Intelligent Systems (IS Fall 2013)

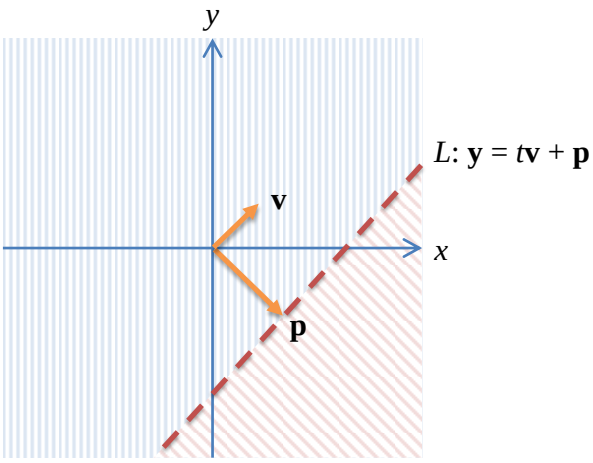
Assignment 2: Support Vector Machines

Student: Fang–Lin He

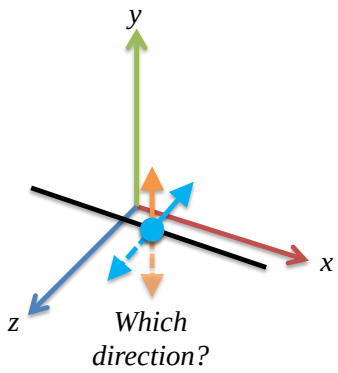
We consider an input vector space $\mathbb{R}^n$ and a classifier $sign(f(x))$ assigning a label −1 or +1 to $\mathbf{x}$ based on a linear function $f(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x} + \mathbf{b}$. Let $(\mathbf{x}^1, \mathbf{d}^1), \ldots, (\mathbf{x}^N, \mathbf{d}^N) \in \mathbb{R}^n \times \{-1, +1\}$ be the dataset. For some set X let $k: X \times X \to \mathbb{R}$ be a kernel function. Let $\phi: X \to V$ be the corresponding feature map into the feature space V. The operator $\cdot$ denotes the inner product.

1. (5 points) Is a hyperplane a line? If yes, why don't we call it line, if no, what else is it?

   >> By the geometric meaning, a hyperplane is a subspace that can separate the vector space into two parts. By this definition, my answer is: A line *may* be a hyperplane; a hyperplane is *not necessarily* a line. It depends on the dimension of the vector space. For a 2–dimensional vector space, a line is a hyperplane since it separates the space into two parts, like the figure shown below:

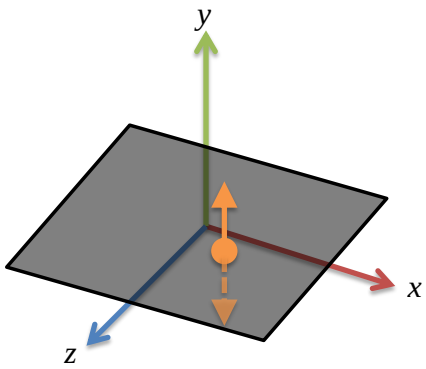


   However, in 3–dimensional vector space, a line is not a hyperplane since it cannot separate the space into two:



   Instead, a plane is a hyperplane in such case since it can separate the space into two,

like the below figure shows:

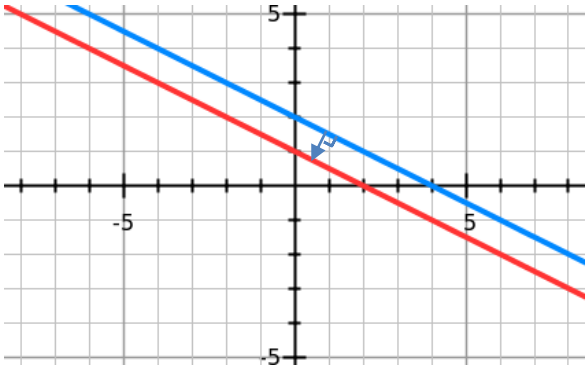


2. (5 points) In $\mathbb{R}^n$, how many dimensions does a subspace need to have in order to split the whole space into two parts? Check your answer at least for $\mathbb{R}^2$ and $\mathbb{R}^3$.
   >> My answer is $(n-1)$–dimensions. As mentioned in answer to the first question, in $\mathbb{R}^2$, a line can split the whole $\mathbb{R}^2$ space into two parts, and by the mathematical definition, a line is one–dimension and can be a subspace in higher dimensional space. Similarly, a plane which is two–dimension and can split the $\mathbb{R}^3$ vector space into two parts.

