Support Vector Machines

Support Vector Machines (SVMs)

- One of the most successful machine learning algorithms
- Three main ideas:
  - **Linearity**
  - **Sparseness**
  - **Kernel** trick
- Also called **Sparse Kernel Machines**

Pattern classification

- Consider the classification problem: $\{(\mathbf{x_1}, t_1), (\mathbf{x_2}, t_2), ..., (\mathbf{x_n}, t_n)\}$
  - n is the number of training patterns
  - $\mathbf{x_i}$ is the attribute vector for pattern i
  - t_i is the class label for pattern i, $t_i \in \{-1, 1\}$
- A classifier is a function $f(\mathbf{x}, \Theta)$ that assigns each $\mathbf{x_i}$ an estimation of its class $y_i = f(\mathbf{x_i}, \Theta)$
- We usually train the classifier parameters Θ in order to minimize a risk function defined over the training data:

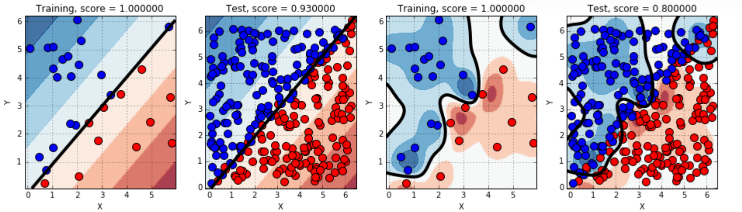
$$R_{train}[f] = \frac{1}{n} \sum_{i=1}^{n} C(y_i, t_i)$$

- Where $C(y, t)$ is a cost function, usually the mean squared error:

$$C(y, t) = (y - t)^2$$

Complexity and overfitting

- When the number of training patterns is small, we may obtain a classifier that **overfits** the training data and has a poor generalization capability
- How can we prevent overfitting?
  - A common approach involves controlling the **model complexity**: a simpler model is preferred over a more complex one as far as they both provide a similar classification accuracy



How to measure the model complexity

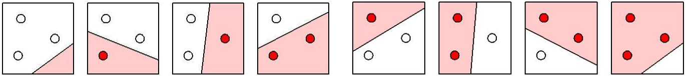
- The Vapnik-Chervonenkis (VC) dimension measures the complexity of a given family of functions $f(\mathbf{x}; \mathbf{\Theta})$
  - f represents the family
  - $\mathbf{\Theta}$ is the set of parameters
- The VC dimension of a family $f(\mathbf{x}; \mathbf{\Theta})$ is defined as the maximum number of patterns that can be explained by this family
- More complex families are able to fit more complex data sets, but they present a lower generalization capability

The VC dimension - Definition

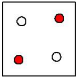
- **Shattering**:
  - Consider a dataset with n patterns $\{\mathbf{x_1}, \mathbf{x_2}, ..., \mathbf{x_n}\}$ belonging to 2 different classes
  - There exist 2^n different ways to assign the class labels
  - For example, if $n = 3$ there are 8 different such class assignments: $\{(-1,-1,-1),(-1,-1,1),...,(1,1,1)\}$
  - The family of functions $f(\mathbf{x};\mathbf{\Theta})$ **shatters** the dataset if for any possible class assignment α there exists a set of parameters $\mathbf{\Theta}_\alpha$ such that $f(\mathbf{x};\mathbf{\Theta}_\alpha)$ solves it
- The **VC dimension** of the family $f(\mathbf{x};\mathbf{\Theta})$ is defined as the size of the largest set which can be shattered by $f(\mathbf{x};\mathbf{\Theta})$
  - If the VC dimension of $f(\mathbf{x};\mathbf{\Theta})$ is h, then there exists at least one set with h points which can be shattered by $f(\mathbf{x};\mathbf{\Theta})$

