Introduction to Machine Learning
Lecture 8 - SVM and The Kernel Trick

1 Linear Separable SVM (Support Vector Machine)

1.1 Introduction

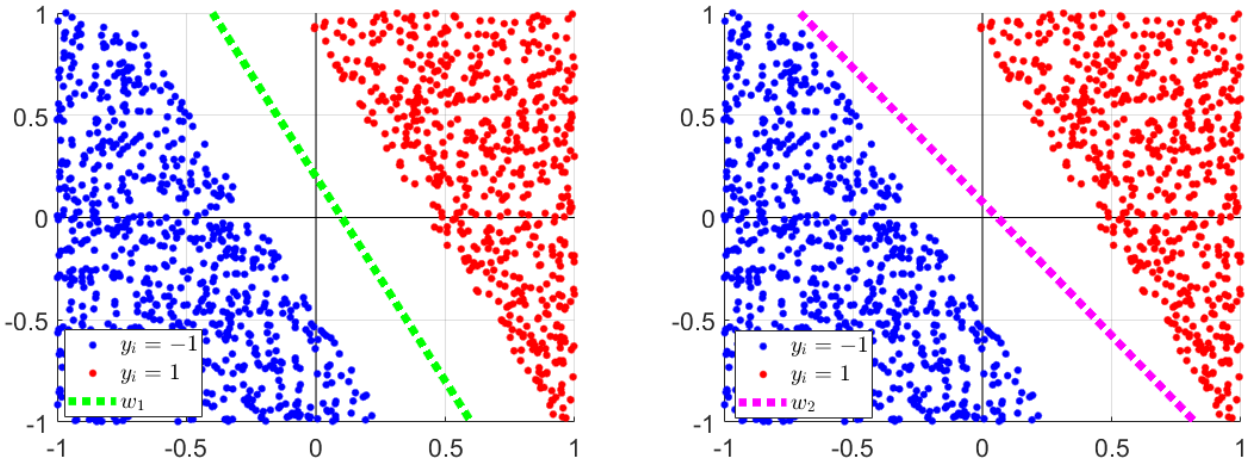
Consider some **linear separable** training set $\mathcal{D} = \{\boldsymbol{x}_i, y_i\}_{i=1}^N$ such that:

$$y_i \in \{-1, 1\}$$

Since that data is linear separable, there may be several linear classifiers $\{\boldsymbol{w}, b\}$ such that:

$$y_i = \text{sign}\left(\boldsymbol{w}^T \boldsymbol{x}_i - b\right), \qquad \forall i$$

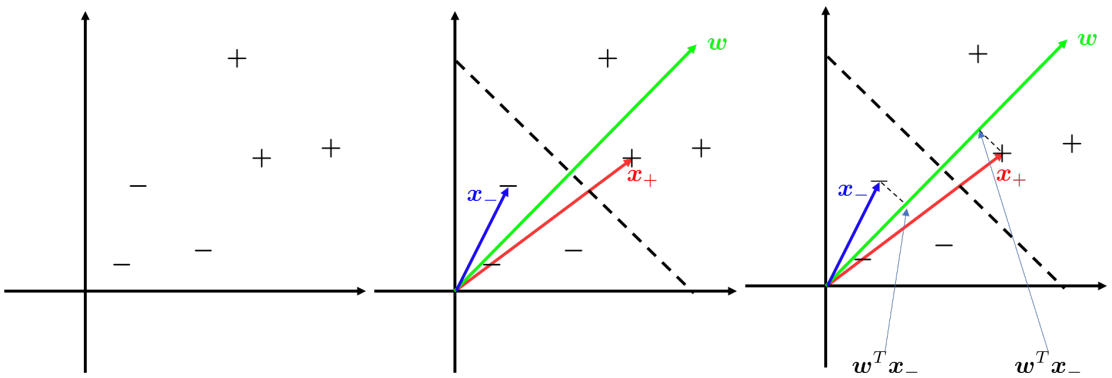
For example, both $\boldsymbol{w}_1$ and $\boldsymbol{w}_2$ provide perfect classification (so which one is better? w_1 or w_2):



The Support Vector Machine (SVM) classifier search for the largest margin between the boundary and the training set. In this figure, w_1 provides a larger margin than w_2.

1.2 Derivation

Consider some $\boldsymbol{w}$ and b:



Since we want large separation we require:

$$\begin{cases} \boldsymbol{w}^T \boldsymbol{x}_+ - b \geq 1 \\ \boldsymbol{w}^T \boldsymbol{x}_- - b \leq -1 \end{cases}$$

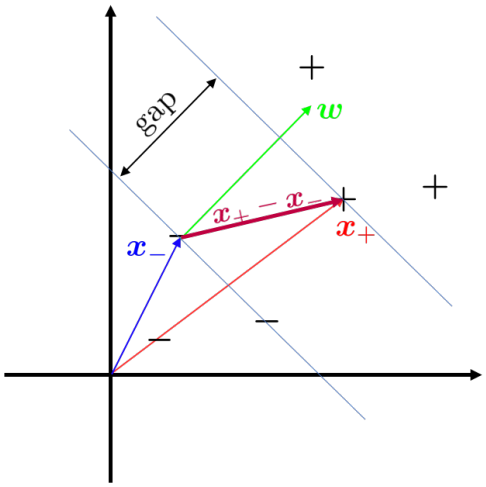
Alternatively, since $y_i = \pm 1$ (1 for the positive class and -1 for the negative class), we can write both equation as:

$$\boxed{y_i \left(\boldsymbol{w}^T \boldsymbol{x}_i - b \right) \geq 1}$$

Notes:

1. The value 1 was chosen arbitrarily since the length of $\boldsymbol{w}$ is not given (yet).
2. The points which are in the gutter (closest to the boundary) result with:

$$\begin{aligned} & y_i \left(\boldsymbol{w}^T \boldsymbol{x}_i - b \right) = 1, \qquad \text{for } \boldsymbol{x}_i \text{ in the gutter} \\ & \Rightarrow \boldsymbol{w}^T \boldsymbol{x}_i = b + y_i \end{aligned}$$



The gap is given by (projection of $(\boldsymbol{x}_+ - \boldsymbol{x}_-)$ to the normalized $\boldsymbol{w}$):

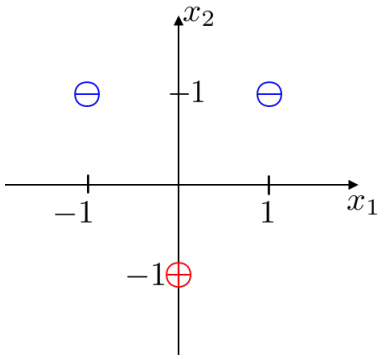
$$\begin{aligned} \text{gap} &= \left\langle \frac{\boldsymbol{w}}{\|\boldsymbol{w}\|}, \boldsymbol{x}_+ - \boldsymbol{x}_- \right\rangle, \qquad \boldsymbol{x}_+ \text{ and } \boldsymbol{x}_- \text{ are in the gutter} \\ &= \frac{\boldsymbol{w}^T}{\|\boldsymbol{w}\|} (\boldsymbol{x}_+ - \boldsymbol{x}_-) \\ &= \frac{1}{\|\boldsymbol{w}\|} \left(\boldsymbol{w}^T \boldsymbol{x}_+ - \boldsymbol{w}^T \boldsymbol{x}_- \right) \\ &= \frac{1}{\|\boldsymbol{w}\|} (b + 1 - b + 1) \\ &= \frac{2}{\|\boldsymbol{w}\|} \end{aligned}$$

Thus, to maximize the gap one should minimize $\|\boldsymbol{w}\|$.
Equivalently, one can minimize $\frac{1}{2} \|\boldsymbol{w}\|^2$:

$$\begin{cases} \min\limits_{\boldsymbol{w},b} \frac{1}{2} \|\boldsymbol{w}\|^2 \\ \text{s.t.} \\ y_i \left(\boldsymbol{w}^T \boldsymbol{x}_i - b \right) \geq 1 \quad \forall i \end{cases}$$

1.3 Example I

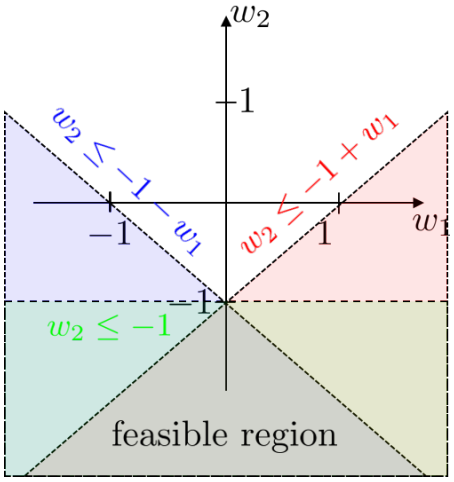
Consider the following training set:



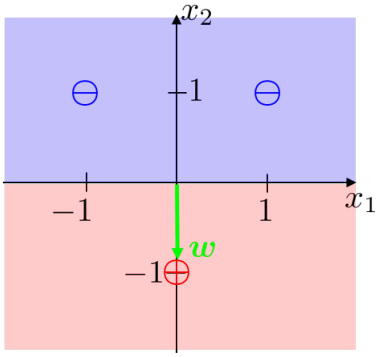
For simplicity, let us assume $b = 0$.
The SVM problem is given by:

$$\begin{cases} \min\limits_{\boldsymbol{w}} \frac{1}{2}\|\boldsymbol{w}\|^2 \\ \text{s.t.} \\ (-1)\cdot\begin{bmatrix}1 & 1\end{bmatrix}\boldsymbol{w} \geq 1 \\ (-1)\cdot\begin{bmatrix}-1 & 1\end{bmatrix}\boldsymbol{w} \geq 1 \\ (+1)\cdot\begin{bmatrix}0 & -1\end{bmatrix}\boldsymbol{w} \geq 1 \end{cases} = \begin{cases} \min\limits_{\boldsymbol{w}} \frac{1}{2}\|\boldsymbol{w}\|^2 \\ \text{s.t.} \\ \begin{bmatrix}1 & 1\end{bmatrix}\boldsymbol{w} \leq -1 \\ \begin{bmatrix}-1 & 1\end{bmatrix}\boldsymbol{w} \leq -1 \\ \begin{bmatrix}0 & -1\end{bmatrix}\boldsymbol{w} \geq 1 \end{cases} = \begin{cases} \min\limits_{\boldsymbol{w}} \frac{1}{2}\|\boldsymbol{w}\|^2 \\ \text{s.t.} \\ w_2 \leq -1 - w_1 \\ w_2 \leq -1 + w_1 \\ w_2 \leq -1 \end{cases}$$

