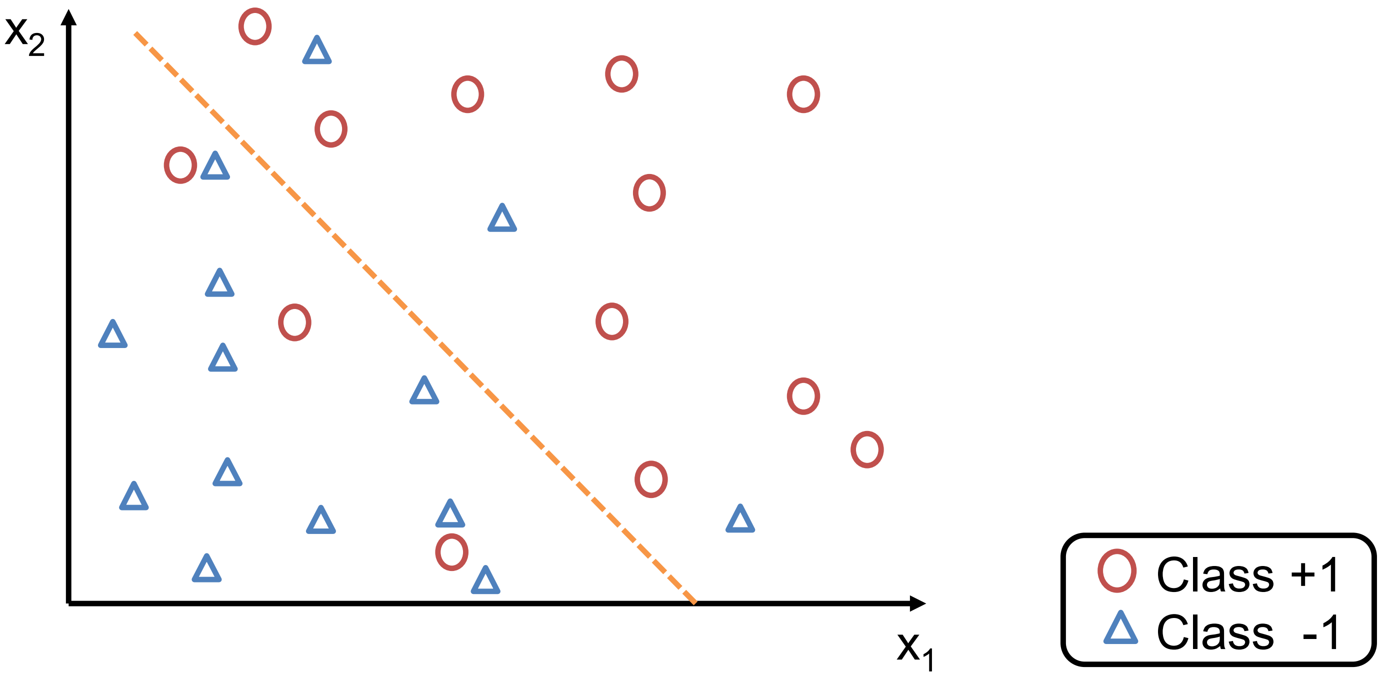
$$\text{maximizar } \mathcal{L}_D = \sum_{i=1}^{l} \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i \cdot \alpha_j \cdot y_i \cdot y_j \cdot \vec{x}_i \cdot \vec{x}_j$$

$$\textit{sujeto a las condiciones } \sum_{i=1}^{l} \alpha_i \cdot y_i = 0 \text{ y } \alpha_i \geq 0, \forall_i$$

- Este problema se puede resolver con SMO **(Sequential Minimal Optimization)**

Pero... ¿y si los datos no son separables linealmente?

- Antes hemos asumido que las clases serían separables linealmente, ¿y si no lo son?



Pero... ¿y si los datos no son separables linealmente?