The VC dimension - Example

- Example
  - Consider the family $f(\mathbf{x}; \boldsymbol{\Theta})$ of hyperplanes in $\mathbb{R}^2$
  - $f(\mathbf{x}; \boldsymbol{\Theta}) = w_0 + w_1 x_1 + w_2 x_2$
  - $\boldsymbol{\Theta} = (w_0, w_1, w_2)$
- It is possible to find a set of $n = 3$ points that is shattered using hyperplanes (all different class assignments are solved)



- But this is not possible for $n = 4$



- So the VC dimension of the family of hyperplanes in $\mathbb{R}^2$ is 3

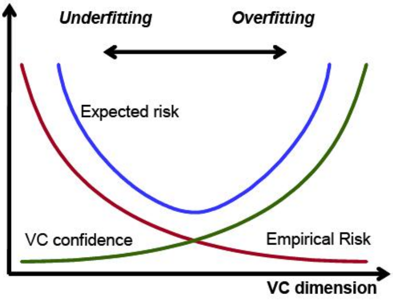
Structural Risk Minimization (I)

- Vapnik & Chervonenkis (1974)
- To obtain an optimal classifier we should balance the empirical risk measured on the training data and the VC dimension of the model
- With probability $1-\eta$, the expected risk is upper bounded by:

$$E[R[f]] \leq R_{train}[f] + \sqrt{\frac{h(\log \frac{2n}{h} + 1) - \log \frac{\eta}{4}}{n}}$$

  where
  - h is the VC dimension of f
  - n is the number of training patterns
  - $n > h$
- The second term is called **VC confidence**
- When n/h increases, VC decreases and the empirical risk becomes a better approximation of the expected risk

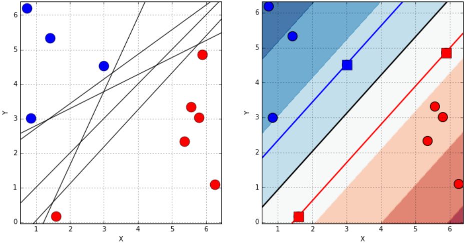
Structural Risk Minimization (II)



- We should select the model with the lowest upper bound to the expected risk
- In practical terms, computing the VC dimension is not feasible in most situations
- Linear models are an exception

Optimal separating hyperplane (I)

- Consider the problem $\{(\mathbf{x_1}, t_1), (\mathbf{x_2}, t_2), ..., (\mathbf{x_n}, t_n)\}$
  - n patterns, 2 classes, $t_i \in \{-1, 1\}$, linearly separable
- Which is the **optimal separating hyperplane**?
- It seems reasonable to maximize the **margin** (minimum distance from any point to the decision boundary)
  - The higher the margin is, the more tolerant our model is to statistical fluctuations (higher generalization capability)



Optimal separating hyperplane (II)

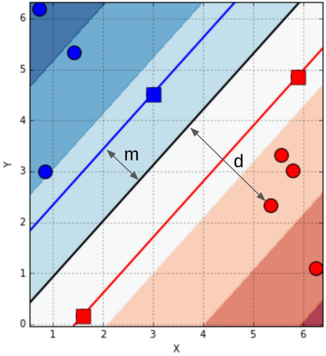
- This intuition is supported by the results of SRM
- The VC dimension of a separating hyperplane with margin m is bounded by the following upper bound:

$$h \leq \text{mín}(\lceil \frac{R^2}{m^2} \rceil, d) + 1$$

  - d is the dimension
  - R is the radius of the smallest hypersphere that contains all data points
- When we maximize the margin we are minimizing the VC dimension, and so increasing the generalization capability of the model
- If the margin is large enough the VC dimension, and so the model complexity, can be small even when the dimension d is very large

Maximum margin hyperplane (I)

- We want to find the separating hyperplane $\mathbf{w}^t\mathbf{x} + b = 0$ that maximizes the margin



- The distance from point $\mathbf{x}_i$ to the hyperplane is given by:

$$d = \frac{|\mathbf{w}^t\mathbf{x}_i + b|}{||\mathbf{w}||}$$

- **Canonical hyperplane**: $|\mathbf{w}^t\mathbf{x} + b| = 1$ for the closest points
- Using this canonical representation, the margin is

$$m = \frac{1}{||\mathbf{w}||}$$

Maximum margin hyperplane (II)

