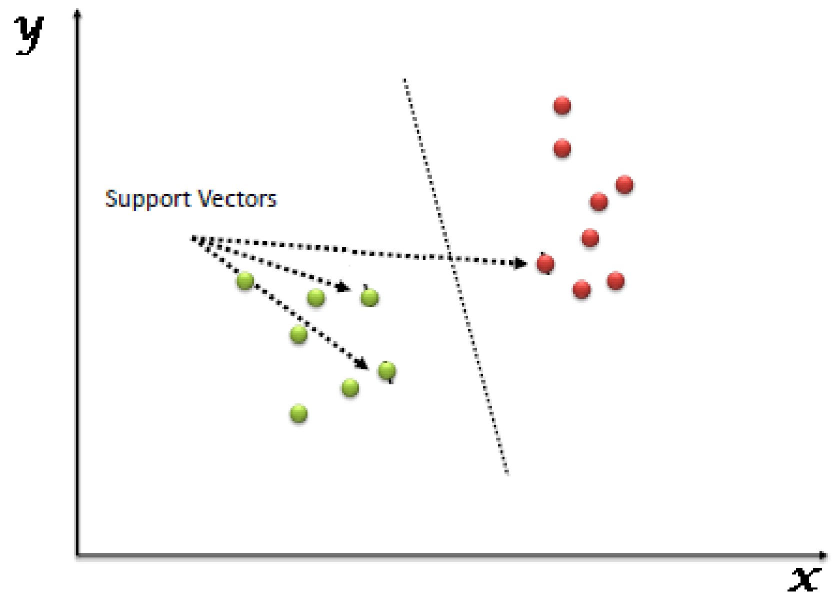
TON DUC THANG UNIVERSITY
Faculty of Information Technology
Computer Science

Support Vector Machines and Kernel Method

LÊ ANH CƯỜNG

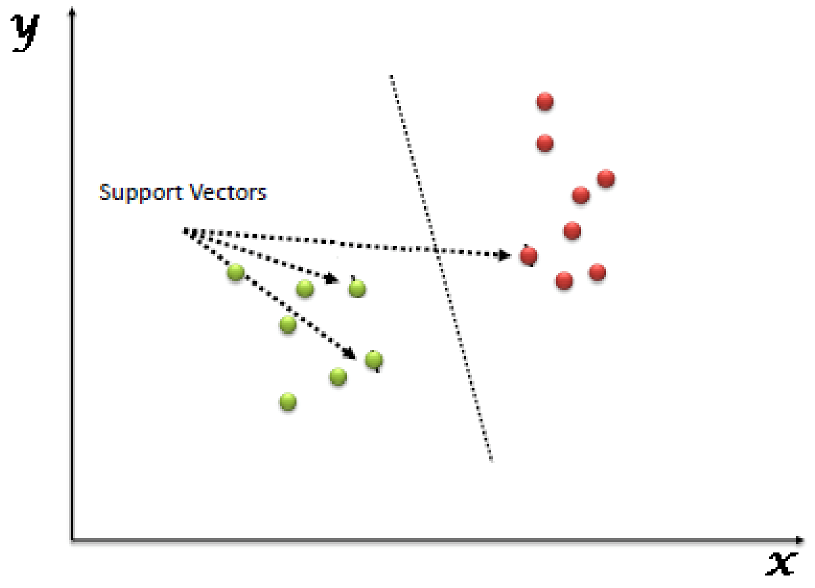
What is SVM?

- "Support Vector Machine" (SVM) is a supervised machine learning algorithm which can be used for both classification or regression challenges.
- However, it is mostly used in classification problems.



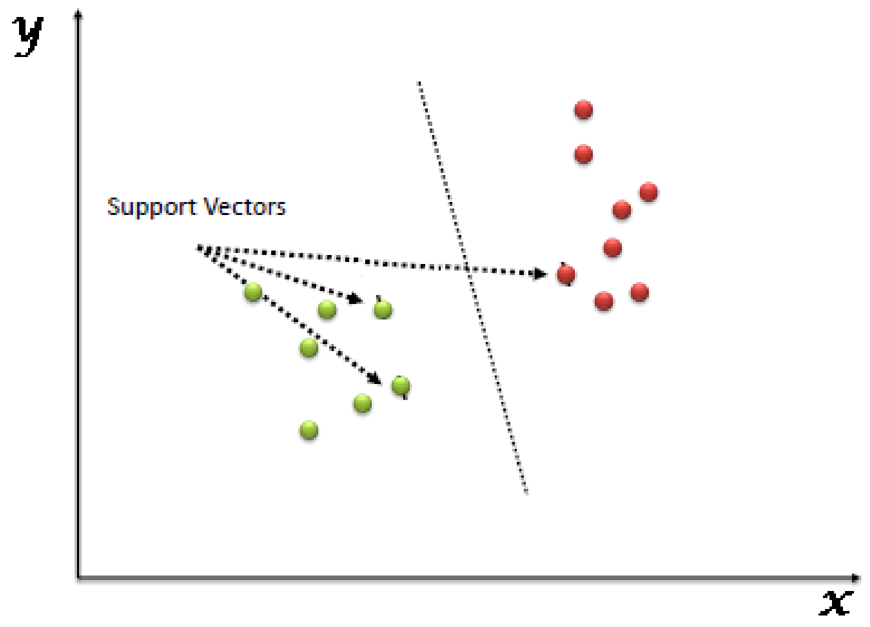
What is SVM?

- In this algorithm, we plot each data item as a point in n-dimensional space (where n is number of features you have) with the value of each feature being the value of a particular coordinate. Then, we perform classification by finding the hyper-plane that differentiate the two classes very well (look at the below snapshot).



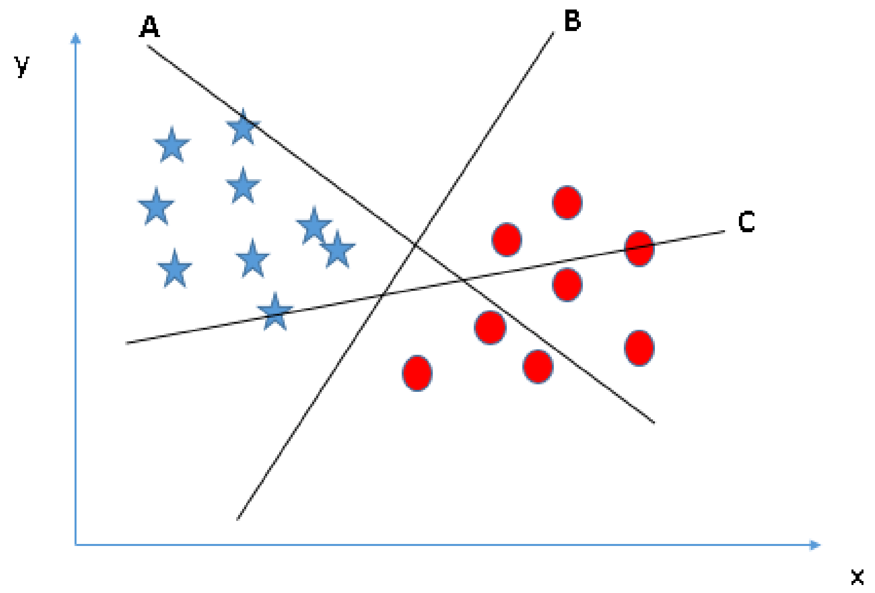
What is SVM?

