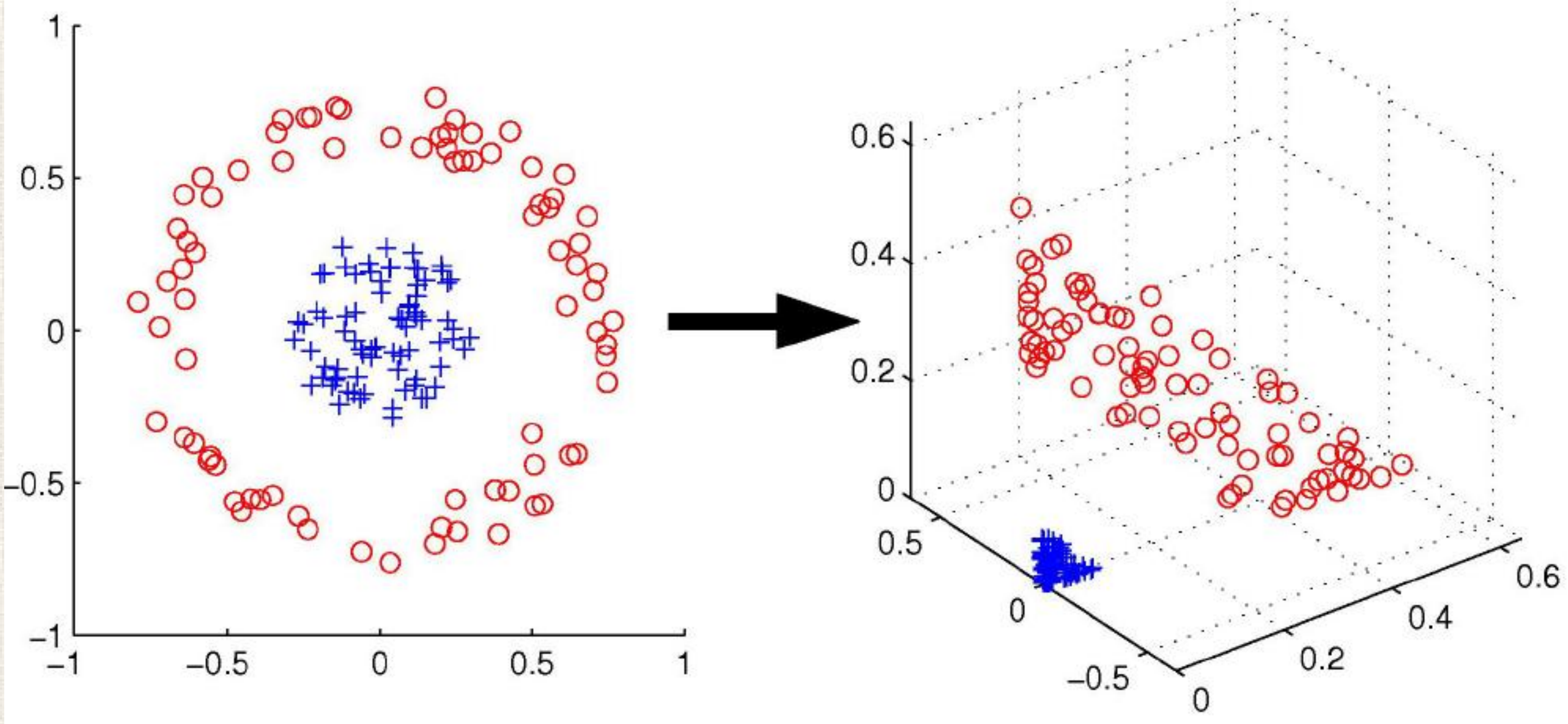
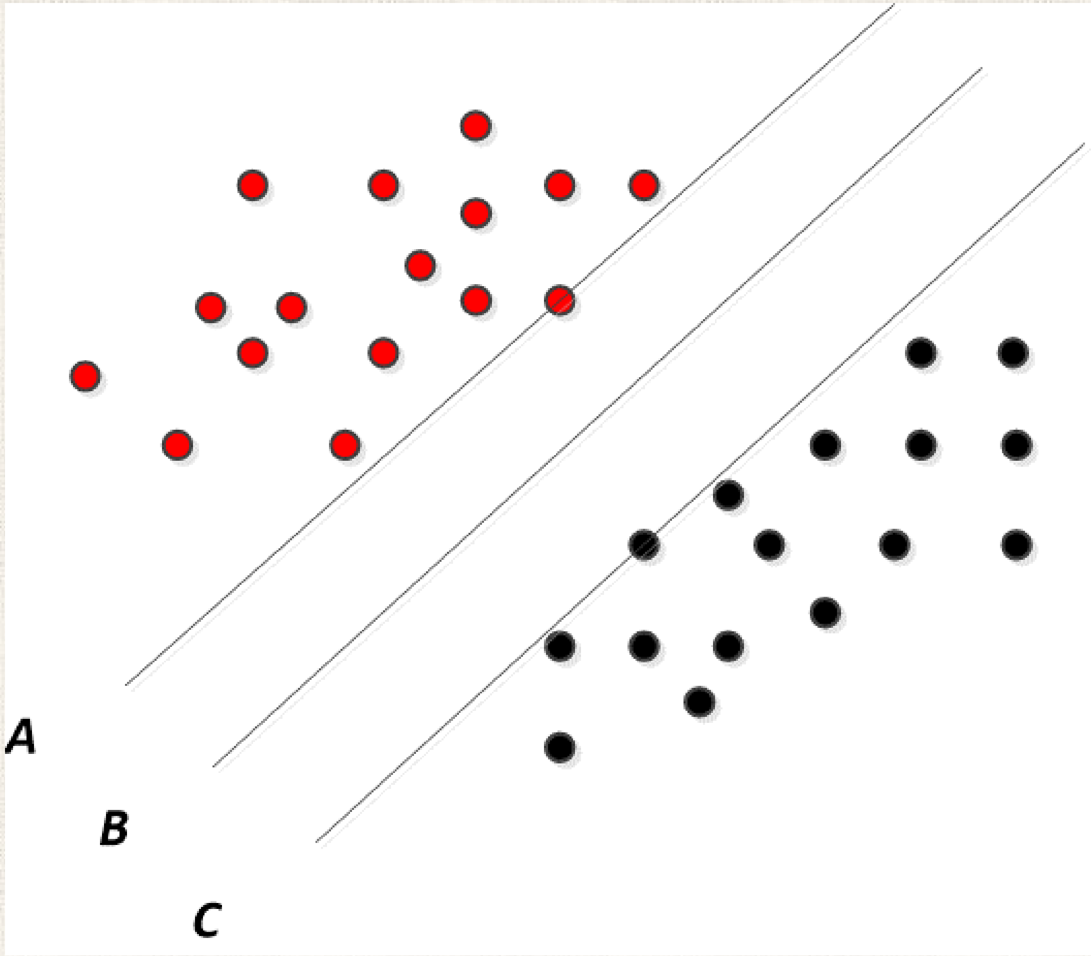
How can we expand the Perceptron model to make it more powerful?

Use Nonlinear Feature Mapping



Introduce the concept of ***margin*** of a hyperplane



We want a hyperplane with the largest margin. What is the definition of margin of a hyperplane?

Support Vector Machine Method

- First, consider the ***binary linearly separable case***: (***hard margin***)

  Given a training data set $\mathcal{D} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{N}, \boldsymbol{x}_i \in \mathcal{R}^d, y_i \in \{-1, +1\}$, we are looking for a linear model classifier in the form:

$$y(\boldsymbol{x}) = \boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x}) + b$$

  Where $\boldsymbol{\phi}(\boldsymbol{x})$ denotes a fixed ***feature-space transformation***, $\boldsymbol{w}$ is the weight vector and we have made the bias (intercept) parameter explicitly. ***No expanded feature vector is used***.

- For a binary linearly separable data set, there exists at least one choice of $\boldsymbol{w}\ and\ b$ to satisfy the following:

$$\boldsymbol{w}^T\boldsymbol{\phi}(\boldsymbol{x}_i) + b > 0, if\ y_i = +1$$

$$\boldsymbol{w}^T\boldsymbol{\phi}(\boldsymbol{x}_i) + b < 0, if\ y_i = -1$$

Which is equivalent to:

$$y_i(\boldsymbol{w}^T\boldsymbol{\phi}(\boldsymbol{x}_i) + b) > 0, \qquad i = 1, \dots, N$$

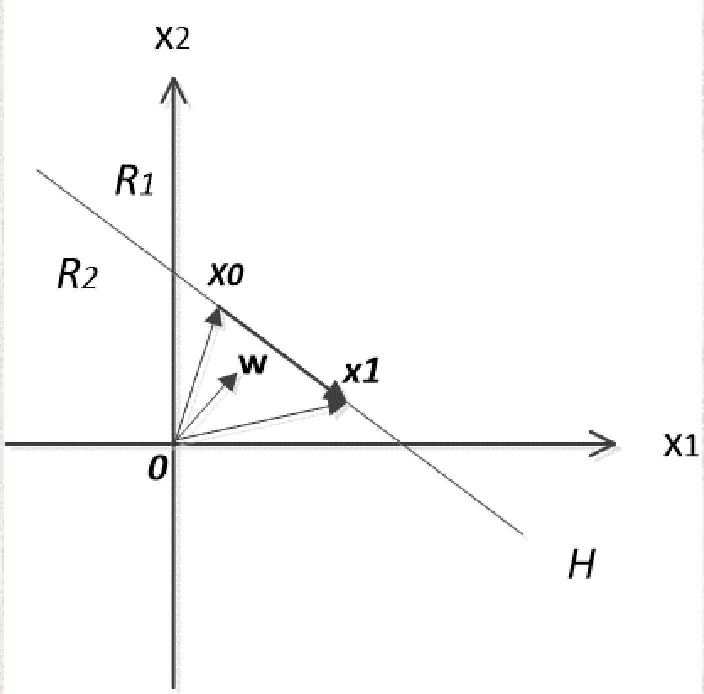
- We look for a hyperplane with the ***maximum margin***.

- Consider the hyperplane $H: \{\boldsymbol{x}: \boldsymbol{w}^T \boldsymbol{x} + b = 0\}$

  First, let's prove that the vector $\boldsymbol{w}$ is normal to the hyperplane H.

  Consider two points $\boldsymbol{x}_0$ and $\boldsymbol{x}_1$ on H. We want to show that $\boldsymbol{w}$ is perpendicular to the vector $\boldsymbol{x}_1 - \boldsymbol{x}_0$

$$\boldsymbol{w}^T(\boldsymbol{x}_1 - \boldsymbol{x}_0) = \boldsymbol{w}^T \boldsymbol{x}_1 - \boldsymbol{w}^T \boldsymbol{x}_0 = b - b = 0$$



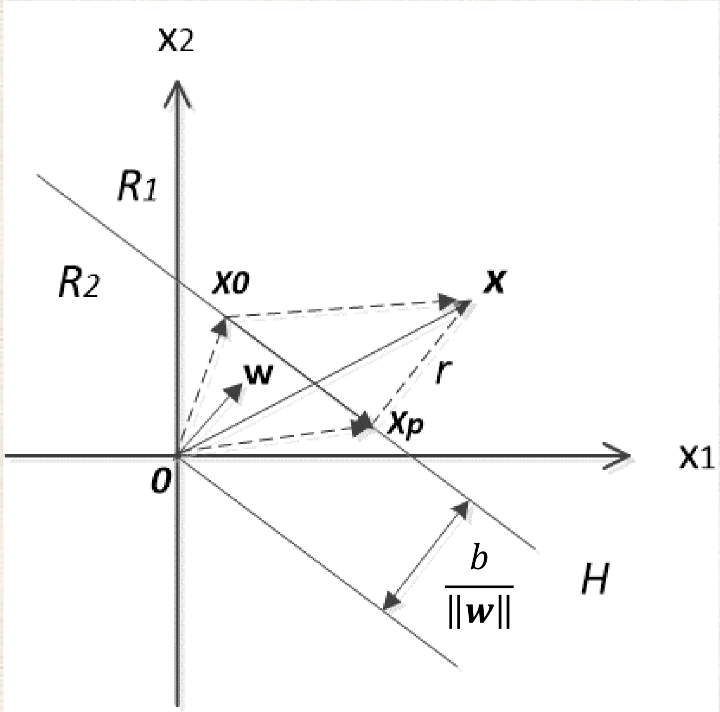
- Consider the hyperplane $H\colon \{\boldsymbol{x}\colon \boldsymbol{w}^T\boldsymbol{x} + b = 0\}$

- Secondly, let's find the ***geometric distance*** from a point $\boldsymbol{x}$ to hyperplane H.

  Let's take any point $\boldsymbol{x}_0$ on H, then, we have,

$$\boldsymbol{x} = \boldsymbol{x}_p + r\frac{\boldsymbol{w}}{||\boldsymbol{w}||}$$

  Where, $\boldsymbol{x}_p - \boldsymbol{x}_0$ is the normal projection *of* $\boldsymbol{x} - \boldsymbol{x}_0$ to H. Hence, $\boldsymbol{x}_p$ is on H. r is the ***algebraic distance*** from $\boldsymbol{x}$ to the hyperplane H (positive if $\boldsymbol{x}$ is on the positive side and negative if $\boldsymbol{x}$ is on the negative side).