The problem of maximizing the margin is then equivalent to the following

Optimization problem

- Minimize (with respect to $\mathbf{w}$ and b):

$$J(\mathbf{w}) = \frac{1}{2}||\mathbf{w}||^2$$

- Subject to the constraints $t_i(\mathbf{w}^t\mathbf{x}_i + b) \geq 1 \ \ \forall i$

- To solve this problem we introduce a Lagrange multiplier $\alpha_i \geq 0$ for each of the constraints and obtain the Lagrangian function:

$$L(\mathbf{w}, b, \alpha) = \frac{1}{2}||\mathbf{w}||^2 - \sum_{i=1}^{n} \alpha_i[t_i(\mathbf{w}^t\mathbf{x}_i + b) - 1]$$

Maximum margin hyperplane (III)

$$L(\mathbf{w}, b, \alpha) = \frac{1}{2}||\mathbf{w}||^2 - \sum_{i=1}^{n} \alpha_i [t_i(\mathbf{w}^t \mathbf{x}_i + b) - 1]$$

The solution to the original optimization problem can be obtained by optimizing the Lagrangian function $L(\mathbf{w}, b, \alpha)$ with respect to $\mathbf{w}$, b and α_i subject to the

Karush-Kuhn-Tucker (KKT) conditions

$$\alpha_i \geq 0$$

$$t_i(\mathbf{w}^t \mathbf{x}_i + b) - 1 \geq 0$$

$$\alpha_i [t_i(\mathbf{w}^t \mathbf{x}_i + b) - 1] = 0$$

- $\alpha_i = 0$ implies $t_i(\mathbf{w}^t \mathbf{x}_i + b) - 1 > 0$ (**inactive** constraint)
- $\alpha_i > 0$ implies $t_i(\mathbf{w}^t \mathbf{x}_i + b) - 1 = 0$ (**active** constraint)

The dual problem (I)

- Setting the gradient of $L(\mathbf{w}, b, \alpha)$ with respect to $\mathbf{w}$ and b equal to 0 we get

$$\mathbf{w} = \sum_{i=1}^{n} \alpha_i t_i \mathbf{x}_i$$

$$\sum_{i=1}^{n} \alpha_i t_i = 0$$

- And substituting these expressions back into $L(\mathbf{w}, b, \alpha)$ we obtain the **dual problem**

The dual problem (II)

Dual problem

- Maximize with respect to α_i:

$$\tilde{L}(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j t_i t_j \mathbf{x}_i \mathbf{x}_j$$

- Subject to the constraints:

$$\alpha_i \geq 0$$

$$\sum_{i=1}^{n} \alpha_i t_i = 0$$

Support vectors (I)

- Recall the KKT conditions:

$$\alpha_i \geq 0$$

$$t_i(\mathbf{w}^t \mathbf{x}_i + b) - 1 \geq 0$$

$$\alpha_i [t_i(\mathbf{w}^t \mathbf{x}_i + b) - 1] = 0$$

- For any $\mathbf{x}_i$, one and only one of the following two conditions holds:
  - $\alpha_i = 0$; these points do not contribute to the definition of the separating hyperplane
  - $t_i(\mathbf{w}^t \mathbf{x}_i + b) = 1$; these points define the separating hyperplane, they are called **support vectors**

Support vectors (II)

- Only support vectors are needed to define the optimal separating hyperplane
- The vector $\mathbf{w}$ is obtained as