- Support Vectors are simply the co-ordinates of individual observation. Support Vector Machine is a frontier which best segregates the two classes (hyper-plane/ line).



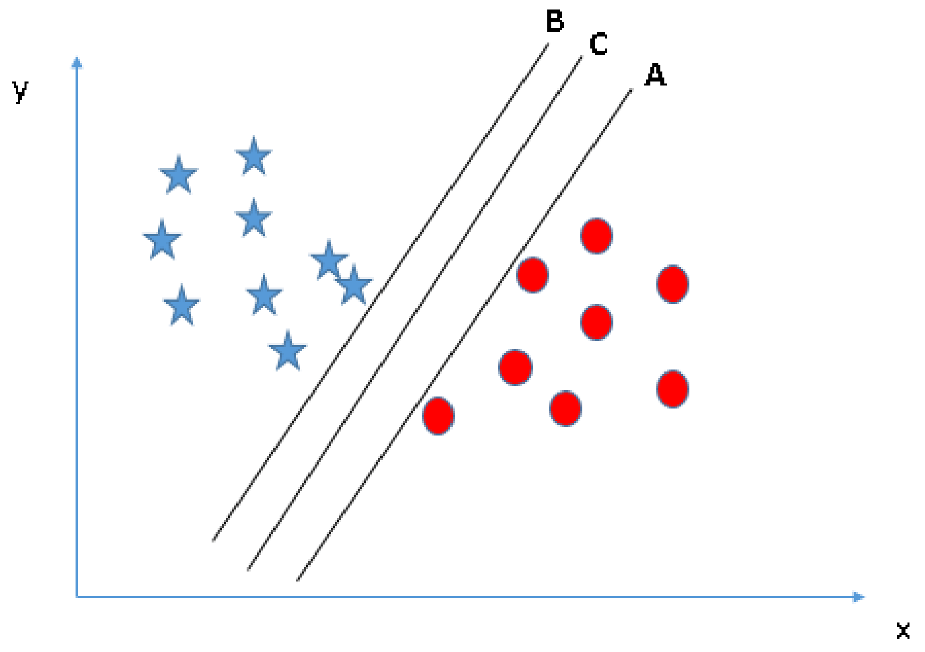
How does it work?

- We got accustomed to the process of segregating the two classes with a hyper-plane. Now the burning question is "How can we identify the right hyper-plane?".



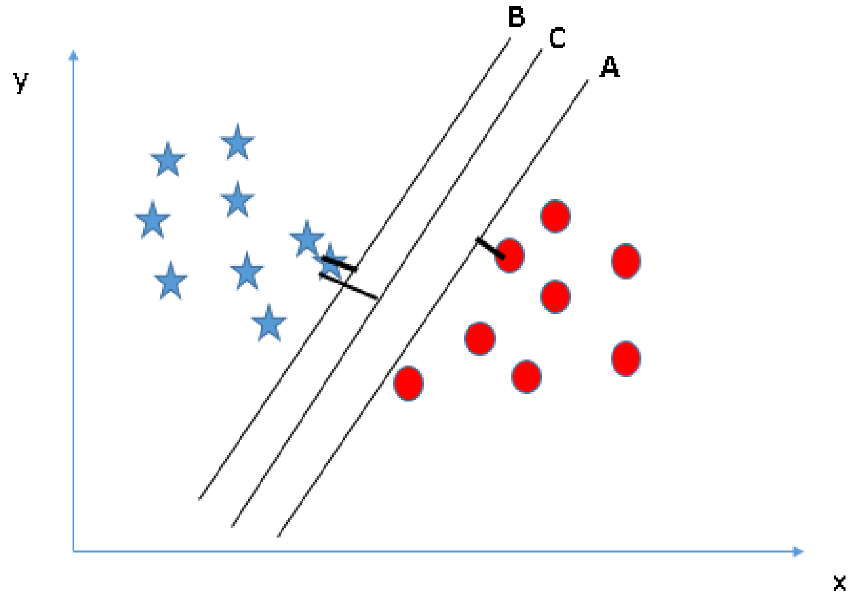
How does it work?

- Here, we have three hyper-planes (A, B and C) and all are segregating the classes well. Now, How can we identify the right hyper-plane?



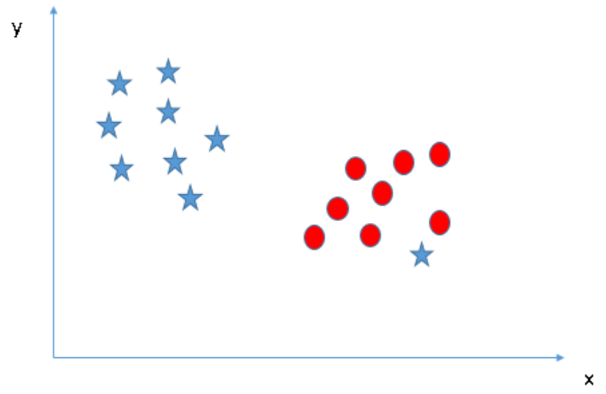
How does it work?

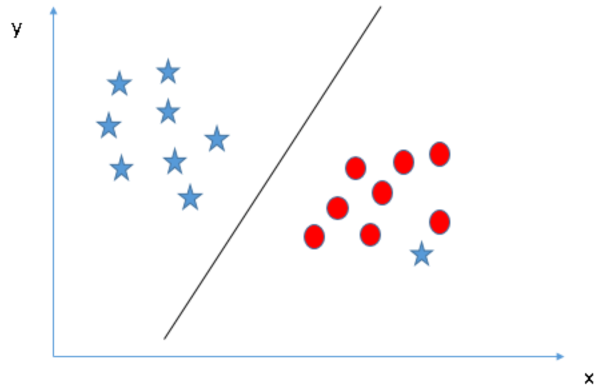
- Maximizing the distances between nearest data point (either class) and hyper-plane will help us to decide the right hyper-plane. This distance is called as **Margin**.