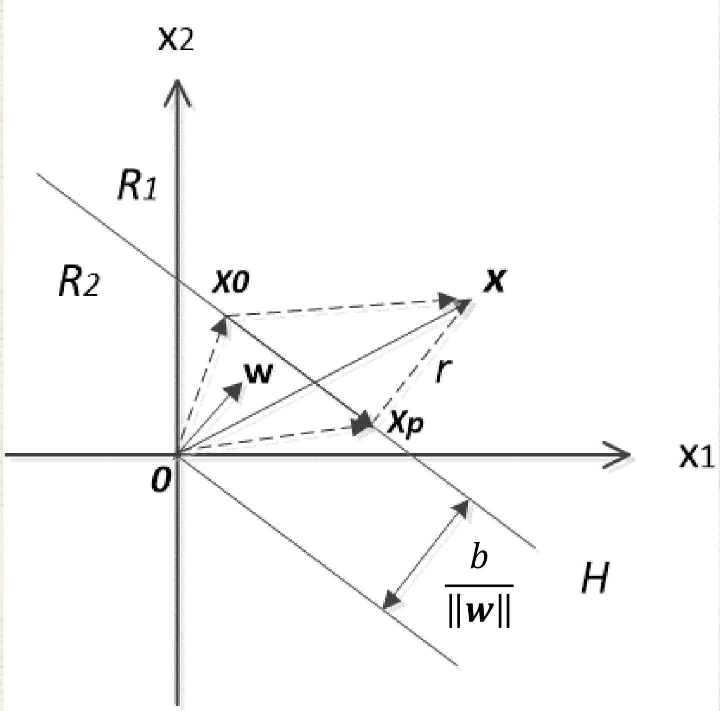
- Hence, we have,

$$\boldsymbol{w}^T\boldsymbol{x} + b = \boldsymbol{w}^T\boldsymbol{x}_p + r\frac{\boldsymbol{w}^T\boldsymbol{w}}{\|\boldsymbol{w}\|} + b = \boldsymbol{w}^T\boldsymbol{x}_p + b + r\frac{\boldsymbol{w}^T\boldsymbol{w}}{\|\boldsymbol{w}\|}$$

$$\Rightarrow \boldsymbol{w}^T\boldsymbol{x} + b = r\|\boldsymbol{w}\|$$

$$\Rightarrow r = \frac{\boldsymbol{w}^T\boldsymbol{x} + b}{\|\boldsymbol{w}\|}$$

- When $\boldsymbol{x} = \boldsymbol{0}, r = \frac{b}{\|\boldsymbol{w}\|}$, this means the distance from the origin to H is $\frac{b}{\|\boldsymbol{w}\|}$



Consider the hyperplane $H: \{\boldsymbol{x}: \boldsymbol{w}^T\boldsymbol{x} + b = 0\}$

- The ***algebraic distance*** of a point $\boldsymbol{x_i}$ to hyperplane H is

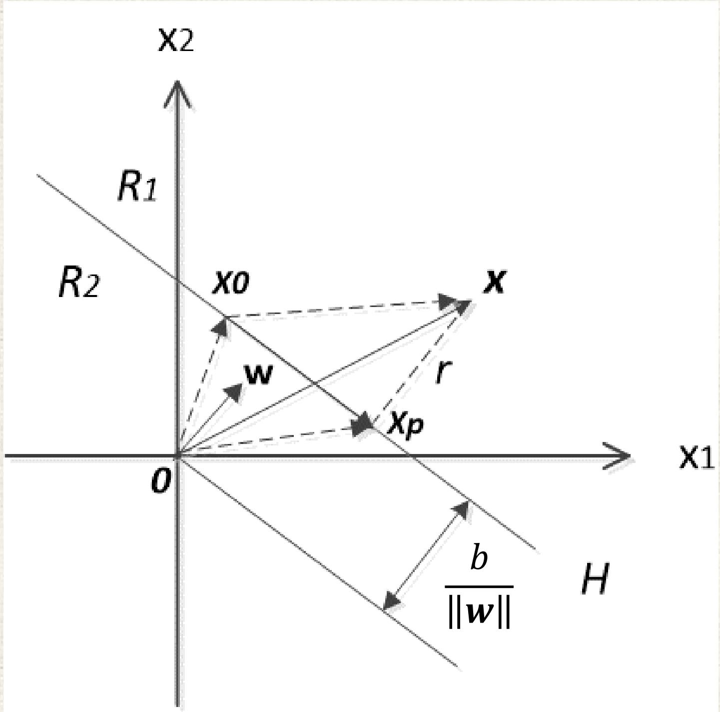
$$r_i = \frac{\boldsymbol{w}^T\boldsymbol{x_i} + b}{||\boldsymbol{w}||}$$

- Then the ***geometric distance*** of a point $\boldsymbol{x}_i$ to hyperplane H is:

$$\frac{y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b)}{||\boldsymbol{w}||}$$

- ***The margin of hyperplane H is the geometric distance of the closest point in the data set to the hyperplane, i.e.,***

$$\min_i \left\{\frac{y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b)}{||\boldsymbol{w}||}\right\}$$



In general, when using feature mapping, let's consider the hyperplane $H: \{\boldsymbol{x}: \boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x}) + b = 0\}$

The margin of hyperplane H is,

$$\min_i \left\{ \frac{y_i(\boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x}_i) + b)}{||\boldsymbol{w}||} \right\}$$

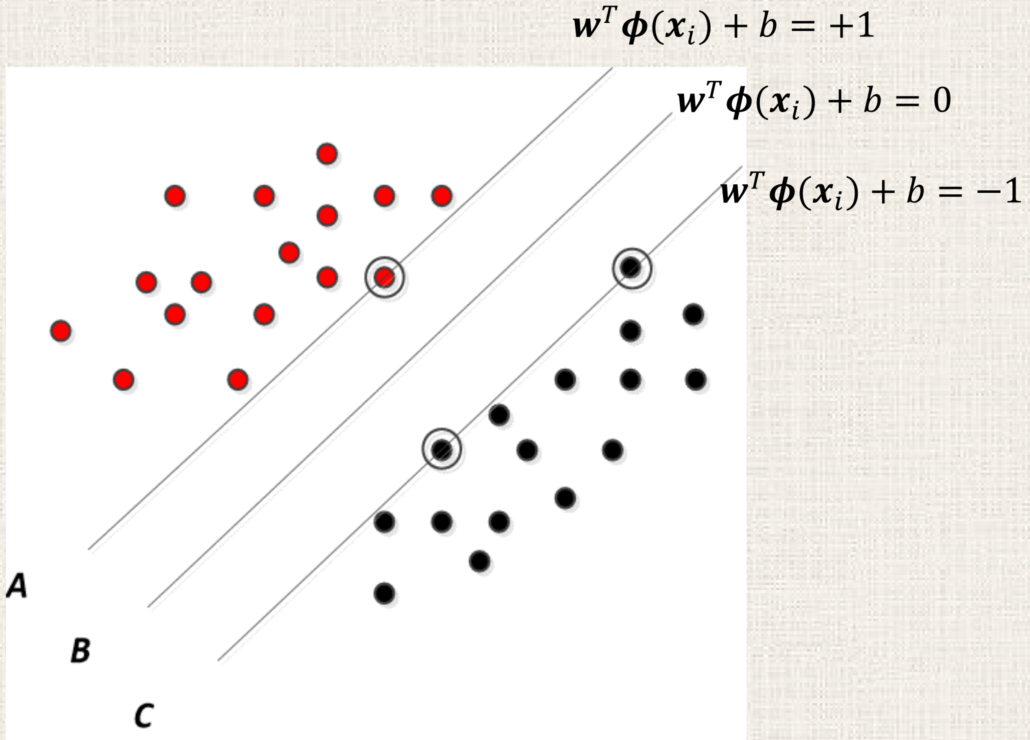
- Since the rescaling of $\boldsymbol{w}\ and\ b$ does not change the hyperplane $H: \{\boldsymbol{x}: \boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x}) + b = 0\}$, we can use this freedom to produce the following constraints:
  - Suppose all data points in the linearly separable data set satisfy the following constraints:

$$\boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x}_i) + b \geq +1,\ \ for\ y_i = +1$$

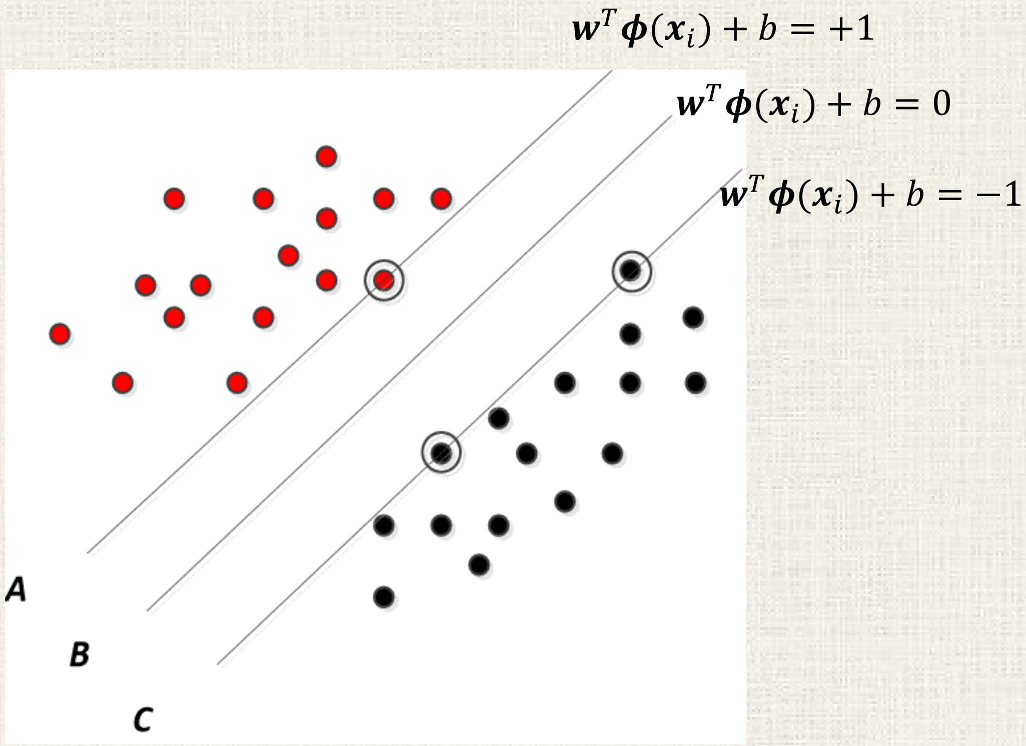
$$\boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x}_i) + b \leq -1,\ \ for\ y_i = -1$$

  - Which can be combined to a set of inequality:

$$y_i(\boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x}_i) + b) \geq +1, \qquad \forall i$$



- Those points $\boldsymbol{x}_i$ that satisfies $\boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x}_i) + b = \pm 1$ are called ***support vectors***.



- Hence, the margin of H is

$$\gamma = \min_i \left\{ \frac{y_i(\boldsymbol{w}^T\boldsymbol{\phi}(\boldsymbol{x}_i) + b)}{\|\boldsymbol{w}\|} \right\} = \frac{1}{\|\boldsymbol{w}\|}$$

- Thus, the maximum margin solution is found by solving the optimization problem:

$$\underset{\boldsymbol{w},b}{arg\,min}\left\{\frac{1}{2}\|\boldsymbol{w}\|^2\right\}$$

$$subject\ to\ (s.t.)\quad y_i(\boldsymbol{w}^T\boldsymbol{\phi}(\boldsymbol{x_i})+b\ )\geq 1,\ \ i=1,2,\dots,N$$

Which is a ***quadratic programming problem***.

- To solve this problem, we need to review some important results from constrained optimization.