The feasible region is given by:



Notice that $\boldsymbol{w} = \begin{bmatrix}0 \\ -1\end{bmatrix}$ is the closet point (in the feasible region) to the origin:



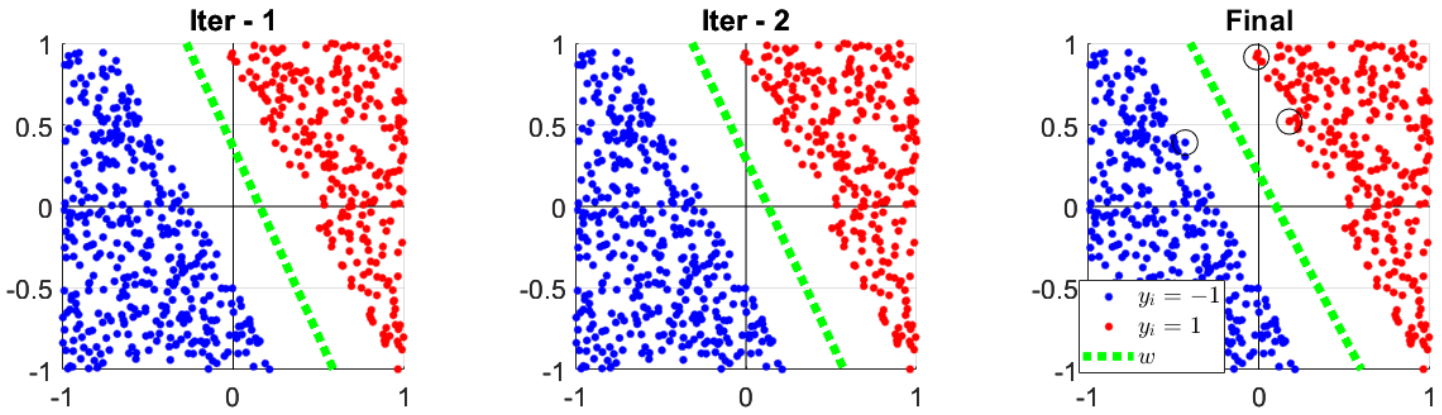
1.4 Example II

1.4.1 The linear separable case

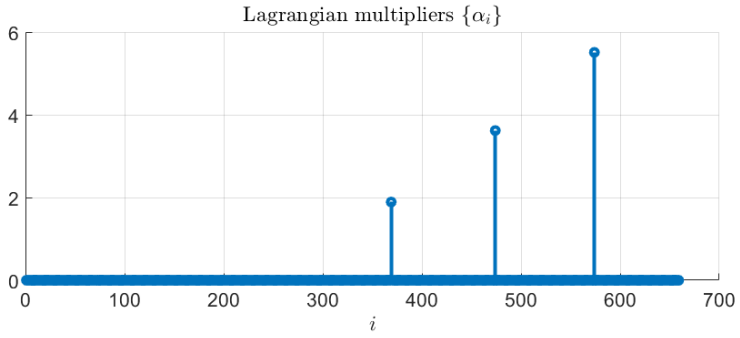
The Lagrangian is given by:

$$\mathcal{L}(\boldsymbol{w}, b) = \frac{1}{2}\|\boldsymbol{w}\|^2 - \sum_i \alpha_i \left(y_i \left(\boldsymbol{w}^T \boldsymbol{x}_i - b\right) - 1\right)$$

where $\{\alpha_i \geq 0\}_i$ are the Lagrangian multipliers.
The Augmented Lagrangian algorithm can solve the given (constrained) optimization problem:



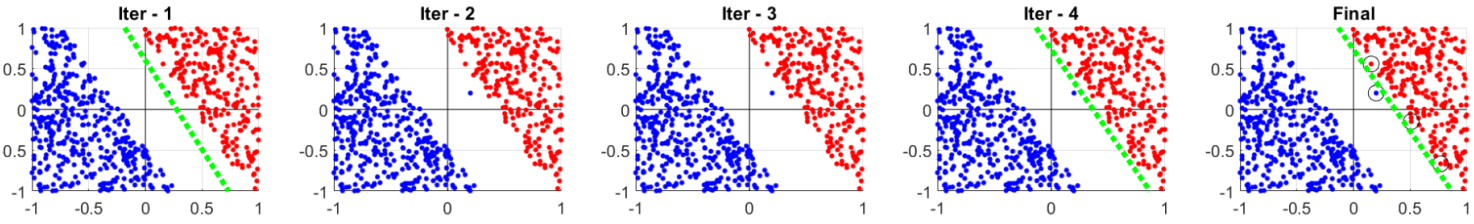
The obtained Lagrangian multipliers $\{\alpha_i\}_i$ are:



The three points correspond to the three $\alpha_i \neq 0$ are marked with black circles in the "Final" figure.
These three points are exactly in the gutter and they are called the support vectors.
The other points do not affect the optimization result (the value of $\boldsymbol{w}$ and b).