Issues: Outliers

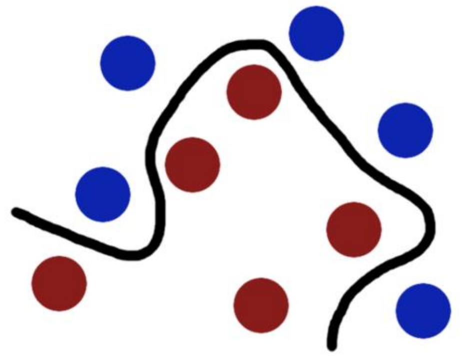
- Below, I am unable to segregate the two classes using a straight line, as one of star lies in the territory of other(circle) class as an outlier.
- one star at other end is like an outlier for star class. SVM has a feature to ignore outliers and find the hyper-plane that has maximum margin. Hence, we can say, SVM is robust to outliers.





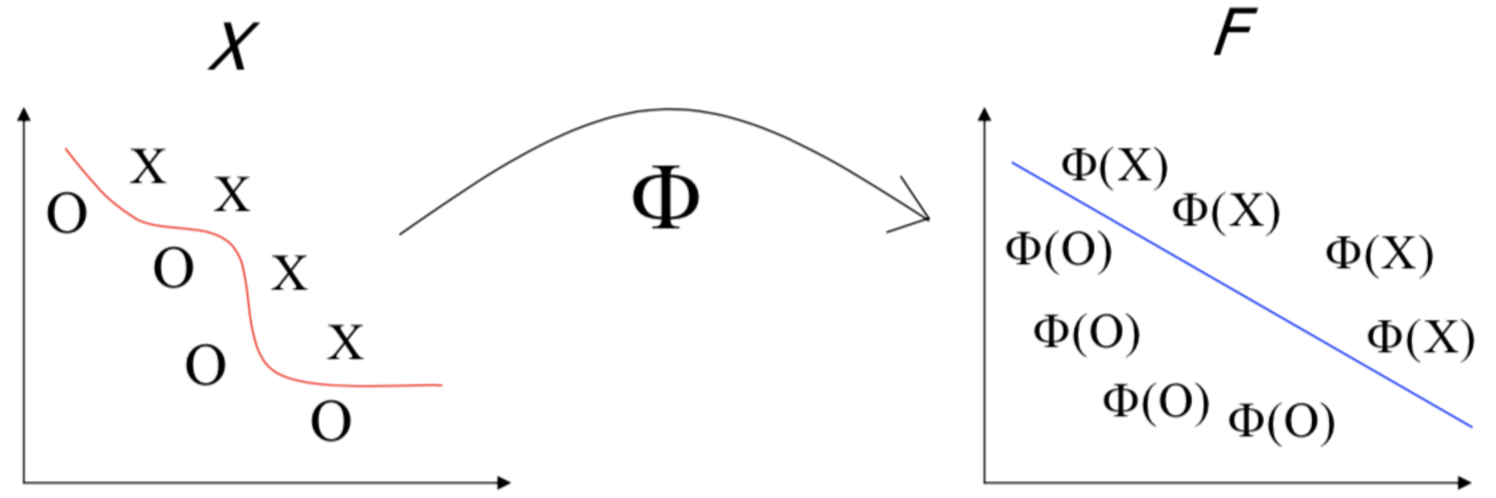
Limitation of linear hyper-plane

- In the scenario below, we can’t have linear hyper-plane between the two classes, so how does SVM classify these two classes? Till now, we have only looked at the linear hyper-plane.

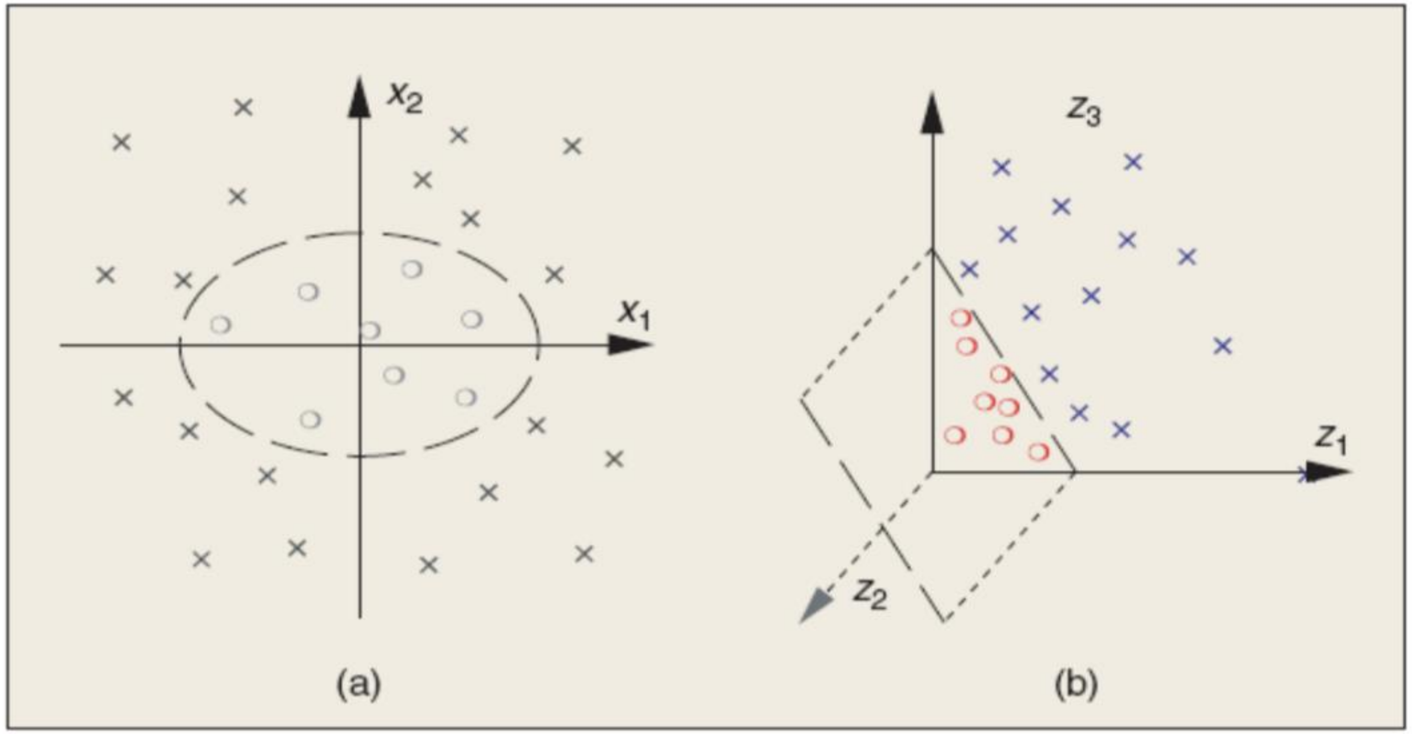


Data Transform from the original space to higher dimension space

- Solution: Map the data into a (possibly high-dimensional) vector space where linear relations exist among the data, then apply a linear algorithm in this space