The Lagrange Multiplier Method for Constrained Optimization

- Let's consider the following *constrained optimization problem*:

$$\min_{\boldsymbol{x}\in\mathcal{R}^n} f(\boldsymbol{x}) \quad subject\ to \begin{cases} c_i(\boldsymbol{x}) = 0, i \in \mathcal{E} \\ c_j(\boldsymbol{x}) \geq 0, j \in \mathcal{I} \end{cases}$$

Where f and functions c_i, c_j are all smooth, real valued functions, and $\mathcal{E}\ and\ \mathcal{I}$ are two finite set of indices. We call f the ***objective function***, $c_i, i \in \mathcal{E}$ are the ***equality constraints*** and $c_j, j \in \mathcal{I}$ are the ***inequality constraints.***

- we define the ***feasible set*** Ω to be the set of points $\boldsymbol{x}$ that satisfy the constraints, that is,

$$\Omega = \{\boldsymbol{x} | c_i(\boldsymbol{x}) = 0, i \in \mathcal{E}; c_j(\boldsymbol{x}) \geq 0, j \in \mathcal{I}\}$$

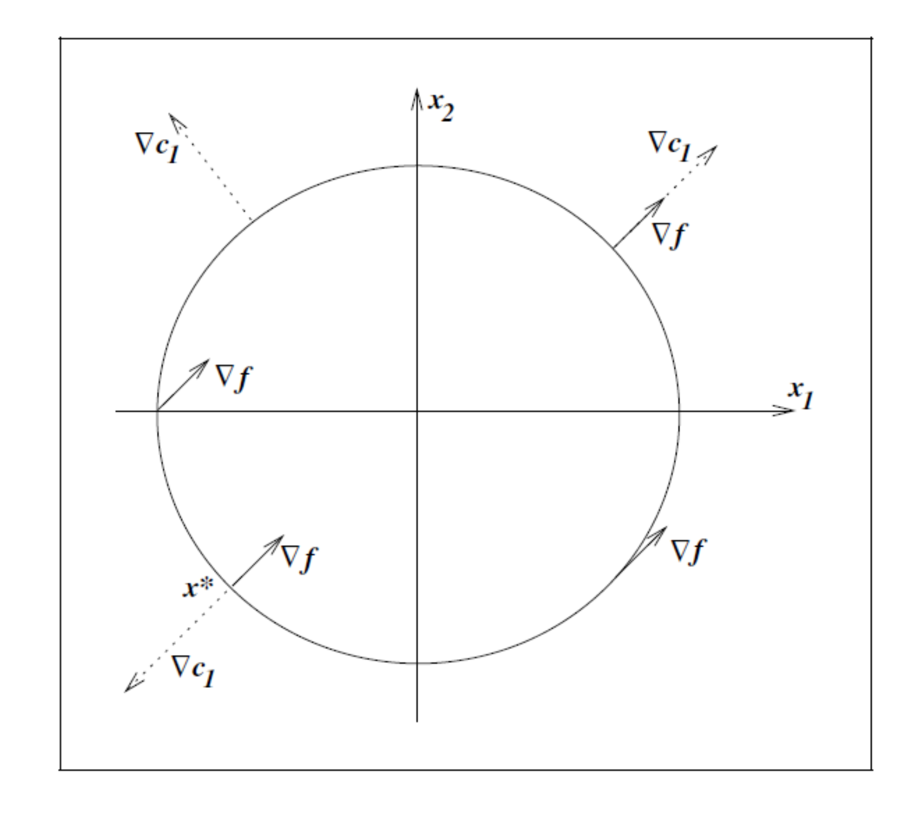
- At a *feasible point* $\boldsymbol{x}$, the inequality constraint $j \in \mathcal{I}$ is said to be ***active*** if $c_j(\boldsymbol{x}) = 0$ and **inactive** if the strict inequality $c_j(\boldsymbol{x}) > 0$ is satisfied.

- The ***active set*** $\mathcal{A}(\boldsymbol{x})$ at any feasible point $\boldsymbol{x}$ consists of the equality constraints indices from $\mathcal{E}$ together with indices of all active inequality constraints.

Example : single equality constraint

$$\min_{x_1,x_2}\{x_1 + x_2\} \quad s.t. \quad x_1^2 + x_2^2 - 2 = 0$$

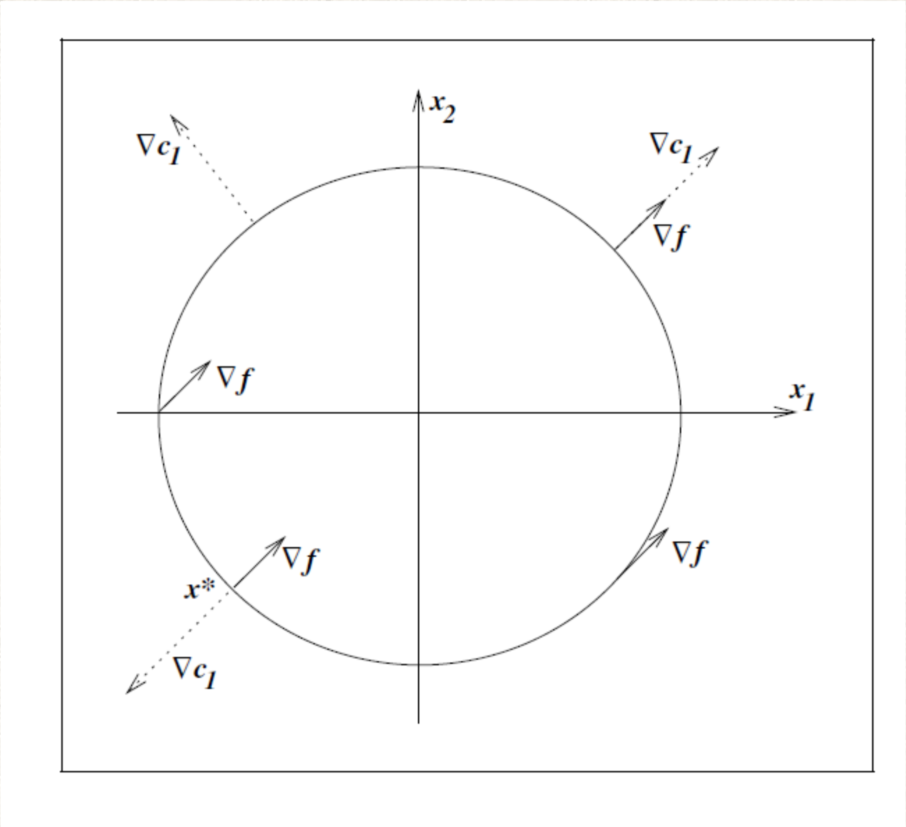
This is a two-variable problem with $f(\boldsymbol{x}) = x_1 + x_2$ and $c_1(\boldsymbol{x}) = x_1^2 + x_2^2 - 2$



- By inspection, we can find that the feasible set of this problem is the circle of radius $\sqrt{2}$ centered at the origin (just the boundary of the circle, not its interior). The solution $\boldsymbol{x}^*$ is $(-1,-1)$

- We also see that at the solution $\boldsymbol{x}^*$, the ***constraint normal*** $\nabla c_1(\boldsymbol{x}^*)$ is parallel to $\nabla f\,(\boldsymbol{x}^*)$. That is, there is a scalar λ_1^* such that

$$\nabla f\,(\boldsymbol{x}^*) = \lambda_1^* \nabla c_1(\boldsymbol{x}^*)$$



- Actually,

$$\nabla f(\boldsymbol{x}) = \begin{bmatrix}1\\1\end{bmatrix}, \nabla c_1(\boldsymbol{x}) = 2\begin{bmatrix}x_1\\x_2\end{bmatrix}, \qquad \nabla f(\boldsymbol{x}^*) = \begin{bmatrix}1\\1\end{bmatrix}, \nabla c_1(\boldsymbol{x}^*) = 2\begin{bmatrix}-1\\-1\end{bmatrix}, \qquad (\lambda_1^* = -\frac{1}{2})$$

- Actually, the necessary condition $\nabla f\ (\boldsymbol{x}^*) = \lambda_1^* \nabla c_1(\boldsymbol{x}^*)$ is true for the entire class of equality constrained optimization problem. Let's prove this:

- Consider the first order Taylor series approximation of $f(\boldsymbol{x})$ and $c_1(\boldsymbol{x})$. To retain feasibility with respect to $c_i(\boldsymbol{x}) = 0$, we require any small step $\boldsymbol{s}$ to satisfy that $c_1(\boldsymbol{x}+\boldsymbol{s}) = 0$, that is,

$$0 = c_1(\boldsymbol{x}+\boldsymbol{s}) \approx c_1(\boldsymbol{x}) + \nabla c_1(\boldsymbol{x})^T \boldsymbol{s} = \nabla c_1(\boldsymbol{x})^T \boldsymbol{s}$$

- Hence, the step $\boldsymbol{s}$ retains feasibility with respect to c_1, when it satisfies

$$\nabla c_1(\boldsymbol{x})^T \boldsymbol{s} = 0$$

- Similarly, if we want $\boldsymbol{s}$ to produce a decrease in f, we should have so that

$$f(\boldsymbol{x}+\boldsymbol{s}) - f(\boldsymbol{x}) \approx \nabla f(\boldsymbol{x})^T \boldsymbol{s} < 0$$

i.e.,

$$\nabla f(\boldsymbol{x})^T \boldsymbol{s} < 0$$

- If at a point $\boldsymbol{x}^*$, there is no such direction of $\boldsymbol{s}$ to satisfy

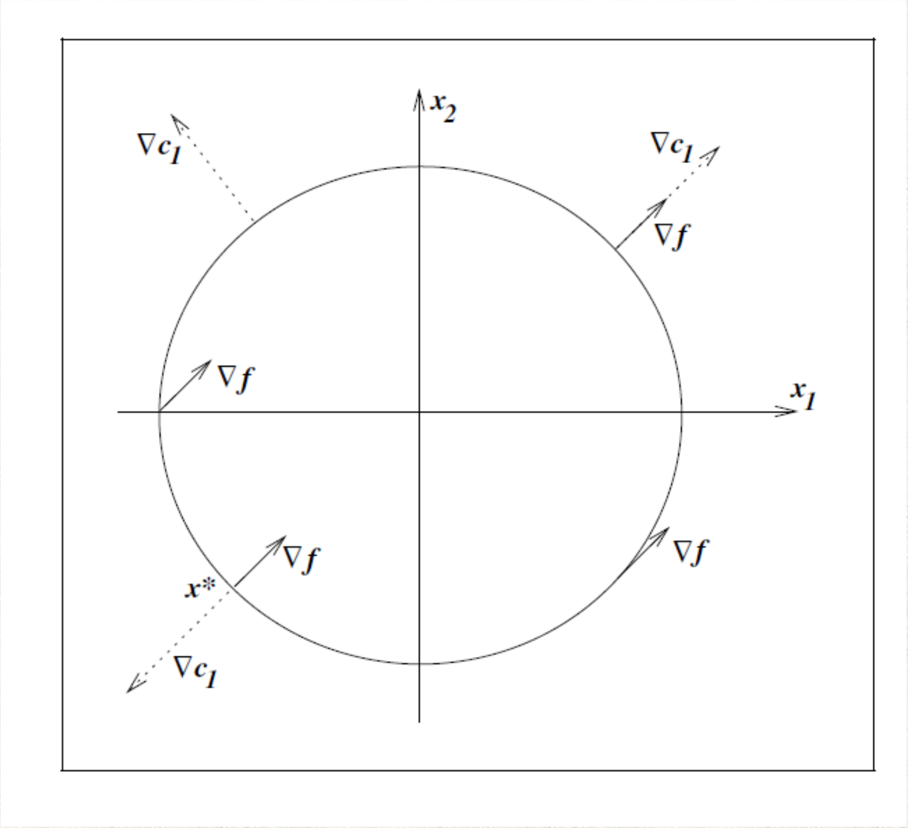
$$\nabla c_1(\boldsymbol{x})^T \boldsymbol{s} = 0 \ \ and \ \ \nabla f(\boldsymbol{x})^T \boldsymbol{s} < 0$$

Then the point $\boldsymbol{x}^*$ appear to be a ***stationary point***

- From the figure, its easy to check that the only way that the above condition can not be met is when $\nabla c_1(\boldsymbol{x})$ and $\nabla f(\boldsymbol{x})$ are parallel, that is

$$\nabla f(\boldsymbol{x}) = \lambda_1 \nabla c_1(\boldsymbol{x})$$

hold at some $\boldsymbol{x}$, for some scalar λ_1.



- Let's introduce the ***Lagrangian function***

$$\mathcal{L}(\boldsymbol{x}, \lambda_1) = f(\boldsymbol{x}) - \lambda_1 c_1(\boldsymbol{x})$$

Notice that,

$$\nabla_{\boldsymbol{x}}\mathcal{L}(\boldsymbol{x}, \lambda_1) = \nabla f(\boldsymbol{x}) - \lambda_1 \nabla c_1(\boldsymbol{x})$$

Then, the condition $\nabla f(\boldsymbol{x}) = \lambda_1 \nabla c_1(\boldsymbol{x})$ is equivalent to

$$\nabla_{\boldsymbol{x}}\mathcal{L}(\boldsymbol{x}, \lambda_1) = 0$$

- This means we can search for solutions of the equality-constrained optimization problem by seeking stationary points of the ***Lagrangian function***.

- The scalar λ_1 is called the ***Lagrange multiplier*** for the equation constraint $c_1(\boldsymbol{x}) = 0$.