$$\mathbf{w} = \sum_{i=1}^{n} \alpha_i t_i \mathbf{x}_i$$

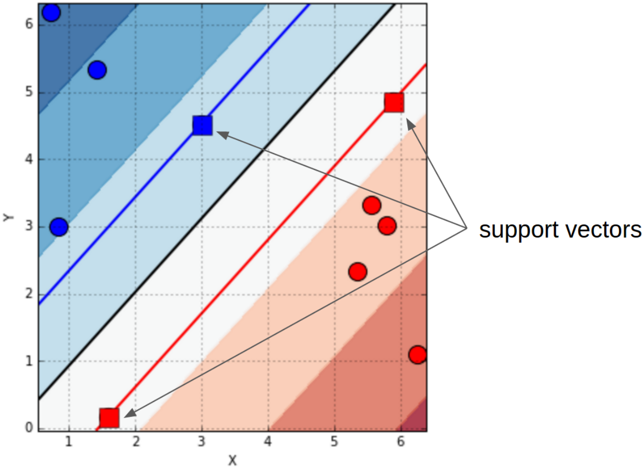
- The parameter b can then be obtained from any support vector using

$$t_i(\mathbf{w}^t \mathbf{x}_i + b) = 1$$

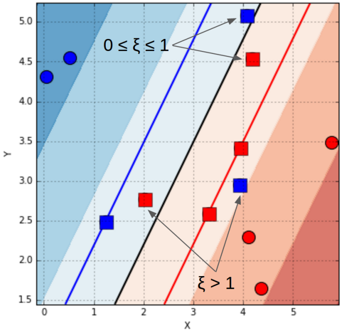
- Note that only support vectors are necessary to perform classification

$$f(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i t_i \mathbf{x}_i \mathbf{x} + b$$

Support vectors (III)



Non linearly separable problems (I)



- We introduce the slack variables $\xi_i \geq 0$
- Now the constraints are

$$t_i(\mathbf{w}^t \mathbf{x}_i + b) \geq 1 - \xi_i$$

- $\xi_i = 0$ for points out of the margin that are correctly classified:

$$t_i(\mathbf{w}^t \mathbf{x}_i + b) \geq 1$$

- $0 \leq \xi_i \leq 1$ for points inside the margin that are correctly classified
- $\xi_i > 1$ for points that are not correctly classified

- **New goal:** to maximize the margin while penalizing wrongly classified patterns

Non linearly separable problems (II)

Optimization problem

- Minimize with respect to $\mathbf{w}$, b and ξ:

$$J(\mathbf{w}, \xi) = C \sum_{i=1}^{n} \xi_i + \frac{1}{2} ||\mathbf{w}||^2$$

- Subject to the constraints:

$$t_i(\mathbf{w}^t \mathbf{x}_i + b) \geq 1 - \xi_i$$

$$\xi_i \geq 0$$

- $\sum_{i=1}^{n} \xi_i$ is an upper bound to the total number of errors
- The C parameter controls the relative weight given to the training classification error and to the complexity (margin)
  - Higher C favours models with smaller error
  - Lower C favours simpler models

Non linearly separable problems (III)

- As before, we introduce Lagrange multipliers α_i and μ_i

$$L(\mathbf{w}, b, \alpha, \mu) =$$

$$\frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{n}\xi_i - \sum_{i=1}^{n}\alpha_i[t_i(\mathbf{w}^t\mathbf{x}_i + b) - 1 + \xi_i] - \sum_{i=1}^{n}\mu_i\xi_i$$

- The KKT conditions are now:

$$\alpha_i \geq 0$$

$$t_i(\mathbf{w}^t\mathbf{x}_i + b) - 1 + \xi_i \geq 0$$

$$\alpha_i[t_i(\mathbf{w}^t\mathbf{x}_i + b) - 1 + \xi_i] = 0$$

$$\mu_i \geq 0$$

$$\xi_i \geq 0$$

$$\mu_i\xi_i = 0$$

Non linearly separable problems (IV)

- Setting the gradient of L wrt $\mathbf{w}$ equal to 0 we get:

$$\mathbf{w} = \sum_{i=1}^{n} \alpha_i t_i \mathbf{x}_i$$

- Setting the derivative of L wrt b equal to 0 we get:

$$0 = \sum_{i=1}^{n} \alpha_i t_i$$

- Setting the derivative of L wrt ξ_i equal to 0 we get:

$$\alpha_i = C - \mu_i$$

- Substituting this expressions in L we get the **dual problem**

The dual problem

Dual problem

- Maximize with respect to α_i:

$$\tilde{L}(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j t_i t_j \mathbf{x}_i \mathbf{x}_j$$