3. (20 points) Let $\mathbb{H}_0 = x \in \mathbb{R}^n | f(x) = 0$ be the separating hyperplane corresponding to *f*. How does the hyperplane change for
   (a) $g = 7 \cdot f$, that is, $g(\mathbf{x}) = (7\mathbf{w}) \cdot \mathbf{x} + (7b)$?
   (b) $h = f + 2$, that is, $g(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x} + (b + 2)$?
   >> I couldn't see the answer at a glance, so I let $f(\mathbf{x}) = x + 2y - 4$, that is, $\mathbf{x} = (x, y)$, $\mathbf{w} = (1, 2)$, $b = -4$ in the formula. The hyperplane $\mathcal{H}_0 = x \in \mathbb{R}^n | f(x) = 0$ is then obtained by $f(\mathbf{x}) = 0 = x + 2y - 4$.
   For (a), $g = 7 \cdot f$, I obtain $g(\mathbf{x}) = 0 = 7x + 14y - 28$. Rearranging $f(\mathbf{x})$ and $g(\mathbf{x})$, I find that these two functions indicate the same line:
   $f(\mathbf{x}) = 0 \rightarrow 0 = x + 2y - 4 \rightarrow y = (4 - x) / 2$
   $g(\mathbf{x}) = 0 \rightarrow 0 = 7x + 14y - 28 \rightarrow y = (28 - 7x) / 14 = (4 - x) / 2$
   In this case, let $x = t$, $y = (4 - t) / 2$, the hyperplane $\mathcal{H}_0$ for both $f(\mathbf{x})$ and $g(\mathbf{x})$ is the set $\mathbf{x} = (t, (4 - t) / 2) \mid -\infty < t < \infty$.
   Therefore, I infer that the hyperplane multiplied by a constant doesn't change the hyperplane itself.

   For (b), $h = f + 2$, that is, $h(\mathbf{x}) = x + 2y - 2$. Similarly, rearranging $f(\mathbf{x})$ and $h(\mathbf{x})$ obtaining:
   $f(\mathbf{x}) = 0 \rightarrow 0 = x + 2y - 4 \rightarrow y = (4 - x) / 2$
   $h(\mathbf{x}) = 0 \rightarrow 0 = x + 2y - 2 \rightarrow y = (2 - x) / 2$
   By plotting these two functions ($f(\mathbf{x}) = 0 \rightarrow$blue line, $h(\mathbf{x}) = 0 \rightarrow$ red line), I found that

adding a constant to the function $f(\mathbf{x})$ moves the line along the direction of the normal vector:

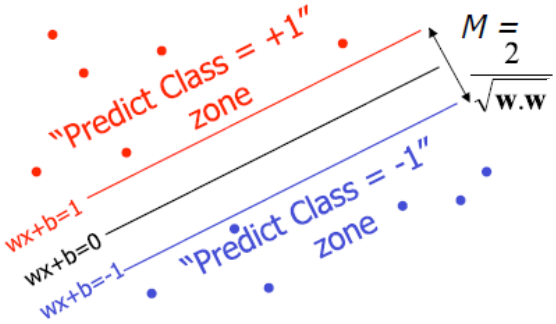


In this case, let $x = t$, $y = (2 - t) / 2$, the hyperplane $\mathcal{H}_0$ for $h(\mathbf{x})$ is the set $\mathbf{x} = (t, (2 - t) / 2) \mid -\infty < t < \infty$.