Let's go back to the example: $min\{x_1 + x_2\} \quad s.t. \quad x_1^2 + x_2^2 - 2 = 0$

- The ***Lagrangian function*** is $\mathcal{L}(\boldsymbol{x}, \lambda_1) = x_1 + x_2 - \lambda_1(x_1^2 + x_2^2 - 2)$. Then,

$$\nabla_{x_1}\mathcal{L}(\boldsymbol{x}, \lambda_1) = 1 - 2\lambda_1 x_1 = 0 \Rightarrow x_1 = \frac{1}{2\lambda_1}$$

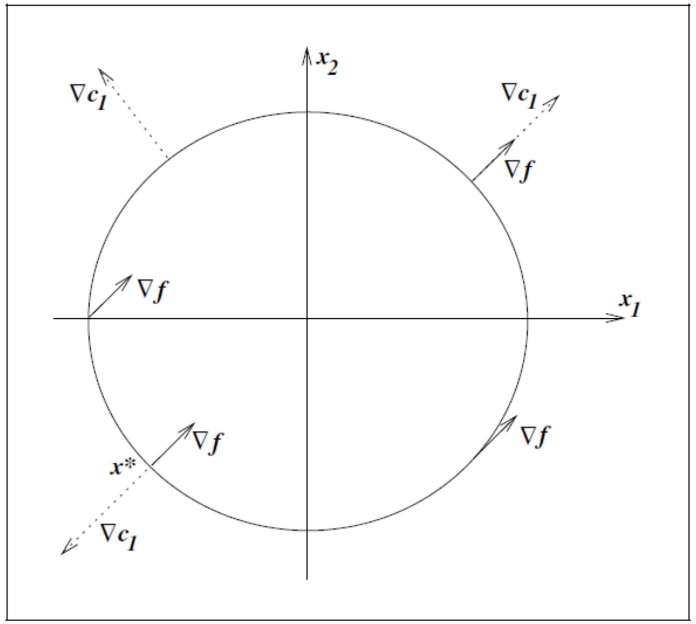
$$\nabla_{x_2}\mathcal{L}(\boldsymbol{x}, \lambda_1) = 1 - 2\lambda_1 x_2 = 0 \Rightarrow x_2 = \frac{1}{2\lambda_1}$$

- Substitute these into $c_1(\boldsymbol{x}) = 0$ gives: $\lambda_1 = \frac{1}{2} \ or \ -\frac{1}{2}$.
  - When $\lambda_1 = \frac{1}{2}$, we have the stationary point as $\boldsymbol{x}^* = [1 \quad 1]^T$
  - When $\lambda_1 = -\frac{1}{2}$, we have the stationary point as $\boldsymbol{x}^* = [-1 \quad -1]^T$
  - The actual solution is $\boldsymbol{x}^* = [-1 \quad -1]^T$.

Example: a single inequality constraint

$$min\{x_1 + x_2\} \quad s.t. \quad 2 - x_1^2 - x_2^2 \geq 0$$

- What is the feasible set?
- What is the solution by visual inspection?



- ***Follow the same argument that a given feasible point x is not optimal if we can find a small step s that both retains feasibility and decreases the objective function $f(x)$ to first order.***

- The step $\boldsymbol{s}$ improves the objective function, to first order, if

$$f(\boldsymbol{x}+\boldsymbol{s}) - f(\boldsymbol{x}) \approx \nabla f(\boldsymbol{x})^T \boldsymbol{s} < 0 \qquad \textit{(svm-1)}$$

- The step $\boldsymbol{s}$ retains feasibility if

$$0 \le c_1(\boldsymbol{x}+\boldsymbol{s}) \approx c_1(\boldsymbol{x}) + \nabla c_1(\boldsymbol{x})^T \boldsymbol{s}$$

$$\Rightarrow c_1(\boldsymbol{x}) + \nabla c_1(\boldsymbol{x})^T \boldsymbol{s} \ge 0 \qquad \textit{(svm-2)}$$

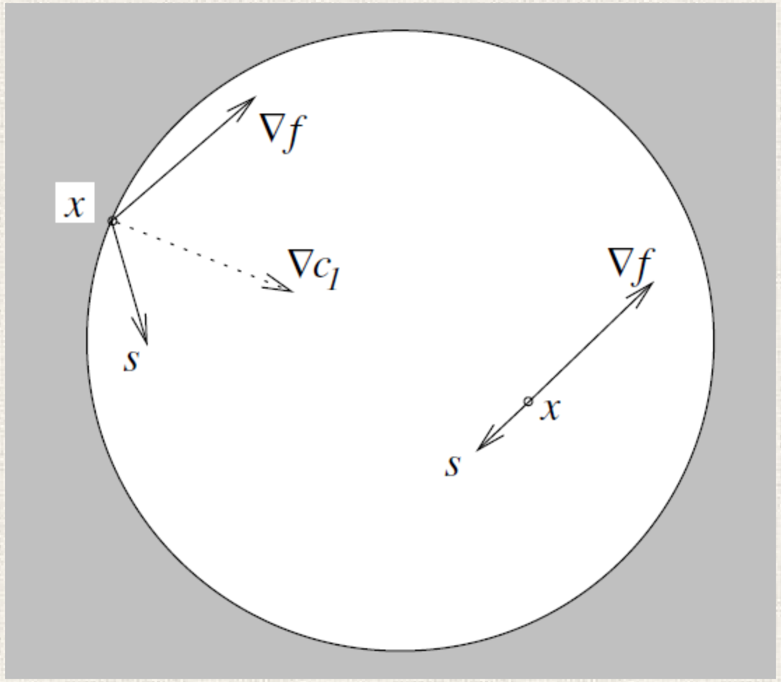
To determine if a step $\boldsymbol{s}$ exists that satisfies both conditions *(svm-1)* and *(svm-2)*, we consider two cases:

- **Case I:** $\boldsymbol{x}$ lies strictly inside the circle so that $c_1(\boldsymbol{x}) > 0$ holds

  In this case, if $\boldsymbol{s}$ is small enough, the condition *(svm-2)* can be satisfied!

  In fact, whenever $\nabla f(\boldsymbol{x}) \neq 0$, we can obtain a step $\boldsymbol{s}$ that satisfies both *(svm-1)* and *(svm-2)* by setting $\boldsymbol{s} = -\alpha \nabla f(\boldsymbol{x})$ for any positive scalar α sufficiently small.

  When $\boldsymbol{\nabla f(x) = 0}$, such a s can not be found. (svm-1 can not be met)



- **Case II:** $\boldsymbol{x}$ lies on the boundary of the circle so that $c_1(\boldsymbol{x}) = 0$
  - In this case, the condition *(svm-2) and (svm-1)* becomes

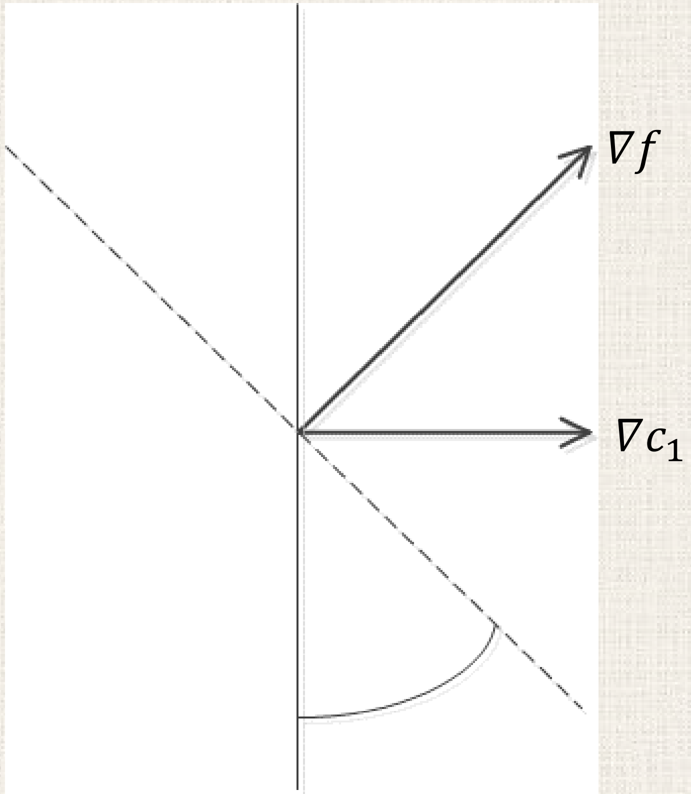
$$\nabla f(\boldsymbol{x})^T \boldsymbol{s} < 0 \;\; and \;\; \nabla c_1(\boldsymbol{x})^T \boldsymbol{s} \geq 0 \qquad \textit{(svm-3)}$$

  - The first part of the condition defines an open half-space and the second part defines a closed half-space. ***It is clear that the intersection of these two regions is empty only when $\boldsymbol{\nabla f(x)}$ and $\boldsymbol{\nabla c_1(x)}$ point in the same direction***, that is, when,

$$\nabla f(\boldsymbol{x}) = \lambda_1 \nabla c_1(\boldsymbol{x}), \;\; for\ some\ \lambda_1 \geq 0 \qquad \textit{(svm-4)}$$

Notice that the sign of the $\boldsymbol{\lambda_1}$ is significant here:

If $\boldsymbol{\lambda_1 < 0}$, then $\boldsymbol{\nabla f(x)}$ and $\boldsymbol{\nabla c_1(x)}$ point in the opposite direction and the sets of directions of s that meet the requirement of (svm-3) will make up an entire open half-plane!



- Define the ***Lagrangian function*** as: $\mathcal{L}(\boldsymbol{x}, \lambda_1) = f(\boldsymbol{x}) - \lambda_1 c_1(\boldsymbol{x})$
- The optimality conditions for both **case I** and **case II** can be expressed using **Lagrangian function** as: When no first order feasible descent direction exist at some point $\boldsymbol{x}^*$, we have that

$$\nabla_{\boldsymbol{x}} \mathcal{L}(\boldsymbol{x}^*, \lambda_1^*) = 0 \Rightarrow \boldsymbol{\nabla f}(\boldsymbol{x}^*) = \lambda_1^* \boldsymbol{\nabla c_1}(\boldsymbol{x}^*),\ \textit{for some } \lambda_1^* \geq \mathbf{0} \qquad \text{(svm-5)}$$

Where we also require that

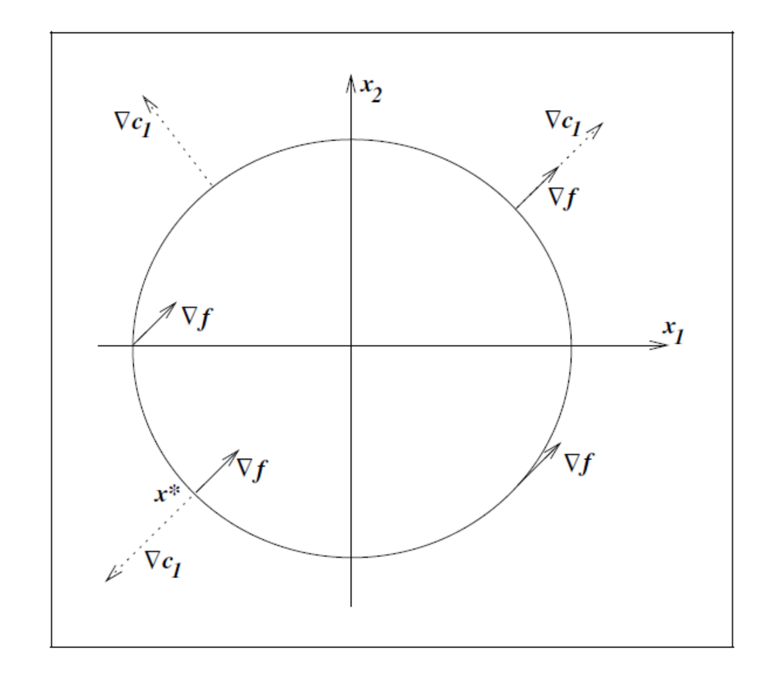
$$\lambda_1^* \boldsymbol{c_1}(\boldsymbol{x}^*) = \mathbf{0} \qquad \text{(svm-6)}$$

- Condition (svm-6) is known as the ***complementarity condition***. ***It implies that the Lagrange multiplier λ_1 can be strictly positive only when the corresponding constraint $\boldsymbol{c_1}$ is active.***
- In case I, we have that $c_1(\boldsymbol{x}^*) > 0$, so (svm-6) requires $\lambda_1^* = 0$, (svm-5) reduces to $\nabla f(\boldsymbol{x}^*) = 0$
- In case II, we have that $c_1(\boldsymbol{x}^*) = 0$, (svm-6) allows λ_1^* to take a nonnegative value. So (svm-5) becomes equivalent to (svm-4)

Example: a single inequality constraint

$$\min\{x_1 + x_2\} \quad s.t. \quad 2 - x_1^2 - x_2^2 \geq 0$$

Can you find the solution by visual inspection?