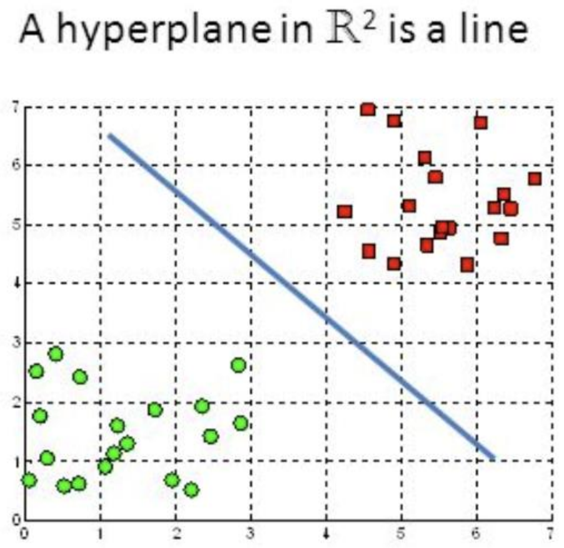


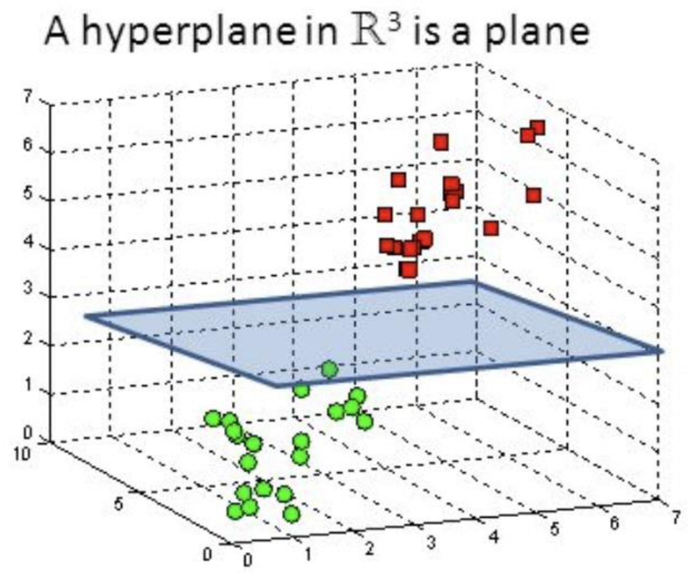
Data Transform from the original space to higher dimension space



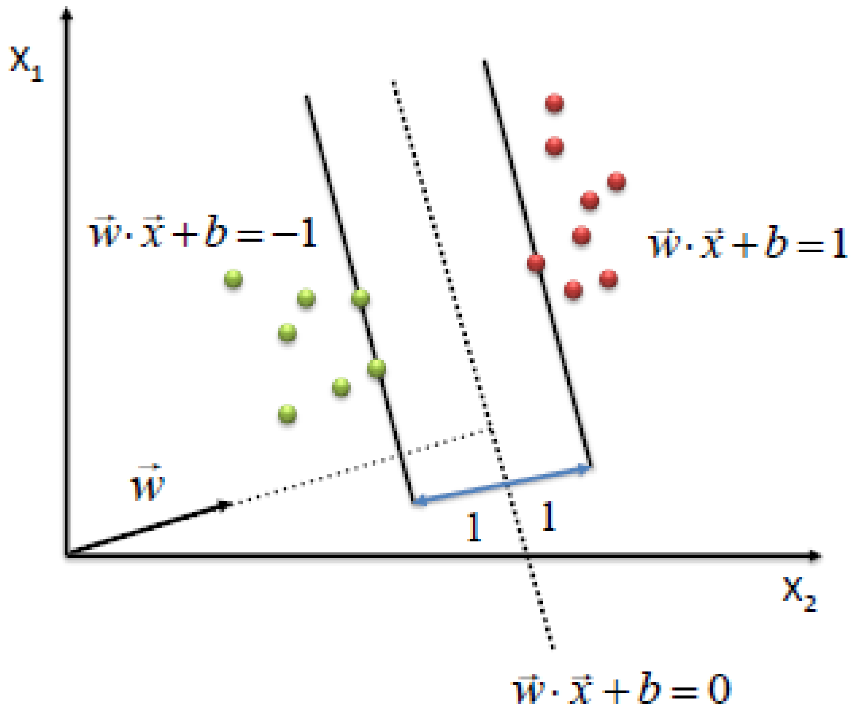
▲ *1. Effect of the map* $\phi(x_1, x_2) = (x_1^2, \sqrt{2}x_1x_2, x_2^2)$. *(a) Input space* $\mathcal{X}$ *and (b) feature space* $\mathcal{H}$.

Data Transform from the original space to higher dimension space





Learning: Maximize Margin



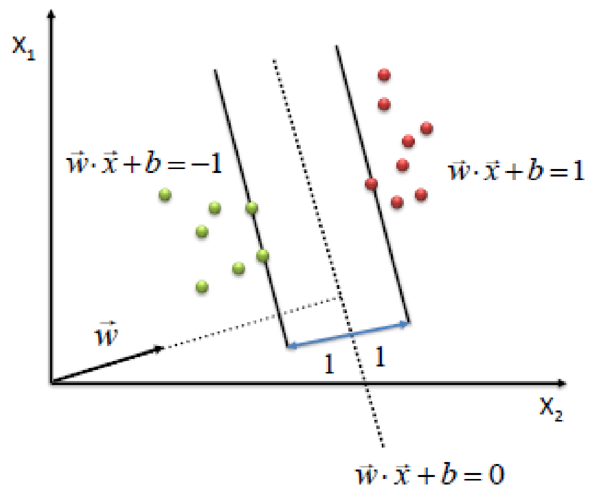
X_1
$\vec{w}\cdot\vec{x}+b=-1$
$\vec{w}\cdot\vec{x}+b=1$
$\vec{w}$
1
1
X_2
$\vec{w}\cdot\vec{x}+b=0$

$$\boxed{\max \frac{2}{\|w\|}}$$

s.t.