Therefore, I infer that the hyperplane added by a constant doesn't change its direction (since adding a constant doesn't affect the normal vector $\mathbf{w}$) but translates its position.

4. (10 points) Explain the objective functions of hard–margin and soft–margin support vector machine training as well as the constraints of the corresponding optimization problems. If you like, refer to a sketch.

>> The plot of hard–margin SVM training:



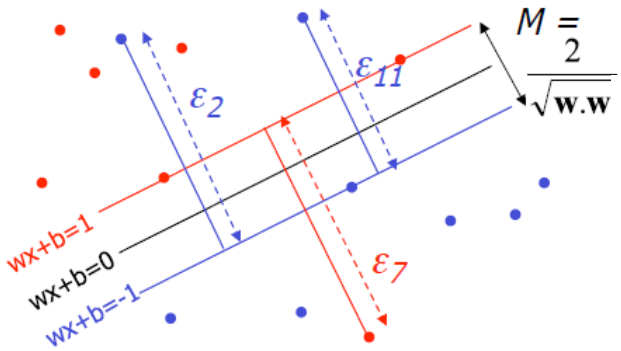
The objective function of hard–margin SVM training: $\mathbf{w} \cdot \mathbf{w}$. Constraints:

$\mathbf{w} \cdot \mathbf{x}_k + b >= 1$ if $y_k = 1$

$\mathbf{w} \cdot \mathbf{x}_k + b <= -1$ if $y_k = -1$

Explanation: The hard–margin SVM training aims to find the best separation line $\mathbf{w} \cdot \mathbf{x} + b = 0$ which separates two classes *perfectly* (all data are classified correctly) yet has the widest space (margin) between *support vectors*. So, the objective function is to minimize $\mathbf{w} \cdot \mathbf{w}$ since we want to maximize the margin, and the margin M is obtained by $\frac{2}{\sqrt{\mathbf{w} \cdot \mathbf{w}}}$. So minimizing $\mathbf{w} \cdot \mathbf{w}$ maximizes the margin. Since SVM aims to classify two classes correctly, so its constraints are $\mathbf{w} \cdot \mathbf{x}_k + b >= 1$ if $y_k = 1$ and $\mathbf{w} \cdot \mathbf{x}_k + b <= -1$ if $y_k = -1$ because it aims to separate positive values where $y_k = 1$ beyond $\mathbf{w} \cdot \mathbf{x}_k + b = 1$ and negative values

$y_k = -1$ beyond $\mathbf{w} \cdot \mathbf{x}_k + b = -1$. Therefore, there is space between data $\mathbf{w} \cdot \mathbf{x}_k + b = -1$ and $\mathbf{w} \cdot \mathbf{x}_k + b = +1$ where we want to maximize the distance between them.



The objective function of soft–margin SVM training: $\frac{1}{2}\mathbf{w}\cdot\mathbf{w} + C\sum_{k=1}^{R}\varepsilon_k$. Constraints:

$\mathbf{w} \cdot \mathbf{x}_k + b >= 1-\varepsilon_k$ if $y_k = 1$
$\mathbf{w} \cdot \mathbf{x}_k + b <= -1+\varepsilon_k$ if $y_k = -1$
$\varepsilon_k >= 0$ for all k

The difference between hard–margin and soft–margin SVM training is that the soft–margin one allows misclassification. That is, in the plot, the two blue dots beyond $\mathbf{w} \cdot \mathbf{x}_k + b = 1$ and one red dots beyond $\mathbf{w} \cdot \mathbf{x}_k + b = -1$. So now the objective function is added by a regularization term $C\sum_{k=1}^{R}\varepsilon_k$ where ε_k is the distance (absolute value, which is as a constraint of the objective function) from the data and separation line ($\mathbf{w} \cdot \mathbf{x}_k + b = 1$ or $\mathbf{w} \cdot \mathbf{x}_k + b = -1$), and C is the parameter controlling the trade-off between weight and tolerance to misclassification. For the constraints, since it allows misclassification, that is, for the positive class data may lie beyond the negative plane.