Solution:

- We can find the ***Lagrangian function*** of this problem as:

$$\mathcal{L}(\boldsymbol{x}, \lambda_1) = f(\boldsymbol{x}) - \lambda_1 c_1(\boldsymbol{x})$$

$$= x_1 + x_2 - \lambda_1(2 - x_1^2 - x_2^2) = x_1 + x_2 + \lambda_1(x_1^2 + x_2^2 - 2)$$

$$= \lambda_1 x_1^2 + x_1 + \lambda_1 x_2^2 + x_2 - 2\lambda_1$$

- Then,

$$\nabla_{x_1}\mathcal{L}(\boldsymbol{x}, \lambda_1) = 2\lambda_1 x_1 + 1 = 0 \Rightarrow x_1 = -\frac{1}{2\lambda_1} \qquad \text{(svm-7)}$$

$$\nabla_{x_2}\mathcal{L}(\boldsymbol{x}, \lambda_1) = 2\lambda_1 x_2 + 1 = 0 \Rightarrow x_2 = -\frac{1}{2\lambda_1} \qquad \text{(svm-8)}$$

Solution (continue):

- With the condition that

$$\lambda_1 \geq 0 \; and \; \lambda_1(x_1^2 + x_2^2 - 2) = 0$$

λ_1 is strictly positive only when $c_1(\boldsymbol{x})$ is ***active***, that is,

$$c_1(\boldsymbol{x}) = 2 - x_1^2 - x_2^2 = 0 \qquad \text{(svm-9)}$$

- Substituting (svm-7) and (svm-8) into (svm-9) gives:

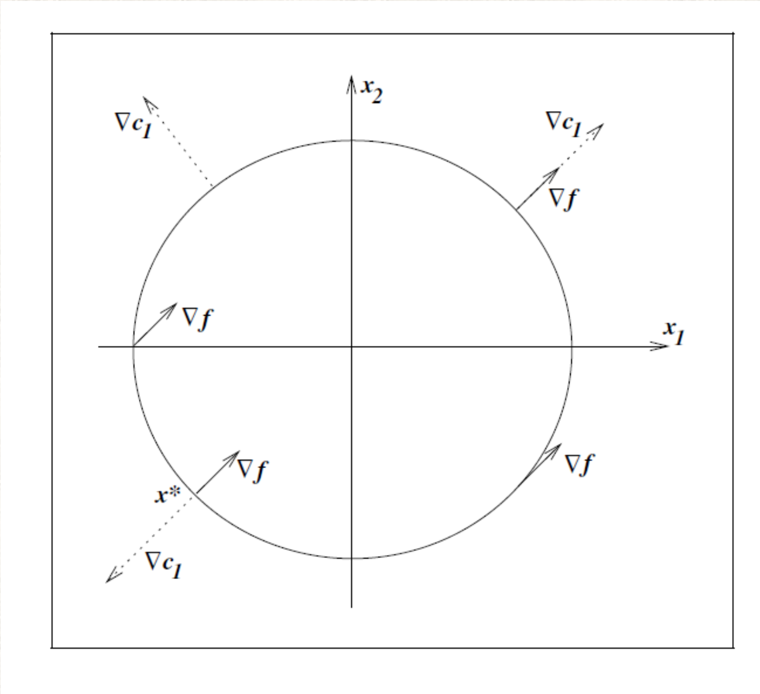
$$\frac{1}{4\lambda_1^2} + \frac{1}{4\lambda_1^2} - 2 = 0 \Rightarrow \lambda_1^* = \frac{1}{2}$$

- Then we have the solution $\boldsymbol{x}^* = \begin{bmatrix} -\frac{1}{2\lambda_1} \\ -\frac{1}{2\lambda_1} \end{bmatrix} = \begin{bmatrix} -1 \\ -1 \end{bmatrix}$

Example: two inequality constraints

$$min\{x_1 + x_2\} \quad s.t. \quad 2 - x_1^2 - x_2^2 \geq 0, x_2 \geq 0$$

Can you find the solution by visual inspection?



Solution:

- We can find the ***Lagrangian function*** of this problem as:

$$\mathcal{L}(\boldsymbol{x},\boldsymbol{\lambda}) = f(\boldsymbol{x}) - \lambda_1 c_1(\boldsymbol{x}) - \lambda_2 c_2(\boldsymbol{x})$$

$$= x_1 + x_2 - \lambda_1(2 - x_1^2 - x_2^2) - \lambda_2 x_2$$

$$= \lambda_1 x_1^2 + x_1 + \lambda_1 x_2^2 + (1-\lambda_2)x_2 - 2\lambda_1$$

Where $\boldsymbol{\lambda} = [\lambda_1 \quad \lambda_2]^T$ is the vector of *Lagrange multipliers*.

- The extension of condition (svm-5) becomes:

$$\nabla_{\boldsymbol{x}}\mathcal{L}(\boldsymbol{x}^*,\boldsymbol{\lambda}^*) = 0,\ \text{for some } \boldsymbol{\lambda}^* \geq 0$$

- That is,

$$\nabla_{x_1}\mathcal{L}(\boldsymbol{x},\boldsymbol{\lambda}) = 2\lambda_1 x_1 + 1 = 0 \Rightarrow x_1 = -\frac{1}{2\lambda_1} \qquad \text{(svm-10)}$$

$$\nabla_{x_2}\mathcal{L}(\boldsymbol{x},\boldsymbol{\lambda}) = 2\lambda_1 x_2 + 1 - \lambda_2 = 0 \Rightarrow x_2 = \frac{\lambda_2 - 1}{2\lambda_1} \qquad \text{(svm-11)}$$

Solution (continue):

- λ_1 is strictly positive only when $c_1(\boldsymbol{x})$ is active, that is,

$$c_1(\boldsymbol{x}) = 2 - x_1^2 - x_2^2 = 0 \qquad \text{(svm-12)}$$

- λ_2 is strictly positive only when $c_2(\boldsymbol{x})$ is active, that is,

$$c_2(\boldsymbol{x}) = x_2 = 0 \qquad \text{(svm-13)}$$

- Substituting (svm-13) into (svm-12) gives:

$$2 - x_1^2 = 0 \Rightarrow x_1^2 = 2 \qquad \text{(svm-14)}$$

- Substituting (svm-10) into (svm-14) gives: $\frac{1}{4\lambda_1^2} = 2 \Rightarrow \lambda_1 = \frac{1}{2\sqrt{2}}$

- Substituting (svm-11) into (svm-13) gives: $\frac{\lambda_2 - 1}{2\lambda_1} = 0 \Rightarrow \lambda_2 = 1$

Solution (continue):

- Hence, we have,

$$\boldsymbol{\lambda}^* = \begin{bmatrix} \frac{1}{2\sqrt{2}} \\ 1 \end{bmatrix}$$

- The corresponding solution point $\boldsymbol{x}^* = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} -\frac{1}{2\lambda_1} \\ \frac{\lambda_2 - 1}{2\lambda_1} \end{bmatrix} = \begin{bmatrix} -\sqrt{2} \\ 0 \end{bmatrix}$

The Lagrangian Dual Problem

- Let's restate the constrained optimization problem (the ***primal problem***):

$$\min_{\boldsymbol{x} \in \mathcal{R}^n} f(\boldsymbol{x}) \quad subject\ to\ c_i(\boldsymbol{x}) \geq 0, i = 1, \dots, m$$

- Let's define the ***Lagrangian function*** *as:*

$$\mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}) = f(\boldsymbol{x}) - \sum_{i=1}^{m} \lambda_i c_i(\boldsymbol{x})$$

Where, $\boldsymbol{\lambda} = [\lambda_1 \quad \dots \quad \lambda_m] \in \mathcal{R}^m$ is the *Lagrange multiplier vector.*

- Then we have the following first-order optimality conditions (***KKT conditions***) of the primal problem:

$$\nabla f(\boldsymbol{x}) = \lambda_1 \nabla c_1(\boldsymbol{x}) + \lambda_2 \nabla c_2(\boldsymbol{x}) + \cdots + \lambda_m \nabla c_m(\boldsymbol{x})$$

$$c_i(x) \geq 0, \qquad i = 1, \dots, m$$

$$\lambda_i c_i(x) = 0, \qquad i = 1, \dots, m$$

- Let's define the ***dual objective function*** $q: \mathcal{R}^n \rightarrow \mathcal{R}$ as:

$$q(\boldsymbol{\lambda}) = \inf_{\boldsymbol{x}} \mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda})$$

  Notice that the calculation of the infimum requires finding the global minimizer of the function $\mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda})$. This is not a problem when the function is convex.

- The ***dual problem*** is defined as:

$$\max_{\boldsymbol{\lambda}} q(\boldsymbol{\lambda}) \quad subject\ to\ \boldsymbol{\lambda} \geq 0$$

- The ***dual problem*** provides a lower bound to the solution of the ***primal problem***. The difference between the optimal value of ***primal problem*** and ***dual problem*** is called the *duality gap*. For convex optimization problem, the duality gap is zero.

Example: consider the problem

$$\min_{x_1,x_2}\left\{\frac{1}{2}(x_1^2+x_2^2)\right\} \quad s.t. \quad x_1-1\geq 0$$

Solution:

- The ***Lagrangian function*** is:

$$\mathcal{L}(x_1, x_2, \lambda_1) = \frac{1}{2}(x_1^2 + x_2^2) - \lambda_1(x_1 - 1)$$

- If we hold λ_1 fixed, this is a convex function of $\boldsymbol{x} = [x_1 \quad x_2]^T$. Hence, the infimum with respect to $\boldsymbol{x}$ is achieved when the partial derivatives with respect to $x_1\ and\ x_2$ are zero, that is,

$$\frac{\partial \mathcal{L}}{\partial x_1} = 0 \Rightarrow x_1 - \lambda_1 = 0 \Rightarrow x_1 = \lambda_1;$$

$$\frac{\partial \mathcal{L}}{\partial x_2} = 0 \Rightarrow x_2 = 0$$

- By substituting these infimal values into $\mathcal{L}(x_1, x_2, \lambda_1)$, we obtain the dual objective function:

$$q(\lambda_1) = \frac{1}{2}(\lambda_1^2 + 0) - \lambda_1(\lambda_1 - 1) = -\frac{1}{2}\lambda_1^2 + \lambda_1$$

Solution (continue):

- Hence the ***dual problem*** is:

$$\max_{\lambda_1 \geq 0} \left(-\frac{1}{2}\lambda_1^2 + \lambda_1 \right)$$

To solve this problem, we let

$$\frac{d\left(-\frac{1}{2}\lambda_1^2 + \lambda_1\right)}{d\lambda_1} = 0 \Rightarrow \lambda_1 = 1$$

This is the solution of the ***dual problem***!!!

- Therefore, we have the solution of the ***primal problem*** as:

$$\boldsymbol{x}^* = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} \lambda_1 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

Support Vector Machine Model

- Thus, the maximum margin solution is found by solving the optimization problem:

$$\underset{\boldsymbol{w},b}{argmin}\left\{\frac{1}{2}\|\boldsymbol{w}\|^2\right\}$$

$$subject\ to \quad y_i(\boldsymbol{w}^T\boldsymbol{\phi}(\boldsymbol{x_i}) + b\ ) \geq 1,\ \ i = 1,2,\dots,N$$

Which is a quadratic programming problem.

- This quadratic optimization problem is called the ***primal problem*** of the support vector machine model and can be solved using standard optimization algorithms.

Instead of directly work on the ***primal problem***, we will solve the ***dual problem*** first.

- To solve this problem, we introduce *Lagrange multipliers* $a_i \geq 0$, with one multiplier for each constraint, giving the ***Lagrangian function***

$$L_P(\boldsymbol{w}, b, \boldsymbol{a}) = \frac{1}{2}\|\boldsymbol{w}\|^2 - \sum_{i=1}^{N} a_i\{y_i(\boldsymbol{w}^T\boldsymbol{\phi}(\boldsymbol{x_i}) + b\ ) - 1\}$$

Where $\boldsymbol{a} = [a_1, a_2, \dots, a_N]^T$ is the *Lagrange multiplier vector*.

- To find the ***dual problem***, we have,

$$L_P(\boldsymbol{w}, b, \boldsymbol{a}) = \frac{1}{2}\|\boldsymbol{w}\|^2 - \sum_{i=1}^{N} a_i\{y_i(\boldsymbol{w}^T\boldsymbol{\phi}(\boldsymbol{x_i}) + b\ ) - 1\}$$

$$= \frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} - \sum_{i=1}^{N}\{a_i y_i\left(\boldsymbol{w}^T\boldsymbol{\phi}(\boldsymbol{x_i})\right) + a_i y_i b - a_i\}$$

$$= \frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} - \sum_{i=1}^{N} a_i y_i\left(\boldsymbol{w}^T\boldsymbol{\phi}(\boldsymbol{x_i})\right) - \sum_{i=1}^{N} a_i y_i b + \sum_{i=1}^{N} a_i$$

- Setting the derivatives of $L_P(\boldsymbol{w}, b, \boldsymbol{a})$ with respect to $\boldsymbol{w}\ and\ b$ equal to zero, we obtain the following conditions:

$$\frac{\partial L_P}{\partial \boldsymbol{w}} = 0 \Rightarrow \boldsymbol{w} = \sum_{i=1}^{N} a_i y_i \boldsymbol{\phi}(\boldsymbol{x}_i)$$

$$\frac{\partial L_P}{\partial b} = 0 \Rightarrow \sum_{i=1}^{N} a_i y_i = 0$$

- Eliminating $\boldsymbol{w}$ and b from $L_P(\boldsymbol{w}, b, \boldsymbol{a})$ using these conditions then gives the dual representation of the maximum margin problem in which we ***maximize***

$$L_D(\boldsymbol{a}) = \sum_{i=1}^{N} a_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} a_i a_j y_i y_j \boldsymbol{\phi}(\boldsymbol{x_i})^T \boldsymbol{\phi}(\boldsymbol{x}_j)$$
$$= \sum_{i=1}^{N} a_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} a_i a_j y_i y_j k(\boldsymbol{x}_i, \boldsymbol{x}_j)$$

Subject to the constraints:

$$a_i \geq 0, all\ i \ \ and \ \sum_{i=1}^{N} a_i y_i = 0$$

- Here the ***kernel function*** is defined by

$$k(\boldsymbol{x}, \boldsymbol{x}') = \boldsymbol{\phi}(\boldsymbol{x})^T \boldsymbol{\phi}(\boldsymbol{x}')$$

- This convex optimization problem is called the ***dual problem*** of the support vector machine method and can be solved using standard optimization software.