1.4.2 The linear barely separable case

Notice how a single point can change the classifier (one blue point close to the red group):



For non-separable problems, the constrains cannot be satisfied.
Thus, a soft (relaxed) version of SVM should be used.

2 Soft SVM

Reminder: The SVM optimization problem is given by:

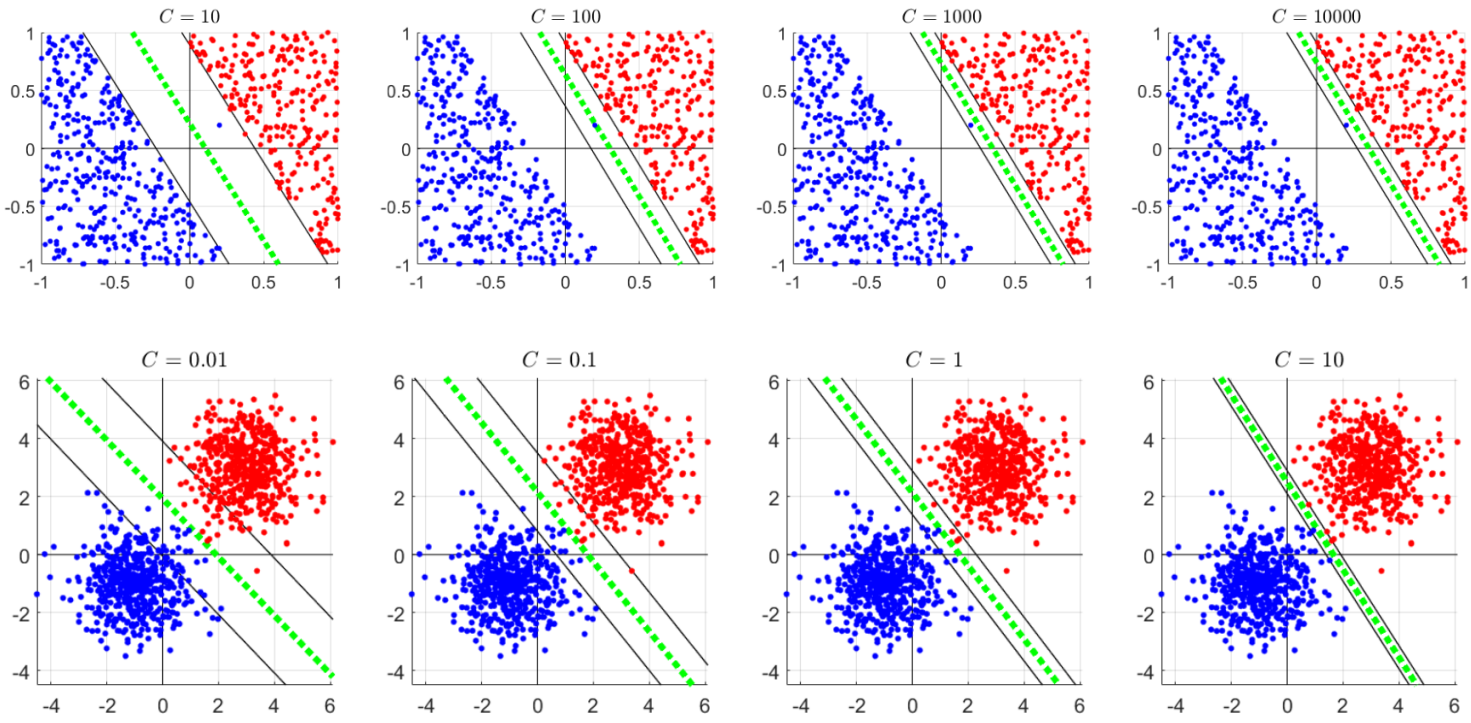
$$\begin{cases} \min\limits_{\boldsymbol{w},b} \frac{1}{2} \|\boldsymbol{w}\|^2 \\ \text{s.t.} \\ y_i \left(\boldsymbol{w}^T \boldsymbol{x}_i - b\right) \geq 1 \quad \forall i \end{cases}$$

Instead of strict constrains we can write a relaxed unconstrained version of SVM:

$$\boxed{\min_{\boldsymbol{w},b} \frac{1}{2} \|\boldsymbol{w}\|^2 + C \sum_i \max\left\{0, 1 - y_i \left(\boldsymbol{w}^T \boldsymbol{x}_i - b\right)\right\}}$$

1. The new term is a cost penalty for each point $\boldsymbol{x}_i$ which does not satisfy the original constrain: $y_i \left(\boldsymbol{w}^T \boldsymbol{x}_i - b\right) \geq 1$.
2. The $C \geq 0$ is the cost factor (regularization factor).
3. For $C = 0$, there is no cost for violating the original constrains.
   Hence, the solution will be $\boldsymbol{w} = 0$
4. For $C \to \infty$, the solution must not violate the constrains, and we obtained the original constrained problem.