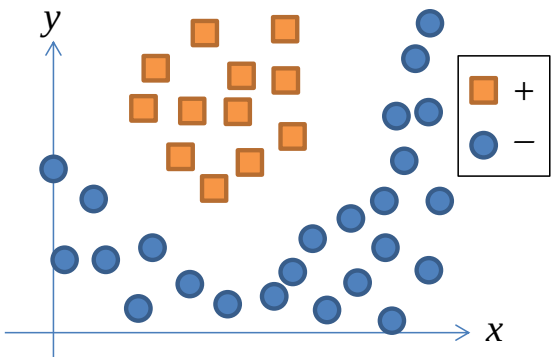
$\mathbf{w} \cdot \mathbf{x}_k + b >= 1-\varepsilon_k$ if $y_k = 1$

Thus the constraints become $\mathbf{w} \cdot \mathbf{x}_k + b <= -1+\varepsilon_k$ if $y_k = -1$. For instance, if $\mathbf{y}_5 = 1$ but the data lies beyond the negative plane $\mathbf{w} \cdot \mathbf{x}_5 + b = -1$ which means misclassification. In this case, suppose $\varepsilon_5 = 2.3$, it still holds the constraints $\mathbf{w} \cdot \mathbf{x}_5 + b = 1 - 2.3 = -1.3$. So with the objective function $\frac{1}{2}\mathbf{w}\cdot\mathbf{w} + C\sum_{k=1}^{R}\varepsilon_k$ and constraints

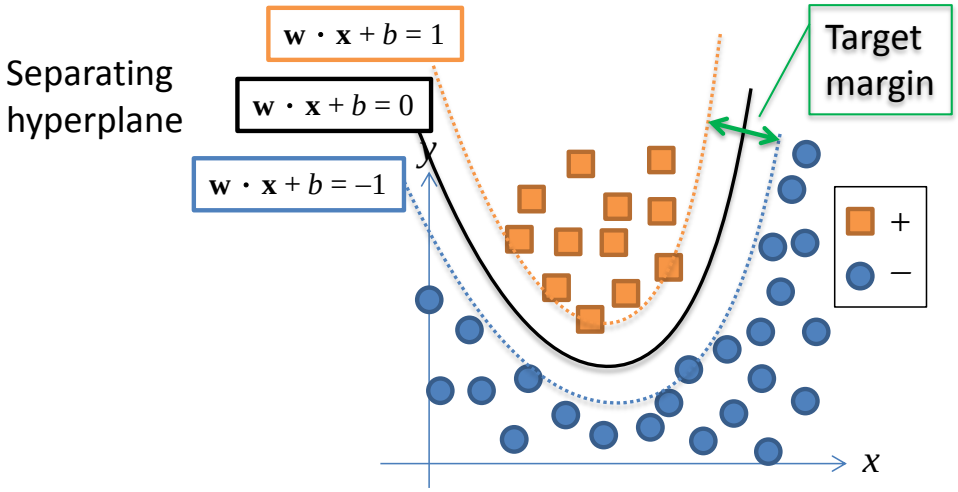
$\mathbf{w} \cdot \mathbf{x}_k + b >= 1-\varepsilon_k$ if $y_k = 1$
$\mathbf{w} \cdot \mathbf{x}_k + b <= -1+\varepsilon_k$ if $y_k = -1$ and $\varepsilon_k >= 0$ for all k, the soft-margin SVM can allow misclassification yet find the best margin between two classes.

5. (10 points) Draw a two-dimensional example of linearly non-separable data. Indicate positive and negative labels for the points. Then draw a (not necessary optimal) separating hyperplane and a target margin into the example. Indicate on which side of the hyperplane the classifier decides for which class.