- Assuming that ***dual problem*** of SVM is solved and the solution $\boldsymbol{a} = [a_1\ a_2\ \ldots a_N]^T$ is obtained, how can we determine the solution $(\boldsymbol{w}, b)$ of the ***primal problem***?

  - $\boldsymbol{w} = \sum_{i=1}^{N} a_i y_i \boldsymbol{\phi}(\boldsymbol{x}_i)$

  - To find the expression for b, we need to revisit the concept of ***support vector***

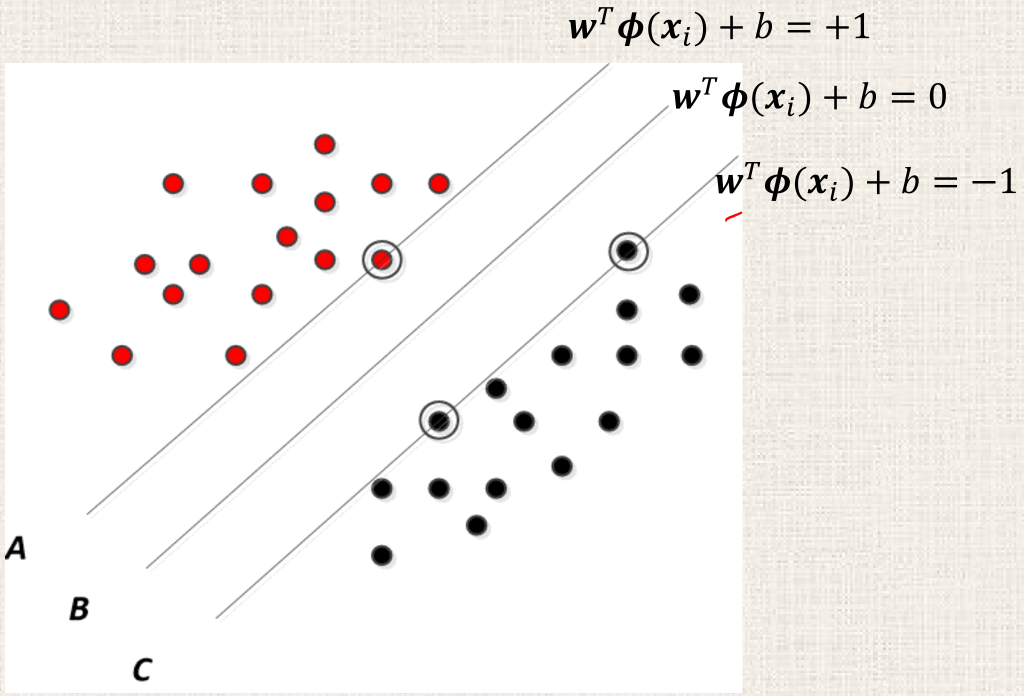
- The Karush-Kuhn-Tucker (***KKT***) conditions play a central role in both the theory and practice of constrained optimization. ***KKT conditions are satisfied at the solutions of any constrained optimization problem (convex or not) with any kind of constraints, provided certain conditions.***
- The following are the ***KKT*** conditions derived from the ***primal problem***:

$$a_i \geq 0, i = 1,2, \dots, N$$

$$y_i[\mathbf{w}^T \boldsymbol{\phi}(\mathbf{x_i}) + b] - 1 \geq 0$$

$$a_i\{y_i[\mathbf{w}^T \boldsymbol{\phi}(\mathbf{x_i}) + b] - 1\} = 0$$

- Thus, for any data point, either $a_i = 0$ or $y_i(\mathbf{w}^T \boldsymbol{\phi}(\mathbf{x_i}) + b) = 1$.
- Those data points satisfying $y_i(\mathbf{w}^T \boldsymbol{\phi}(\mathbf{x_i}) + b) = 1$ are called ***support vectors*** and they correspond to points on the hyperplanes $\mathbf{w}^T \boldsymbol{\phi}(\mathbf{x_i}) + b = +1 \; and \; \mathbf{w}^T \boldsymbol{\phi}(\mathbf{x_i}) + b = -1$.



- In order to classify new data point using trained model, we evaluate the sign of the prediction $y(\boldsymbol{x})$.
  Notice that $\boldsymbol{w} = \sum_{i=1}^{N} a_i y_i \boldsymbol{\phi}(\boldsymbol{x}_i)$, we have,

$$y(\boldsymbol{x}) = \boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x}) + b = \sum_{i=1}^{N} a_i y_i k(\boldsymbol{x}, \boldsymbol{x}_i) + b$$

- Having solved the quadratic optimization problem and find a value of $\boldsymbol{a}$, we can then determine the value of the intercept parameter b by noting that any ***support vector*** $\boldsymbol{x}_i$ satisfies

$$y_i y(\boldsymbol{x}_i) = 1 \Rightarrow y_i \left(\sum_{j=1}^{N} a_j y_j k(\boldsymbol{x}_i, \boldsymbol{x}_j) + b \right) = 1$$

  Where S is the set of indices of the ***support vectors***.

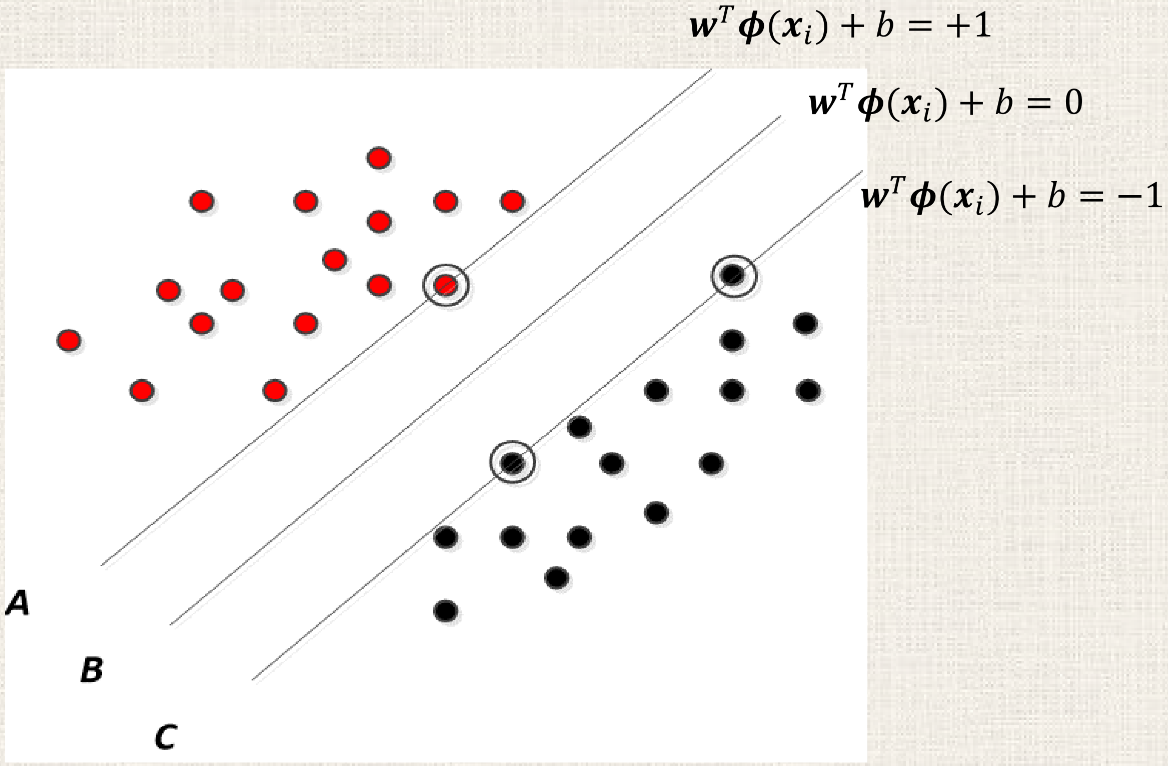
- Multiplying both sides of the above equality by y_i and notice that $y_i^2 = 1$, and then averaging over these equalities, we have,

$$b = \frac{1}{N_S} \sum_{i \in S} \left(y_i - \sum_{j=1}^{N} a_j y_j k(\boldsymbol{x}_i, \boldsymbol{x}_j) \right)$$

  Where N_S is the total number of support vectors.

Support Vector Machine Method (hard margin)

(Linearly separable binary data set)



Support Vector Machine Method (hard margin)

- The ***primal problem***:

$$\underset{\boldsymbol{w},b}{argmin}\left\{\frac{1}{2}\|\boldsymbol{w}\|^2\right\}$$

$$subject\ to \quad y_i(\boldsymbol{w}^T\boldsymbol{\phi}(\boldsymbol{x_i}) + b\ ) \geq 1,\ \ i = 1,2,\dots,N$$

- The Lagrangian function of the primal problem is:

$$L_P(\boldsymbol{w}, b, \boldsymbol{a}) = \frac{1}{2}\|\boldsymbol{w}\|^2 - \sum_{i=1}^{N} a_i\{y_i(\boldsymbol{w}^T\boldsymbol{\phi}(\boldsymbol{x_i}) + b\ ) - 1\}$$

- The ***dual problem***:

$$\underset{\boldsymbol{a}}{argmax}\left\{L_D(\boldsymbol{a}) = \sum_{i=1}^{N} a_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} a_i a_j y_i y_j \boldsymbol{\phi}(\boldsymbol{x_i})^T \boldsymbol{\phi}(\boldsymbol{x}_j)\right\}$$

$$subject\ to\ a_i \geq 0, all\ i\ \ and\ \sum_{i=1}^{N} a_i y_i = 0$$

The solution of the ***dual problem*** can be found using some optimization software:

$$\boldsymbol{a} = [a_1, a_2, \ldots, a_N]^T$$

Support Vector Machine Method (hard margin)

- With the solution $a_i, i = 1,2,\dots,N$, of the dual problem found, the ***decision function*** for a new input $\boldsymbol{x}$ is:

$$y(\boldsymbol{x}) = \boldsymbol{w}^T\boldsymbol{\phi}(\boldsymbol{x}) + b = \sum_{i=1}^{N} a_i y_i \boldsymbol{\phi}(\boldsymbol{x_i})^T\boldsymbol{\phi}(\boldsymbol{x}) + b = \sum_{i=1}^{N} a_i y_i k(\boldsymbol{x_i}, \boldsymbol{x}) + b$$

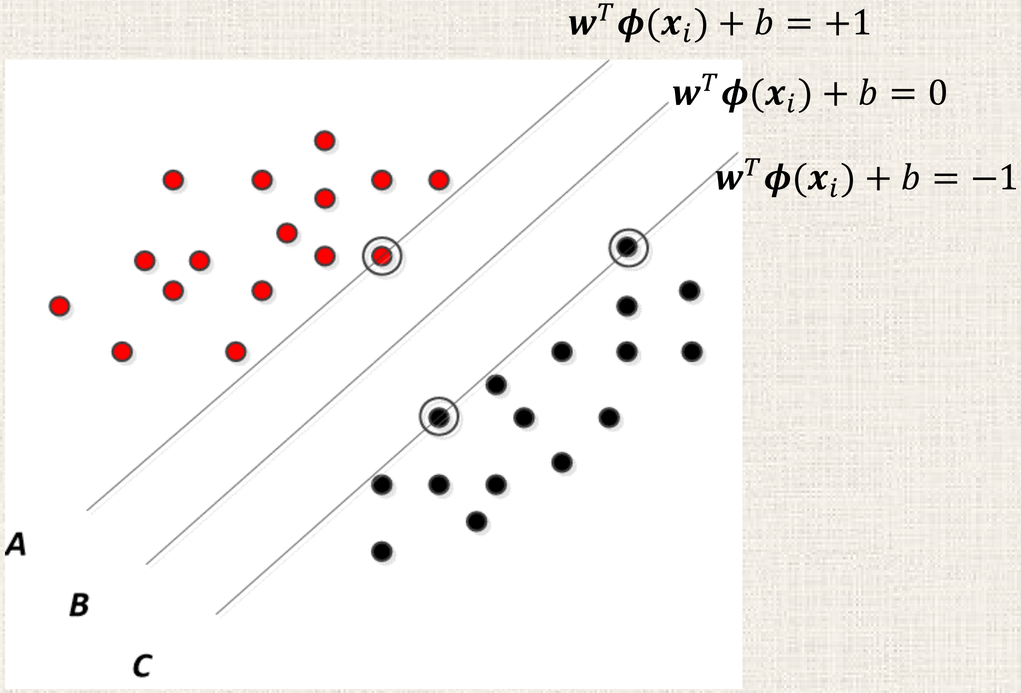
$$b = \frac{1}{N_S}\sum_{i\in S}\left(y_i - \sum_{j=1}^{N} a_j y_j k(\boldsymbol{x}_i, \boldsymbol{x}_j)\right)$$