- Subject to the constraints:

$$0 \leq \alpha_i \leq C$$

$$\sum_{i=1}^{n} \alpha_i t_i = 0$$

- The problem is esentially the same as in the linearly separable case, but with different constraints

Support vectors (I)

As before, we have:

- $\alpha_i = 0$ for points out of the margin that are correctly classified
  - These points do not contribute to the definition of the separating hyperplane
- The rest of the points are **support vectors**
  - They satisfy:

$$t_i(\mathbf{w}^t \mathbf{x}_i + b) = 1 - \xi_i$$

$$\alpha_i > 0$$

Support vectors (II)

- Support vectors satisfy $t_i(\mathbf{w}^t\mathbf{x}_i + b) = 1 - \xi_i$, with $\alpha_i > 0$
- Two possibilities:
  - $\alpha_i < C$, $\mu_i > 0$ and $\xi_i = 0$; these points are **on** the margin
  - $\alpha_i = C$, $\mu_i = 0$ and $\xi_i > 0$; these points are **inside** the margin (correctly classified if $\xi_i \leq 1$, wrongly classified if $\xi_i > 1$)
- The separating hyperplane is given by:

$$\mathbf{w} = \sum_{i=1}^{n} \alpha_i t_i \mathbf{x}_i$$

- With b obtained from any support vector with $\alpha_i < C$

$$t_i(\mathbf{w}^t\mathbf{x}_i + b) = 1$$

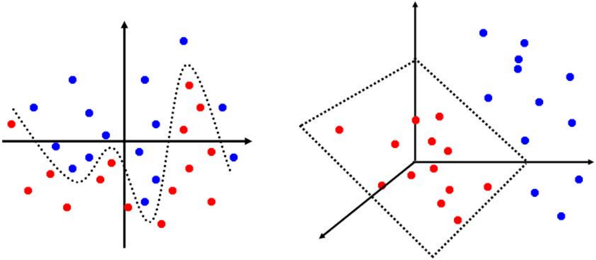
- We only need the support vectors to perform classification

Support vectors (III)

α	$\mu = C - \alpha$	ξ	$t(\mathbf{w}^t\mathbf{x} + b)$	Type
$\alpha = 0$	$\mu > 0$	$\xi = 0$	$t(\mathbf{w}^t\mathbf{x} + b) > 1$	Well classified, out of the margin
$0 < \alpha < C$	$\mu > 0$	$\xi = 0$	$t(\mathbf{w}^t\mathbf{x} + b) = 1$	Well classified, on the margin
$\alpha = C > 0$	$\mu = 0$	$0 < \xi \leq 1$	$t(\mathbf{w}^t\mathbf{x} + b) \geq 0$	Well classified, inside the margin
		$\xi > 1$	$t(\mathbf{w}^t\mathbf{x} + b) < 0$	Wrongly classified point

Non-linear problems (I)

- **Cover's theorem:** A classification problem which is projected onto a high dimensional space is more likely to be linearly separable
- Using this idea, the SVMs perform two steps:
  1. They make a non linear projection of the data onto a high dimensional space
  2. They find the best separating hyperplane in that space



Non-linear problems (II)

Projecting onto a high dimensional space presents two main problems:

1. "Curse of dimensionality"
    - Much more patterns are needed to train the models
    - The models are more prone to overfitting
    - SVMs overcome this problem by maximizing the margin; note that the model complexity depends only on the margin, not on the dimension
2. Much higher computational cost
    - SVMs overcome this problem by making the projection only implicitly (thanks to the **kernel** trick)

Kernel methods (I)