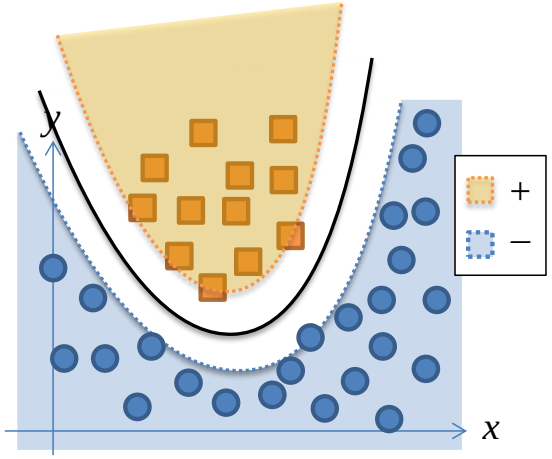
>> My data:



My separating hyperplane and a target margin:



The side of hyper plane the classifier decides for positive class are shown in orange background; for negative class are shown in blue background.



6. (10 points) Explain the relation between kernel function and feature map. What does the kernel function compute? Why should we use a kernel in the first place?

   >> The kernel function and the feature mapping are both used to create non-linear classifier. After applying them, we can apply linear classification on the mapped vector space. The kernel function *implicitly* computes a higher-dimensional feature space as feature map explicitly does but only requires *dot product* to apply the transformation. That is why so called *kernel trick*. The benefit of kernel function is its low computational costs. By comparing the complexity as page 53 in the course slides, the complexity of explicitly computing quatric polynomial (feature map) is 39200 times of the complexity of implicitly computing it by applying dot product (kernel function). So naturally, we would choose the lower complexity method.

7. (40 points) Consider the cubic kernel

$$k : \mathbb{R}^2 \times \mathbb{R}^2 \to \mathbb{R}; \quad k(x, y) = (x \bullet y)^3 = (x_1 y_1 + x_2 y_2)^3 \qquad (1)$$

   and the feature map

$$\phi : \mathbb{R}^2 \to \mathbb{R}^4; \quad \phi(x_1, x_2) = (x_1^3, \sqrt{3}x_1^2 x_2, \sqrt{3}x_1 x_2^2, x_2^3). \qquad (2)$$

   In the feature space $\mathbb{R}^4$ we use the standard inner product:

$$v \bullet w = v_1 w_1 + v_2 w_2 + v_3 w_3 + v_4 w_4$$

   Show that ϕ is a feature map for the cubic kernel, i.e., show that the relation

$$k(x, y) = \phi(x) \bullet \phi(y) \qquad (3)$$

   holds

   >> By extending $k(\mathbf{x}, \mathbf{y})$, where $\mathbf{x} = (x_1, x_2)$, $\mathbf{y} = (y_1, y_2)$, we obtain:

$$\begin{aligned} k(\mathbf{x}, \mathbf{y}) &= (x_1 y_1 + x_2 y_2)^3 \\ &= (x_1 y_1)^3 + 3(x_1 y_1)^2 (x_2 y_2) + 3(x_1 y_1)(x_2 y_2)^2 + (x_2 y_2)^3 \end{aligned}$$

   By extending $\phi(\mathbf{x}) \cdot \phi(\mathbf{y})$, we obtain:

$$\begin{aligned} \phi(\mathbf{x}) \cdot \phi(\mathbf{y}) &= \left(x_1^3, \sqrt{3}x_1^2 x_2, \sqrt{3}x_1 x_2^2, x_2^3\right) \cdot \left(y_1^3, \sqrt{3}y_1^2 y_2, \sqrt{3}y_1 y_2^2, y_2^3\right) \\ &= x_1^3 y_1^3 + \sqrt{3}x_1^2 x_2 \sqrt{3}y_1^2 y_2 + \sqrt{3}x_1 x_2^2 \sqrt{3}y_1 y_2^2 + x_2^3 y_2^3 \\ &= (x_1 y_1)^3 + 3x_1^2 y_1^2 x_2 y_2 + 3x_1 y_1 x_2^2 y_2^2 + (x_2 y_2)^3 \\ &= (x_1 y_1)^3 + 3(x_1 y_1)^2 (x_2 y_2) + 3(x_1 y_1)(x_2 y_2)^2 + (x_2 y_2)^3 \end{aligned}$$

   From above two final equations, we can clear see that $k(\mathbf{x}, \mathbf{y}) = \phi(\mathbf{x}) \cdot \phi(\mathbf{y})$. In this case, the kernel function is a homogeneous polynomial kernel.