$(w\cdot x+b)\geq 1, \forall x$ of class 1

$(w\cdot x+b)\leq -1, \forall x$ of class 2



X_1
$\vec{w}\cdot\vec{x}+b=-1$
$\vec{w}\cdot\vec{x}+b=1$
$\vec{w}$
1
1
X_2
$\vec{w}\cdot\vec{x}+b=0$

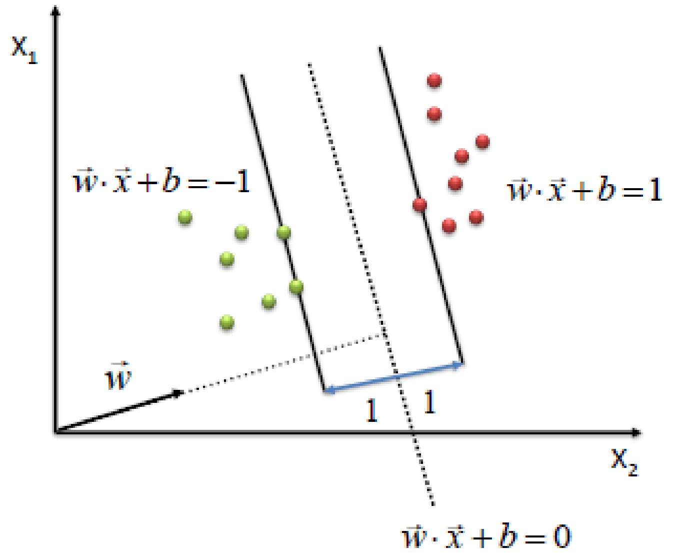
$$\boxed{\max \frac{2}{\|w\|}}$$

$s.t.$
$(w\cdot x+b)\geq 1, \forall x$ of class 1
$(w\cdot x+b)\leq -1, \forall x$ of class 2

Goal: we want to find the hyperplane (i.e. decision boundary) linearly separating our classes. Our boundary will have equation: $\mathbf{w}^T\mathbf{x}+b=0$.

Anything above the decision boundary should have label 1.
i.e., $\mathbf{x_i}$ s.t. $\mathbf{w}^T\mathbf{x_i}+b>0$ will have corresponding $y_i=1$.
Similarly, anything below the decision boundary should have label -1.
i.e., $\mathbf{x_i}$ s.t. $\mathbf{w}^T\mathbf{x_i}+b<0$ will have corresponding $y_i=-1$.

You'll notice that we will now have some space between our decision boundary and the nearest data points of either class. Thus, let's rescale the data such that anything on or above the boundary $\mathbf{w}^T\mathbf{x} + b = 1$ is of one class (with label 1), and anything on or below the boundary $\mathbf{w}^T\mathbf{x} + b = -1$ is of the other class (with label -1).


X1
$\vec{w}\cdot\vec{x}+b=-1$
$\vec{w}\cdot\vec{x}+b=1$
$\vec{w}$
1
1
X2
$\vec{w}\cdot\vec{x}+b=0$

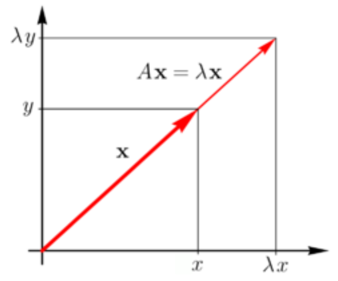
$$\boxed{\max \frac{2}{\|w\|}}$$

$s.t.$

$(w \cdot x + b) \geq 1, \forall x \text{ of class } 1$

$(w \cdot x + b) \leq -1, \forall x \text{ of class } 2$

First note that the two lines are parallel, and thus share their parameters $\mathbf{w}, b$. Pick an arbirary point $\mathbf{x_1}$ to lie on line $\mathbf{w}^T\mathbf{x} + b = -1$. Then, the closest point on line $\mathbf{w}^T\mathbf{x} + b = 1$ to $\mathbf{x_1}$ is the point $\mathbf{x_2} = \mathbf{x_1} + \lambda\mathbf{w}$ (since the closest point will always lie on the perpendicular; recall that the vector $\mathbf{w}$ is perpendicular to both lines). Using this formulation, $\lambda\mathbf{w}$ will be the line segment connecting $\mathbf{x_1}$ and $\mathbf{x_2}$, and thus, $\lambda\|\mathbf{w}\|$, the distance between $\mathbf{x_1}$ and $\mathbf{x_2}$, is the shortest distance between the two lines/boundaries. Solving for λ:



$$\Rightarrow \mathbf{w}^T\mathbf{x_2} + b = 1 \text{ where } \mathbf{x_2} = \mathbf{x_1} + \lambda\mathbf{w}$$

$$\Rightarrow \mathbf{w}^T(\mathbf{x_1} + \lambda\mathbf{w}) + b = 1$$

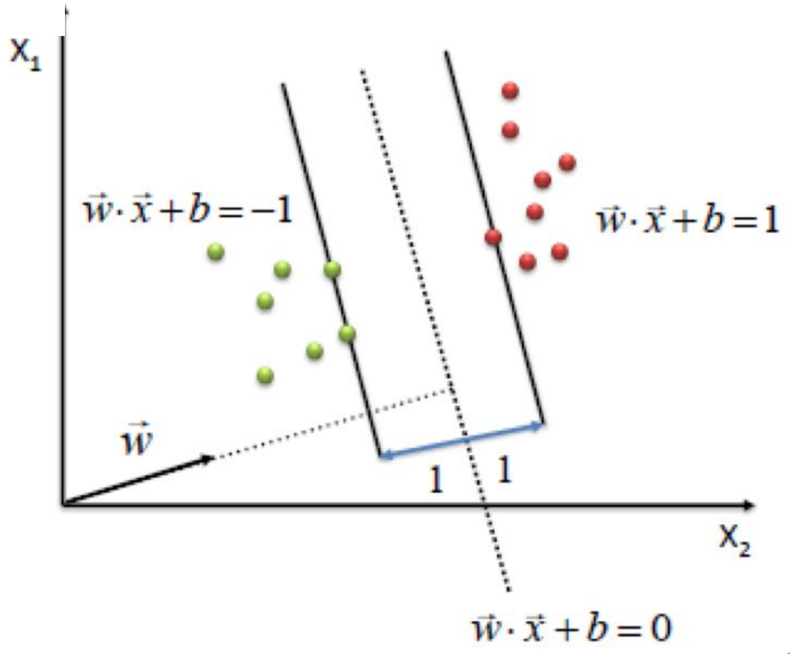
$$\Rightarrow \mathbf{w}^T\mathbf{x_1} + b + \lambda\mathbf{w}^T\mathbf{w} = 1 \text{ where } \mathbf{w}^T\mathbf{x_1} + b = -1$$

$$\Rightarrow -1 + \lambda\mathbf{w}^T\mathbf{w} = 1$$

$$\Rightarrow \lambda\mathbf{w}^T\mathbf{w} = 2$$

$$\Rightarrow \lambda = \frac{2}{\mathbf{w}^T\mathbf{w}} = \frac{2}{\|\mathbf{w}\|^2}$$

$$\Rightarrow \boxed{\max \frac{2}{\|w\|}}$$


X₁
X₂
$\vec{w}\cdot\vec{x}+b=-1$
$\vec{w}\cdot\vec{x}+b=1$
$\vec{w}$
1
1
$\vec{w}\cdot\vec{x}+b=0$

This *quadratic programming* problem is expressed as:

$$min_{\mathbf{w},b} \frac{\mathbf{w^T w}}{2}$$

subject to: $y_i(\mathbf{w}^T\mathbf{x_i} + b) \geq 1$ ($\forall$ data points $\mathbf{x_i}$).