- **Kernel:** function $k(\mathbf{x}_i, \mathbf{x}_j)$ that can be expressed as the dot product
$$k(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{\Phi}(\mathbf{x}_i)^t \mathbf{\Phi}(\mathbf{x}_j)$$
for some transformation $\mathbf{\Phi}(\mathbf{x})$
- Example: the kernel $k(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i^t \mathbf{x}_j)^2$, with $\mathbf{x}_i \in \mathbb{R}^2$, can be expressed as
$$k(\mathbf{x}_i, \mathbf{x}_j) = (x_{i1}^2, \sqrt{2} x_{i1} x_{i2}, x_{i2}^2)(x_{j1}^2, \sqrt{2} x_{j1} x_{j2}, x_{j2}^2)^t$$
  - The associated transformation is
$$\mathbf{\Phi}(\mathbf{x}) = (x_1^2, \sqrt{2} x_1 x_2, x_2^2)^t$$

Kernel methods (II)

The SVM general strategy:

1. $\mathbf{x}_i \in \mathbb{R}^d$, con $i = 1, 2, ..., n$
2. Find a non linear transformation $\mathbf{z} = \mathbf{\Phi}(\mathbf{x})$, with $\mathbf{z} \in \mathbb{R}^T$ and $T > d$, such that $\mathbf{\Phi}(\mathbf{x})^t \mathbf{\Phi}(\mathbf{y}) = k(\mathbf{x}, \mathbf{y})$ for a given kernel k
3. In this T-dimensional space the two classes are more likely to be linearly separated
4. Find the optimal separating hyperplane in this transformed space
$$\mathbf{w}^t \mathbf{\Phi}(\mathbf{x}) + b = 0$$

Kernel methods (III)

- As before, $\mathbf{w}$ is given by the support vectors ($\alpha_i \neq 0$)

$$\mathbf{w} = \sum_{i=1}^{n} \alpha_i t_i \mathbf{\Phi}(\mathbf{x}_i)$$

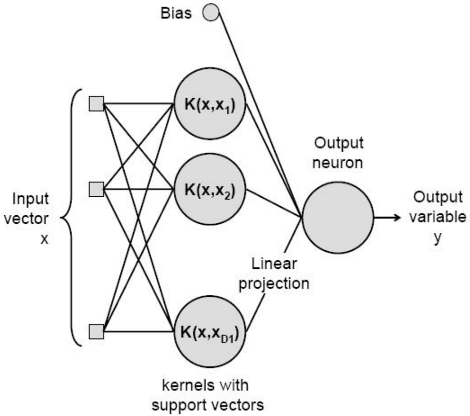
- The b coefficient is obtained from a support vector with $\alpha_i < C$

$$t_i(\mathbf{w}^t \mathbf{\Phi}(\mathbf{x}_i) + b) = 1$$

- Finally, to classify a new pattern $\mathbf{x}$ we must evaluate

$$f(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i t_i \mathbf{\Phi}(\mathbf{x}_i) \mathbf{\Phi}(\mathbf{x}) + b = \sum_{i=1}^{n} \alpha_i t_i k(\mathbf{x}_i, \mathbf{x}) + b$$

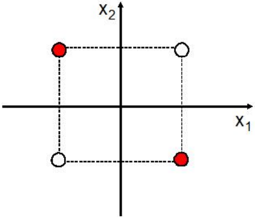
General structure of a SVM



$$y = \sum_{i=1}^{n} \alpha_i t_i k(\mathbf{x}_i, \mathbf{x}) + b$$

A simple example (I)

- XOR in 2D
  - Class 1: $\mathbf{x}_1 = (-1, -1)$, $\mathbf{x}_2 = (1, 1)$, $t = 1$
  - Class 2: $\mathbf{x}_3 = (1, -1)$, $\mathbf{x}_4 = (-1, 1)$, $t = -1$



- We use the kernel $k(\mathbf{x}, \mathbf{y}) = (\mathbf{x}^t\mathbf{y} + 1)^2$
  - The associated transformation is

$$\mathbf{\Phi}(\mathbf{x}) = (1, \sqrt{2}x_1, \sqrt{2}x_2, \sqrt{2}x_1x_2, x_1^2, x_2^2)^t$$

- We take $C = \infty$ to favour small error models

A simple example (II)