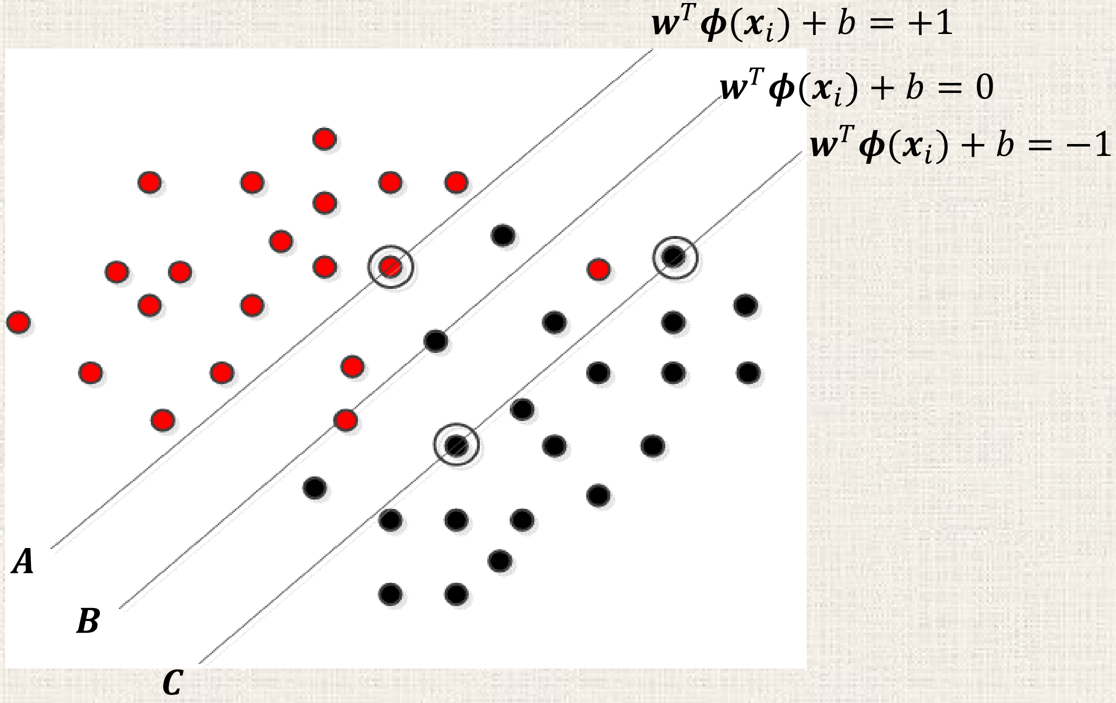
Where $k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \boldsymbol{\phi}(\boldsymbol{x_i})^T\boldsymbol{\phi}(\boldsymbol{x_i'})$ is the ***kernel function***. S is the set of all ***support vector*** points.

Support Vector Machine Method *(Soft margin)*

(Non-linearly-separable binary data sets)

- The hard margin SVM method will find no feasible solution on non-linearly separable data set!
- How can we extend the ideas of hard margin SVM to handle non-linearly separable data set?


$\boldsymbol{w}^T\boldsymbol{\phi}(\boldsymbol{x}_i) + b = +1$
$\boldsymbol{w}^T\boldsymbol{\phi}(\boldsymbol{x}_i) + b = 0$
$\boldsymbol{w}^T\boldsymbol{\phi}(\boldsymbol{x}_i) + b = -1$
A
B
C


$\boldsymbol{w}^T\boldsymbol{\phi}(\boldsymbol{x}_i) + b = +1$
$\boldsymbol{w}^T\boldsymbol{\phi}(\boldsymbol{x}_i) + b = 0$
$\boldsymbol{w}^T\boldsymbol{\phi}(\boldsymbol{x}_i) + b = -1$
A
B
C

- This can be done by introducing positive ***slack variables*** $\xi_i \geq 0, i = 1, \dots, N$ in the constraints on the data set:

$$\boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x}_i) + b \geq +1 - \xi_i, \qquad for\ y_i = +1$$

$$\boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x}_i) + b \leq -1 + \xi_i, \qquad for\ y_i = -1$$

- Or equivalently,

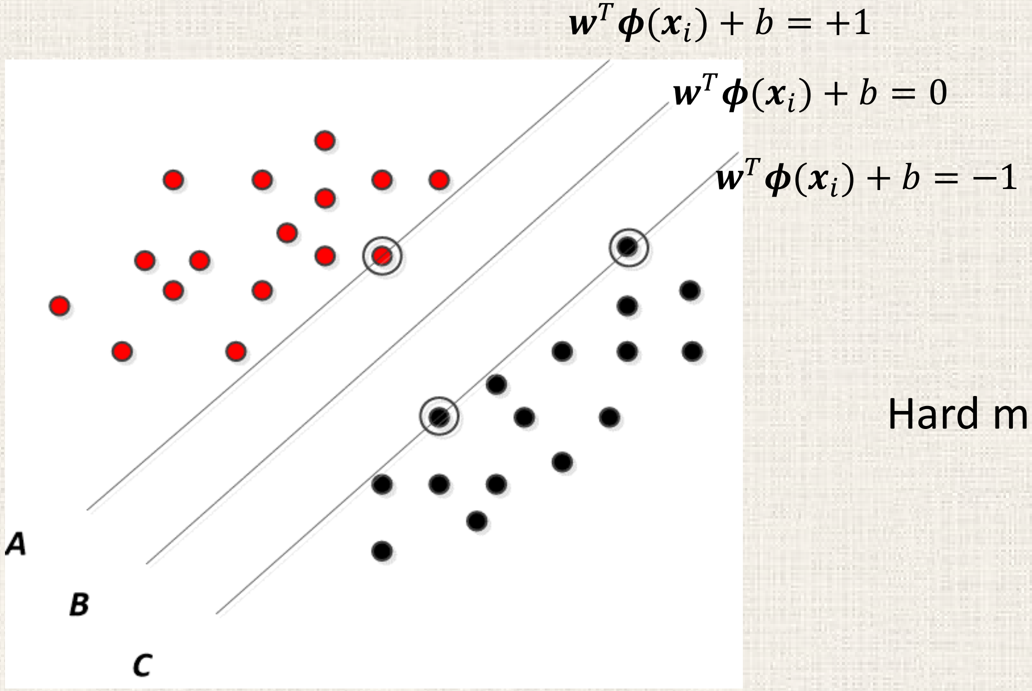
$$y_i\{\boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x}_i) + b\} \geq 1 - \xi_i, \forall i$$

  - $\xi_i = 0$: the data point $\boldsymbol{x}_i$ is out of the margin and is properly classified;
  - $0 < \xi_i \leq 1$: the data point $\boldsymbol{x}_i$ is inside the margin but on the correct side of the decision boundary;
  - $\xi_i > 1$: the data point $\boldsymbol{x}_i$ is on the wrong side of the decision boundary.

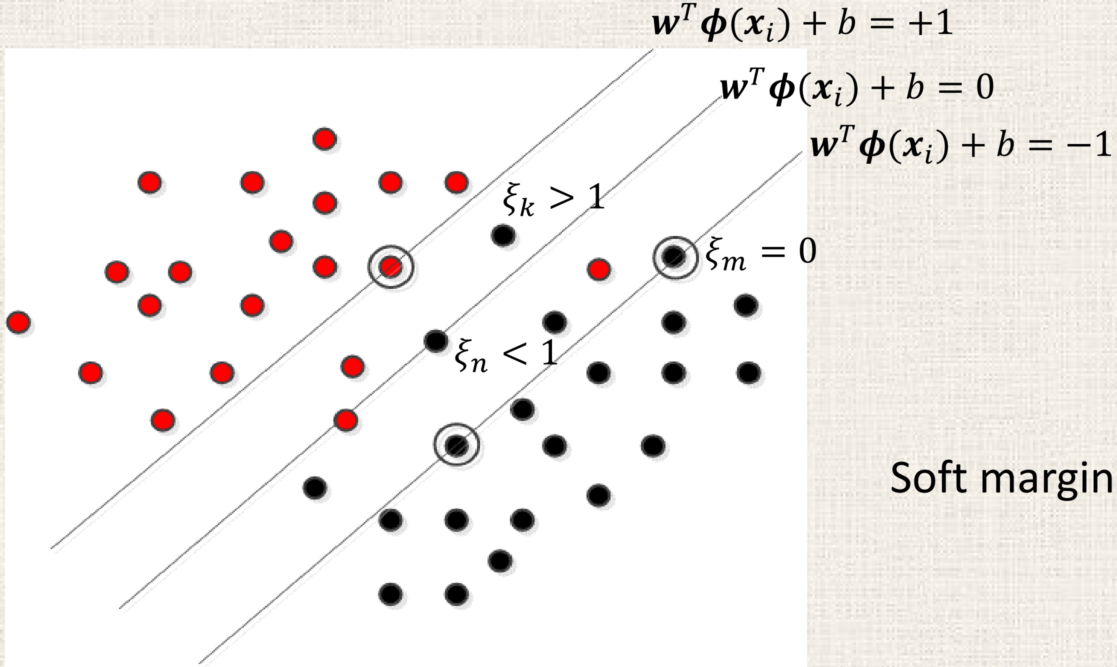
Soft margin:

$$y_i\{\boldsymbol{w}^T\boldsymbol{\phi}(\boldsymbol{x}_i)+b\} \geq +1-\xi_i, \forall i$$

- $\xi_i = 0$,: the data point $\boldsymbol{x}_i$ is properly classified;
- $0 < \xi_i \leq 1$: the data point $\boldsymbol{x}_i$ is inside the margin but on the correct side of the decision boundary;
- $\xi_i > 1$: the data point $\boldsymbol{x}_i$ is on the wrong side of the decision boundary (miss-classified).
- $\sum_{i=1}^{N} \xi_i$ is the ***upper bound*** of the number of misclassified instances



Hard margin



Soft margin

Soft margin method:

- Our goal is now to maximize the margin while softly penalizing points that lie on the wrong side of the margin boundary. We therefore minimize:

$$\frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{N}\xi_i$$

Subject to the constraints: $y_i\{\boldsymbol{w}^T\boldsymbol{\phi}(\boldsymbol{x}_i) + b\} \geq +1 - \xi_i, \forall i$ and $\xi_i \geq 0$.

- Where the parameter C control the tradeoff between the slack variable penalty and the margin.
- When $C \rightarrow \infty$, this becomes the hard margin method.
- The corresponding ***Lagrangian function*** is given by:

$$L_P(\boldsymbol{w}, b, \boldsymbol{a}) = \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{N}\xi_i - \sum_{i=1}^{N}a_i\{y_i y(\boldsymbol{x}_i) - 1 + \xi_i\} - \sum_{i=1}^{N}\mu_i\xi_i$$

Where, $a_i \geq 0\ and\ \mu_i \geq 0$ are ***Lagrange multipliers***.

- This is the ***primal problem*** of soft margin method.

Support Vector Machine Method

Soft margin method:

- The ***KKT conditions*** of the primal problem are given by:

$$a_i \geq 0$$
$$y_i y(\boldsymbol{x}_i) - 1 + \xi_i \geq 0$$
$$a_i\{y_i y(\boldsymbol{x}_i) - 1 + \xi_i\} = 0$$
$$\mu_i \geq 0$$
$$\xi_i \geq 0$$
$$\mu_i \xi_i = 0$$

Where $i = 1, \dots, N$

- $a_i > 0$, for points of $\boldsymbol{x}_i$ that satisfy $y_i y(\boldsymbol{x}_i) - 1 + \xi_i = 0 \Rightarrow y_i y(\boldsymbol{x}_i) = 1 - \xi_i$. These points of $\boldsymbol{x}_i$ are ***support vectors***. (includes all the data points on the margin boundary as well as those on the wrong side of the margin boundary)

Soft margin method:

- The dual problem can be derived as:

$$\frac{\partial L_p}{\partial \boldsymbol{w}} = 0 \Rightarrow \boldsymbol{w} = \sum_{i=1}^{N} a_i y_i \boldsymbol{\phi}(\boldsymbol{x}_i)\,;\ \frac{\partial L_p}{\partial b} = 0 \Rightarrow \sum_{i=1}^{N} a_i y_i = 0;\ \frac{\partial L_p}{\partial \xi_i} = 0 \Rightarrow a_i = C - \mu_i$$

- Using these results to eliminate $\boldsymbol{w}, b\ and\ \xi_i$ from $L_P(\boldsymbol{w}, b, \boldsymbol{a})$, we obtain the ***dual Lagrangian*** as:

$$L_D(\boldsymbol{a}) = \sum_{i=1}^{N} a_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} a_i a_j y_i y_j k(\boldsymbol{x}_i, \boldsymbol{x}_j)$$

Subject to the following constraints:

$$0 \le a_i \le C$$

$$\sum_{i=1}^{N} a_i y_i = 0$$

Where $i = 1, \dots, N$

- This is the ***dual problem*** of the soft margin method. This is a quadratic programming problem.

