Introductory Notes on Support Vector Machines

Anca Ralescu
Machine Learning and Computational Intelligence Laboratory
Department of Computer Science
University of Cincinnati
Cincinnati, Ohio 45221, USA
ancaralescu@gmail.com

- Introduce the notion of **margin** and **support vectors** working out directly from the formulation of the classification problem.
- Objective: build classifiers for data, that is **Algorithms for partitioning the data**.
- Two main approaches:
  - **Statistical**(parametric/classical): Data to be classified comes from a population whose underlying distribution (its parameters) are known: this means that we know the **probability density/mass function** which can be used for classification of a new data point. However, in real world problems we do not know the distribution underlying the data. In fact in theoretical studies of probability/statistics this is a much researched problem. We can say that **it** is a (machine) learning problem;
  - **Non-parametric**: Given known classifications (training data) $\Longrightarrow$ derive a rule (**decision functions**) for classifying unknown/new data points: $g(x)$ In a two class problem we have:

$$\begin{cases} g(x) = 0 & \text{boundary} \\ g(x) > 0 & \text{class 1} \\ g(x) < 0 & \text{class 2} \end{cases} \tag{1}$$

- We will be concerned with the Non-parametric case.

1 Linear Classifiers

To begin with, we will be concerned only with linear classifiers. This means that we start by assuming that the data set **is linearly separable**.

Then, in the case when this is not true, the data set is NOT linearly separable, we will use a device which makes this data linearly separable. This device maps data implicitly into a higher dimension where it becomes linearly separable.

In the case of **linear classifiers** the decision function $g(\mathbf{x})$ is linear in $\mathbf{x}$, that is

$$g(\mathbf{x}) = sign(\mathbf{w} \cdot \mathbf{x} + b) \tag{2}$$

where *sign* (*signum*)is defined as

$$sign(a) = \begin{cases} -1 & \text{if } a < 0 \\ 0 & \text{if } a = 0 \\ +1 & \text{if } a > 0 \end{cases}$$

The job of the classifier is to find the weight vector $\mathbf{w}$ and the constant term b (bias).

2 Example

Consider the data from figure 1

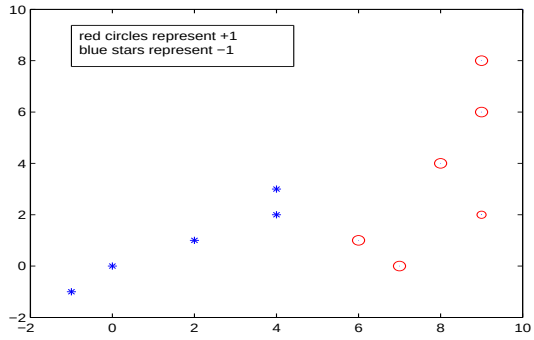


Figure 1: A two class example set

How do we separate the two classes represented there? Figure 2 shows three decision functions (lines) for separating these points.

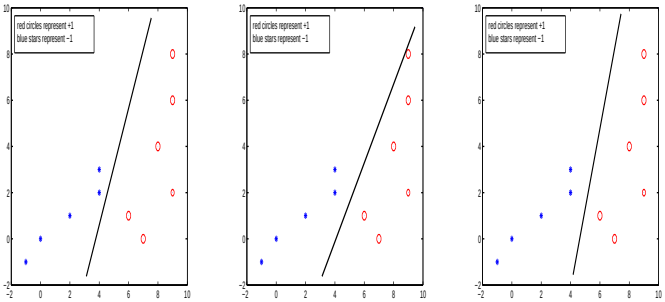


Figure 2: Three decision functions (lines) for the data in Figure 1.

Obviously there are **infinitely many such decision functions** for this data set!!!! So, the question that arises is **which is the best decision function and how to determine it**.

The use to the word best suggest that we must come up with some criterion with respect to which a solution (here a decision function) is judged to be best.

3 Margin

To do this we introduce the concept of **margin**. Informally we will define the margin as follows: let $\mathbf{w} \cdot \mathbf{x} + b$ be the boundary. Then the margin is the **width** by which the boundary can be increased before hitting a data point, illustrated in Figure 3.

The data points **hit** by the margin are called **support vectors**. Obviously for each decision function/boundary we have different margins and therefore possibly different support vectors. The **maximum margin linear classifier** is the classifier with the largest/widest margin. The resulting classifier is called **Linear Support Vector Machine** (LSVM), the simplest kind of SVM.

Next we need to actually find the classifier, which means finding $\mathbf{w}$ and b in equation (2).

Remark 1 *It is fair to ask why we look for the maximum margin. There are answers both from intuitive point of view and from theoretical point of view. From intuitive point of view:*

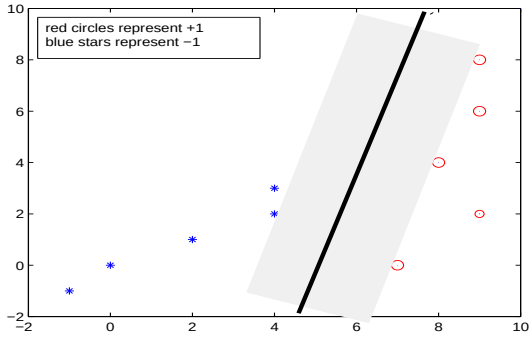


Figure 3: Margin for a two class example set

- *it looks like this will give the best behavior with respect to possible mistakes (best generalization error) for both classes;*
- *experimental results showed that this works VERY well;*
- *the model is immune to removal, addition to data points which are outside the margin;*

From theoretical point of view the arguments come mainly from the PAC/VC dimension learning theory.

The planes delimiting the margin are actually very similar to the boundary, as shown in figure 4. Let $D_c(\mathbf{x}) \equiv \mathbf{w} \cdot \mathbf{x} + b = c$.

$$\begin{cases} \text{the } +1 \text{ plane:} & D_{+1}(\mathbf{x}) \\ \text{boundary:} & D_0(\mathbf{x}) \\ \text{the -1 plane:} & D_{-1}(\mathbf{x}) \end{cases} \tag{3}$$

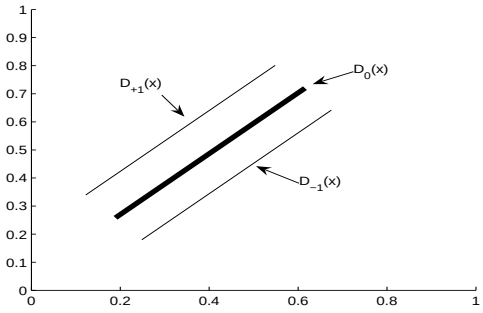


Figure 4: Sketch of $D_0(\mathbf{x}), D_{+1}(\mathbf{x}), D_{-1}(\mathbf{x})$

The classification is then done according to the following rules: For the unknown vector $\mathbf{x}_0$

$$Class(\mathbf{x}_0) = \begin{cases} +1 & \text{if} \quad \mathbf{w} \cdot \mathbf{x} + b \geq 1 \\ -1 & \text{if} \quad \mathbf{w} \cdot \mathbf{x} + b \leq -1 \\ \text{can't tell} & \text{if} \quad -1 < \mathbf{w} \cdot \mathbf{x} + b < +1 \end{cases} \tag{4}$$

4 Computing the margin

We want to compute the margin width, M, in terms of $\mathbf{w}$ and b.

Claim 1 *The vector* $\mathbf{w}$ *is perpendicular on both* D_{+1} *and* D_{-1}.

It is easy to see this from computing $\mathbf{w} \cdot (\mathbf{u} - \mathbf{v})$ for each case, $\mathbf{u}, \mathbf{v} \in D_{+1}$, and $\mathbf{u}, \mathbf{v} \in D_{-1}$. Figure 5 illustrates this fact:

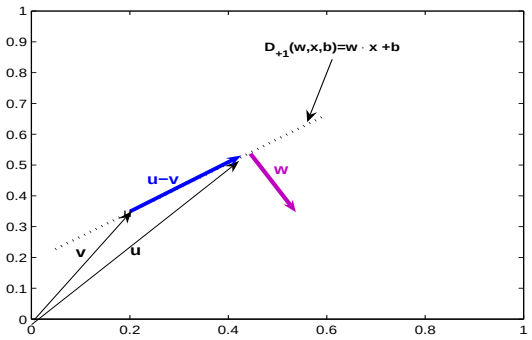


Figure 5: w is perpendicular on the margin planes

Next let $\mathbf{u}^- \in D_{-1}$ an arbitrary vector (need not be support vector). Let $\mathbf{u}^+ \in D_{+1}$ nearest to $\mathbf{u}^-$: Obviously this is on the perpendicular from $\mathbf{u}^-$ on D_{+1}.

Claim 2 *There exists* λ *such that*

$$\mathbf{u}^+ = \mathbf{u}^- + \lambda \mathbf{w}$$

Since the line connecting $\mathbf{u}^+$ and $\mathbf{u}^-$ is perpendicular on all the planes, and $\mathbf{w}$ is also perpendicular, to get from $\mathbf{u}^-$ to $\mathbf{u}^+$ one has to travel some distance in the direction $\mathbf{w}$.

Thus we have

$$M = |\mathbf{u}^+ - \mathbf{u}^-| = \mathbf{u}^- = |\mathbf{u}^- + \lambda \mathbf{w} - \mathbf{u}^-| = |\lambda \mathbf{w}|$$

$$\begin{aligned} &\mathbf{w} \cdot \mathbf{u}^+ + b = +1 \\ &\Longrightarrow \mathbf{w} \cdot (\mathbf{u}^- + \lambda \mathbf{w}) + b = +1 \\ &\Longrightarrow -1 + \lambda \mathbf{w} \cdot \mathbf{w} = 1 \\ &\Longrightarrow \lambda = \tfrac{2}{\mathbf{w} \cdot \mathbf{w}} \end{aligned}$$

Thus

$$M = \lambda |\mathbf{w}| = \lambda \sqrt{\mathbf{w} \cdot \mathbf{w}} = \frac{2}{\mathbf{w} \cdot \mathbf{w}} \sqrt{\mathbf{w} \cdot \mathbf{w}} = \frac{2}{\sqrt{\mathbf{w} \cdot \mathbf{w}}}$$

The next issue is how to learn the maximum margin linear classifier. There are several approaches. We will pursue that using Quadratic Programming(QP).

5 Quadratic Programming

Constrained optimization problems:

- Objective function: the function to be optimized (maximize)
- Constraints: inequalities / equalities

When the objective function is quadratic (degree 2) we call this quadratic optimization problem.

A trivial (quadratic) optimization problem subject to constraints: find the maximum area that can be surrounded by a fence of fixed length(l): Let L, W be the unknown length and the width of the area we are looking for: $A = L \times W$. We know that $2(L + W) = l$. The optimization problem is

$$\begin{array}{l}\text{Maximize } A = L \times W \\ \text{subject to} \\ 2(L+W) = l\end{array}$$

From the formula for the margin, $M = \frac{2}{\sqrt{\mathbf{w}\cdot\mathbf{w}}}$ it follows that to maximize M we need to minimize $\sqrt{\mathbf{w}\cdot\mathbf{w}}$, or equivalently

$$\textbf{Minimize } \mathbf{w}\cdot\mathbf{w}$$

which is a quadratic (degree 2) expression. Now this minimization is subject to the conditions that the training points must be correctly classified. That is, if $\mathbf{x}^+$ is in the positive class, it must satisfy

$$D_{+1}(\mathbf{x}) = \mathbf{w}\cdot\mathbf{x} + b \geq +1$$

and if it is in the negative class, then

$$D_{-1}(\mathbf{x}) = \mathbf{w}\cdot\mathbf{x} + b \leq -1$$

Now denoting by $y = \pm 1$ the label for $\mathbf{x}$ in each case, and multiplying the corresponding D with it we obtain the expression

$$y(\mathbf{w}\cdot\mathbf{x} + b) \geq 1$$

Thus, given n training points, we will have the constraints

$$y_i(\mathbf{w}\cdot\mathbf{x}_i + b) \geq 1 \text{ for } i = 1, \ldots, n$$

To summarize then, our quadratic programming problem is

$$\begin{array}{l}\text{Minimize } \frac{1}{2}||\mathbf{w}||^2 \\ \text{subject to} \\ y_i(\mathbf{w}\cdot\mathbf{x}_i + b) \geq 1 \text{ for } i = 1, \ldots, n\end{array}$$

where $||\mathbf{w}||^2 = \mathbf{w}\cdot\mathbf{w}$

Let us consider some very simple examples:

Example 1 *One dimensional case: let the training be*

$$S = \{((-3,0),-1),((-2,0),-1),((0,0),-1),((1,0),-1),((3,0),+1),((4,0),+1),((5,0),+1)\}$$

Let $\mathbf{w} =< w_1, w_2 >$ *and* b *the weight and bias respectively. We want to determine them so as to minimize* $||\mathbf{w}||^2$ *subject to the conditions*

$$\begin{array}{rcrcl} < w_1, w_2 > & \cdot & < -3, 0 > + b & \leq & -1 \\ < w_1, w_2 > & \cdot & < -2, 0 > + b & \leq & -1 \\ < w_1, w_2 > & \cdot & < 0, 0 > + b & \leq & -1 \\ < \mathbf{w_1}, \mathbf{w_2} > & \cdot & < \mathbf{1}, \mathbf{0} > + \mathbf{b} & = & \mathbf{-1} \\ < \mathbf{w_1}, \mathbf{w_2} > & \cdot & < \mathbf{3}, \mathbf{0} > + \mathbf{b} & = & \mathbf{+1} \\ < w_1, w_2 > & \cdot & < 4, 0 > + b & \geq & +1 \\ < w_1, w_2 > & \cdot & < 5, 0 > + b & \geq & +1 \end{array} \tag{5}$$

I used the notation $< a, b >$ *to emphasize that we use the pair* (a, b) *as a vector!!!! Use* $< a_1, a_2 > \cdot < b_1, b_2 >= a_1 b_1 + a_2 b_2$ *to obtain from (5)*

$$\begin{array}{l} -3w_1 + b \leq -1 \\ -2w_1 + b \leq -1 \\ b \leq -1 \\ \mathbf{w_1 + b = -1} \\ \mathbf{3w_1 + b = +1} \\ 4w_1 + b \geq +1 \\ 5w_1 + b \geq +1 \end{array} \tag{6}$$

Solving the equations in bold of (6) we obtain the solution:

$$\begin{cases} w_1 = 1 \\ b = -2 \end{cases} \tag{7}$$

It is easy to verify that with these values of w_1 and b all the inequalities in (6) are satisfied.

We now proceed to find w_2 such that $\frac{1}{2}||\mathbf{w}||^2$ is minimum. Let

$$f(w_2) = \frac{1}{2}||\mathbf{w}||^2 = \frac{1}{2}(w_1^2 + w_2^2) = \frac{1}{2}(1 + w_2^2)$$

Obviously this is minimized for $w_2 = 0$. Thus the planes determined are

$$D_c(\mathbf{w}, \mathbf{x}, b) = \mathbf{w} \cdot \mathbf{x} + b =< 1, 0 > \cdot < x_1, x_2 > -2 = x_1 - 2 = c$$

That is, the boundary plane is $x_1 - 2 = 0$, or $x_1 = 2$. Figure 6 illustrates this example.

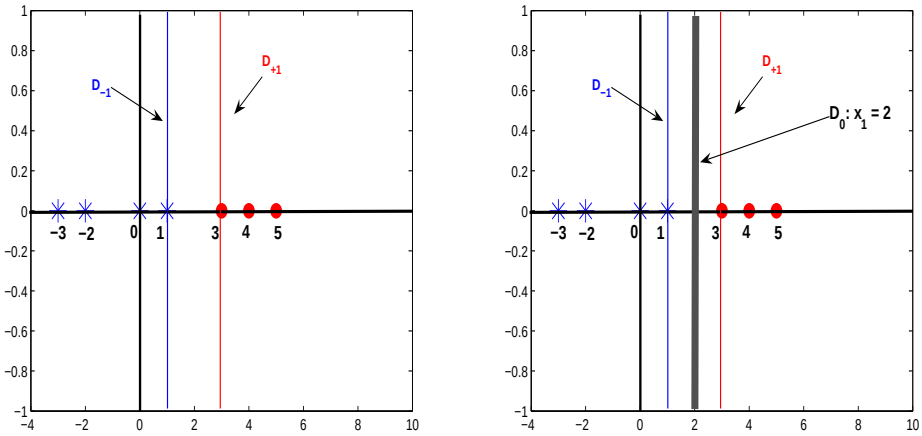


Figure 6: Simple example of deriving the boundary hyperplane in one dimensional case.

6 Digression:Lagrange Multipliers

Start with a small/trivial example:

Example 2 *Find the largest rectangular box without a lid that can be made from a piece of cardboard with total surface area of $12m^2$. Let x, y, z denote the length, width and height/depth of the box. Then the volume is $V(x, y, z) = xyz$. The total surface area of the box without the lid is $A(x, y, z) = xy + 2(yz + xz)$. Thus our problem is*

$$\begin{aligned} &\textit{Maximize } V(x, y, z) = xyz \\ &\textit{subject to } A(x, y, z) = xy + 2(yz + xz) = 12 \end{aligned}$$

We can solve this directly by reducing the problem to an **unconstrained** maximization problem. Or, we can use Lagrange Multipliers Method tat can be stated as follows:

The maximum/minimum values of a function $f(x, y, z)$ subject to constraint $g(x, y, z) = k$ (if they exist) are found as follows:

- Form the quantity $Q(x, y, z, \alpha) = \nabla f(x, y, z) - \alpha \nabla g(x, y, z)$
- Find all the values x, y, z, α such that

- $Q(x, y, z, \alpha) = 0$, and
- $g(x, y, z) = k$

- Evaluate f at the solutions (x, y, z) that result from the previous step; The largest value of f is the maximum; the smallest value of f is the minimum.

Let us return to the example above. We set

$$Q(x, y, z, \alpha) = xyz - \alpha(xy + 2yz + 2xz)$$

Next take the partial derivatives and set equal to zero and use $g(x, y, z) = 12$:

$$\begin{cases} Q_x(x, y, z, \alpha) = yz - \alpha(y + 2z) = 0 \\ Q_y(x, y, z, \alpha) = xz - \alpha(x + 2z) = 0 \\ Q_z(x, y, z, \alpha) = xy - \alpha(2x + 2y) = 0 \\ xy + 2xz + 2yz = 12 \end{cases}$$

Note that $\alpha \neq 0$ (for if it is zero the constraint is violated) Solving the system of equations we obtain $x = 2, y = 2, z = 1$.

In the above α is called Lagrange multiplier. Note that we had one constraint (equality) and that we introduced one lagrange Multiplier for it. If we have more than one constraint we introduce a Lagrange Multiplier for each of these. So, if we have

$$\text{Maximize } f(x, y, z) \text{ subject to } g(x, y, z) = k \text{ and } h(x, y, z) = c$$

we form

$$Q(x, y, z, \alpha, \beta) = f(x, y, z) - \alpha g(x, y, z) - \beta h(x, y, z)$$

and solve the following system for x, y, z, α, and β:

$$\begin{cases} Q_x(x, y, z, \alpha, \beta) = 0 \\ Q_y(x, y, z, \alpha, \beta) = 0 \\ Q_z(x, y, z, \alpha, \beta) = 0 \\ g(x, y, z) = k \\ h(x, y, z) = c \end{cases}$$

7 Applying Lagrange Multipliers(LM) for the SVM problem

So we want to minimize $\frac{1}{2}||\mathbf{w}||^2$ subject to the constraints $y_i(\mathbf{w} \cdot \mathbf{x}_i + b) - 1 \geq 0$. Note that we have n constraints, where n is the size of the data set (so usually very large). Following the above discussion on the LM we form

$$Q(\mathbf{w}, b, \alpha) = \frac{1}{2}||\mathbf{w}||^2 - \sum_{i=1}^{n} \alpha_i \left[y_i(\mathbf{w} \cdot \mathbf{x}_i + b) - 1\right] \tag{8}$$

Note that in (8) α denotes the vector $\alpha_1, \ldots, \alpha_n)$ of the LM corresponding to each constraint.

We need to solve

$$\frac{\delta Q}{\delta \mathbf{w}} = 0 \tag{9}$$

and

$$\frac{\delta Q}{\delta b} = 0 \tag{10}$$

Solving (10) we obtain

$$\sum_{i=1}^{n} \alpha_i y_i = 0 \tag{11}$$

To solve (9) let us actually do it in detail: Assume we are in dimension 2, that is $\mathbf{x} =< x_1, x_2 >$, and $\mathbf{w} =< w_1, w_2 >$.

Then we rewrite (8) as

$$Q(w_1, w_2, x_1, x_2, \alpha) = \frac{1}{2}(w_1^2 + w_2^2) - \sum_{i=1}^{n} [\alpha_i y_i (w_1 x_1 + w_2 x_2) + \alpha_i y_i b - \alpha_i y_i] \tag{12}$$

Take now the derivatives with respect to w_1, w_2:

$$\frac{\delta Q}{\delta w_1} = \frac{1}{2} 2 w_1 - \sum_{i=1}^{n} \alpha_i y_i x_1 = 0 \Longrightarrow w_1 = \sum_{i=1}^{n} \alpha_i y_i x_1 \tag{13}$$

$$\frac{\delta Q}{\delta w_2} = \frac{1}{2} 2 w_2 - \sum_{i=1}^{n} \alpha_i y_i x_2 = 0 \Longrightarrow w_2 = \sum_{i=1}^{n} \alpha_i y_i x_2 \tag{14}$$

Writing (13) and (14) together yields

$$\mathbf{w} = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x_i} \tag{15}$$

We now add another set of conditions that come from a result in optimization theory, the Kuhn-Tucker Theorem which states that **the Lagrange parameters/multipliers can be non-zero only if the corresponding inequality constraint is and equality at the solution**.

So what was our inequality constraint?

$$y_i(\mathbf{w} \cdot \mathbf{x_i} + b) \geq 1 \iff y_i(\mathbf{w} \cdot \mathbf{x_i} + b) - 1 \geq 0, i = 1, \ldots, n$$

So, the solution must satisfy

$$\alpha_i \left[y_i(\mathbf{w} \cdot \mathbf{x_i} + b) - 1 \right] = 0, \ i = 1, \ldots, n \tag{16}$$

Equations (16) are known as the Karush-Kuhn-Tucker complementarity conditions. We will refer to them as the KKT conditions.

It follows from (16) that $\alpha_i = 0$ or $\alpha_i \neq 0$. In the latter case, the KKT conditions hold if $y_i(\mathbf{w} \cdot \mathbf{x_i} + b) - 1 = 0$.

The training points $\mathbf{x_i}$ for which $\alpha_i \neq 0$ are called **support vectors**.

8 Simple one dimensional example revisited

Recall the data set was

$$S = \{((-3, 0), -1), ((-2, 0), -1), ((0, 0), -1), ((1, 0), -1), ((3, 0), +1), ((4, 0), +1), ((5, 0), +1)\}$$

Thus we have $\mathbf{y} = [-1, -1, -1, -1, 1, 1, 1]$. Let $\alpha = [\alpha_1, \alpha_2, \alpha_3, \alpha_3, \alpha_5, \alpha_6, \alpha_7]$ Then condition (11) becomes

$$-\alpha_1 - \alpha_2 - \alpha_3 - \alpha_4 + \alpha_5 + \alpha_5 + \alpha_7 = 0$$

The KKT conditions become:

$$\begin{cases} \alpha_1 \left[-1(< w_1, w_2 > \cdot < -3, 0 > + b) - 1 \right] = 0 \\ \alpha_2 \left[-1(< w_1, w_2 > \cdot < -2, 0 > + b) - 1 \right] = 0 \\ \alpha_3 \left[-1(< w_1, w_2 > \cdot < 0, 0 > + b) - 1 \right] = 0 \\ \alpha_4 \left[-1(< w_1, w_2 > \cdot < 1, 0 > + b) - 1 \right] = 0 \\ \alpha_5 \left[+1(< w_1, w_2 > \cdot < 3, 0 > + b) - 1 \right] = 0 \\ \alpha_6 \left[+1(< w_1, w_2 > \cdot < 4, 0 > + b) - 1 \right] = 0 \\ \alpha_7 \left[+1(< w_1, w_2 > \cdot < 5, 0 > + b) - 1 \right] = 0 \end{cases}$$

which further becomes

$$\begin{cases} \alpha_1 [3w_1 - b - 1] = 0 \\ \alpha_2 [2w_1 - b - 1] = 0 \\ \alpha_3 [-b - 1] = 0 \\ \alpha_4 [-w_1 - b - 1] = 0 \\ \alpha_5 [3w_1 + b - 1] = 0 \\ \alpha_6 [4w_1 + b - 1] = 0 \\ \alpha_7 [5w_1 + b - 1] = 0 \end{cases}$$

Use now (15) to substitute

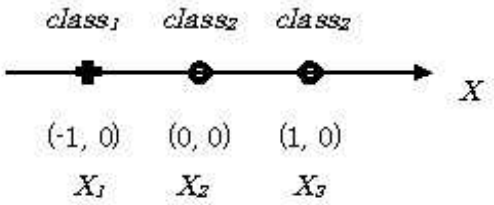
$$w_1 = \sum_{i=1}^{7} \alpha_i y_i x_1 = 3\alpha_1 + 2\alpha_2 + 0\alpha_3 - \alpha_4 + 3\alpha_5 + 4\alpha_6 + 5\alpha 7$$

to obtain

$$\begin{cases} \alpha_1 [3(3\alpha_1 + 2\alpha_2 + 0\alpha_3 - \alpha_4 + 3\alpha_5 + 4\alpha_6 + 5\alpha 7) - b - 1] = 0 \\ \alpha_2 [2(3\alpha_1 + 2\alpha_2 + 0\alpha_3 - \alpha_4 + 3\alpha_5 + 4\alpha_6 + 5\alpha 7) - b - 1] = 0 \\ \alpha_3 [-b - 1] = 0 \\ \alpha_4 [-(3\alpha_1 + 2\alpha_2 + 0\alpha_3 - \alpha_4 + 3\alpha_5 + 4\alpha_6 + 5\alpha 7) - b - 1] = 0 \\ \alpha_5 [3(3\alpha_1 + 2\alpha_2 + 0\alpha_3 - \alpha_4 + 3\alpha_5 + 4\alpha_6 + 5\alpha 7) + b - 1] = 0 \\ \alpha_6 [4(3\alpha_1 + 2\alpha_2 + 0\alpha_3 - \alpha_4 + 3\alpha_5 + 4\alpha_6 + 5\alpha 7) + b - 1] = 0 \\ \alpha_7 [5(3\alpha_1 + 2\alpha_2 + 0\alpha_3 - \alpha_4 + 3\alpha_5 + 4\alpha_6 + 5\alpha 7) + b - 1] = 0 \end{cases}$$

We obtained a system of $n = 7$ **quadratic** equations, which has 2^n possible solutions! For small n we could go ahead and solve this, but, as n increases solving this directly is not an option!!!! We are NOT going to solve this... Instead we look at other ways.

Example 3 *This example illustrates all the steps to solve a simple case with linearly separable, one-dimensional training data (Figure 7).*



X	Y	
-1	+1	class1
0	-1	class2
1	-1	class2

Figure 7: Training data in Example 3 (linearly separable)

training data points	(-1,0)	(0,0)	(1,1)
label y	-1	+1	+1
α	α_1	α_2	α_3

Then equation (??) becomes

$$-\alpha_1 + \alpha_2 + \alpha_3 = 0 \quad \textbf{\textit{(A)}}$$

From (??) it follows

$$w_1 = \sum \alpha_i y_i \mathbf{x_{i1}} = \alpha_1 + \alpha_3 \quad \textbf{(B)}$$

The dot product between $\mathbf{w}$ *and* $\mathbf{x_i}$ *is*

$$\mathbf{w} \cdot \mathbf{x_1} = -\alpha_1 - \alpha_3; \;\; \mathbf{w} \cdot \mathbf{x_2} = 0; \;\; \mathbf{w} \cdot \mathbf{x_3} = \alpha_1 + \alpha_3;$$

The KKT conditions become

$$\begin{cases} \alpha_1\{-1(-\alpha_1 - \alpha_3 + b) - 1\} = 0 \\ \alpha_2\{+1(0 + b) - 1\} = 0 \\ \alpha_3\{+1(\alpha_1 + \alpha_3 + b) - 1\} = 0 \end{cases}$$

which becomes:

$$\begin{cases} \alpha_1\{\alpha_1 + \alpha_3 - b - 1\} = 0 \\ \alpha_2\{b - 1\} = 0 \\ \alpha_3\{\alpha_1 + \alpha_3 + b - 1\} = 0 \end{cases}$$

Of all the possible assignments/values for α_i *and* b *only*

$$\alpha_1 = 2; \alpha_2 = 2; \alpha_3 = 0; b = 1$$

satisfies the KKT conditions and condition (A)*. Then,* $\mathbf{w} = (w_1, 0) = (2, 0)$*, and hence the planes we are looking for are:*

$$D_c(\mathbf{w}, \mathbf{x}, b) : 2\mathbf{x} + 1 = c$$

- $c = -1$*:* $D_{-1}(\mathbf{w}, \mathbf{x}, b) : 2\mathbf{x} + 1 = -1 : \mathbf{x} = -1$
- $c = 0$*:* $D_0(\mathbf{w}, \mathbf{x}, b) : 2\mathbf{x} + 1 = 0 : \mathbf{x} = \frac{-1}{2}$
- $c = 1$*:* $D_{+1}(\mathbf{w}, \mathbf{x}, b) : 2\mathbf{x} + 1 = +1 : \mathbf{x} = 0$

Again, I cut corners: I should have considered all the possible case for $\alpha_i = 0$ *and eliminate those which lead to contradictions (please try to work out as an excercise)..*

9 Elements of Optimization Theory

We start by stating the general problem - minimize/maximize a function subject to constraints.

9.1 Primal Optimization Problem

$$\begin{array}{lll} \text{mimimize} & f(\mathbf{w}), \text{ with } \mathbf{w} \text{ a vector in } \Re^n & \leftarrow \text{objective function} \\ \text{subject to} & g_i(\mathbf{w}) \leq 0, \;\; i = 1, \ldots, k & \leftarrow \text{constraint} \\ & h_j(\mathbf{w}) = 0, \;\; j = 1, \ldots, m & \leftarrow \text{constraint} \end{array} \tag{17}$$

Note that any maximization problem can be turned into a minimization problem by simply multiplying by -1.

Feasible region:

the collection of vectors $\mathbf{w}$ for which the constraints hold:

$$R = \{\mathbf{w} : \;\; \mathbf{g}(\mathbf{w}) \leq \mathbf{0}, \mathbf{h}(\mathbf{w}) = \mathbf{0}\}$$

Note $\mathbf{g} = (g_1, \ldots, g_k)$, $\mathbf{h} = (h_1, \ldots, h_m)$, and $\mathbf{0} = (0, \ldots, 0)$.

Solution to (17):

- **Global minimum:** $\mathbf{w}^*$ is a solution (global minimum) to (17) if there is no other $\mathbf{w}$ such that $f(\mathbf{w}) < f(\mathbf{w}^*)$
- **Local minimum:** $\mathbf{w}^*$ is a local minimum to (17) if there exists $\epsilon > 0$ such that $f(\mathbf{w}^*) \leq f(\mathbf{w})$ for $\|\mathbf{w} - \mathbf{w}^*\| < \epsilon$.

Quadratic Program:

- An optimization problem in which the objective function, inequality, and equality constraints are linear is called **linear program**
- If the objective function is quadratic while all the constraints are all linear the problem is called **quadratic program**.

Convex function:

$f : D \longrightarrow \Re^n$ is called **convex** if for $\mathbf{w}, \mathbf{u} \in D$, and for any $\theta \in (0, 1)$

$$f(\theta\mathbf{w} + (1-\theta)\mathbf{u}) \leq \theta f(\mathbf{w}) + (1-\theta) f(\mathbf{u})$$

If $\leq$ is replaced by $<$ the function is called strictly convex.

Fact

Any local minimum $\mathbf{w}^*$ for the convex objective function f in an unconstrained optimization problem is also a global minimum: let $\mathbf{u} \neq \mathbf{w}^*$. Let$\epsilon, \theta > 0$ such that

$$\theta > 1 - \frac{\epsilon}{\|w^* - u\|} > 0$$

Then

$$\|\mathbf{w}^* - (\theta\mathbf{w}^* + (1-\theta)\mathbf{u})\| \leq \epsilon$$

and

$$f(\mathbf{w}^*) \overset{\mathbf{w}^* \text{ is local min}}{\leq} \leq f(\theta\mathbf{w}^* + (1-\theta)\mathbf{u}) \overset{\text{by convexity}}{\leq} \theta f(\mathbf{w}^*) + (1-\theta) f(\mathbf{u})$$

Convex Optimization Problem:

The objective function, its domain and all of the constraints are convex. **The SVM problem is a convex, quadratic program**.

Lagrangian Theory: Fermat (1629), Lagrange (1797), Kuhn and Tucker (1951)

We have the following results:

Theorem 1 *(Fermat) A* necessary *condition for* $\mathbf{w}^*$ *to be a minimum of* $f(\mathbf{w})$*, where* f *is differentiable and has continuous derivatives, is*

$$\frac{\delta f(\mathbf{w}^*)}{\delta \mathbf{w}} = 0 \tag{18}$$

If f *is convex then (18) condition is also sufficient.*

For constrained problems: introduce Lagrange multipliers and Lagrangian.

Lagrangian:

Given the objective function $f(\mathbf{w})$, with equality constraints $h_i(\mathbf{w}) = 0$, $i = 1, \ldots, m$ the *Lagrangian function* is defined as

$$L(\mathbf{w}, \beta) = f(\mathbf{w}) + \sum_{i=1}^{m} \beta_i h_i(\mathbf{w})$$

β_i are called *Lagrange multipliers.*

Theorem 2 *(Lagrange) A necessary condition for a normal vector* $\mathbf{w}^*$ *to be the minimum of* $f(\mathbf{w})$ *subject to* $h_i(\mathbf{w}) = 0$, $i = 1, \ldots, m$ *(with* $f, h_i \in C^1$*) is*

$$\begin{array}{rcl} \frac{\delta L(\mathbf{w}^*, \beta^*)}{\delta \mathbf{w}} & = & 0 \\ \frac{\delta L(\mathbf{w}^*, \beta^*)}{\delta \beta^*} & = & 0 \end{array} \tag{19}$$

for some values $\beta^* = (\beta_1, \ldots, \beta_m)$.

If $L(\mathbf{w}^*, \beta^*)$ *is convex then (19) are also sufficient.*

Example 4 *(Maximum entropy distribution) Let* $\mathbf{p} = (p_1, \ldots, p)n)$ *be a discrete probability distribution:*

$$p_i \geq 0; \ \sum_{i=1}^{n} p_i = 1$$

Define its entropy as

$$H(\mathbf{p}) = -\sum_{i=1}^{n} p_i \log p_i$$

We want to find the distribution (assignments of p_i*) which has the largest entropy¿ This means that we need to solve the following optimization problem:*

$$\begin{array}{ll} \textit{maximize} & H(\mathbf{p}) \\ \textit{subject to} & \sum_{i=1}^{n} p_i = 1 \end{array}$$

For the Lagrangian

$$L(\mathbf{p}, \beta) = \sum_{i=1}^{n} p_i \log p_i + \beta(\sum_{i=1}^{n} p_i - 1)$$

β *is the Lagrange multiplier for the constraint* $\sum_{i=1}^{n} p_i = 1$ *equivalent to* $\sum_{i=1}^{n} p_i - 1 = 0$. *It is easy to see that*

$$\frac{\delta L(\mathbf{p}, \beta)}{\delta p_i} = \log p_i + p_i \frac{1}{p_i} \log e + \beta = \log p_i + \log e + \beta = \log e p_i + \beta = 0, \textit{for all } i = 1, \ldots, n$$

leads to

$$p_i = \frac{2^{-\beta}}{e} = c = \textit{constant}$$

Thus

$$1 = \sum_{i=1}^{n} p_i = \sum_{i=1}^{n} c = nc \Longrightarrow c = \frac{1}{n}$$

Most general optimization problem: equality and inequality constraints

The *Lagrangian* of the optimization problem

$$\begin{array}{ll} \text{minimize} & f(\mathbf{w}),\ \mathbf{w} \in \Re^n \\ \text{subject to} & g_i(\mathbf{w}) \leq 0,\ i = 1, \ldots, k \\ & h_j(\mathbf{w}) = 0,\ j = 1, \ldots, m \end{array}$$

is defined as

$$L(\mathbf{w}, \alpha, \beta) = f(\mathbf{w}) + \sum_{i=1}^{k} \alpha_i g_i(\mathbf{w}) + \sum_{j=1}^{m} \beta_j h_j(\mathbf{w}) \stackrel{\text{vector notation}}{=} f(\mathbf{w}) + \alpha' \mathbf{g}(\mathbf{w}) + \beta' \mathbf{h}(\mathbf{w})$$

Here α' denotes transpose of α.

10 Lagrangian Dual Problem

$$\begin{array}{ll} \text{maximize} & \Theta(\alpha, \beta) \\ \text{subject to} & \alpha > \mathbf{0} \end{array} \tag{20}$$

where

$$\Theta(\alpha, \beta) = \inf_{\mathbf{w}} L(\mathbf{w}, \alpha, \beta)$$

Theorem 3 *(Kuhn-Tucker) The necessary and sufficient conditions for a normal vector* $\mathbf{w}^*$ *to be the solution of the optimization problem with convex domain*

$$\begin{array}{ll} \textit{minimize} & f(\mathbf{w}) \\ \textit{subject to} & g_i(\mathbf{w}) \leq 0,\ i = 1, \ldots, k, \\ & h_j(\mathbf{w}) = 0,\ j == 1, \ldots, m, \end{array} \tag{21}$$

with $f \in C^1$, g_i, h_j *linear in* $\mathbf{w}$, *are the existence of* α^*, β^*, *such that*

$$\begin{array}{lcl} \frac{\delta L(\mathbf{w}^*, \alpha^*, \beta^*)}{\delta \mathbf{w}} & = & 0, \\ \frac{\delta L(\mathbf{w}^*, \alpha^*, \beta^*)}{\delta \beta} & = & 0, \\ \alpha_i^* g_i(\mathbf{w}) & = & 0,\ i = 1, \ldots, k, \leftarrow \textbf{Karush-Kuhn-Tucker complementarity conditions} \\ g_i(\mathbf{w}) & \leq & 0,\ i = 1, \ldots, k, \\ \alpha_i^* & \geq & 0,\ i = 1, \ldots, k. \end{array}$$

11 Practical Use of Duality

- The dual problem is obtained by introducing the Lagrange multipliers (one for each constraint) and then solving an unconstrained optimization problem. This is easier (although it may be computationally expensive) to solve, then to handle the inequality constraints directly.
- the Lagrange multipliers are called dual variables and they become the unknowns of the problem.
- The primal problem is transformed into the dual problem using the following steps:
  1. Form the Lagrangian: Objective function + LM $\times$ constraints.
  2. Set to 0 the derivatives of the Lagrangian with respect to the primal variables $(\mathbf{w}, b)$ and solve for the primal variables.
  3. Substitute these back into the Lagrangian: there are no more primal variables.

S	-3	-1	0
Y	-1	+1	-1

S'	(-3, 9)	(-1, 1)	(0, 0)
Y	-1	+1	-1

4. The result is a function of the dual variables only,

$$\Theta(\alpha, \beta) = \inf_{\mathbf{w}} L(\mathbf{w}, \alpha\beta)$$

which must be maximized under simpler constraints.

When we do this for the SVM (hard margin) problem, the solution is found as follows:

$$\begin{array}{ll} \text{maximize} & \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \mathbf{x}_i \cdot \mathbf{x}_j \\ \text{subject to} & \alpha_i \geq 0, i = 1, \ldots, n, \\ & \sum_{i=1}^{n} \alpha_i y_i = 0 \end{array} \tag{22}$$

Define

$$\begin{array}{l} \mathbf{w} = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i \\ \text{need to select } b\text{: } b = y_K - \mathbf{x}_K \cdot_K \\ \text{where } K = \arg\max_i \alpha_i \end{array} \tag{23}$$

To classify a data point $\mathbf{x}$ use $sign(\mathbf{w} \cdot \mathbf{x} + b)$ which because of (**??**) becomes

$$sign(\sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i \cdot \mathbf{x} + b) \tag{24}$$

Note that because of the theory behind this approach we know that the SVM solution is optimal! That is, it has highest generalization power.

12 Introduce Kernels

Let us look at (22) and(24) and note that the basic operation is the **dot product** between data points in the training set, or between these and a test data point.

On the other hand, we have seen in small(low dimensional) examples, that the data are not linearly separable we can transform it into a linearly separable set by mapping into a higher dimension. For example, the training set which is NOT linearly separable in $\Re$, can be mapped using

$$x \mapsto (x, x^2)$$

into the 2-dimensional linearly separable set

In this example we found and **explicit mapping**. We can calculate from this mapping the dot product in the new (feature) space. The "dot product" in the original feature space is simply the product in that space:

In the new feature space the dot product is

Table 1: The dot product of the data in S

	-3	-1	0
-3	9	3	0
-1	3	1	0
0	0	0	0

Table 2: The dot product for the data in S'

	(-3,9)	(-1,1)	(0,0)
(-3, 9)	90	12	0
(-1, 1)	12	2	0
(0,0)	0	0	0

But we can actually show that the dot product in the new feature space can be expressed in terms of (is as function of) the dot product in the original space: Let us denote the mapping above by ϕ. That is

$$\phi(x) = \mathbf{z} = (x, x^2)$$

Then, for two image vectors $\mathbf{z}_1, \mathbf{z}_2$, the dot product is

$$\phi(x_1) \cdot \phi(x_2) = \mathbf{z}_1 \cdot \mathbf{z}_2 = x_1 x_2 + x_1^2 x_2^2 = x_1 x_2 + (x_1 x_2)^2$$

where $x_1 x_2$ is the dot product in $\Re$.

Thus the dot product in the new feature space is a polynomial of the dot product in the old feature space.

In this example we computed the dot product in the new feature space from the **explicit mapping** into this space.

However, and this is, perhaps the most important aspect of the SVM mechanism, we do not need the explicit mapping!!!! Rather, we use an **implicit mapping** by specifying that the dot product in the new (unknown) feature space is a **function** of the dot product in the original (old) feature space.

The functions which achieve mappings between dot products are called **kernels**.

$$K(\mathbf{x}, \mathbf{y}) = g(\mathbf{x} \cdot \mathbf{y}) = \phi(\mathbf{x}) \cdot \phi(\mathbf{y})$$

Let us look at a small example.

Example 5 *Assume* $\mathbf{x} = (x_1, x_2) \in \Re^2$. *Define*

$$K(\mathbf{x}, \mathbf{y}) = p(\mathbf{x} \cdot \mathbf{y})$$

where $p(a) = (a+1)^d$, $d = 1, 2, 3...$ *Can we calculate the explicit mapping* ϕ *which leads to a feature space in which* $K(\mathbf{x}, \mathbf{y}) = \phi(\mathbf{x} \cdot \phi(\mathbf{y})$*?*