This *quadratic programming* problem is expressed as:

$min_{\mathbf{w},b} \frac{\mathbf{w^T w}}{2}$

subject to: $y_i(\mathbf{w}^T\mathbf{x_i} + b) \geq 1$ ($\forall$ data points $\mathbf{x_i}$).

2 Soft-margin extention

Consider the case that your data isn't perfectly linearly separable. For instance, maybe you aren't guaranteed that all your data points are correctly labelled, so you want to allow some data points of one class to appear on the other side of the boundary.

We can introduce *slack variables* - an $\epsilon_i \geq 0$ for each $\mathbf{x_i}$. Our quadratic programming problem becomes:

$min_{\mathbf{w},b,\epsilon} \frac{\mathbf{w^T w}}{2} + C\sum_i \epsilon_i$

subject to: $y_i(\mathbf{w}^T\mathbf{x_i} + b) \geq 1 - \epsilon_i$ and $\epsilon_i \geq 0$ ($\forall$ data points $\mathbf{x_i}$).

Kernel definition

- A function that takes as its inputs vectors in the original space and returns the dot product of the vectors in the feature space is called a *kernel function*
- More formally, if we have data $\mathbf{x}, \mathbf{z} \in X$ and a map $\phi : X \rightarrow \Re^N$ then

$$k(\mathbf{x}, \mathbf{z}) = \left\langle \phi(\mathbf{x}), \phi(\mathbf{z}) \right\rangle$$

is a kernel function

Kernel definition: an important point

- Using kernels, we do not need to embed the data into the space $\Re^N$ explicitly, because a number of algorithms only require the inner products between image vectors!
- <u>We never need the coordinates of the data in the feature space!</u>

Kernel Example

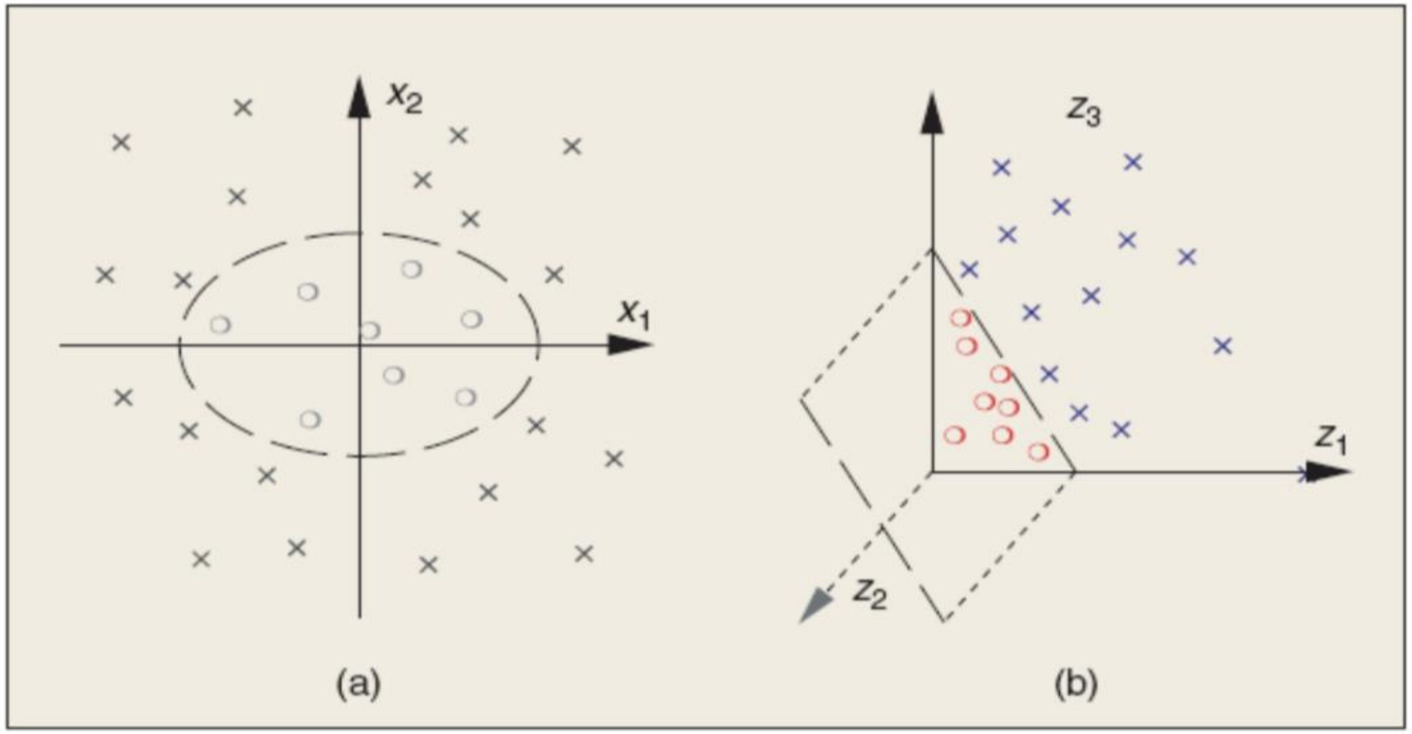
- Consider a two-dimensional input space $X \subseteq \Re^2$ with the feature map:

$$\phi : \mathbf{x} = (x_1, x_2) \mapsto \phi(\mathbf{x}) = (x_1^2, x_2^2, \sqrt{2}x_1x_2) \in F = \Re^3$$

Now consider the inner product in the feature space:

$$\begin{aligned}\langle \phi(\mathbf{x}), \phi(\mathbf{z}) \rangle &= \left\langle (x_1^2, x_2^2, \sqrt{2}x_1x_2), (z_1^2, z_2^2, \sqrt{2}z_1z_2) \right\rangle \\ &= x_1^2 z_1^2 + x_2^2 z_2^2 + 2x_1x_2z_1z_2 = (x_1z_1 + x_2z_2)^2 \\ &= \boxed{\langle \mathbf{x}, \mathbf{z} \rangle^2}\end{aligned}$$

Kernel example



▲ 1. *Effect of the map* $\phi(x_1, x_2) = (x_1^2, \sqrt{2}x_1x_2, x_2^2)$ *(a) Input space* $\mathcal{X}$ *and (b) feature space* $\mathcal{H}$.