Examples:



3 Dual Problem

3.1 Derivation

The Lagrangian is given by:

$$\mathcal{L}\left(\boldsymbol{w}, b, \{\alpha_i\}\right) = \frac{1}{2}\left\|\boldsymbol{w}\right\|^2 - \sum_i \alpha_i \left(y_i \left(\boldsymbol{w}^T \boldsymbol{x}_i - b\right) - 1\right)$$

Comparing the gradient to zero:

$$\begin{aligned} \nabla_{\boldsymbol{w}} \mathcal{L} &= 0 \\ \boldsymbol{w} - \sum_i \alpha_i y_i \boldsymbol{x}_i &= 0 \\ \Rightarrow \boxed{\boldsymbol{w}^* = \sum_i \alpha_i y_i \boldsymbol{x}_i} & \end{aligned}$$

$$\begin{aligned} \nabla_b \mathcal{L} &= 0 \\ \sum_i \alpha_i y_i &= 0 \\ \Rightarrow \boxed{\sum_i \alpha_i y_i = 0} & \end{aligned}$$

Plugging these relations back to $\mathcal{L}$ results in:

$$\begin{aligned} \mathcal{L}\left(\boldsymbol{w}^*, b, \{\alpha_i\}\right) &= \frac{1}{2}\left\|\boldsymbol{w}\right\|^2 - \sum_i \alpha_i \left(y_i \left(\boldsymbol{w}^T \boldsymbol{x}_i - b\right) - 1\right) \\ &= \frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} - \sum_i \alpha_i y_i \boldsymbol{w}^T \boldsymbol{x}_i + b\underbrace{\sum_i \alpha_i y_i}_{=0} + \sum_i \alpha_i \\ &= \frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} - \underbrace{\left(\sum_i \alpha_i y_i \boldsymbol{x}_i^T\right)}_{=\boldsymbol{w}^T}\boldsymbol{w} + \sum_i \alpha_i \\ &= \sum_i \alpha_i - \frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} \\ &= \sum_i \alpha_i - \frac{1}{2}\left(\sum_i \alpha_i y_i \boldsymbol{x}_i\right)^T \left(\sum_j \alpha_j y_j \boldsymbol{x}_j\right) \\ &= \sum_i \alpha_i - \frac{1}{2}\sum_i \sum_j \alpha_i \alpha_j y_i y_j \boldsymbol{x}_i^T \boldsymbol{x}_j \\ &= \sum_i \alpha_i - \frac{1}{2}\sum_i \sum_j \alpha_i \alpha_j y_i y_j \left\langle \boldsymbol{x}_i, \boldsymbol{x}_j \right\rangle \end{aligned}$$

3.2 Formulation

Overall, the dual problem can be written as:

$$\begin{cases} \max\limits_{\{\alpha_i\}} \sum_i \alpha_i - \frac{1}{2} \sum_i \sum_j \alpha_i \alpha_j y_i y_j \langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle \\ \text{s.t.} \\ \alpha_i \geq 0 & \forall i \\ \sum_i \alpha_i y_i = 0 \end{cases}$$

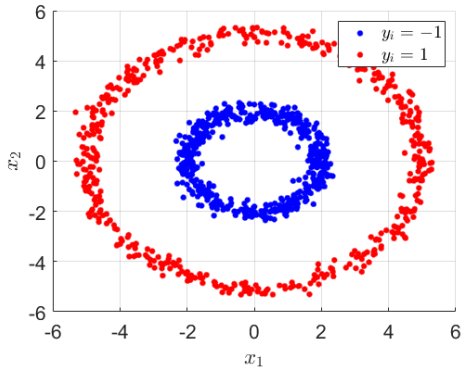
Remarks

1. The maximum value of the dual problem is the minimal value of the primal problem (not always true for general constrained problems).
2. The dual problem dependent only on the inner products $\langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle$ and not the vector $\{\boldsymbol{x}_i\}_i$ themselves.
3. The vector $\boldsymbol{w}$ is obtained by:
$$\boxed{\boldsymbol{w} = \sum_i \alpha_i y_i \boldsymbol{x}_i}$$
4. b can be obtained by using a point in the gutter:
$$y_i \left(\boldsymbol{w}^T \boldsymbol{x}_i - b \right) = 1, \qquad \text{for } \boldsymbol{x}_i \text{ in the gutter}$$
$$\Rightarrow \boxed{b = \boldsymbol{w}^T \boldsymbol{x}_i - y_i}$$
5. The decision rule (of a new point $\boldsymbol{x}_0$) is given by:
$$\begin{aligned} \hat{y}_0 &= \text{sign} \left(\boldsymbol{w}^T \boldsymbol{x}_0 - b \right) \\ &= \text{sign} \left(\sum_i \alpha_i y_i \boldsymbol{x}_i^T \boldsymbol{x}_0 - b \right) \\ &= \text{sign} \left(\sum_i \alpha_i y_i \langle \boldsymbol{x}_i, \boldsymbol{x}_0 \rangle - b \right) \end{aligned}$$
6. The decision rule dependent only on the inner product.