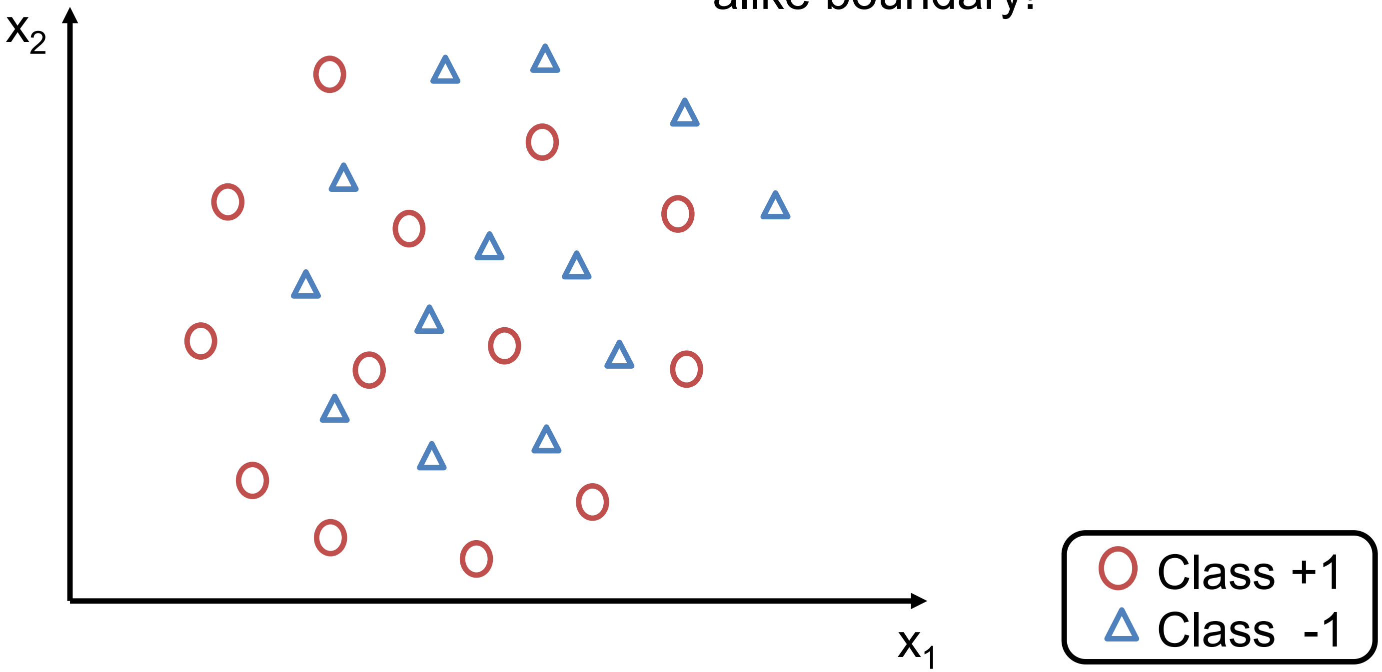
- Lo primero que podríamos pensar es en relajar un poco los márgenes y asumir algo de error. A este error, le vamos a llamar C.
- Ahora, el problema dual se transforma en:

$$\text{maximize } \mathcal{L}_D = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \cdot \alpha_j \cdot y_i \cdot y_j \cdot \vec{x}_i \cdot \vec{x}_j$$

$$\text{S}ujeto\ a\ las\ condiciones\ \sum_{i=1}^{l} \alpha_i \cdot y_i = 0 \text{ y } 0 \leq \alpha_i \leq C, \forall_i$$

Pero... ¿y si los escenarios NO son lineales?

- **No** hyperplane will separate this spiral-alike boundary!



¡Que no cunda el pánico!