Kernel example

- Then $k(\mathbf{x},\mathbf{z}) = \langle \mathbf{x},\mathbf{z} \rangle^2$
- But $k(\mathbf{x},\mathbf{z})$ is also the kernel that computes the inner product of the map

$$\psi(\mathbf{x}) = (x_1^2, x_2^2, x_1 x_2, x_2 x_1) \in F = \Re^4$$

Prove it?

- This shows that a given feature space is not unique to a given kernel function

Lagrangian Methods for Constrained Optimization

$$P: \underset{x \in X}{\text{maximize}}\, f(x), \quad \text{subject to } g(x) = b\,.$$

The solution of a constrained optimization problem can often be found by using the so-called **Lagrangian method**. We define the **Lagrangian** as

$$L(x, \lambda) = f(x) + \lambda(b - g(x))\,.$$

https://en.wikipedia.org/wiki/Lagrange_multiplier

https://www.cmi.ac.in/~madhavan/courses/dmml2018/literature/Lagrangian_Methods_for_Constrained_Optimization.pdf

Lagrangian Methods for Constrained Optimization

$$P: \underset{x \in X}{\text{maximize}}\, f(x), \quad \text{subject to } g(x) = b\,.$$

The solution of a constrained optimization problem can often be found by using the so-called **Lagrangian method**. We define the **Lagrangian** as

$$L(x, \lambda) = f(x) + \lambda(b - g(x))\,.$$

Theorem 5 (Lagrangian Sufficiency Theorem) *Suppose there exist $x^* \in X$ and λ^*, such that x^* maximizes $L(x, \lambda^*)$ over all $x \in X$, and $g(x^*) = b$. Then x^* solves P.*

$$P: \underset{x \in X}{\text{maximize}}\, f(x), \quad \text{subject to } g(x) = b\,.$$

The solution of a constrained optimization problem can often be found by using the so-called **Lagrangian method**. We define the **Lagrangian** as

$$L(x, \lambda) = f(x) + \lambda(b - g(x))\,.$$

Theorem 5 (Lagrangian Sufficiency Theorem) *Suppose there exist* $x^* \in X$ *and* λ^**, such that* x^* *maximizes* $L(x, \lambda^*)$ *over all* $x \in X$*, and* $g(x^*) = b$*. Then* x^* *solves* P*.*

Theorem 6 *If* f *and* g *are convex functions,* X *is a convex set, and* x^* *is an optimal solution to* P*, then there exist Lagrange multipliers* $\lambda \in \mathbb{R}^m$ *such that* $L(x^*, \lambda) \leq L(x, \lambda)$ *for all* $x \in X$*.*

SVM Optimization

$$\begin{array}{lll} min & \frac{1}{2}w^T w + C\sum_{i=1}^{m}\xi_i & \\ s.t. & y_i(x_i^T w + b) \geq 1 - \xi_i & \xi_i \geq 0 \end{array}$$

https://people.eecs.berkeley.edu/~jordan/courses/281B-spring04/lectures/lec6.pdf

SVM Optimization

$$
\begin{aligned}
&min \qquad && \frac{1}{2}w^T w + C\sum_{i=1}^{m}\xi_i \\
&s.t. \qquad && y_i(x_i^T w + b) \geq 1 - \xi_i \qquad \xi_i \geq 0
\end{aligned}
$$

We take the lagrangian in the usual manner:

$$\mathcal{L}(w, \xi, b, \alpha) = \frac{1}{2}w^T w + C\sum_{i=1}^{m}\xi_i + \sum_{i=1}^{m}\alpha_i\left[1 - \xi_i - y_i(x_i^T w + b)\right]$$

SVM Optimization

To find the dual form of the problem, we first need to minimize $\mathcal{L}(w, \xi, b, \alpha)$ with respect to w, ξ, and b (for fixed α), to get Θ_D.

$$\underset{w,\xi,b}{min} \; \mathcal{L}(w, \xi, b, \alpha), \quad \xi_i \geq 0 \tag{5}$$

Since the Lagrangian function is linear in α, we cannot set the gradient with respect to α to zero. We obtain the following dual optimization problem:

$$D: \; \underset{\alpha}{max} \quad \Theta(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} y_i y_j \alpha_i \alpha_j \left\langle \phi(x^{(i)}), \phi(x^{(j)}) \right\rangle . \tag{6}$$

$$s.t. \quad 0 \leq \alpha_i \leq C \tag{7}$$

$$\sum_{i=1}^{m} \alpha_i y_i = 0 \tag{8}$$

SVM Optimization

Indeed, we can still use

$$w = \sum_{i=1}^{m} \alpha_i x_i^T y_i$$

to give us the optimal value of w in terms of the optimal value of α

The Sequential Minimal Optimization (SMO) algorithm 2 introduced by John Platt provides an efficient algorithm for solving the dual problem. The dual optimization problem we wish to solve is stated in (6),(7), (8). This can be a very large QP optimization problem. S

Dual optimization problem

- Constrained optimization:

$$\max_{\alpha} \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j (\mathbf{x}_i \cdot \mathbf{x}_j)$$

$$\text{subject to: } \alpha_i \geq 0 \wedge \sum_{i=1}^{m} \alpha_i y_i = 0, i \in [1, m].$$

- Solution:

$$h(x) = \operatorname{sgn}\Big(\sum_{i=1}^{m} \alpha_i y_i (\mathbf{x}_i \cdot \mathbf{x}) + b\Big),$$

with $b = y_i - \sum_{j=1}^{m} \alpha_j y_j (\mathbf{x}_j \cdot \mathbf{x}_i)$ for any SV $\mathbf{x}_i$.

SVM with Kernel

- **Constrained optimization:**

$$\max_{\alpha} \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j K(x_i, x_j)$$

$$\text{subject to: } 0 \leq \alpha_i \leq C \wedge \sum_{i=1}^{m} \alpha_i y_i = 0, i \in [1, m].$$

- **Solution:**

$$h(x) = \operatorname{sgn}\Big(\sum_{i=1}^{m} \alpha_i y_i K(x_i, x) + b\Big),$$

with $b = y_i - \sum_{j=1}^{m} \alpha_j y_j K(x_j, x_i)$ for any x_i with $0 < \alpha_i < C$.