- Prediction on new instance $\boldsymbol{x}$ is again made by the sign of $y(\boldsymbol{x})$:

$$y(\boldsymbol{x}) = \sum_{i=1}^{N} a_i y_i k(\boldsymbol{x}, \boldsymbol{x}_i) + b$$

- To determine the value of b, we note that those support vectors $\boldsymbol{x}_i$ have $\xi_i = 0$, so that $y_i y(\boldsymbol{x}_i) = 1$, and hence will satisfy

$$y_i y(\boldsymbol{x}_i) = 1 \Rightarrow y_i \left(\sum_{j=1}^{N} a_j y_j k(\boldsymbol{x}_i, \boldsymbol{x}_j) + b \right) = 1$$

  Where S is the set of indices of the ***support vectors***.

- Multiplying both sides of the above equality by y_i and notice that $y_i^2 = 1$, and then averaging over these equalities, we have ,

$$b = \frac{1}{N_S} \sum_{i \in S} \left(y_i - \sum_{j=1}^{N} a_j y_j k(\boldsymbol{x}_i, \boldsymbol{x}_j) \right)$$

  Where N_S is the total number of support vectors.

Kernels and Kernel Trick

- Recall the ***dual optimization problem*** of SVM:

$$\max_{\boldsymbol{a}} \left\{ L_D(\boldsymbol{a}) = \sum_{i=1}^{N} a_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} a_i a_j y_i y_j k(\boldsymbol{x}_i, \boldsymbol{x}_j) \right\}$$

- With the solution $a_i, i = 1,2, \dots, N$, of the dual problem found, the ***dual decision function***:

$$y(\boldsymbol{x}) = \boldsymbol{w}^T \boldsymbol{x} + b = \sum_{i=1}^{N} a_i y_i k(\boldsymbol{x}, \boldsymbol{x}_i) + b$$

$$b = \frac{1}{N_S} \sum_{i \in S} \left(y_i - \sum_{j \in S} a_j y_j k(\boldsymbol{x}_i, \boldsymbol{x}_j) \right)$$

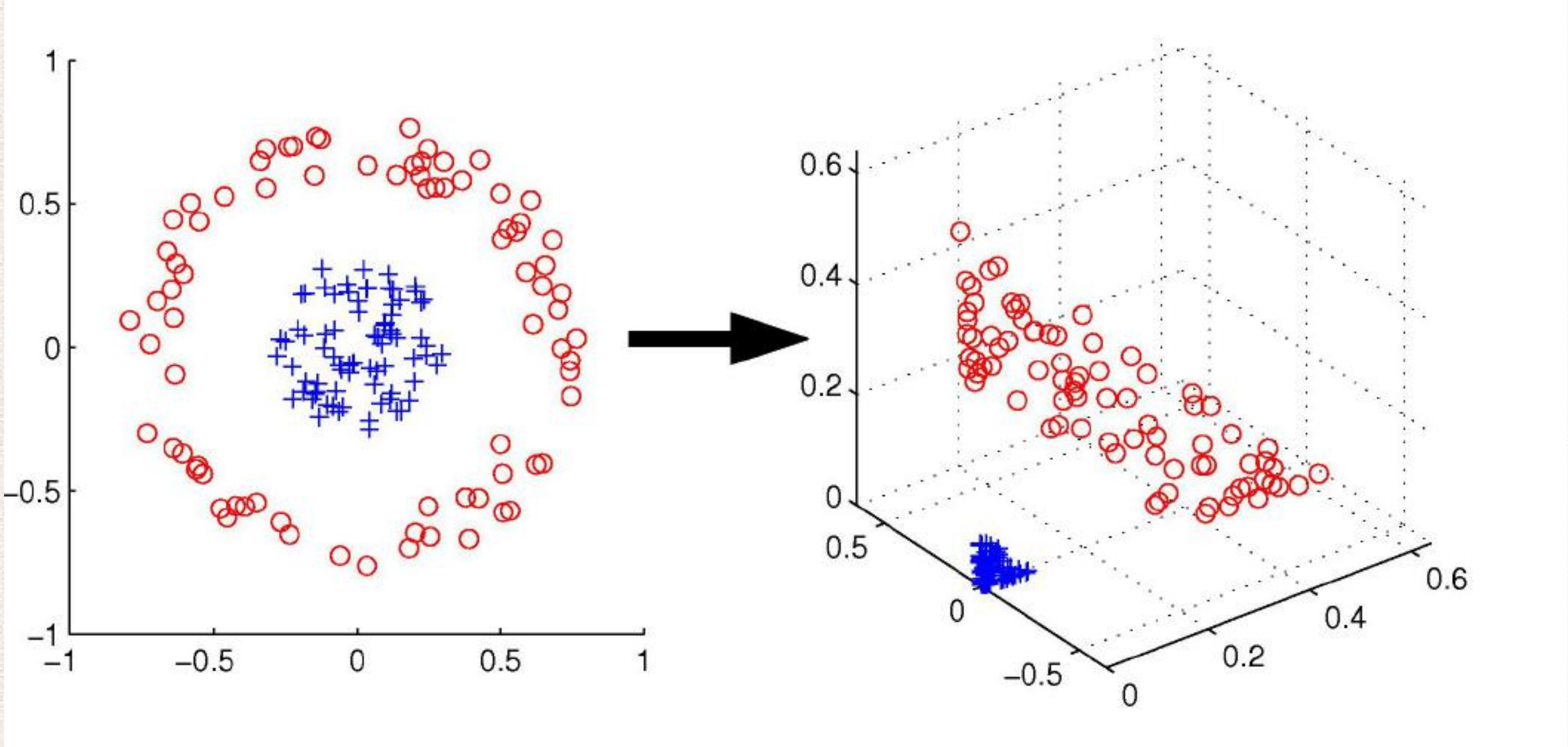
Where $k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \boldsymbol{\phi}(\boldsymbol{x}_i)^T \boldsymbol{\phi}(\boldsymbol{x}_j)$ is the ***kernel function***.

- Originally we define the ***kernel function*** as:

$$k(\boldsymbol{x}, \boldsymbol{x}') = \boldsymbol{\phi}(\boldsymbol{x})^T \boldsymbol{\phi}(\boldsymbol{x}')$$

  Which involve transforming the input $\boldsymbol{x}$ from original input space to higher dimensional feature space and then carry out inner product operation.

- By using the kernel function with some properties, we can make the operations in ***input space*** and leave the mapping completely ***implicit***.



1
0.5
0
−0.5
−1
−1
−0.5
0
0.5
1
0.6
0.4
0.2
0
0.5
0
−0.5
0
0.2
0.4
0.6

- Clearly, the data in the left is not linearly separable.
- If we map it to a 3-dimensional feature space using

$$\phi: \mathcal{R}^2 \rightarrow \mathcal{R}^3$$

$$(x_1, x_2) \rightarrow (z_1, z_2, z_3) = (x_1^2, \sqrt{2}x_1x_2, x_2^2)$$

- Then, in the new feature space, the data set is linearly separable by a hyperplane $\boldsymbol{w}^T\boldsymbol{Z} + b = 0$,

$$w_1z_1 + w_2z_2 + w_3z_3 + b = 0$$

i.e.,

$$w_1x_1^2 + w_2\sqrt{2}x_1x_2 + w_3x_2^2 = -b$$

Which is an ellipse in the original input space.

- Consider two vectors $\boldsymbol{x} = [x_1 \quad x_2]^T$ *and* $\boldsymbol{x}' = [x'_1 \quad x'_2]^T$ in original feature space. We have,

$$\boldsymbol{\phi}(\boldsymbol{x})^T\boldsymbol{\phi}(\boldsymbol{x}') = \begin{bmatrix} x_1^2 & \sqrt{2}x_1x_2 & x_2^2 \end{bmatrix} \begin{bmatrix} (x_1')^2 \\ \sqrt{2}x_1'x_2' \\ (x_2')^2 \end{bmatrix} = (x_1x_1')^2 + 2(x_1x_1')(x_2x_2') + (x_2x_2')^2$$

- Now, let's define $k(\boldsymbol{x}, \boldsymbol{x}') = (\boldsymbol{x}^T\boldsymbol{x}')^2$, then,

$$k(\boldsymbol{x}, \boldsymbol{x}') = (\boldsymbol{x}^T\boldsymbol{x}')^2 = \left([x_1 \quad x_2] \begin{bmatrix} x_1' \\ x_2' \end{bmatrix} \right)^2 = (x_1x_1' + x_2x_2')^2$$

$$= (x_1x_1')^2 + 2(x_1x_1')(x_2x_2') + (x_2x_2')^2$$

- This means, with the kernel function $k(\boldsymbol{x}, \boldsymbol{x}') = (\boldsymbol{x}^T\boldsymbol{x}')^2$, instead of carry out the mapping on the data set and then calculate $\boldsymbol{\phi}(\boldsymbol{x})^T\boldsymbol{\phi}(\boldsymbol{x}')$ in ***transformed high-dimensional feature space***, we only need to calculate $k(\boldsymbol{x}, \boldsymbol{x}') = (\boldsymbol{x}^T\boldsymbol{x}')^2$ in the ***input space***!!!

Important properties of *kernel functions*:

A kernel is a two arguments real-valued function over $\mathcal{R}^d \times \mathcal{R}^d \to \mathcal{R}$:

$$k(\boldsymbol{x}, \boldsymbol{y}) = \boldsymbol{\phi}(\boldsymbol{x})^T \boldsymbol{\phi}(\boldsymbol{y})$$

- A ***kernel*** must be ***symmetric***, that is,

$$k(\boldsymbol{x}, \boldsymbol{y}) = k(\boldsymbol{y}, \boldsymbol{x})$$

- The *kernel matrix* is ***positive semi-definite***:

  A kernel matrix K is the matrix results from applying kernel k to all pairs of data points in data set $\{\boldsymbol{x}_i\}_{i=1}^N$:

$$K = \begin{bmatrix} k(\boldsymbol{x}_1, \boldsymbol{x}_1) & k(\boldsymbol{x}_1, \boldsymbol{x}_2) & \dots & k(\boldsymbol{x}_1, \boldsymbol{x}_N) \\ k(\boldsymbol{x}_2, \boldsymbol{x}_1) & k(\boldsymbol{x}_2, \boldsymbol{x}_2) & \cdots & k(\boldsymbol{x}_2, \boldsymbol{x}_N) \\ \vdots & \vdots & \ddots & \vdots \\ k(\boldsymbol{x}_N, \boldsymbol{x}_1) & k(\boldsymbol{x}_N, \boldsymbol{x}_2) & \dots & k(\boldsymbol{x}_N, \boldsymbol{x}_N) \end{bmatrix}$$

Kernel construction rules:

- Clearly, the linear kernel defined as $k(\boldsymbol{x}, \boldsymbol{y}) = \boldsymbol{x}^T \boldsymbol{y}$ is a valid kernel.
- For any positive semi-definite matrix $\boldsymbol{B}_{d \times d}$, $k(\boldsymbol{x}, \boldsymbol{y}) = \boldsymbol{x}^T \boldsymbol{B} \boldsymbol{y}$ is a valid kernel.
- Suppose k_1 , k_2 and k_3 are valid kernels, then, the following are all valid kernels
  - $k(\boldsymbol{x}, \boldsymbol{y}) = k_1(\boldsymbol{x}, \boldsymbol{y}) + k_2(\boldsymbol{x}, \boldsymbol{y})$
  - $k(\boldsymbol{x}, \boldsymbol{y}) = \alpha k_1(\boldsymbol{x}, \boldsymbol{y})$
  - $k(\boldsymbol{x}, \boldsymbol{y}) = k_1(\boldsymbol{x}, \boldsymbol{y}) k_2(\boldsymbol{x}, \boldsymbol{y})$
  - $k(\boldsymbol{x}, \boldsymbol{y}) = k_3(\boldsymbol{\phi}(\boldsymbol{x}), \boldsymbol{\phi}(\boldsymbol{y}))$
- Suppose k_1 is a kernel and p is a polynomial with ***non-negative*** coefficients, then the following are kernels
  - $k(\boldsymbol{x}, \boldsymbol{y}) = p(k_1(\boldsymbol{x}, \boldsymbol{y}))$
  - $k(\boldsymbol{x}, \boldsymbol{y}) = e^{k_1(\boldsymbol{x}, \boldsymbol{y})}$

The following are some of the commonly used kernels in Machine Learning:

- ***Linear Kernel:***

$$k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \boldsymbol{x}_i^T \boldsymbol{x}_j$$

- ***Polynomial Kernel:***

$$k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \left(\boldsymbol{x}_i^T \boldsymbol{x}_j + \theta\right)^d$$

- ***Gaussian Radial Basis Function (RBF): (universal kernel)***

$$k(\boldsymbol{x}_i, \boldsymbol{x}_j) = e^{-\gamma \left\| \boldsymbol{x}_i - \boldsymbol{x}_j \right\|^2}$$

- ***Sigmoid Kernel:***

$$k(\boldsymbol{x}_i, \boldsymbol{x}_j) = tanh(\eta \boldsymbol{x}_i^T \boldsymbol{x}_j + \theta)$$

Advantages of kernel based methods:

- Kernel methods learn nonlinear functions with the algorithms for learning linear function
- Kernel allows for learning in high-dimensional feature spaces without explicit mapping into feature space
- Kernels make learning in high-dimensional feature space computationally feasible
- Kernels provide an abstraction that separates data representations and learning

Power of SVM = linear model with max margin + nonlinear feature mapping (kernel functions)