- *If the world does not fit you; just move to a* ***higher-dimensional space****!*
- Podemos hacerlo añadiendo un tipo de función especial: un kernel K(·).

$$K\left(\vec{x}_i, \vec{x}_j\right) = \Phi(\vec{x}_i) \cdot \Phi(\vec{x}_j)$$

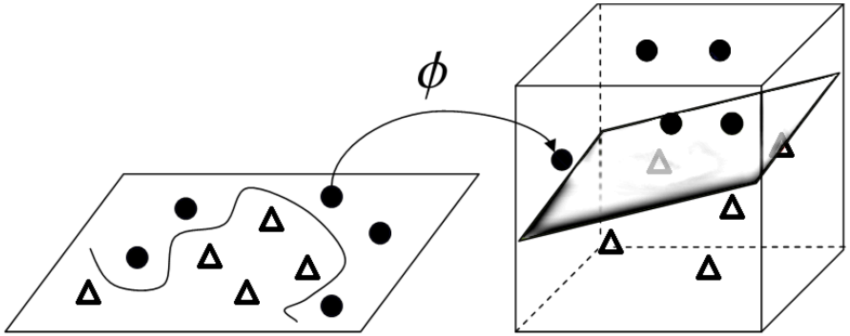
- El teorema de Mercer nos dice que podemos expresar este mapeo como un **producto escalar**
- La función de kernel mapeará los datos “de nuestro mundo” a otro con infinitos números de dimensiones: nuestros datos del mundo real

$$\text{maximize } \mathcal{L}_D = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \cdot \alpha_j \cdot y_i \cdot y_j \cdot K\left(\vec{x}_i, \vec{x}_j\right)$$

Subject to constraints $\sum_{i=1}^{l} \alpha_i \cdot y_i = 0$ and $0 \leq \alpha_i \leq C, \forall_i$

- Las funciones de kernel son computacionalmente baratas de calcular.
  - SMO can handle this for you; nice!

Visualmente



- Las **funciones de Kernel** pueden ser muchas. Se podría buscar una función de Kernel para cada problema concreto de clasificación, pero se ha demostrado que hay funciones que generalmente proporcionan buenos resultados y son muy habituales. Por ejemplo:

1. *Lineal*: $\mathrm{K}(\mathbf{X}, \mathbf{Z}) = <\mathbf{X}, \mathbf{Z}>$.
2. *RBF*: *Radial Basis Function Kernel*: $\mathrm{K}(\mathbf{X}, \mathbf{Z}) = e^{-\gamma ||\mathbf{X}-\mathbf{Z}||^2}$.
3. Sigmoide: $\mathrm{K}(\mathbf{X}, \mathbf{Z}) = \tanh(\gamma < \mathbf{X}, \mathbf{Z} > +r)$
4. Polinomial: $\mathrm{K}(\mathbf{X}, \mathbf{Z}) = (\gamma < \mathbf{X}, \mathbf{Z} > +r)^d$.