Different kernel

Linear kernels $k(\mathbf{x}, \mathbf{x}') = \mathbf{x}^\top \mathbf{x}'$

Polynomial kernels $k(\mathbf{x}, \mathbf{x}') = \left(1 + \mathbf{x}^\top \mathbf{x}'\right)^d$ for any $d > 0$

– Contains all polynomials terms up to degree d

Gaussian kernels $k(\mathbf{x}, \mathbf{x}') = \exp\left(-||\mathbf{x} - \mathbf{x}'||^2/2\sigma^2\right)$ for $\sigma > 0$

– Infinite dimensional feature space

Radial Basis Function (RBF) SVM

$$f(\mathbf{x}) = \sum_i^N \alpha_i y_i \exp\left(-||\mathbf{x} - \mathbf{x}_i||^2/2\sigma^2\right) + b$$

Kernel Method

- Idea:
  - Define $K : X \times X \to \mathbb{R}$, called kernel, such that:

$$\Phi(x) \cdot \Phi(y) = K(x, y).$$

  - K often interpreted as a similarity measure.
- Benefits:
  - Efficiency: K is often more efficient to compute than Φ and the dot product.
  - Flexibility: K can be chosen arbitrarily so long as the existence of Φ is guaranteed (Mercer's condition).

Example - Polynomial Kernels

- Definition:

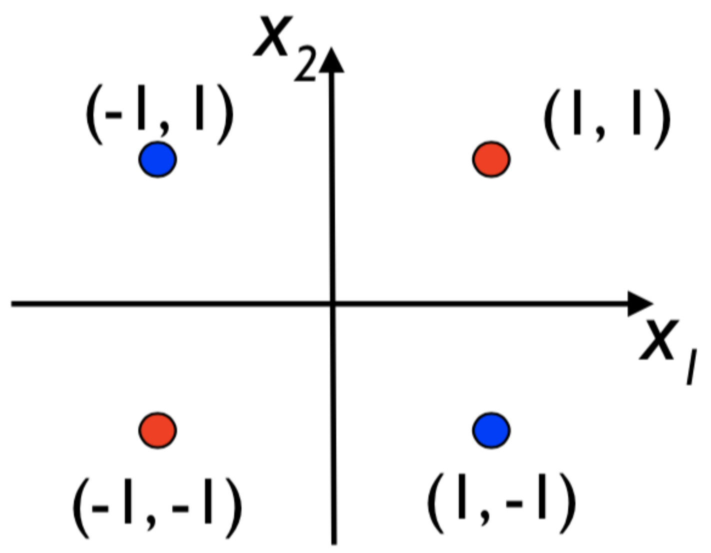
$$\forall x, y \in \mathbb{R}^N, \; K(x,y) = (x \cdot y + c)^d, \quad c > 0.$$

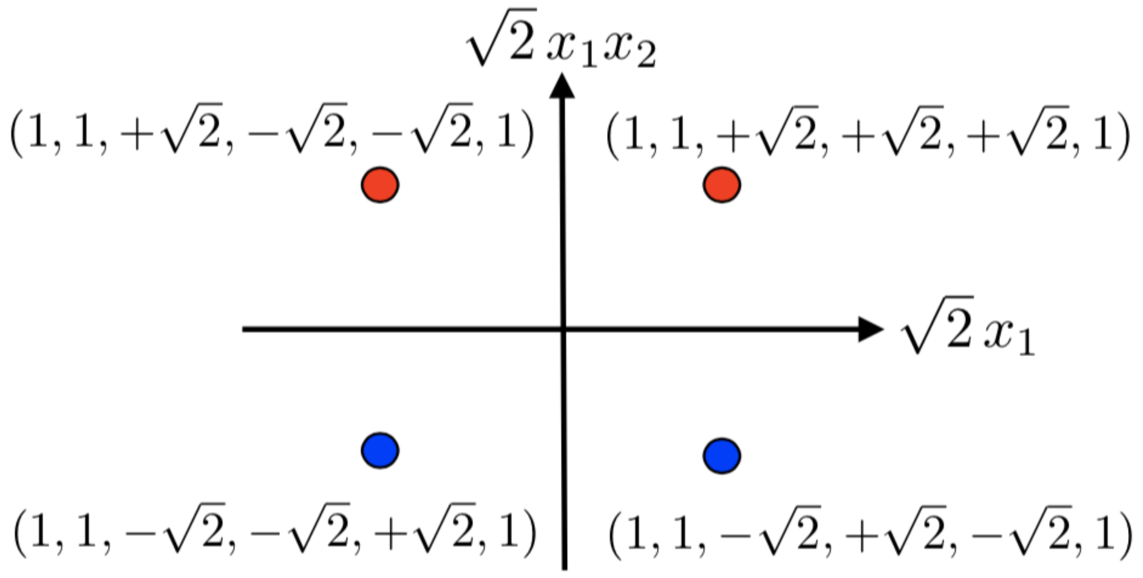
- Example: for $N = 2$ and $d = 2$,

$$\begin{aligned} K(x,y) &= (x_1 y_1 + x_2 y_2 + c)^2 \\ &= \begin{bmatrix} x_1^2 \\ x_2^2 \\ \sqrt{2}\, x_1 x_2 \\ \sqrt{2c}\, x_1 \\ \sqrt{2c}\, x_2 \\ c \end{bmatrix} \cdot \begin{bmatrix} y_1^2 \\ y_2^2 \\ \sqrt{2}\, y_1 y_2 \\ \sqrt{2c}\, y_1 \\ \sqrt{2c}\, y_2 \\ c \end{bmatrix}. \end{aligned}$$

XOR Problem

- Use second-degree polynomial kernel with $c = 1$:





$\sqrt{2}\,x_1x_2$
$\sqrt{2}\,x_1$
$(1, 1, +\sqrt{2}, -\sqrt{2}, -\sqrt{2}, 1)$
$(1, 1, +\sqrt{2}, +\sqrt{2}, +\sqrt{2}, 1)$
$(1, 1, -\sqrt{2}, -\sqrt{2}, +\sqrt{2}, 1)$
$(1, 1, -\sqrt{2}, +\sqrt{2}, -\sqrt{2}, 1)$

Linearly non-separable

Linearly separable by $x_1x_2 = 0.$

Other Standard PDS Kernels

positive definite symmetric (**PDS**)

- Gaussian kernels:

$$K(x, y) = \exp\left(-\frac{||x-y||^2}{2\sigma^2}\right), \quad \sigma \neq 0.$$

- Sigmoid Kernels:

$$K(x, y) = \tanh(a(x \cdot y) + b), \quad a, b \geq 0.$$

Thank you