- The dual problem is

$$\tilde{L}(\alpha) = \sum_{i=1}^{4} \alpha_i + -\frac{1}{2} \sum_{i=1}^{4} \sum_{j=1}^{4} \alpha_i \alpha_j t_i t_j k(\mathbf{x}_i, \mathbf{x}_j)$$

- With the constraints

$$\alpha_i \geq 0$$

$$\alpha_1 + \alpha_2 - \alpha_3 - \alpha_4 = 0$$

- The kernel can be expressed as $k(\mathbf{x}_i, \mathbf{x}_j) = K_{ij}$, with

$$K = \begin{pmatrix} 9 & 1 & 1 & 1 \\ 1 & 9 & 1 & 1 \\ 1 & 1 & 9 & 1 \\ 1 & 1 & 1 & 9 \end{pmatrix}$$

- Then

$$\tilde{L}(\alpha) = \sum_{i=1}^{4} \alpha_i - \frac{9}{2} \sum_{i=1}^{4} \alpha_i^2 - \alpha_1\alpha_2 + \alpha_1\alpha_3 + \alpha_1\alpha_4 + \alpha_2\alpha_3 + \alpha_2\alpha_4 - \alpha_3\alpha_4$$

A simple example (III)

- We optimize with respect to the multipliers α_i

$$\frac{\partial \tilde{L}(\alpha)}{\partial \alpha_1} = 0 \Longrightarrow 9\alpha_1 + \alpha_2 - \alpha_3 - \alpha_4 = 1$$

$$\frac{\partial \tilde{L}(\alpha)}{\partial \alpha_2} = 0 \Longrightarrow \alpha_1 + 9\alpha_2 - \alpha_3 - \alpha_4 = 1$$

$$\frac{\partial \tilde{L}(\alpha)}{\partial \alpha_3} = 0 \Longrightarrow -\alpha_1 - \alpha_2 + 9\alpha_3 + \alpha_4 = 1$$

$$\frac{\partial \tilde{L}(\alpha)}{\partial \alpha_4} = 0 \Longrightarrow -\alpha_1 - \alpha_2 + \alpha_3 + 9\alpha_4 = 1$$

- To obtain the solution

$$\alpha_1 = \alpha_2 = \alpha_3 = \alpha_4 = \frac{1}{8}$$

- Note that all the points are support vectores and they are on the margin

A simple example (IV)

- The classification function is given by

$$f(\mathbf{x}) = \frac{1}{8}\sum_{i=1}^{4} t_i k(\mathbf{x}_i, \mathbf{x}) + b$$

- We obtain b from

$$\mathbf{w} = \frac{1}{8}(\mathbf{\Phi}(\mathbf{x}_1) + \mathbf{\Phi}(\mathbf{x}_2) - \mathbf{\Phi}(\mathbf{x}_3) - \mathbf{\Phi}(\mathbf{x}_4))$$

$$t_i(\mathbf{w}^t \mathbf{\Phi}(\mathbf{x}_i) + b) = 1$$

- Which leads to $b = 0$
- Operating, we finally obtain

$$f(\mathbf{x}) = x_1 x_2$$

which, as we already know, solves the XOR problem

Summary: Advantages of the SVMs

- No local minima (quadratic problem)
- The optimal solution can be found in polynomial time
- Small number of free parameters: C, kernel type and kernel parameters. They can be automatically adjusted using **cross-validation**
- Stable result (it does not depend on initial random values)
- Sparse solution (it only takes into account the support vectors)
- Maximizing the margin allows to control the complexity independently of the number of dimensions
- Good generalization capability

Bibliography

- *Pattern Classification.* R.O. Duda, P.E. Hart, D.G. Stork. Wiley, 2001.
- *Pattern Recognition and Machine Learning.* C. Bishop. Springer, 2006.
- *Machine Learning in Python: Essential Techniques for Predictive Analysis.* M. Bowles. Wiley, 2015.
- *Introduction to Machine Learning with Python. A Guide for Data Scientists.* A.C. Mueller, S. Guido. O'Reilly, 2016.