4 The Kernel Trick (Non-linear SVM)

4.1 Feature transform

Consider the following problem:

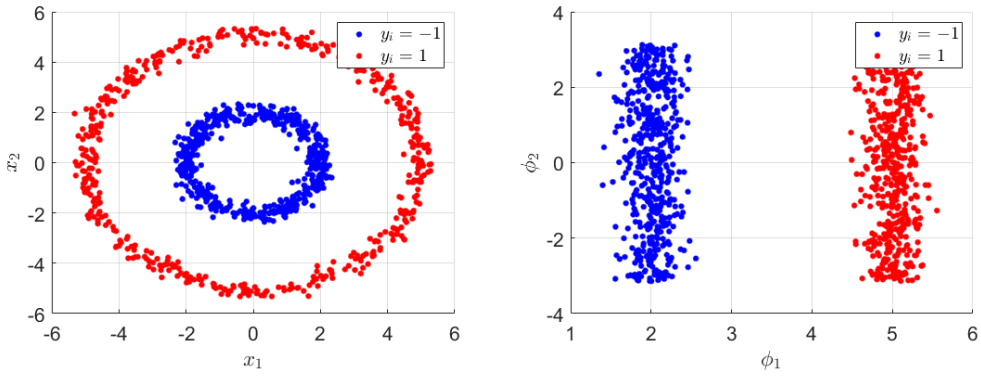


A linear SVM classifier is not able to separate these two clusters.
Reminder: The dual problem is given by:

$$\begin{cases} \max\limits_{\{\alpha_i\}} \sum_i \alpha_i - \frac{1}{2} \sum_i \sum_j \alpha_i \alpha_j y_i y_j \left\langle \boldsymbol{x}_i, \boldsymbol{x}_j \right\rangle \\ \text{s.t.} \\ \alpha_i \geq 0 & \forall i \\ \sum_i \alpha_i y_i = 0 \end{cases}$$

To solve the problem above, one can use the following feature transform (polar coordinates):

$$\boldsymbol{\phi}\left(\boldsymbol{x}_i\right) = \begin{bmatrix} \|\boldsymbol{x}_i\| \\ \angle \boldsymbol{x}_i \end{bmatrix}$$



The dual optimization problem using the new features is given by:

$$\begin{cases} \max\limits_{\{\alpha_i\}} \sum_i \alpha_i - \frac{1}{2} \sum_i \sum_j \alpha_i \alpha_j y_i y_j \left\langle \boldsymbol{\phi}\left(\boldsymbol{x}_i\right), \boldsymbol{\phi}\left(\boldsymbol{x}_j\right) \right\rangle \\ \text{s.t.} \\ \alpha_i \geq 0 & \forall i \\ \sum_i \alpha_i y_i = 0 \end{cases}$$

with the decision rule:

$$\hat{y}_0 = \text{sign}\left(\sum_i \alpha_i y_i \left\langle \boldsymbol{\phi}\left(\boldsymbol{x}_i\right), \boldsymbol{\phi}\left(\boldsymbol{x}_j\right) \right\rangle - b \right)$$

Notice again that the algorithm needs to compute only the inner products $\langle \boldsymbol{\phi}(\boldsymbol{x}_i), \boldsymbol{\phi}(\boldsymbol{x}_j) \rangle$.
Lets define the following function (**kernel**):

$$K(\boldsymbol{x}_i, \boldsymbol{x}_j) \triangleq \langle \boldsymbol{\phi}(\boldsymbol{x}_i), \boldsymbol{\phi}(\boldsymbol{x}_j) \rangle$$

Hence, we can write the optimization problem as follows:

$$\begin{cases} \max\limits_{\{\alpha_i\}} \sum_i \alpha_i - \frac{1}{2} \sum_i \sum_j \alpha_i \alpha_j y_i y_j K(\boldsymbol{x}_i, \boldsymbol{x}_j) \\ \text{s.t.} \\ \alpha_i \geq 0 & \forall i \\ \sum_i \alpha_i y_i = 0 \end{cases}$$

Definition 1. A **Kernel** function $K : \mathbb{R}^d \times \mathbb{R}^d \longrightarrow \mathbb{R}$ satisfies:

1. Symmetry: $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = K(\boldsymbol{x}_j, \boldsymbol{x}_i)$
2. K is positive definite, that is, the matrix $\boldsymbol{K}[i,j] = K(\boldsymbol{x}_i, \boldsymbol{x}_j)$ is SPD for any set $\{\boldsymbol{x}_i\}_i$

Theorem 2. *Under some technical conditions, the kernel function K can be written as:*

$$K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sum_{m=1}^{M} \phi_m(\boldsymbol{x}_i) \phi_m(\boldsymbol{x}_j) = \langle \boldsymbol{\phi}(\boldsymbol{x}_i), \boldsymbol{\phi}(\boldsymbol{x}_j) \rangle$$

where M might be infinity and $\{\boldsymbol{\phi}_m\}_m$ are the basis of K.

4.1.1 Common Kernels

Polynomial Kernel Consider for example the following kernel:

$$K\left(\boldsymbol{x}_i,\boldsymbol{x}_j\right)=\left(1+\boldsymbol{x}_i^T\boldsymbol{x}_j\right)^2$$

where $\boldsymbol{x}_i,\boldsymbol{x}_j\in\mathbb{R}^d$.
This kernel can also be written as:

$$\begin{aligned}K\left(\boldsymbol{x}_i,\boldsymbol{x}_j\right)&=\left(1+\boldsymbol{x}_i^T\boldsymbol{x}_j\right)^2\\&=1+2\boldsymbol{x}_i^T\boldsymbol{x}_j+\left(\boldsymbol{x}_i^T\boldsymbol{x}_j\right)^2\\&=1+2\sum_{k=1}^{d}\boldsymbol{x}_i\left[k\right]\boldsymbol{x}_j\left[k\right]+\left(\sum_{k=1}^{d}\boldsymbol{x}_i\left[k\right]\boldsymbol{x}_j\left[k\right]\right)\left(\sum_{m=1}^{d}\boldsymbol{x}_i\left[m\right]\boldsymbol{x}_j\left[m\right]\right)\\&=1+\sum_{k=1}^{d}\sqrt{2}\boldsymbol{x}_i\left[k\right]\cdot\sqrt{2}\boldsymbol{x}_j\left[k\right]+\sum_{k=1}^{d}\sum_{m=1}^{d}\left(\boldsymbol{x}_i\left[k\right]\boldsymbol{x}_i\left[m\right]\right)\left(\boldsymbol{x}_j\left[k\right]\boldsymbol{x}_j\left[m\right]\right)\\&=\left\langle\boldsymbol{\phi}\left(\boldsymbol{x}_i\right),\boldsymbol{\phi}\left(\boldsymbol{x}_j\right)\right\rangle\end{aligned}$$

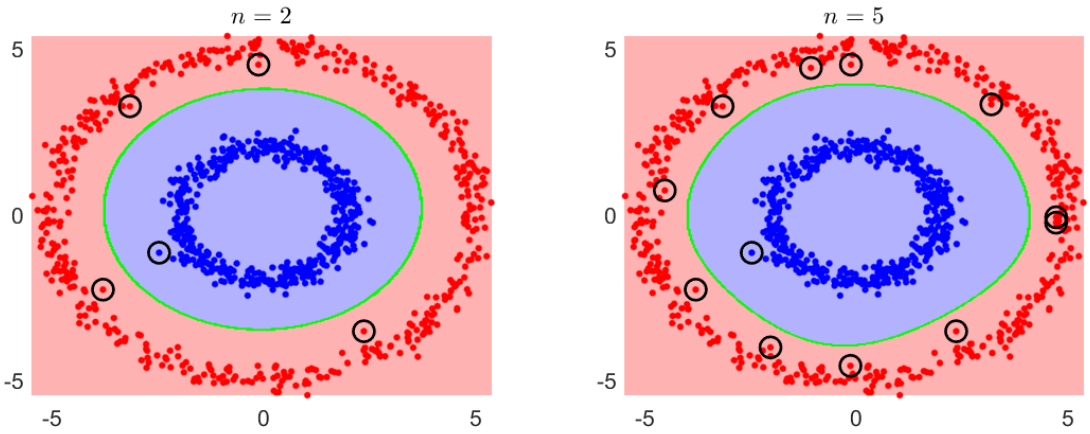
where:

$$\boldsymbol{\phi}\left(\boldsymbol{x}\right)\triangleq\begin{bmatrix}1\\\sqrt{2}\boldsymbol{x}\left[1\right]\\\vdots\\\sqrt{2}\boldsymbol{x}\left[d\right]\\\boldsymbol{x}\left[1\right]\boldsymbol{x}\left[1\right]\\\vdots\\\boldsymbol{x}\left[1\right]\boldsymbol{x}\left[d\right]\\\boldsymbol{x}\left[2\right]\boldsymbol{x}\left[1\right]\\\vdots\\\boldsymbol{x}\left[d\right]\boldsymbol{x}\left[d\right]\end{bmatrix}$$

One can also use higher order polynomials:

$$K\left(\boldsymbol{x}_i,\boldsymbol{x}_j\right)=\left(1+\boldsymbol{x}_i^T\boldsymbol{x}_j\right)^n,\qquad n\geq 1$$

Examples:



The black circles indicate the support vectors.

Gauss Kernel For $\boldsymbol{x}_i, \boldsymbol{x}_j \in \mathbb{R}^d$

$$K\left(\boldsymbol{x}_i, \boldsymbol{x}_j\right) = \exp\left(-\frac{\left\|\boldsymbol{x}_i - \boldsymbol{x}_j\right\|^2}{2\sigma^2}\right), \qquad \sigma > 0$$

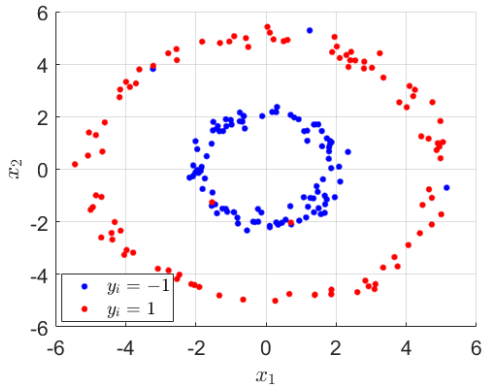
For simplicity lets assume $x_i, x_j \in \mathbb{R}$ (scalars).
Using Taylor series:

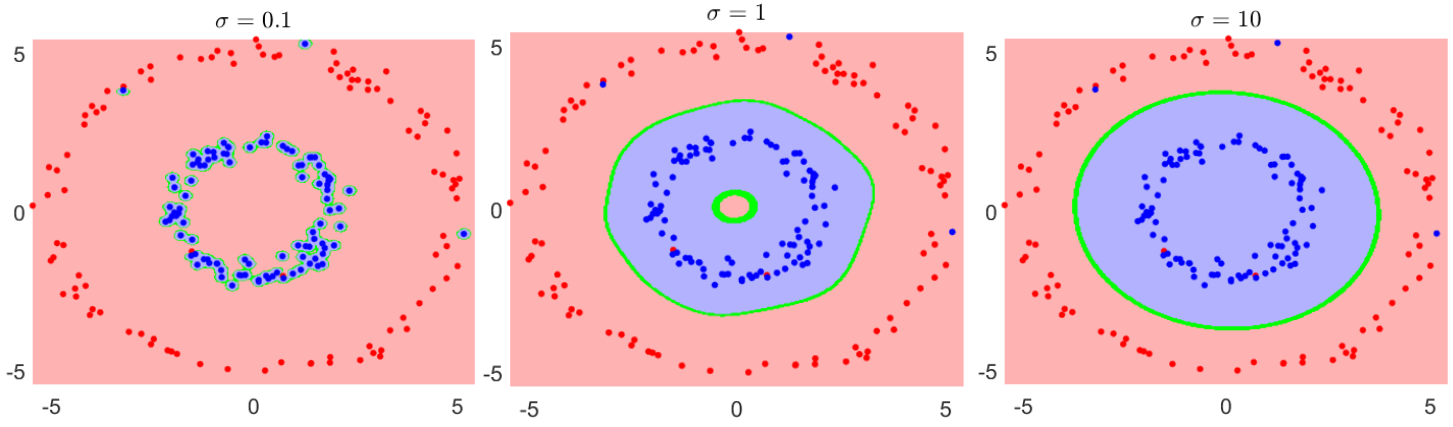
$$\begin{aligned}
K\left(x_i, x_j\right) &= \exp\left(-\frac{\left(x_i - x_j\right)^2}{2\sigma^2}\right) = \exp\left(-\frac{x_i^2 - 2x_i x_j + x_j^2}{2\sigma^2}\right) \\
&= \exp\left(-\frac{x_i^2}{2\sigma^2}\right) \exp\left(-\frac{x_j^2}{2\sigma^2}\right) \exp\left(\frac{x_i x_j}{\sigma^2}\right) \\
&= \exp\left(-\frac{x_i^2}{2\sigma^2}\right) \exp\left(-\frac{x_j^2}{2\sigma^2}\right) \sum_{n=0}^{\infty} \frac{\left(\frac{x_i x_j}{\sigma^2}\right)^n}{n!} \\
&= \exp\left(-\frac{x_i^2}{2\sigma^2}\right) \exp\left(-\frac{x_j^2}{2\sigma^2}\right) \sum_{n=0}^{\infty} \frac{x_i^n}{\sigma^n\sqrt{n!}} \cdot \frac{x_j^n}{\sigma^n\sqrt{n!}} \\
&= \left\langle \boldsymbol{\phi}\left(x_i\right), \boldsymbol{\phi}\left(x_j\right) \right\rangle
\end{aligned}$$

where:

$$\boldsymbol{\phi}\left(x\right) \triangleq \exp\left(-\frac{x^2}{2\sigma^2}\right) \cdot \begin{bmatrix} 1 \\ \frac{x}{\sigma} \\ \frac{x^2}{\sigma^2\sqrt{2!}} \\ \frac{x^3}{\sigma^3\sqrt{3!}} \\ \vdots \end{bmatrix} \in \mathbb{R}^{\infty}$$

Similar training set with a few mixed label points:





5 Extra

5.1 A simple example for the dual problem

Consider the following convex constrained optimization:

$$\begin{cases} \min f(x) \\ \text{s.t.} \\ g(x) \leq 0 \end{cases} = \begin{cases} \min x^2 \\ \text{s.t.} \\ x+1 \leq 0 \end{cases}$$

One can easily notice that the optimal solution is $x^* = -1$ and $f(x^*) = 1$.

The Lagrangian is given by:

$$\mathcal{L}(x, \lambda) = x^2 + \lambda(x+1)$$

$$\begin{aligned} \frac{\partial}{\partial x}\mathcal{L} &= 0 \\ 2x + \lambda &= 0 \\ \Rightarrow x^* &= -\frac{\lambda}{2} \end{aligned}$$

$$\begin{aligned} \Rightarrow \mathcal{L}(x^*, \lambda) &= \left(-\frac{\lambda}{2}\right)^2 + \lambda\left(-\frac{\lambda}{2}+1\right) \\ &= \frac{\lambda^2}{4} + -\frac{\lambda^2}{2} + \lambda \\ &= -\frac{\lambda^2}{4} + \lambda \end{aligned}$$

The dual problem is given by:

$$\begin{cases} \max \mathcal{L}(x^*, \lambda) \\ \text{s.t.} \\ \lambda \geq 0 \end{cases}$$

The solution is given by:

$$\begin{aligned} \frac{\partial}{\partial \lambda}\mathcal{L} &= 0 \\ -\frac{\lambda}{2} + 1 &= 0 \\ \Rightarrow \lambda^* &= 2 \end{aligned}$$

$$\Rightarrow \mathcal{L}(x^*, \lambda^*) = -1 + 2 = 1$$

$$\Rightarrow x^*(\lambda^*) = -\frac{2}{2} = -1$$

Note that the solution of the dual problem is exactly the same solution of the primal problem:

$$\begin{cases} \mathcal{L}(x^*, \lambda^*) = f(x^*) = 1 \\ x^* = -1 \end{cases}$$

Thus, one can solve the dual problem instead of the primal problem.