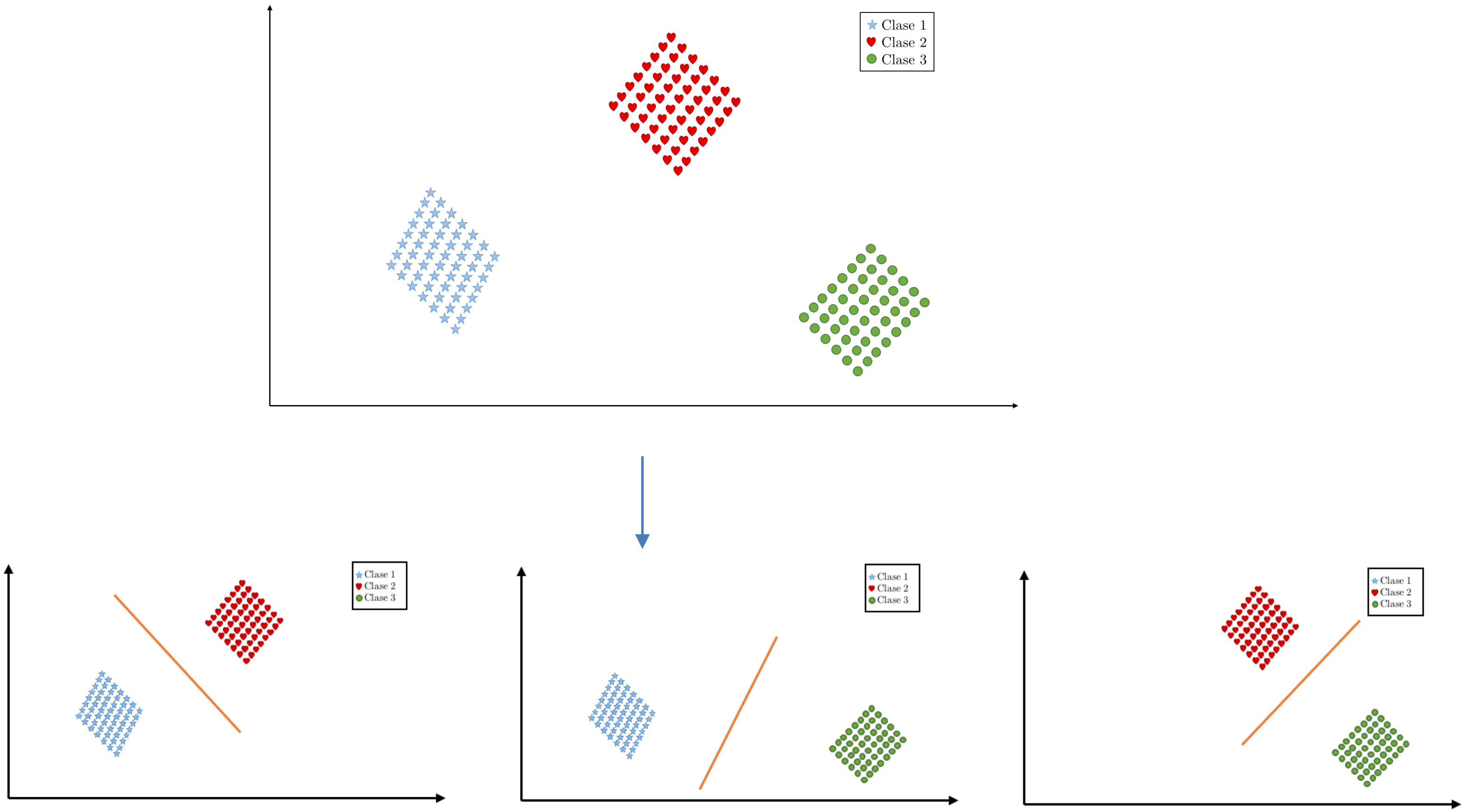
* Los parámetros r, d y gamma se pueden ajustar.

¿Y si tenemos más de dos clases?

- Hay dos tipos de técnicas que se pueden aplicar para llevar a cabo una clasificación **multiclase:**

1. One vs. One: Se aplica un clasificador binario para cada combinación posible de dos clases.
   - Se tienen que utilizar n(n-1)/2 classificadores binarios.
   - Tras aplicar los distintos clasificadores, se va a tener un total de n(n-1)/2 3 resultados. Para saber qué resultado es el ganador, se hace por majority voting.
2. One vs. All: Más habitual.
   - Se entrena el clasificador con n clasificadores distintos.
   - Para cada clasificador, se utiliza una clase separada por un hiperplano del resto de clases y se realiza el proceso de clasificación.
   - En este caso, en el proceso de clasificación que lleva a cabo cada clasificador no se guarda únicamente una etiqueta de clase como ganadora, sino que se obtiene un valor de clasificación normalizado según si el valor a clasificar se encuentra en la clase que está sola o si se encuentra en la clase del clasificador que contiene el resto de clases.
   - También se tiene en cuenta la distancia entre el hiperplano y la muestra.

Ejemplo One vs. One:



Ejemplo One vs. All: