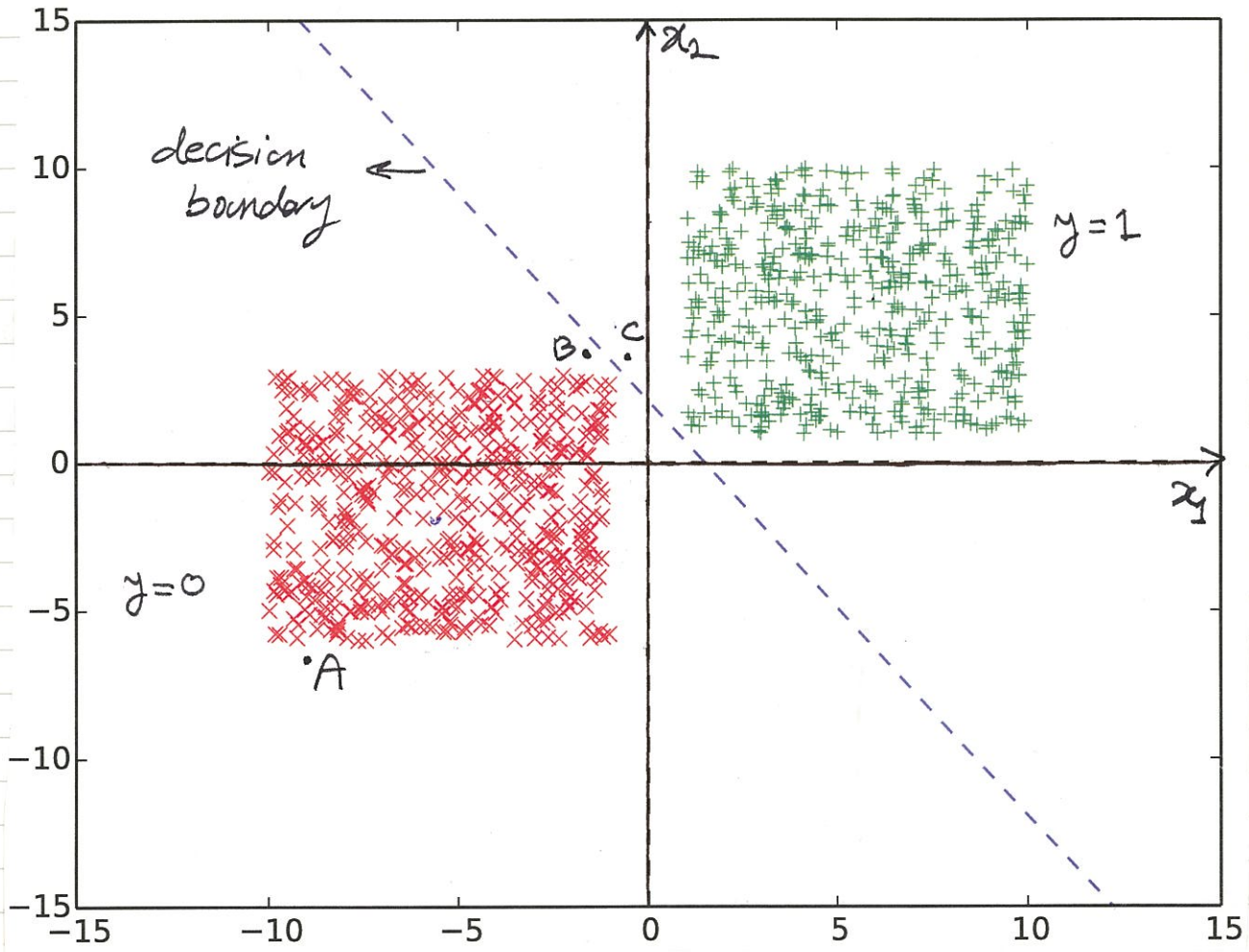
SUPPORT VECTOR MACHINES

Lecture 71: Optimization Objective

The Support Vector Machine (SVM) algorithm is another popular and powerful supervised learning algorithm. Why do we need yet another supervised learning algorithm for classification? To motivate the discussion, look at the following classification, done using logistic regression:



Let's remind ourselves that logistic regression uses the following hypothesis: $h_\theta(x) = g(\theta^T x)$; $g(z) = \frac{1}{1+e^{-z}}$

and that we interpret $h_\theta(x)$ as the propability that $y=1$ at point x. When making a concrete prediction, we choose our confidence level & predict accordingly:

$$\begin{cases} h_\theta(x) \geq \text{confidence level} & \Rightarrow y=1 \\ h_\theta(x) < \text{confidence level} & \Rightarrow y=0 \end{cases}$$

By default we set the confidence level to 0.5:

$$\begin{cases} h_\theta(x) \geq 0.5 \leftrightarrow \theta^T x \geq 0 & \Rightarrow y=1 \\ h_\theta(x) < 0.5 \leftrightarrow \theta^T x < 0 & \Rightarrow y=0 \end{cases}$$

Our decision boundary is then defined by $\theta^T x = 0$.

Consider the figure on the previous page. For point A, we must have that $\text{prob}(y=0) = 1 - h_\theta(x_A) \approx 1$; i.e a very confident prediction of $y=0$. What about point B? It's too close to the decision boundary, so our confidence can't be too much better than 50%. In fact, if we move over just a little bit to point C, we would predict $y=1$. Given the data, though, this is a bad prediction for point C; it's much closer to $y=0$ points than $y=1$.

What would have been a better decision boundary? One for which we would confidently predict $y=0$ for point C? Intuitively, it seems as though the most natural separator of the two data sets is the line $x_1=0$, i.e the x_2 axis. Why? Because it separates the two classes with the largest possible margin.

Can we build our classifier around this idea? That, indeed, is the main motivation behind SVM.

The lecture videos do not do SVM justice in my opinion, so I will go through a lot more detail and discussion here.

Notation

We will consider a linear classifier for a binary classification problem with lables y & features x:

$$y = \begin{cases} +1 & \text{positive} \\ -1 & \text{negative} \end{cases}$$

We will use $y=-1$ instead of $y=0$ for negative examples.

Our classifier will either predict -1 or $+1$:

$$h_{w,b}(x) = g(w^T x + b)$$

where

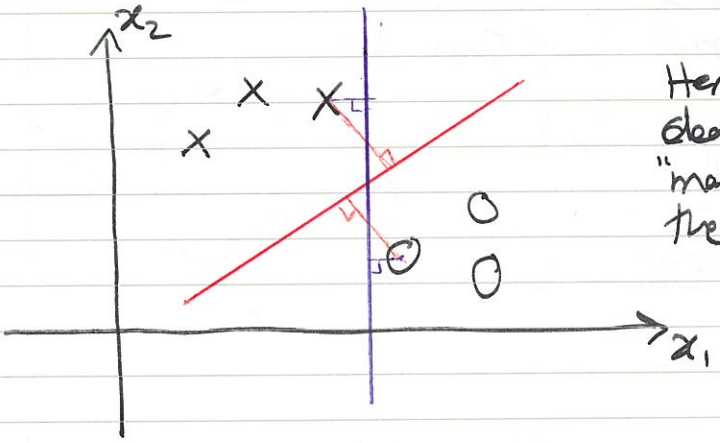
$$g(z) = \begin{cases} +1 & z \geq 0 \\ -1 & z < 0 \end{cases}$$

Note that in our new notation we're using $w = \begin{bmatrix} \theta_1 \\ \vdots \\ \theta_n \end{bmatrix}$ & $b = \theta_0$.

Also, our prediction is no longer probabilistic, as it was for logistic regression.

Large Margin Classifier

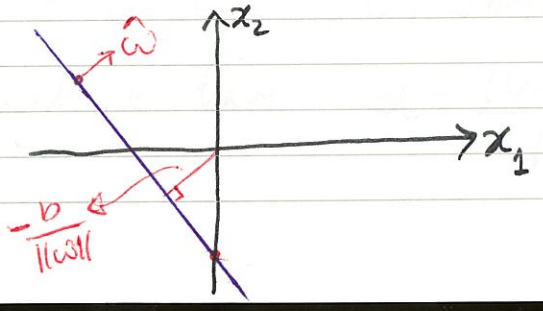
Suppose $y=1$ and $y=-1$ classes are linearly separable. How can we separate them by a hyperplane that "maximizes" the margin between them?



Here the red line is clearly creating a larger "margin" or "gap" between the two classes.

Intuitively, we'd want to select the hyperplane that has the largest distance from the nearest point from either group. Let's formalize this. First, remember that a hyperplane can be characterized as the set of points satisfying:

$$w \cdot x + b = 0$$

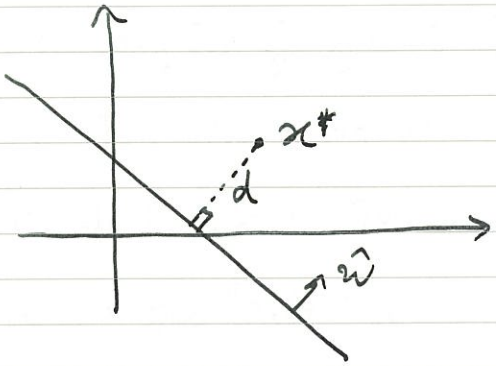


$$\hat{w} = \frac{w}{\|w\|}$$

Note that any hyperplane is characterized by a normal vector $\hat{w}$ & the distance it has from origin. The representation $w \cdot x + b = 0$ is redundant by a scaling of w & b:

$w \cdot x + b = 0$ & $\tilde{w} \cdot x + \tilde{b} = 0$ are the exact same plane if $\tilde{w} = \alpha w$ & $\tilde{b} = \alpha b$.

What is the distance of a point x^* from the line $w \cdot x + b = 0$?



Note that $x^* - d\hat{w}$ is on the line itself. Therefore:

$$w \cdot \left(x^* - d\frac{w}{\|w\|}\right) + b = 0$$

$$\Rightarrow \boxed{d = \frac{w \cdot x^* + b}{\|w\|}}$$

Note that d, being a geometric quantity, is invariant under the rescaling $w \to \alpha w$ & $b \to \alpha w$. Indeed, it only depends on $\hat{w}$ & $b/\|w\|$, which are the normal vector the plane & the distance of the plane from the origin.

Note that our derivation was assuming x^* is "above" the plane, i.e. $w \cdot x^* + b > 0$, otherwise d would be negative. So in general we have $d = \frac{|w \cdot x^* + b|}{\|w\|}$

From our hypothesis on page 38, and of course assuming that the two classes are linearly separable, we require the existance of w & b such that:

$$\begin{cases} w \cdot x^{(i)} + b > 0 & y^{(i)} = 1 \\ w \cdot x^{(i)} + b < 0 & y^{(i)} = -1 \end{cases}$$

This requirement can be combined in the following expression:

$$y^{(i)} (w \cdot x^{(i)} + b) > 0$$

We now see why using $y = -1$ for the negative class is more convenient than $y = 0$. The above requirement just formalizes that all positive examples should be "above" the decision boundary $w \cdot x + b = 0$ & all negative examples "below" it.

Furthermore, of all hyperplanes that separate the two classes, we want the one that maximizes the distance from either class.

The following algorithm accomplishes this:

① pick any hyperplane

② If it separates the two classes:

calculate the distance of all $x^{(i)}$ from the hyperplane and pick the smallest one. Let's call this distance δ.

else:

pass

③ Repeat ① & ② for all hyperplanes & the

Lets translate this algorithm to the wonderful language of mathematics.

① Pick a hyperplane: we take an equivalence class of w, b which define the same hyperplane, i.e those for which $\frac{w_2}{w_1} = \frac{b_2}{b_1}$ is true, since in that case $w_1 x + b_1 = 0$ & $w_2 x + b_2 = 0$ define the exact same plane.

② place the following constraint on w, b: $y^{(i)}(w \cdot x^{(i)} + b) > 0$.

Note that this constraint is valid for any member of the equivalence class, since it just enforces that $y^{(i)} = 1$ are above the hyperplane and $y^{(i)} = -1$ below it.

③ Pick the smallest distance from $x^{(i)}$ to the hyperplane:

$$\min_i \frac{|w \cdot x^{(i)} + b|}{\|w\|}.$$

Again, this is true for any member of the equivalence class, since distances are independent of the choice of scaling. Why don't we then choose a scaling, or member of the equivalence class, that makes our life easier? Let $x^{(i)}_{min}$ be the point with the smallest distance to our hyperplane. We can pick a member of the equivalence class which satisfies: $\boxed{y^{(i)}(w \cdot x^{(i)}_{min} + b) = 1}$

This is a convenient choice, because now the smallest distance from the hyperplane is always $1/\|w\|$.

Also, since all other distances (to points other than $x^{(i)}_{min}$) will be greater than or equal to $1/\|w\|$ (This is true by definition), we have that:

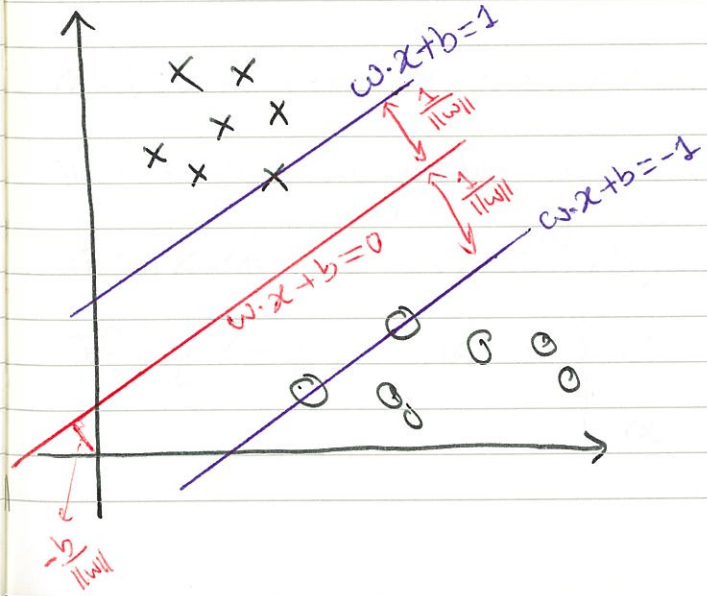
$$\text{distance to } x^{(i)} = \frac{y^{(i)}(w \cdot x^{(i)} + b)}{\|w\|} \geq \frac{1}{\|w\|}$$

$$\Rightarrow \boxed{y^{(i)}(w \cdot x^{(i)} + b) \geq 1} \quad \forall\, i = 1, \ldots, m$$

(m = size of training set, as usual.)

which is a stronger inequality than that we imposed in (2) on the previous page. Finally, our optimization problem is as follows: maximize $1/\|w\|$ s.t. $y^{(i)}(w \cdot x^{(i)} + b) \geq 1$. Maximizing $1/\|w\|$ is the same as minimizing $\|w\|$, which is in turn the same as minimizing $\|w\|^2$:

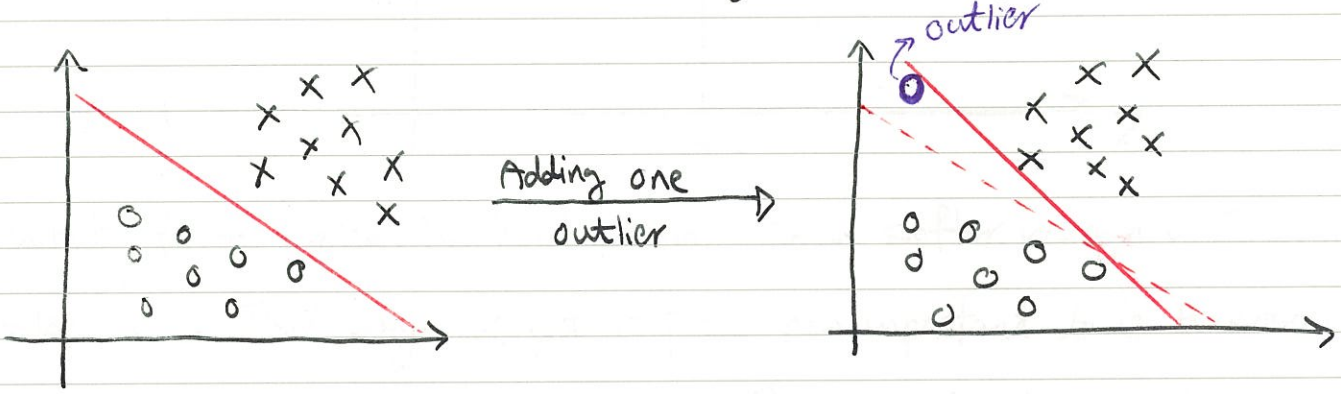
$$\boxed{\begin{aligned} &\min \frac{1}{2}\|w\|^2 \\ &\text{s.t. } y^{(i)}(w \cdot x^{(i)} + b) \geq 1 \quad \forall\, i = 1, \ldots, m \end{aligned}}$$



What we've done is to select two parallel hyperplanes $w \cdot x + b = \pm 1$ which separate the two classes such that the distance between them is as large as possible. Our decision boundary lies half-way between them.

It's interesting to note that the max-margin hyperplane is completely determined by those $x^{(i)}$ which lie nearest to it. These $x^{(i)}$ are called Support Vectors.

So far we've been assuming that the two classes are linearly separable. There's no guarantee that will always be the case. Also, in some cases it's not clear that finding a separating hyperplane is what we'd want, since that might be susceptible to outliers:



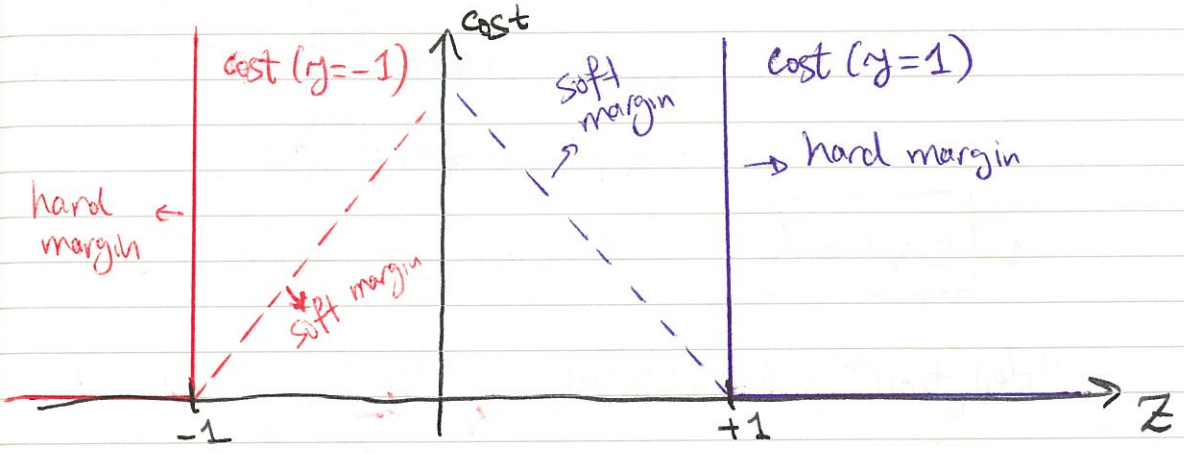
When a single outlier is added, it causes the decision boundary to make a dramatic swing, & the resulting classifier has a much smaller margin. How can we relax our assumption of linear separability and also be less susceptible to outliers. We can think about the constraints $y^{(i)}(w \cdot x^{(i)} + b) \geq 1$ as follows:

If $y^{(i)} = 1$: $\text{cost} = \begin{cases} 0 & \text{if } w \cdot x^{(i)} + b \geq 1 \\ \infty & \text{if } w \cdot x^{(i)} + b < 1 \end{cases}$

And similarly for $y^{(i)} = -1$:

$$\text{cost} = \begin{cases} 0 & \text{if } \omega \cdot x^{(i)} + b \leq -1 \\ \infty & \text{if } \omega \cdot x^{(i)} + b > -1 \end{cases}$$

Let $z \equiv \omega x + b$, then:



What if we replace the hard margin with a softer margin with some slope C? That way, we get to pick how important it is to enforce strict separation by the value of $C > 0$. We then end up with the following objective function:

$$J(\omega, b) = C \sum_{i=1}^{m} \max(0, 1 - y^{(i)}(\omega \cdot x^{(i)} + b)) + \frac{1}{2} \|\omega\|^2$$

For sufficiently large values of C, this will behave very similarly to the hard margin case, if the input data are linearly separable. As it turns out, this objective function is not all that different from

mat of regularized logistic regression. Let's go back the old notation:

$$J(\theta) = C \sum_{i=1}^{m} y^{(i)} \text{Cost}_1^{(SVM)}(\theta^T x^{(i)}) + (1-y^{(i)}) \text{Cost}_0^{(SVM)}(\theta^T x^{(i)}) + \frac{1}{2} \sum_{j=1}^{n} \theta_j^2$$

where we're now using $y=0$ to denote negative examples and as mentioned on page 39: $w_i = \theta_i$ $(i=1 \cdots n)$ & $\theta_0 = b$.

Also:
$$\begin{cases} \text{Cost}_1^{(SVM)} = \max(0, 1-z) \\ \text{Cost}_0^{(SVM)} = \max(0, 1+z) \end{cases}$$

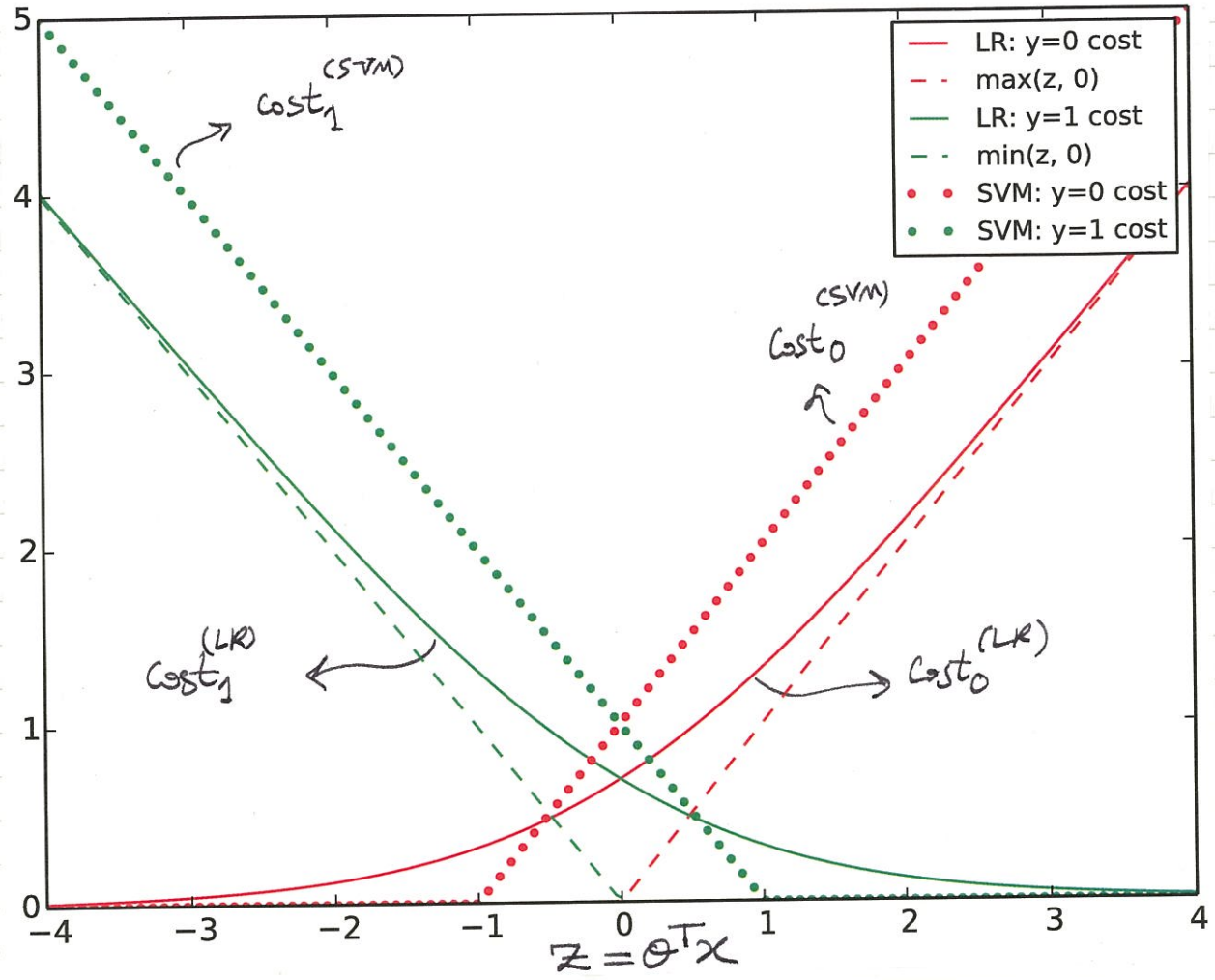
Let $\tilde{J}(\theta) = \frac{1}{mC} J(\theta)$ & $\boxed{\lambda \equiv 1/C}$:

$$\tilde{J}(\theta) = \frac{1}{m} \sum_{i=1}^{m} \left[y^{(i)} \text{Cost}_1(\theta^T x^{(i)}) + (1-y^{(i)}) \text{Cost}_0(\theta^T x^{(i)}) \right] + \frac{\lambda}{2m} \sum_{j=1}^{n} \theta_j^2.$$

Of course, minimization of $J(\theta)$ & $\tilde{J}(\theta)$ will yield the exact same optimal value of θ. This is exactly the form of the objective function of logistic regression (see page 52 of Notebook #1), except:

$$\begin{cases} \text{Cost}_1^{(LR)} = \ln(1+e^{-z}) \\ \text{Cost}_0^{(LR)} = \ln(1+e^{z}) \end{cases}$$

Next page shows a graph of $\text{Cost}^{(SVM)}$ vs. $\text{Cost}^{(LR)}$. We see that SVM assigns a higher cost than LR when you get the prediction wrong.



The correspondence $\lambda = \frac{1}{C}$ is also quite revealing, because we now know large values of C cause overfitting ($\leftrightarrow$ small λ) & small values of C can cause underfitting ($\leftrightarrow$ large λ). We can see this on the figure on page 44 when there's an outlier, if we try too hard to get all examples on the right side of the decision boundary (i.e. large C), we will not generalize well to new examples, i.e we will overfit.

Lecture 72: Large Margin Intuition

We've already motivated the need for large margin classifiers and derived the corresponding optimization problems, both soft and hard margins, using geometric arguments. In this lecture, Andrew gives some intuition for how SVM behaves for large values of C, which of course is as a large margin classifier. One point that's interesting to note about the hard-margin optimization problem is that although we only need $\theta^T x \geq 0$ to predict $y=1$, SVM requires $\theta^T x \geq 1$. Similarly with predicting $y=0$: we only need $\theta^T x < 0$, but SVM requires $\theta^T x \leq -1$. We know from the geometric picture that it's precisely this feature that creates the margin between the two classes. $\theta^T x = \pm 1$ are the parallel hyperplanes that pass through the closest points to the decision boundary.

Lecture 73: Mathematics Behind Large Margin Classification

In this lecture Andrew explains why the hard-margin optimization problem leads to a classifier that creates a large gap between the two classes. We derived that optimization problem by requiring the largest possible margin between two linearly separable classes.

Let's discuss how we can solve the hard-margin optimization problem. The theory that will allow us to do so is called Lagrange Duality.

Let's first talk about constrained optimization problems in general. Consider the following problem:

$$\min_{w} f(w)$$
$$\text{s.t. } h_i(w) = 0 \quad i = 1, \dots, l.$$

We know the method of Lagrange Multipliers can be used to solve it. We define the Lagrangian:

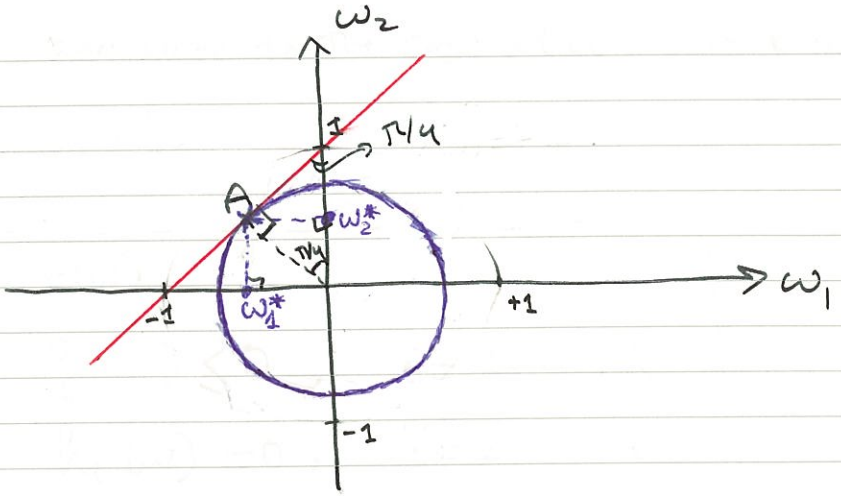
$$\mathcal{L}(w, \beta) = f(w) + \sum_{i=1}^{l} \beta_i h_i(w)$$

where β_i's are called Lagrange multipliers, and solve the following equations:

$$\text{(i)} \quad \frac{\partial \mathcal{L}}{\partial w} = 0$$

$$\text{(ii)} \quad \frac{\partial \mathcal{L}}{\partial \beta_i} = 0$$

Let's go through a simple example. Suppose we want to minimize $f(w) = w_1^2 + w_2^2$ with the constraint $w_2 = w_1 + 1$. Geometrically, what we want to achieve is the following:



We should find the circle centred at the origin with the minimum radius which still intersects $w_2 = w_1 + 1$. The coordinates of the point of intersection (w_1^*, w_2^*) are the solution to our optimization problem. It's easy to see that the length of A from origin is $1 \sin(\pi/4)$, from which it follows that $-w_1^* = w_2^* = 1 \times \sin(\pi/4) \times \cos(\pi/4) = 1/2$.

Do we get the same answer from the method of Lagrange multipliers?

we have: $f(w) = w_1^2 + w_2^2$

$h(w) = w_2 - w_1 - 1$

$\Rightarrow L = f(w) + \beta h(w) = w_1^2 + w_2^2 + \beta(w_2 - w_1 - 1)$

$\frac{\partial L}{\partial w_1} = 0 \Rightarrow 2w_1 - \beta = 0$

$\frac{\partial L}{\partial w_2} = 0 \Rightarrow 2w_2 + \beta = 0$

$\Rightarrow w_1 + w_2 = 0$

$\Rightarrow 2w_2 = 1$

$\frac{\partial L}{\partial \beta} = 0 \Rightarrow w_2 - w_1 - 1 = 0$

$\Rightarrow \boxed{-w_1 = w_2 = 1/2}$ ✓

Let's now generalize to the case where we have inequality as well as equality constraints:

$$\min_w f(w)$$
$$\text{s.t. } g_i(w) \leq 0, \quad i = 1, \dots, k$$
$$h_i(w) = 0, \quad i = 1, \dots, l.$$

If any of $f(w)$, $g_i(w)$, or $h_i(w)$ are nonlinear functions of w, this is called a non-linear programming problem. If all $f(w)$, $g_i(w)$, and $h_i(w)$ are linear functions of w, this is called a linear programming problem. The hard-margin SVM optimization problem defined on page 43 is a non-linear programming problem. Actually, when $f(w)$ is quadratic in w, and g_i & h are linear in w, it's a Quadratic Programming (QP) problem. This is the case for SVM.

Consider now the generalized Lagrangian:

$$\mathcal{L}(w, \alpha, \beta) = f(w) + \sum_{i=1}^{k} \alpha_i g_i(w) + \sum_{i=1}^{l} \beta_i h_i(x)$$

Consider also the quantity:

$$\theta_P(w) \equiv \max_{\substack{\alpha,\beta \\ \alpha_i \geq 0}} \mathcal{L}(w, \alpha, \beta)$$

Let's explain what we just wrote. First of all, the subscript "P" stands for "primal". This is because the non-linear programming problem we defined on the last page is called the primal problem in the context of Lagrange duality. For a fixed value of w, we maximize $\mathcal{L}$ over the variables α & β, subject to only positive values of α_i. In other words, for a fixed value of w, we compute $\mathcal{L}$ for all β_i & $\alpha_i \geq 0$, and assign the maximum to $\theta_P(w)$. What does this have to do with our optimization problem? Well, let's think more about the computation of $\theta_P(w)$. For a fixed value of w, $\mathcal{L}$ is linear in α_i & β_i. If we have $g_i(w) > 0$ for any i, or $h_i(w) \neq 0$ for any i, $\theta_P(w)$ would be infinite. Say $g_1(w^*) = +3$. Then we have $\mathcal{L} = +3\alpha_1 + \ldots$. Maximizing $\mathcal{L}$ in the region $\alpha_1 > 0$ gives us ∞. It then follows that:

$$\theta_P(w) = \begin{cases} f(w) & \text{if } w \text{ satisfies primal constraints} \\ \infty & \text{otherwise.} \end{cases}$$

Therefore, minimizing $\theta_P(w)$ w.r.t w is the same as our original

problem. In other words, if w^* solves our original optimization problem, we have:

$$p^* \equiv \Theta_p(w^*) = \min_w \Theta_p(w) = \min_w \max_{\substack{\alpha,\beta \\ \alpha_i \geq 0}} \mathcal{L}(w,\alpha,\beta).$$

Let's now look at a slightly different problem:

$$\Theta_D(\alpha,\beta) \equiv \min_w \mathcal{L}(w,\alpha,\beta)$$

The subscript "D" stands for "dual". The Lagrangian dual problem is the following:

$$\max_{\alpha,\beta} \Theta_D(\alpha,\beta) \quad \text{s.t. } \alpha_i \geq 0$$

Suppose α^*, β^* solves the dual problem:

$$d^* \equiv \Theta_D(\alpha^*,\beta^*) = \max_{\substack{\alpha,\beta \\ \alpha_i \geq 0}} \Theta_D(\alpha,\beta) = \max_{\substack{\alpha,\beta \\ \alpha_i \geq 0}} \min_w \mathcal{L}(w,\alpha,\beta)$$

We now see something interesting: p^* & d^* are the same except that the order of "max" & "min" operations on $\mathcal{L}$ are switched. They are related by the max-min inequality:

$$d^* = \max_{\substack{\alpha,\beta \\ \alpha_i \geq 0}} \min_w \mathcal{L}(w,\alpha,\beta) \leq \min_w \max_{\substack{\alpha,\beta \\ \alpha_i \geq 0}} \mathcal{L}(w,\alpha,\beta) = p^*$$

The max-min inequality is the following general statement:

$$\max_x \min_y f(x,y) \leq \min_y \max_x f(x,y)$$

To see why this is true, first consider the matrix version:

$p^* \equiv \min_y \max_x f_{xy}$: • for every column y, take the max of all rows. You end up with a row vector.
• Take the min of this row vector.

$d^* \equiv \max_x \min_y f_{xy}$: • for every row x, take the min of all colums. You end up with a colum vector.
• Take the max of this column vector.

Example:

$$\begin{array}{c} \\ x \downarrow \end{array} \begin{array}{c} \overset{y \rightarrow}{\begin{bmatrix} 1 & \textcircled{5} & 8 \\ 2 & 1 & 4 \\ 6 & 3 & \textcircled{2} \end{bmatrix}} \end{array}$$

$p^* = 5$

$d^* = 2$

Here's why: * 2 is less than or equal to all number in its row, which the 3rd row here ($2<3$ & $2<6$) & $2 \leq 2$)

* All numbers in the 3rd row, in turn, are less than or equal to the max # in their corresponding columns ($3<5$ & $6 \leq 6$. & $2<8$). But p^* is the max of at least one of the columns. Therefore, d^* will always be less than or equal to p^*. ($2<3$ & $3<5 \Rightarrow 2<5$).

Proof: $\min_y f(x,y) \leq f(x,y) \quad \forall x, y$ (min of a row is smaller than or equal to all #'s in that row)

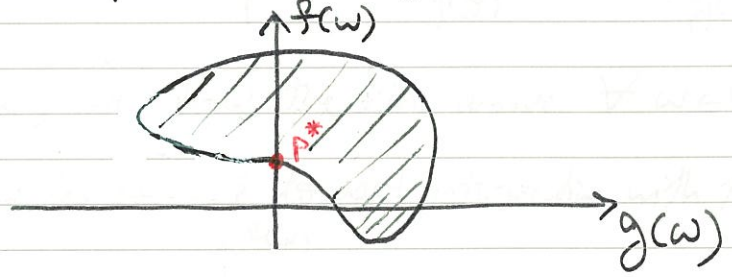
$\Rightarrow \max_x \min_y f(x,y) \leq \max_x f(x,y) \quad \forall y$ (max of all mins of rows is $\leq$ max of all columns)

$\Rightarrow \max_x \min_y f(x,y) \leq \min_y \max_x f(x,y)$ (max of all mins of rows is $\leq$ min of max of all columns)

Lagrange duality has a nice geometric interpretation. Consider the following primal problem:

$$\begin{aligned} &\underset{w}{\text{minimize}} \quad f(w) \\ &\text{s.t.} \quad g(w) \leq 0 \qquad w \in W \rightarrow \text{domain of } w \end{aligned}$$

For every $w \in W$, compute $f(w)$ & $g(w)$, and plot it as follows:



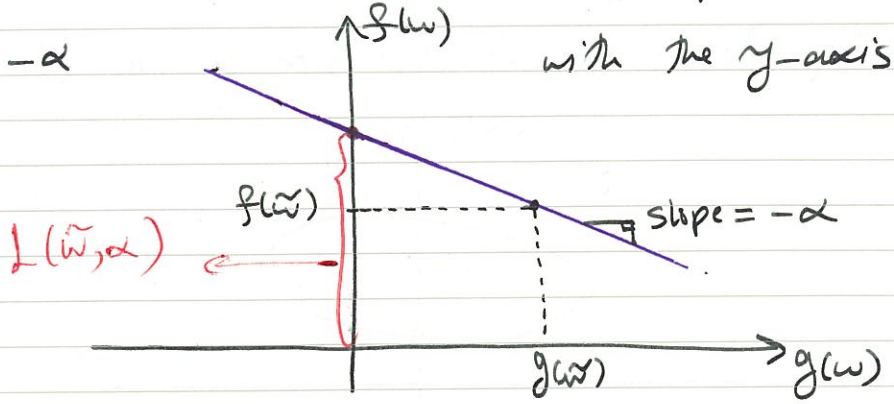
To solve the primal problem, we are looking for the minimum of $f(w)$ when $g(w)$ is negative or zero. In the figure above, this corresponds the red point, since there $f(w)$ takes its minimum in the $g \leq 0$ region. In fact, the distance of this point from the x-axis, i.e its y coordinate, is p^*.

Consider now the dual problem:

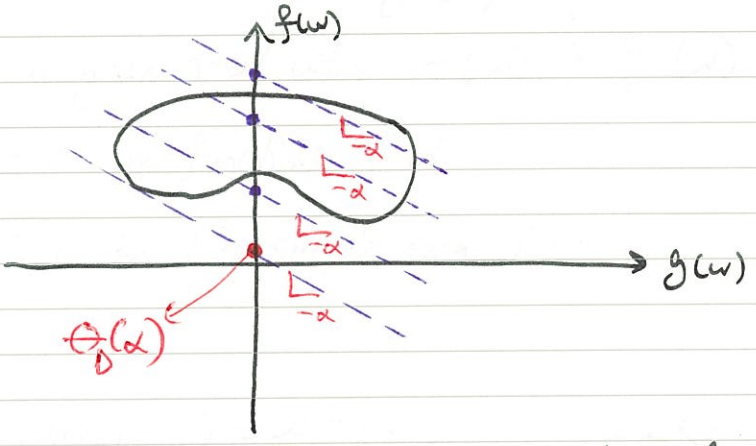
$$\begin{aligned} &\underset{\alpha}{\text{maximize}} \quad \theta_D(\alpha) \qquad \text{where } \theta_D(\alpha) = \min_w \{f(w) + \alpha g(w)\} \\ &\text{s.t.} \quad \alpha \geq 0 \end{aligned}$$

How can we think about $\theta(\alpha)$ geometrically? For a given α, we need to compute $f(w) + \alpha g(w)$ $\forall$ $w \in W$ & take the minimum. Consider first $h(w, \alpha) = f(w) + \alpha g(w)$. Given $\tilde{w}$ & α, $h(\tilde{w}, \alpha)$

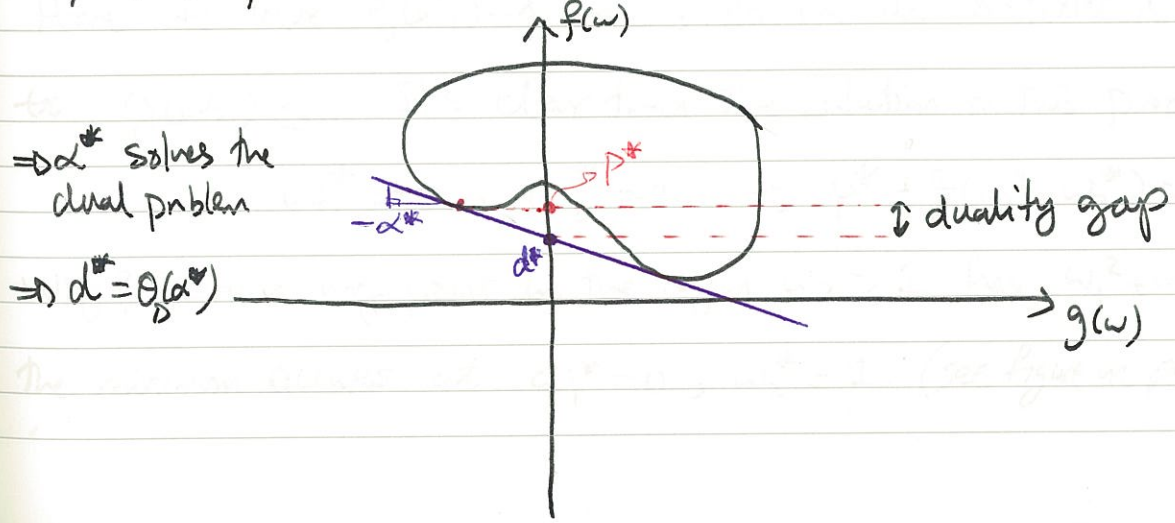
is actually the intersection of the line which passes through $(g(\tilde{w}), f(\tilde{w}))$ with slope $-\alpha$ with the y-axis.



Therefore, give α, we draw the line above $\forall w \in W$, & pick the one which results in the lowest intersection with the y-axis:

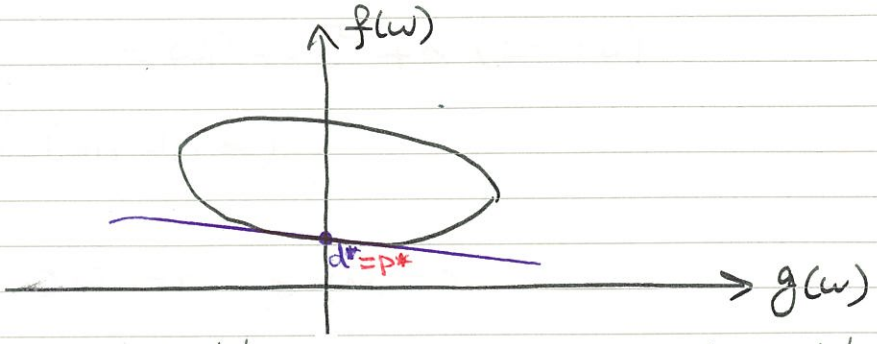


We then repeat this for all positive or zero values of α, i.e different slopes, & pick the maximum $\Theta(\alpha)$:



Are there cases where the dual problem is equivalent to the primal problem? In other words, can we have situations where $d^* = p^*$?

Yes:



This is called strong duality, as opposed to weak duality, which is the general case where a duality gap exists. As it turns out, strong duality is guaranteed when $f(w)$ & $g_i(w)$'s are all convex, and h_i's are linear functions of w.

Let's verify this in two simple examples:

Example #1

$$\min_{w_1, w_2} f(w_1, w_2) = w_1^2 + w_2^2 \qquad -2 \leq w_1, w_2 \leq 2$$
$$\text{s.t.} \quad w_2 \geq 1$$

Here we have $g(w_2) = 1 - w_2$, so that our constraint is equivalent to $g(w_2) \leq 0$. It's clear that the solution to this problem is:

$$w_1^* = 0, \; w_2^* = 1 \Rightarrow p^* = f(w_1^*, w_2^*) = 1.$$

Why? Because any point in the region $w_2 \geq 1$ has $w_1^2 + w_2^2 \geq 1$ and the minimum occurs at $w_1^* = 0$, $w_2^* = 1$. (see figure on page 59.)

Let's now solve the dual problem:

$$L(w_1, w_2, \alpha) = f(w_1, w_2) + \alpha\, g(w_1, w_2)$$
$$= w_1^2 + w_2^2 + \alpha(1 - w_2)$$

$$\theta_D(\alpha) = \min_{w_1, w_2} L(w_1, w_2, \alpha)$$

$$\frac{\partial L}{\partial w_1} = 0 \Rightarrow 2w_1 = 0 \Rightarrow w_1 = 0$$

$$\frac{\partial L}{\partial w_2} = 0 \Rightarrow 2w_2 - \alpha = 0 \Rightarrow \boxed{w_2 = \frac{\alpha}{2}} \quad (\#)$$

$$\theta_D(\alpha) = \left(\frac{\alpha}{2}\right)^2 + \alpha\left(1 - \frac{\alpha}{2}\right) = \frac{\alpha^2}{4} + \alpha - \frac{\alpha^2}{2} = -\frac{\alpha^2}{4} + \alpha$$

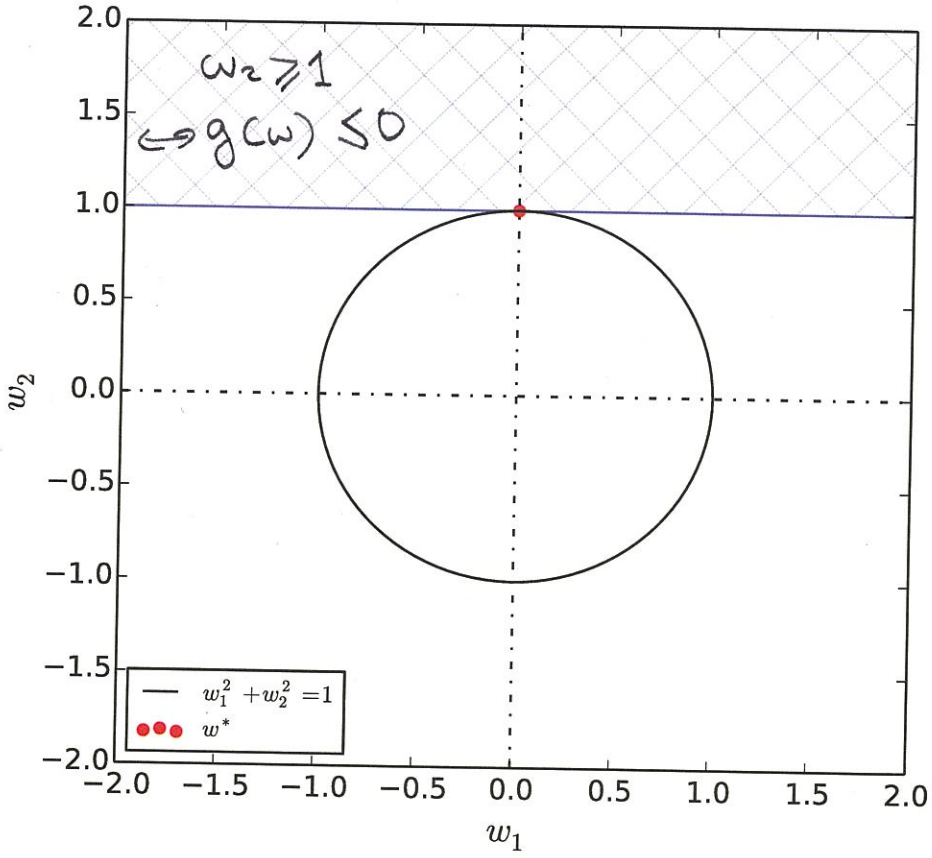
Now we maximize α:

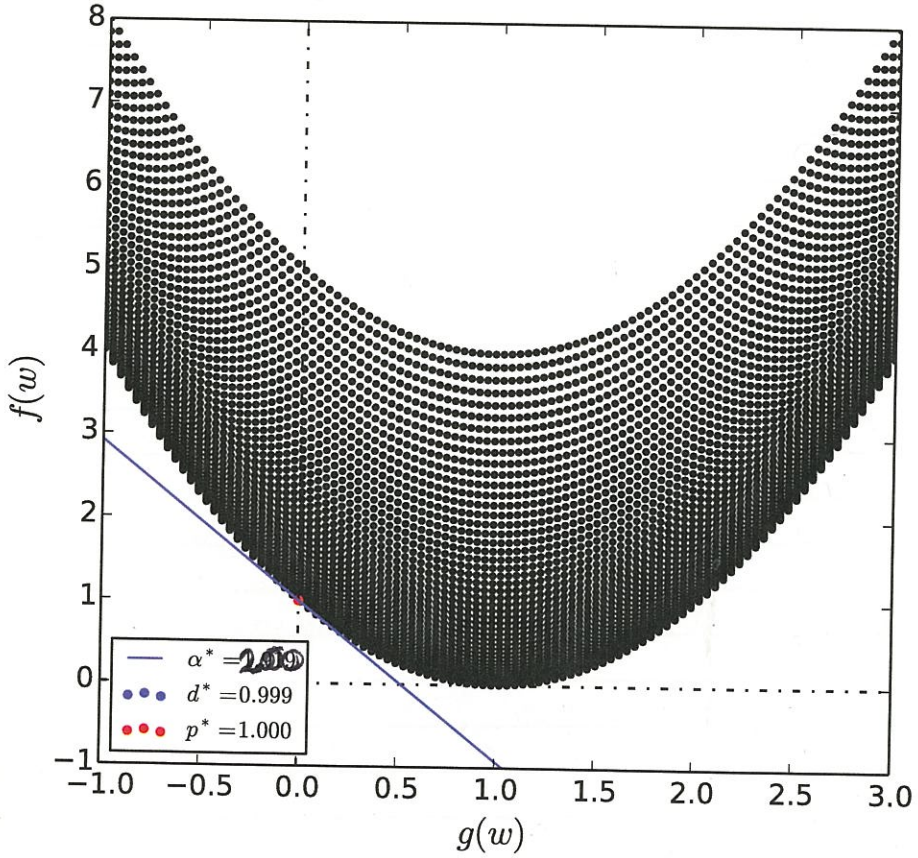
$$\frac{d\theta_D}{d\alpha} = 0 \Rightarrow -\frac{\alpha}{2} + 1 = 0 \Rightarrow \boxed{\alpha^* = 2}$$

$$d^* = \theta_D(\alpha^*) = 1 \Rightarrow \boxed{d^* = p^*}$$

Also, we have that $w_2^* = \frac{\alpha^*}{2}$, as can be verified using (#).

We see that strong duality holds in this case, since f is convex and so is g. This is precisely the situation for SVM! We will use Lagrange duality to solve the SVM optimization problem. The next page shows the geometric interpretation of Lagrange duality for this problem.


$w_2 \geq 1$
$\Leftrightarrow g(w) \leq 0$
$w_1^2 + w_2^2 = 1$
w^*
w_1
w_2


$\alpha^* = 2.999$
$d^* = 0.999$
$p^* = 1.000$
$g(w)$
$f(w)$

Let's now consider the same problem, except with a non-convex constraint:

Example #2

$$\min_{w_1, w_2} f(w_1, w_2) = w_1^2 + w_2^2 \qquad -2 \leq w_1, w_2 \leq +2$$
$$\text{s.t. } w_2 \geq -\frac{1}{2} w_1^4 + w_1^2 + 1$$

We have: $g(w_1, w_2) = -w_2 - \frac{1}{2} w_1^4 + w_1^2 + 1$

The primal problem has the exact same solution: (see page 62)

$$w_1^* = 0, \quad w_2^* = 1, \quad p^* = f(w_1^*, w_2^*) = 1$$

Let's now look at the dual problem:

$$h(w_1, w_2, \alpha) = f(w_1, w_2) + \alpha g(w_1, w_2)$$
$$= w_1^2 + w_2^2 + \alpha(-w_2 - \tfrac{1}{2} w_1^4 + w_1^2 + 1)$$

$$\frac{\partial h}{\partial w_2} = 2w_2 - \alpha = 0 \Rightarrow \boxed{w_2 = \frac{\alpha}{2}}$$

$$\frac{\partial h}{\partial w_1} = 2w_1 - 2\alpha w_1^3 + 2\alpha w_1 = 0$$
$$\Rightarrow 2w_1[1 + \alpha - \alpha w_1^2] = 0 \Rightarrow \begin{cases} w_1 = 0 \\ w_1 = \pm\sqrt{\frac{1+\alpha}{\alpha}} \end{cases}$$

$\pm\sqrt{\frac{1+\alpha}{\alpha}}$ are actually local maxima, so we don't care about them.

$w_1 = 0$ is not always a global minimum, though. For large enough values of α, the minima of h are on the boundaries. Let's see when this the case.

Local minimum: $\begin{cases} w_1 = 0 \\ w_2 = \frac{\alpha}{2} \end{cases} \Rightarrow h = \frac{\alpha^2}{4} - \frac{\alpha^2}{2} + \alpha = -\frac{\alpha^2}{4} + \alpha$

At boundary: $\begin{cases} w_1 = \pm 2 \\ w_2 = \frac{\alpha}{2} \end{cases} \Rightarrow h = 4 + \frac{\alpha^2}{4} + \alpha\left(-\frac{\alpha}{2} - 8 + 4 + 1\right)$

$= -\frac{\alpha^2}{4} + 4 - 3\alpha$

$-\frac{\alpha^2}{4} + 4 - 3\alpha \overset{?}{\leq} -\frac{\alpha^2}{4} + \alpha \quad \Rightarrow \alpha \geq 1.$

Therefore $\theta_D(\alpha) = \begin{cases} -\frac{\alpha^2}{4} + \alpha & 0 \leq \alpha \leq 1 \\ -\frac{\alpha^2}{4} - 3\alpha + 4 & \alpha \geq 1 \end{cases}$

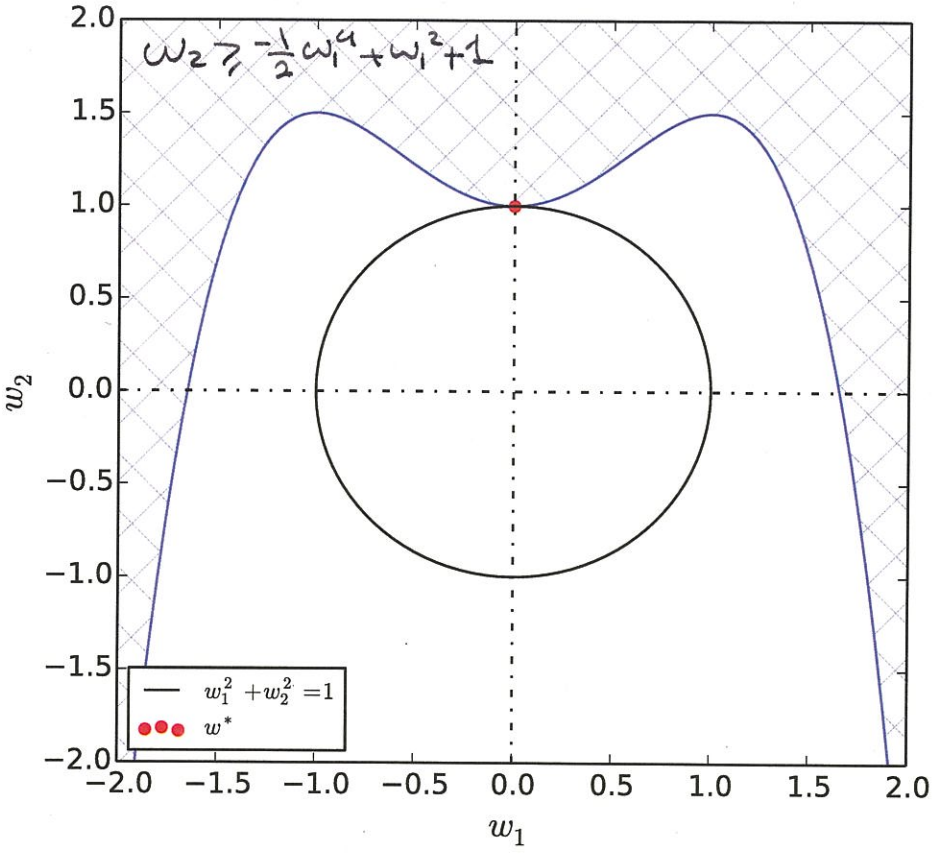
It can be checked that $\theta_D(\alpha)$ takes its maximum at $\alpha = 1$:

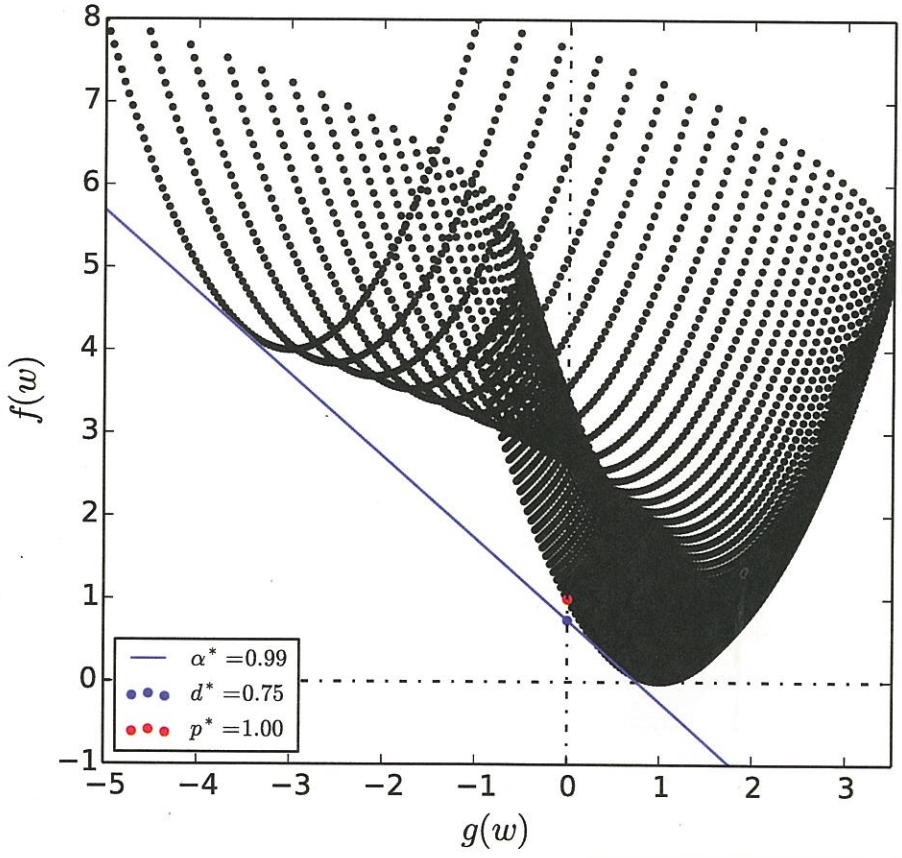
$$\alpha^* = 1 \Rightarrow d^* = \theta_D(\alpha^*) = \frac{3}{4}$$

$$d^* < p^*$$

Strong duality does not hold in this & there's a duality gap of $1/4$.

Next page shows the geometric interpretation. Note how important the domain of w was in this problem. Perhaps this isn't terribly surprising, since minimization is a global problem.


$w_2 \geq -\frac{1}{2}w_1^4 + w_1^2 + 1$
$w_1^2 + w_2^2 = 1$
w^*
w_1
w_2


$\alpha^* = 0.99$
$d^* = 0.75$
$p^* = 1.00$
$g(w)$
$f(w)$

Let's state the conditions for strong duality more precisely:

* f and g_i's are convex
* h_i's are affine, i.e. $h_i(w) = a_i^T w + b_i$
* Constraints g_i are feasable, ie. $\exists\, w$ s.t $g_i(w) < 0\ \forall i$.

Then: there must exist w^*, α^*, β^* so that w^* is the solution to the primal problem, α^* & β^* are the solution to the dual problem, and moreover $p^* = d^* = L(w^*, \alpha^*, \beta^*)$. Also, w^*, α^* and β^* satisfy the Karush-Kuhn-Tucker (KKT) conditions, which are as follows:

(i) $\frac{\partial L}{\partial w_i}(w^*, \alpha^*, \beta^*) = 0$ $\quad i = 1, \ldots, n$

(ii) $\frac{\partial L}{\partial \beta_i}(w^*, \alpha^*, \beta^*) = 0$ $\quad i = 1, \ldots, l$

(iii) $\alpha_i^* g_i(w^*) = 0$ (dual-complementarity conditions) $\quad i = 1, \ldots, k$

(iv) $g_i(w^*) \leq 0$ $\quad i = 1, \ldots, k$

(v) $\alpha_i^* \geq 0$ $\quad i = 1, \ldots, k$

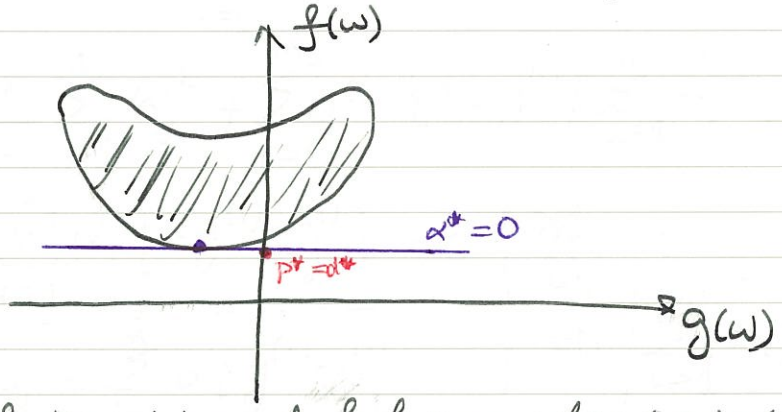
Let's go over these one by one. Of course, (iv) and (v) ought to be true if there's any solution at all, by definition. Also, (ii) just means that $h_i(w^*) = 0\ \forall i$, which again should be the case if there's any solution w^*. The interesting conditions are (i) & (iii). In the strong duality

example we worked out, we saw that to solve the dual problem, we solve $\frac{\partial h}{\partial w_i} = 0$ to express the minima of h w.r.t w, in terms of α_i & β_i. Once α_i^* & β_i^* are found by maximizing $\Theta_D(\alpha, \beta)$, with the constraints $\alpha_i \geq 0$, w_i^* can be easily calculated from α_i^* & β_i^*, and of course $\frac{\partial h}{\partial w_i}(w^*, \alpha^*, \beta^*) = 0$.

Condition (iii) is very curious:

$$\alpha_i^* g_i(w^*) = 0 \qquad \forall\ i$$

It's saying that if either $g_i(w^*)$ is strictly negative, α_i^* will always be zero. Why should this be? Let's go back to our geometric interpretation, and look at a case where $g_i(w^*) < 0$.



We see that if the minimum of f happens when $g(w) < 0$, and there's also strong duality, there's really no other way but to have $\alpha^* = 0$, i.e the linear line defining $\Theta_D(\alpha^*)$ is always horizontal.

Let's now get back to the hard-margin optimization of SVM. The primal problem is as follows (see page 43):

$$\min_{w,b} f(w,b) = \frac{1}{2}\sum_{j=1}^{n} w_j^2$$

$$\text{s.t. } g_i(w,b) = 1 - y^{(i)}\left[\sum_{j=1}^{n} w_j x_j^{(i)} + b\right] \leq 0 \quad \forall\, i = 1,\dots,m.$$

We have an inequality constraint for every training example. Since f & g_i's are convex, we can use Lagrange duality to solve the primal problem, assuming ofcourse that the two classes $y^{(i)} = \pm 1$ are linearly separable to begin with.

$$\mathcal{L}(w,b,\alpha) = f(w,b) + \sum_{i=1}^{m} \alpha_i g_i(w,b)$$

$$= \frac{1}{2}\sum_{j=1}^{n} w_j^2 + \sum_{i=1}^{m} \alpha_i \left\{1 - y^{(i)}\left[\sum_{j=1}^{n} w_j x_j^{(i)} + b\right]\right\}$$

$$\frac{\partial \mathcal{L}}{\partial w_j} = w_j + \sum_{i=1}^{m} \alpha_i \left\{-y^{(i)} x_j^{(i)}\right\} = 0$$

$$\Rightarrow \boxed{w_j = \sum_{i=1}^{m} \alpha_i y^{(i)} x_j^{(i)}} \quad (*)$$

$$\frac{\partial \mathcal{L}}{\partial b} = 0 \Rightarrow \boxed{\sum_{i=1}^{m} \alpha_i y^{(i)} = 0} \quad (\#)$$

$$\theta_D(\alpha) = \frac{1}{2}\sum_{j=1}^{n}\left(\sum_{i=1}^{m} \alpha_i y^{(i)} x_j^{(i)}\right)^2 + \sum_{i=1}^{m} \alpha_i \left\{1 - y^{(i)}\left[\sum_{j=1}^{n} x_j^{(i)}\left(\sum_{i=1}^{m} \alpha_i y^{(i)} x_j^{(i)}\right) + b\right]\right\}$$

$$\theta_D(\alpha) = \frac{1}{2}\sum_{i,i'=1}^{m} \alpha_i \alpha_{i'} y^{(i)} y^{(i')} \left(\sum_{j=1}^{n} x_j^{(i)} x_j^{(i')}\right)$$

$$+ \sum_{i=1}^{m} \alpha_i$$

$$- \sum_{i,i'=1}^{m} \alpha_i \alpha_{i'} y^{(i)} y^{(i')} \left(\sum_{j=1}^{n} x_j^{(i)} x_j^{(i')}\right)$$

$$- b \sum_{i=1}^{m} \alpha_i y^{(i)}$$

Using ⊞ on the previous page, the last term drops out:

$$\theta_D(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,i'=1}^{m} \alpha_i \alpha_{i'} y^{(i)} y^{(i')} \langle x^{(i)}, x^{(i')} \rangle$$

The dual optimization problem is then:

$$\max_{\alpha} \theta_D(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,i'=1}^{m} \alpha_i \alpha_{i'} y^{(i)} y^{(i')} \langle x^{(i)}, x^{(i')} \rangle \quad (\#)$$

$$\text{s.t.} \quad \alpha_i \geq 0 \qquad i = 1, \ldots, m$$

$$\sum_{i=1}^{m} \alpha_i y^{(i)} = 0$$

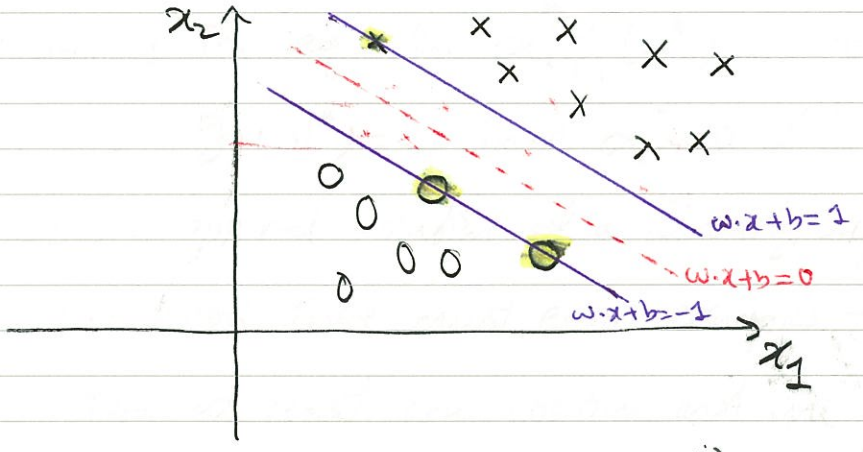
Once we solve this problem, ⊛ on the previous page can be used to find the optimal values of w:

$$w^* = \sum_{i=1}^{m} \alpha_i^* y^{(i)} x^{(i)} \quad (*)$$

We know from KKT condition (iii) on page 63 that α_i^* are only

non-zero when $g_i(w^*) = 0$, which is precisely the equation that defines the support vectors: w^* is determined entirely by the support vectors. This confirms our geometric interpretation.

Once we have w^*, what's b^*? Let's revisit the geometric picture:



The $y=1$ support vectors satisfy $w \cdot x^{(i)}_{support} \leq w \cdot x^{(i)} \quad \forall\, y^{(i)} = 1$

The $y=0$ support vectors satisfy $w \cdot x^{(i)}_{support} \geq w \cdot x^{(i)} \quad \forall\, y^{(i)} = -1$

We can then fix b^* as follows:

$$\min_{i \,:\, y^{(i)}=1} w^* \cdot x^{(i)} + b^* = 1 \Rightarrow \boxed{b^* = 1 - \min_{i \,:\, y^{(i)}=1} w^* \cdot x^{(i)}}$$

or $$\max_{i \,:\, y^{(i)}=-1} w^* \cdot x^{(i)} + b^* = -1 \Rightarrow \boxed{b^* = -1 - \max_{i \,:\, y^{(i)}=-1} w^* \cdot x^{(i)}}$$

or combining the two:

$$\boxed{b^* = -\frac{1}{2}\left[\min_{i \,:\, y^{(i)}=1} w^* \cdot x^{(i)} + \max_{i \,:\, y^{(i)}=-1} w^* \cdot x^{(i)}\right]}$$

This derivation is based on our geometric intuition, though. How do the dual Lagrange equations (##) on page 66 know about this? Can we derive b^* strictly from those equations?

First of all, how are we sure there are examples for which $g_i(w^*)=0$? The reason is the equation $\alpha_i^* g_i(w^*)=0 \ \forall i$. If there's no i for which $g_i(w^*)=0$, then $\alpha_i^*=0 \ \forall i$ & $w^*=0$, which cannot be the optimal solution of a linearly separable data set. So now we know there must exist examples for which $g_i(w^*)=0$. But is there at least one positive and one negative example for which this holds? (In other words, how does the optimization problem know there should be at least one positive and one negative support vector?) The key is equation $\sum_{i=1}^{m} \alpha_i^* y^{(i)} = 0$ (*). We know that $\alpha_i^* \neq 0$ only when $g_i(w^*)=0$. Let's say there is only one example, call it i^*, for which $g_{i^*}(w^*)=0$. But (*) would imply $\alpha_{i^*}^* = 0$, so this can't be. How about we have two such examples, but they're both either positive or negative? (*) would imply $\alpha_{i_1^*}^* + \alpha_{i_2^*}^* = 0$. This means at least one of them has to be negative, which is against the constraint $\alpha_i^* \geq 0$. So there must be at least one positive AND negative example for which $g_i(w^*)=0$. Nice! This will lead to

the same derivation of b^* as page 67.

The dual optimization problem can be cast entirely in terms of the inner-products $\langle x^{(i)}, x^{(j)} \rangle$. This is apparent from (#) on page 66. But how about the prediction itself? For a new example x:

$$w^{*T} x + b^* = \sum_{i=1}^{m} \alpha_i^* y^{(i)} \langle x^{(i)}, x \rangle + b^*$$

where we have used (*) on page 66. Same goes with b^*:

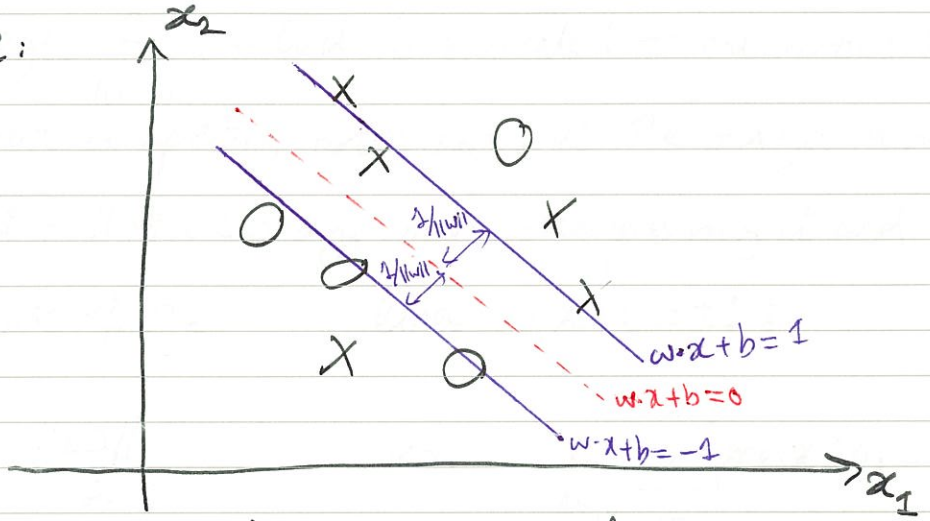
$$b^* = 1 - \min_{y^{(i)}=1} w^* \cdot x^{(i)}$$

$$= 1 - \min_{y^{(i)}=1} \sum_{i=1}^{m} \alpha_i^* y^{(i)} \langle x^{(i)}, x^{(i)} \rangle$$

This is remarkable. The dual optimization problem solves for α_i. There are as many α_i's as training examples. The dimension of our features only enters through the inner-products $\langle x, z \rangle$. So even if we have really high dimensional features, as long as we can compute $\langle x, z \rangle$ efficiently, we'll have an algorithm that scales extremely mildly with the dimension "n" of our features. This was not the case for linear & logistic regression. As we argued in Lecture 4 (see Notebook #1), including high order terms makes makes linear & logistic regression intractable. This is not the case with SVM. We'll come back to this when we

discuss kernels.

Let's work out the dual optimization problem for the soft-margin primal problem. Let's revisit our formulation on pages 44-45 from a more geometric angle. Consider a data set that's not linearly separable:



Pick a unit vector $\hat{w}$ & b. This defines a plane. By picking $\|w\|$, we control the distance between $w\cdot x+b=0$ & $w\cdot x+b=\pm 1$. Let's work out the cost associated with an example being on the wrong side:

$$y^{(i)}=1: \begin{cases} \text{if } w\cdot x^{(i)}+b\geq 1: & \text{cost}=0 \\ \text{if } w\cdot x^{(i)}+b<1: & \text{cost}=C\times \text{distance of example to line } w\cdot x^{(i)}+b=1 \end{cases}$$

$$= C\times \frac{w\cdot x^{(i)}+b-1}{\|w\|}$$

$$y^{(i)}=-1: \begin{cases} \text{if } w\cdot x^{(i)}+b\leq -1: & \text{cost}=0 \\ \text{if } w\cdot x^{(i)}+b>-1: & \text{cost}=C\times \text{distance of example to line } w\cdot x^{(i)}+b=1 \end{cases}$$

$$= C\times (w\cdot x^{(i)}+b+1)/\|w\|$$

C is a positive number which controls how much we want to penalize examples on the wrong side. We can write a more concise expression:

$$\text{Cost of being on wrong side} = C \times \max\left(0, \frac{1 - y^{(i)}(w \cdot x^{(i)} + b)}{\|w\|}\right)$$

While we'd want to minimize this cost, we'd want to maximize the margin $\frac{2}{\|w\|}$, which is equivalent to minimizing $\frac{\|w\|}{2}$. So we have two competing forces here: as the margin increases, which is equivalent to $\|w\|$ decreasing, more examples might end up on the wrong side of the lines $w \cdot x + b = \pm 1$:

$$\frac{\|w\|}{2} \quad \text{vs.} \quad \frac{C}{\|w\|} \sum_{i=1}^{m} \max\left(0, 1 - y^{(i)}(w \cdot x^{(i)} + b)\right)$$

Equivalently we can compare:

$$\frac{\|w\|^2}{2} \quad \text{vs.} \quad C \sum_{i=1}^{m} \max\left(0, 1 - y^{(i)}(w \cdot x^{(i)} + b)\right)$$

Which leads us to the objective function on page 45:

$$J(w, b) = \frac{1}{2}\|w\|^2 + C \sum_{i=1}^{m} \max\left(0, 1 - y^{(i)}(w \cdot x^{(i)} + b)\right)$$

The higher C, the more we penalize examples on the wrong side, and the more our algorithm tries to keep the two classes separate. As we argued before, $C \to \infty$ recovers the hard-margin case.

We can rewrite the soft-margin optimization in a way that makes it suitable for the use of Lagrange duality.

Let $\xi_i \equiv \max(0, 1 - y^{(i)}(w \cdot x^{(i)} + b))$

of course: * $\xi_i \geq 0$

* $\xi_i \geq 1 - y^{(i)}(w \cdot x^{(i)} + b)$

So we can rewrite the optimization problem as follows:

$$\min_{w,b,\xi} J(w,b,\xi) = \frac{1}{2}\sum_{j=1}^{n} w_j^2 + C\sum_{i=1}^{m}\xi_i$$

$$\text{s.t.} \quad g_i(w,b,\xi) = 1 - y^{(i)}(w \cdot x^{(i)} + b) - \xi_i \leq 0 \quad \forall i = 1, \dots, m$$

$$\tilde{g}_i(w,b,\xi) = -\xi_i \leq 0 \quad \forall i = 1, \dots, m$$

Just like the hard-margin case, we can use the dual problem to solve the primal problem, because J, g_i & $\tilde{g}_i$ are convex.

$$\mathcal{L}(w,b,\xi,\alpha,\tilde{\alpha}) = \frac{1}{2}\sum_{j=1}^{n} w_j^2 + C\sum_{i=1}^{m}\xi_i + \sum_{i=1}^{m}\alpha_i\left[1 - y^{(i)}(w \cdot x^{(i)} + b) - \xi_i\right] - \sum_{i=1}^{m}\tilde{\alpha}_i\xi_i$$

$$\frac{\partial \mathcal{L}}{\partial w_j} = 0 \Rightarrow \boxed{w_j = \sum_{i=1}^{m}\alpha_i y^{(i)} x_j^{(i)}}$$

$$\frac{\partial h}{\partial b} = 0 \Rightarrow \boxed{\sum_{i=1}^{m} \alpha_i y^{(i)} = 0} \quad (\#)$$

$$\frac{\partial h}{\partial \xi_i} = 0 \Rightarrow \boxed{C - \alpha_i - \tilde{\alpha}_i = 0} \quad (*)$$

Let's plug these back into the Lagrangian to get $\Theta_D(\alpha, \tilde{\alpha})$:

$$\Theta_D(\alpha, \tilde{\alpha}) = \frac{1}{2} \sum_{j=1}^{n} \left(\sum_{i=1}^{m} \alpha_i y^{(i)} x_j^{(i)} \right)^2$$

$$+ \sum_{i=1}^{m} \xi_i (C - \alpha_i - \tilde{\alpha}_i) + \sum_{i=1}^{m} \alpha_i + b \sum_{i=1}^{m} \alpha_i y^{(i)}$$

$$- \sum_{i=1}^{m} \alpha_i y^{(i)} \sum_{j=1}^{n} \left(\sum_{i'=1}^{m} \alpha_{i'} y^{(i')} x_j^{(i')} \right) x_j^{(i)}$$

$$= \frac{1}{2} \sum_{i,i'=1}^{m} \alpha_i \alpha_{i'} y^{(i)} y^{(i')} \left(\sum_{j=1}^{n} x_j^{(i)} x_j^{(i')} \right) + \sum_{i=1}^{m} \alpha_i$$

$$- \sum_{i,i'=1}^{m} \alpha_i \alpha_{i'} y^{(i)} y^{(i')} \left(\sum_{j=1}^{n} x_j^{(i)} x_j^{(i')} \right)$$

$$= \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,i'=1}^{m} \alpha_i \alpha_{i'} y^{(i)} y^{(i')} \langle x^{(i)}, x^{(i')} \rangle$$

The dual problem also requires that $\alpha_i \geq 0$ & $\tilde{\alpha}_i \geq 0$. Since we can solve $\tilde{\alpha}_i$ in terms of α_i using $(*)$:

$$\tilde{\alpha}_i = C - \alpha_i \geq 0 \iff \alpha_i \leq C$$

The dual problem then reads:

$$\max_{\alpha} \Theta_D(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,i'=1}^{m} \alpha_i \alpha_{i'} y^{(i)} y^{(i')} \langle x^{(i)}, x^{(i')} \rangle$$

$$\text{s.t. } 0 \leq \alpha_i \leq C \quad \forall i = 1, \cdots, m \;, \quad \sum_{i=1}^{m} \alpha_i y^{(i)} = 0$$

This looks exactly the same as the hard-margin dual problem (see page 66), except that α_i are bounded by C.

What do the dual-complementarity conditions tell us?

$$\textcircled{*}\ \tilde{\alpha}_i^* \tilde{g}_i(w^*, b^*, \xi^*) = 0 \qquad \forall\ i = 1, \dots, m$$

$$\Rightarrow \boxed{\tilde{\alpha}_i^* \xi_i^* = 0} \qquad \forall\ i = 1, \dots, m$$

Since we know $\tilde{\alpha}_i$ in terms of α_i:

$$\tilde{\alpha}_i^* \xi_i^* = (C - \alpha_i^*)\xi_i^* = 0$$

Therefore, $\xi_i^* \neq 0$ only when $\alpha_i^* = C$. We know from the geometric picture that $\xi_i^* \neq 0$ only when there's an example on the wrong side, so we should be able to prove that $\alpha_i^* = C$ for all examples on the wrong side.

$$\textcircled{*}\ \alpha_i^* g_i(w^*, b^*, \xi^*) = 0 \qquad \forall\ i = 1, \dots, m$$

$$\Rightarrow \alpha_i^* \left(1 - y^{(i)}(w^* \cdot x^{(i)} + b^*) - \xi_i^*\right) = 0$$

Let's consider three different cases:

(i) $\alpha_i^* = 0$: From above result we have that $\xi_i^* = 0$.

$$g_i \leq 0 \Rightarrow y^{(i)}(w^* \cdot x^{(i)} + b^*) \geq 1$$

(ii) $\alpha_i^* = C$: $g_i = 0 \Rightarrow y^{(i)}(w^* \cdot x^{(i)} + b^*) = 1 - \xi_i^* \leq 1$

(ii) $0 < \alpha_i^* < C$: $g_i = 0$ & $\xi_i^* = 0 \Rightarrow y^{(i)}(w^* \cdot x^{(i)} + b^*) = 1$

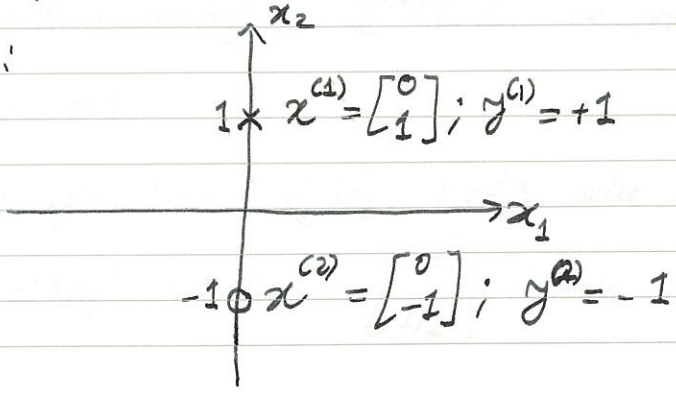
Let's summarize the dual-complementarity consequences:

- $\alpha_i^* = 0 \Rightarrow y^{(i)}(w^* \cdot x^{(i)} + b^*) \geq 1$ (no margin violation)
- $\alpha_i^* = C \Rightarrow y^{(i)}(w^* \cdot x^{(i)} + b^*) \leq 1$ (margin violation)
- $0 < \alpha_i^* < C \Rightarrow y^{(i)}(w^* \cdot x^{(i)} + b^*) = 1$ (support vectors)

This is pretty interesting. For all margin-violating examples: $\alpha_i^* = C$. Note that in these cases $\xi_i^* = \max(0, 1 - y^{(i)}(w^* \cdot x^{(i)} + b^*))$. For non-support non-margin violating examples: $\alpha_i^* = 0$. Support vectors take on values of α_i in between 0 & C. We can use them to fix b^*:

$$b^* = y^{(i)} - w^* \cdot x^{(i)} \text{ for any } i \text{ where } 0 < \alpha_i < C$$

Is it possible to have a case where for all examples either $\alpha_i^* = 0$ or $\alpha_i^* = C$, in which case we wouldn't be able to solve for b^*? In principle, if there are an equal number of margin-violating examples from both classes, $\sum_{i=1}^{m} \alpha_i^* y^{(i)} = 0$ can be satisfied. So, yes, that scenario is possible. In fact, here's a concrete (but very unrealistic) example:



$\langle x^{(1)}, x^{(1)} \rangle = \langle x^{(2)}, x^{(2)} \rangle = 1$ & $\langle x^{(1)}, x^{(2)} \rangle = \langle x^{(2)}, x^{(1)} \rangle = -1$.

$$\mathcal{O}_D(\alpha_1, \alpha_2) = \alpha_1 + \alpha_2 - \frac{1}{2}\left[\alpha_1\alpha_1 + \alpha_1\alpha_2 + \alpha_2\alpha_1 + \alpha_2\alpha_2\right]$$

$$\sum_{i=1}^{m} \alpha_i y^{(i)} = 0 \Rightarrow \alpha_1 - \alpha_2 = 0 \Rightarrow \alpha_1 = \alpha_2 \equiv \alpha$$

$$\Rightarrow \begin{cases} \mathcal{O}_D(\alpha) = 2\alpha - \frac{1}{2} \times 4\alpha^2 = 2(\alpha - \alpha^2) \\ 0 \le \alpha \le C \end{cases}$$

$$\alpha^* = \begin{cases} +1/2 & C \ge 1/2 \\ C & C < 1/2 \end{cases}$$

Consider now the case where $C > 1/2$, where $\alpha^* = 1/2$.

$$w^* = \alpha_1^* y^{(1)} x^{(1)} + \alpha_2^* y^{(2)} x^{(2)}$$

$$= \frac{1}{2}\begin{bmatrix} 0 \\ 1 \end{bmatrix} - \frac{1}{2}\begin{bmatrix} 0 \\ -1 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \rightarrow w^* = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

Since $0 < \alpha_1 < C \Rightarrow b^* = y^{(1)} - w^* \cdot x^{(1)}$

$$= 1 - \begin{bmatrix} 0 \\ 1 \end{bmatrix} \cdot \begin{bmatrix} 0 \\ 1 \end{bmatrix} = 0$$

$w^* \cdot x + b = \pm 1 \leftrightarrow x_2 = \pm 1$. This is what we expect: $x_2 = 0$ is our decision boundary & $x^{(1)}$ & $x^{(2)}$ are maximally separated & are support vectors.

But now consider $C \le 1/2$. Let's take $C = 1/4$ as an example. Then $\alpha^* = 1/4 \Rightarrow w^* = \begin{bmatrix} 0 \\ 1/2 \end{bmatrix}$. Since $\alpha^* \not< C$, we cannot solve

for b^* exactly. We do have the inequality:

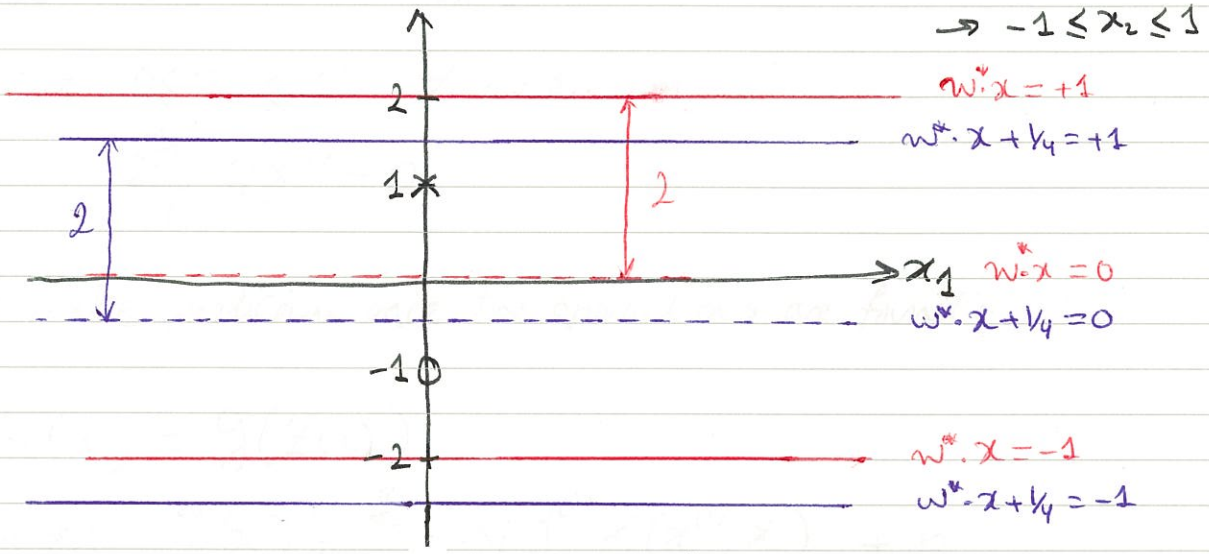
$$y^{(i)}(w^* \cdot x^{(i)} + b^*) \geq 1$$

$i=1$: $1\left(\begin{bmatrix}0\\ 1/2\end{bmatrix} \cdot \begin{bmatrix}0\\ 1\end{bmatrix} + b^*\right) \geq 1 \Rightarrow b^* \geq 1/2$

$i=2$: $-1\left(\begin{bmatrix}0\\ 1/2\end{bmatrix} \cdot \begin{bmatrix}0\\ -1\end{bmatrix} + b^*\right) \geq 1 \Rightarrow b^* \leq -1/2$

So we have: $-\frac{1}{2} \leq b^* \leq 1/2$. $w^* \cdot x + b = 0 \rightarrow \frac{x_2}{2} + b = 0$

$\rightarrow -1 \leq x_2 \leq 1$



Here I've plotted two cases: $b=0$ (red) & $b=1/4$ (blue). Why doesn't the optimization problem distinguish between the two? Well, because the objective function has the exact same value in both cases:

$$\begin{cases} b=0: & \xi_1 = 1/2,\ \xi_2 = 1/2 \Rightarrow (\xi_1 + \xi_2) \times C = 1/4 \\ b=1/4: & \xi_1 = 1/4,\ \xi_2 = 3/4 \Rightarrow (\xi_1 + \xi_2) \times C = 1/4 \end{cases}$$

The penalty of our positive example decreases when b is $1/4$, but the penalty of the negative example increases by the exact same amount. This symmetry only holds when $-1/2 \leq b^* \leq 1/2$. Cute!

Lecture 74: Kernels I ⊕ Lecture 75: Kernels II

As we've seen, and pointed out on page 69, SVM's optimization can be written entirely in terms of the inner-products $\langle x, z \rangle$ of input features.

Let's summarize the optimization problem below: (see page 73)

$$\max_{\alpha} \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,i'=1}^{m} \alpha_i \alpha_{i'} y^{(i)} y^{(i')} K(x^{(i)}, x^{(i')})$$

$$\text{s.t. } 0 \leq \alpha_i \leq C \quad \forall\, i = 1, \dots, m$$

$$\sum_{i=1}^{m} \alpha_i y^{(i)} = 0$$

To make predictions once the optimal α's are found:

$$h(x) = g(z(x))$$

$$\text{where } z(x) = \sum_{i=1}^{m} \alpha_i y^{(i)} K(x^{(i)}, x) + b$$

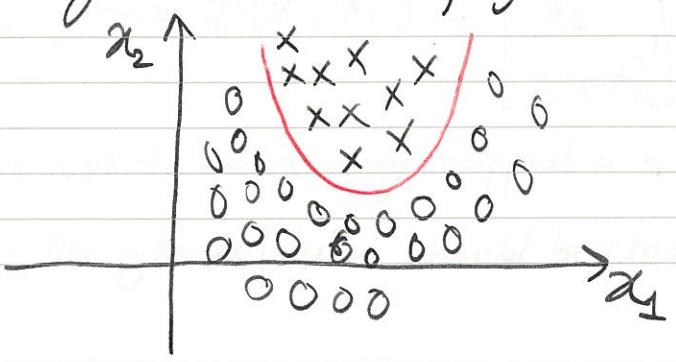
$$b = y^{(i)} - \sum_{i'=1}^{m} \alpha_{i'} y^{(i')} K(x^{(i')}, x^{(i)}) \text{ for any } i \text{ where: } 0 < \alpha_i < C.$$

$$g(z) = \begin{cases} +1 & z \geq 0 \\ -1 & z < 0 \end{cases}$$

$K(x, z) \equiv \langle x, z \rangle$ is called the Kernel.

As we also argued on page 69, the size of input features n only enters through the inner products $\langle x, y \rangle$. Therefore, if we can efficiently compute the inner-product, we can have an algorithms that scales very mildly with the # of features. We've already seen in the discussion of linear and logistic regression that fitting non-linear hypotheses is equivalent to adding more features which correspond to the non-linear terms. Only then, because those algorithms depend crucially on the feature size, learning non-linear functions becomes intractable quite quickly. As we've argued, SVM doesn't have this limitation: if we have 10 features, 100 features, 10000 features, at the end of the day we are fitting for as many parameters as the # of our training set on. This is the beauty of solving the dual Lagrangian instead of the primal problem. This assumes, though, that $\langle x, z \rangle$ can be computed efficiently in high dimensions.

Suppose the two data sets are not linearly-separable, but they can be separated by a second order polynomial:

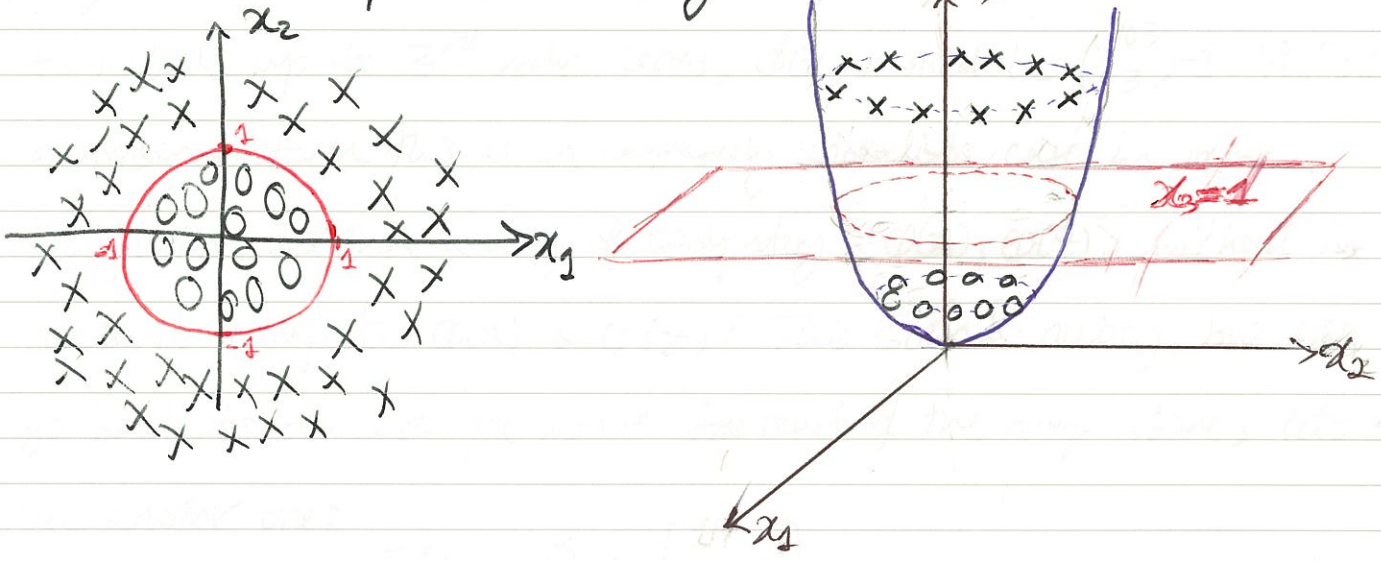


Can we apply SVM in this case? Certainly not with $n=2$, since the whole formulation is about separating linear hyperplanes. But what if we map our original features to a higher dimension?

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \longrightarrow \phi(x_1, x_2) = \begin{bmatrix} x_1 \\ x_2 \\ x_1^2 \end{bmatrix} \quad \text{where now } x_3 = x_1^2$$

Then, if in the original dimension ($n=2$) we have the following non-linear decision boundary: $x_2 - a x_1^2 - b x_1 - c = 0$, in $n=3$ we have a linear decision boundary: $x_2 - a x_3 - b x_1 - c = 0$.

Here's an attempt at visualizing this:



$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \longrightarrow \phi(x_1, x_2) = \begin{bmatrix} x_1 \\ x_2 \\ x_1^2 + x_2^2 \end{bmatrix}$$

In the figure above, we've only mapped a select few points from each class; the general idea should be clear though.

In general, if we know some 2nd order polynomial in x_1 & x_2 can separate our data set, we could make a transformation as follows:

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \xrightarrow{\Phi} \begin{bmatrix} x_1 \\ x_2 \\ x_1 x_2 \\ x_1^2 \\ x_2^2 \end{bmatrix}$$

Now remember that SVM only depends on $\langle x, z \rangle$. So all we need to do is replace $\langle x, z \rangle$ with $\langle \Phi(x), \Phi(z) \rangle$ & we're done. Computation of $\Phi(x)$, however, could be time-consuming. For instance, say we have 10×10 pixel images as input, i.e $n = 100$. If we want to include up to 3rd order terms, $\Phi(x)$ would be $\binom{103}{3} - 1 = 176{,}850$ dimensional. Even this is an extremely unrealistic case in image recognition. Is there a way of computing $\langle \Phi(x), \Phi(z) \rangle$ without the needh to compute $\Phi(x)$ & $\Phi(z)$? This sounds nutty, but let's go back to our example above. Instead of the map above, let's use another one:

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \xrightarrow{\Phi} \begin{bmatrix} 1 \\ \sqrt{2}\, x_1 \\ \sqrt{2}\, x_2 \\ \sqrt{2}\, x_1 x_2 \\ x_1^2 \\ x_2^2 \end{bmatrix}$$

Why in earth would we add another dimension to encode 1 & multiply

some terms by $\sqrt{2}$? Because of the following:

$$\langle \phi(x), \phi(z) \rangle = 1 + 2x_1 z_1 + 2x_2 z_2 + 2x_1 x_2 z_1 z_2 + x_1^2 z_1^2 + x_2^2 z_2^2$$

$$= (x_1 z_1 + x_2 z_2 + 1)^2$$

$$= (x \cdot z + 1)^2$$

Although our map $\phi(x)$ is 6-dimensional, we can compute $\langle \phi(x), \phi(z) \rangle$ by knowing only the inner product $x \cdot z$ in the original 2D feature space. Now imagine we have a 100-dimensional feature space and want to include at least up to 3rd order terms. Instead of creating a 176,850 dimensional map $\phi(x)$, which includes all possible terms like $x_1, x_2, \ldots, x_{100}, x_1 x_1, x_1 x_2, \ldots, x_1 x_{100}, x_2 x_2, \ldots, x_2 x_{100}, \ldots, x_{99} x_{100}, x_{100} x_{100}$, $x_1 x_1 x_2, x_1 x_1 x_3, \ldots, x_{100} x_{100} x_{100}$, we can simply use $K(x, z) = (x \cdot z + 1)^3$, knowing that there exists some $\binom{103}{3}$-dimensional $\phi(x)$, which has all the terms we care about, and with the property that $\langle \phi(x), \phi(y) \rangle$ is equal to $(x \cdot z + 1)^3$. In general, if we have an n-dimensional feature space & want to keep terms up to order d, we can use:

$$K(x, z) = (x \cdot z + 1)^d$$

There's always a map $\phi(x)$ of dimension $\binom{n+d}{d}$, which contains all the terms up to order d we care about, & with the property that

$$\langle \phi(x), \phi(z) \rangle = K(x, z) = (x \cdot z + 1)^d.$$

Let's go one step further: we can pick any Kernel $K(x,y)$ so long as there exists some mapping $\phi(x)$ with the property $K(x,y) = \langle \phi(x), \phi(y) \rangle$. We don't even need to know $\phi(x)$ explicitly, as long as we know such a map exists. Let's remind ourselves of the correspondence between positive semi-definite matrices and inner products. Assume K_{ij} is a symmetric positive semi-definite matrix: $K_{ij} = K_{ji}$ & $z^T K z \geq 0 \ \forall z$. Then, there exist vectors $v^{(k)}$ such that $K_{ij} = \langle v^{(i)}, v^{(j)} \rangle$. In fact, we can find $v^{(k)}$ explicitly. Since K_{ij} is symmetric & positive semi-definite, it has eigenvectors with eigenvalues greater or equal to zero:

$$K e^{(k)} = \lambda_k e^{(k)}$$

$$\text{where } \begin{cases} \lambda_k \geq 0 \\ \langle e^{(k)}, e^{(j)} \rangle = \delta_{kj} \Rightarrow \text{orthonormal basis} \end{cases}$$

It can then be checked that K_{ij} can be written as the following sum:

$$K_{ij} = \sum_{k=1}^{m} \lambda_k e_i^{(k)} e_j^{(k)}$$

($e_i^{(k)}$: i^{th} component of the k^{th} eigenvector.)

Check:

$$[Ke]_i = \sum_{j=1}^{m} K_{ij} e_j^{(k')} = \sum_{j=1}^{m} \left[\sum_{k=1}^{m} \lambda_k e_i^{(k)} e_j^{(k)} \right] e_j^{(k')}$$

$$= \sum_{k=1}^{m} \lambda_k e_i^{(k)} \underbrace{\langle e^{(k)}, e^{(k')} \rangle}_{\delta_{kk'}} = \lambda_{k'} e_i^{(k')} \quad \checkmark$$

Since $e^{(k)}$ form an orthonormal basis, this is all we need to check.

So if we let $V_k^{(i)} \equiv e_i^{(k)} \times \sqrt{\lambda_k}$:

$$K_{ij} = \sum_{k=1}^{m} \lambda_k e_i^{(k)} e_j^{(k)} = \sum_{k=1}^{m} (\sqrt{\lambda_k} e_i^{(k)})(\sqrt{\lambda_k} e_j^{(k)})$$

$$= \sum_{k=1}^{m} V_k^{(i)} V_k^{(j)} = \langle V^{(i)}, V^{(j)} \rangle.$$

In the case of function spaces, a very similar result, by the name of Mercer's theorem holds:

Let $\mathcal{X} \subseteq \mathbb{R}^n$ and $K: \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ be a symmetric continuous function: $K(x, z) = K(z, x)$. Furthermore, assume K is positive semi-definite: $\sum_{i=1}^{n} \sum_{j=1}^{n} K(x_i, x_j) c_i c_j \geq 0$ $\forall$ finite sequences of points $x_1, \ldots, x_n \in \mathcal{X}$ & all real numbers $c_1, \ldots, c_n$.

Consider the associated linear operator T_K:

$$(T_K f)(x) = \int_{\mathcal{X}} K(x, y) f(y) \rho(y) \, d^n y$$

(ρ: strictly positive continuous function (weight); $d^n y$.)

where $f \in L^2(\mathcal{X}, \rho)$. Inner products take the form: $\langle f, g \rangle = \int_{\mathcal{X}} f(x) g(x) \rho(x) \, d^n x$.

Theorem: There exists an orthonormal basis $\{e^{(k)}(x)\}$ of $L^2(\mathcal{X}, \rho)$ consisting of eigenfunctions of T_K such that the corresponding eigenvalues λ_k are non-negative. The eigenfunctions corresponding to non-zero eigenvalues give the following representation of K:

$$K(x, z) = \sum_{k=1}^{\infty} \lambda_k e^{(k)}(x) e^{(k)}(z)$$

where:
$$\begin{cases} (T_K e^{(k)})(x) = \lambda_k e^{(k)}(x) \\ \lambda_k \geq 0 \\ \langle e^{(k)}, e^{(k')} \rangle = \delta_{kk'} \end{cases}$$

Check the representation:

$$\int_X K(x,y)\, e^{(k')}(y)\, \rho\, dy = \int_X \left[\sum_{k=1}^{\infty} \lambda_k e^{(k)}(x) e^{(k)}(y) \right] e^{(k')}(y) \rho(y)\, dy$$

$$= \sum_{k=1}^{\infty} \lambda_k e^{(k)}(x) \times \underbrace{\langle e^{(k)}, e^{(k')} \rangle}_{\delta_{kk'}} = \lambda_{k'} e^{(k')}(x) \quad \checkmark$$

Mercer's condition, i.e. the definition given in the previous page for a symmetric positive semi-definite kernel $K(x,y)$, is equivalent to:

$$\langle f, T_K f \rangle = \int_X \int_X K(x,y) f(x) f(y) \rho(x) \rho(y)\, dx\, dy \geq 0.$$

$$\forall\ f \in L^2(\mathbb{R}^n, \rho).$$

Therefore, if we pick a symmetric continuous function $K(x,y)$ which satisfies the above condition, we can always find a mapping $\phi(x)$ such that $K(x,y) = \langle \phi(x), \phi(y) \rangle$:

$$[\phi(x)]_k = \sqrt{\lambda_k}\, e^{(k)}(x)$$

Note that in general, $\phi(x)$ will be infinite dimensional. This is what

people mean when they say SVM can handle an infinite-dimensional feature space. One of the most popular kernels, called the Gaussian or radial basis function (RBF) kernel, has precisely this feature:

$$K(x, z) = e^{-\frac{\|x-z\|^2}{2\sigma^2}}$$

In fact, it's not hard to write down the feature mapping:

$$e^{-\frac{\|x-z\|^2}{2\sigma^2}} = e^{-\frac{\langle x-z, x-z\rangle}{2\sigma^2}} = e^{-\frac{1}{2\sigma^2}(\|x\|^2+\|z\|^2-2\langle x,z\rangle)}$$

$$= e^{-\frac{1}{2\sigma^2}(\|x\|^2+\|z\|^2)} e^{\frac{1}{\sigma^2}\langle x,z\rangle}$$

$$= \exp\left[\frac{-1}{2\sigma^2}(\|x\|^2+\|z\|^2)\right] \times \sum_{j=0}^{\infty} \frac{(\langle x,z\rangle/\sigma^2)^j}{j!}$$

$$\overset{(*)}{=} \exp\left[\frac{-1}{2\sigma^2}(\|x\|^2+\|z\|^2)\right] \times \sum_{j=0}^{\infty} \frac{1}{j!\sigma^{2j}} \sum_{k_1+\cdots+k_n=j} \frac{j!}{k_1!\cdots k_n!} (x_1z_1)^{k_1}\cdots(x_nz_n)^{k_n}$$

$$= \sum_{j=0}^{\infty} \sum_{k_1+\cdots+k_n=j} \left\{ \frac{x_1^{k_1}\cdots x_n^{k_n}}{\sqrt{k_1!\cdots k_n!}} \cdot \frac{1}{\sigma^j} \exp\left[\frac{-1}{2\sigma^2}\|x\|^2\right] \right\}$$

$$\times \left\{ \frac{z_1^{k_1}\cdots z_n^{k_n}}{\sqrt{k_1!\cdots k_n!}} \frac{1}{\sigma^j} \exp\left[\frac{-1}{2\sigma^2}\|z\|^2\right] \right\}$$

where in $(*)$ we have used the Multinomial theorem. It should also be clear from the expansion above that $\langle f, T_K f\rangle \geq 0 \quad \forall f$.

The Gaussian kernel can be interpreted as a similarity measure: when x & z are far apart, $K(x,z) \to 0$ & when x & z are close, $K(x,z) \sim 1$.

This is best seen when we look at how SVM makes predictions (see page 78):

$$h(x) = \text{sgn}(z(x))$$

where

$$z(x) = \sum_{i=1}^{m} \alpha_i y^{(i)} K(x^{(i)}, x) + b$$

$$= \sum_{i=1}^{m} \alpha_i y^{(i)} e^{\frac{-\|x - x^{(i)}\|^2}{2\sigma^2}} + b$$

← We loop through all training examples and measure how 'close' to x they are. Points closer to x make a larger contribution

We can tune σ to determine the degree of similarity. If σ is small, only points very close to a training example $x^{(i)}$ would be considered similar. On the other hand, if σ is large, even points far from $x^{(i)}$ would be considered similar. It's reasonable to expect that:

* Large σ ⇒ underfitting (high bias)
* Small σ ⇒ overfitting (high variance)

Having chosen a Kernel, all that's left to do is have an algorithm for solving the optimization problem on page 78.

Lecture 76: Using an SVM

The sequential minimal optimization (SMO) algorithm, due to John Platt, gives an efficient way of solving the dual problem on page 78. Before discussing SMO, let's talk about the coordinate ascent algorithm.

Coordinate Ascent

Consider the following unconstrained maximization problem:

$$\max_{\alpha} W(\alpha_1, \alpha_2, \ldots, \alpha_m)$$

We already know how to do this using gradient ascent (the obvious twin of gradient descent, which is used for minimization). Coordinate ascent is another algorithm

Loop until convergence {

  For $i = 1, \ldots, m$ {

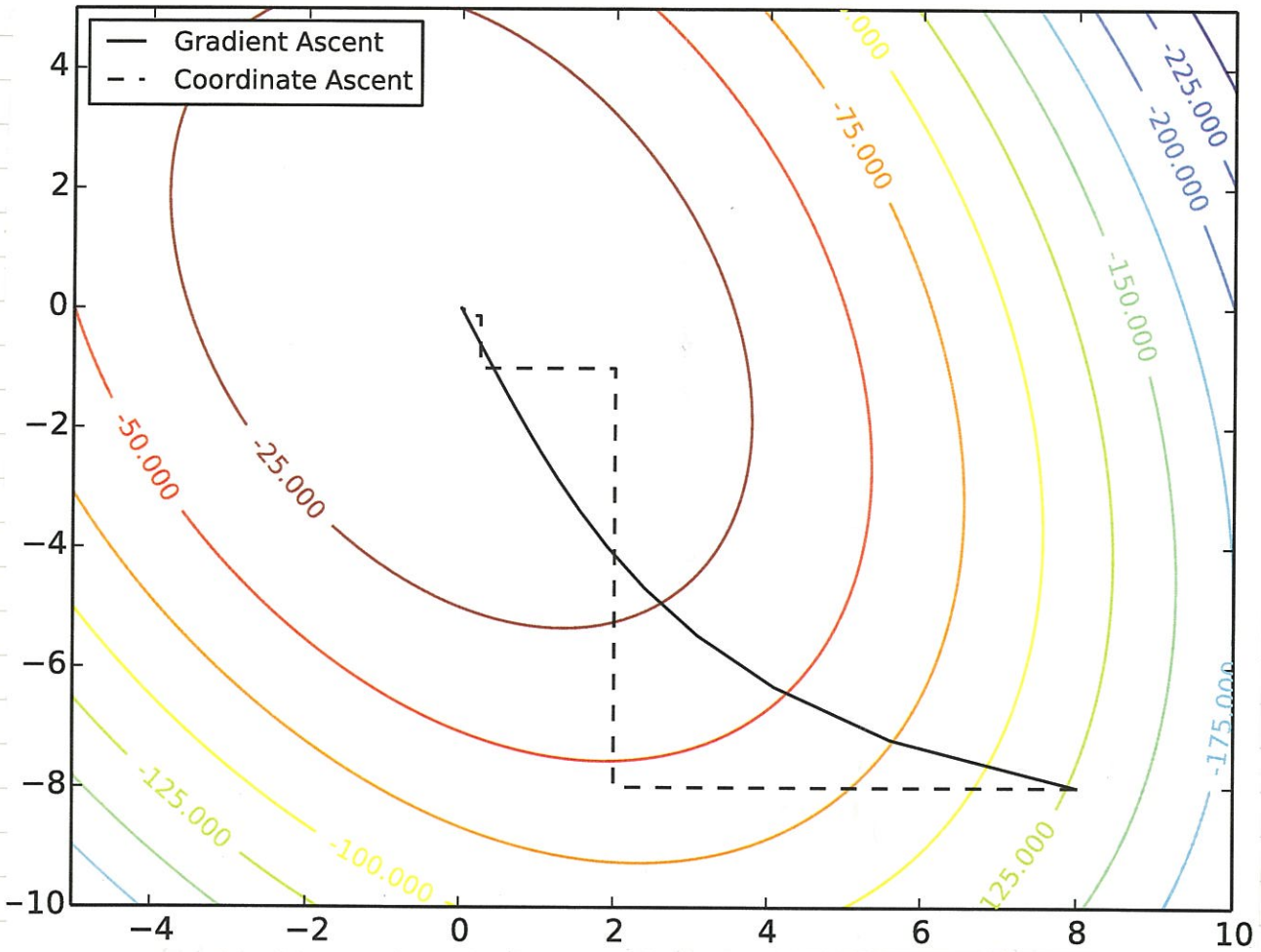
    $\alpha_i := \arg\max_{\hat{\alpha}_i} W(\alpha_1, \ldots, \alpha_{i-1}, \hat{\alpha}_i, \alpha_{i+1}, \ldots, \alpha_m)$

  }

}

This is what the algorithm does: (i) pick a starting point $(\alpha_1, \ldots, \alpha_m)$. (ii) keeping $\alpha_2, \alpha_3, \ldots, \alpha_m$ constant, find $\hat{\alpha}_1$ that maximizes W along coordinate 1. Then keeping $\hat{\alpha}_1, \alpha_3, \ldots \alpha_m$ constant & find $\hat{\alpha}_2$ that maximizes W along coordinate

2. Continue this until convergence. In the version presented here, the inner loop reoptimizes the variables in order $\alpha_1, \alpha_2, \ldots, \alpha_m, \alpha_1, \alpha_2, \ldots$. A more sophisticated version might choose other orderings; for instance, we may choose the next variable to update according to which one we expect to allow us to make the largest increase in W. The key to the efficiency of the coordinate ascent algorithm is how fast "arg max" can be computed, which in turn depends on the form of W.



$W(\alpha_1, \alpha_2) = -2\alpha_1^2 - \alpha_2^2 - \alpha_1\alpha_2$ was used in this example. Argmax was computed analytically to $\underset{\hat{\alpha}_1}{\text{argmax}}\, W(\hat{\alpha}_1, \alpha_2) = -\alpha_2/4$ & $\underset{\hat{\alpha}_2}{\text{argmax}}\, W(\alpha_1, \hat{\alpha}_2) = -\alpha_1/2$.

Sequential Minimal Optimization

Can we solve the optimization problem on page 78 using coordinate ascent? Imagine we start with α's that satisfy the constraints, say $\alpha_i = 0 \; \forall \, i = 1, \ldots, m$. If we then try to maximize the objective function along α_1, whatever new value we obtain will violate the equality constraint $\sum_{i=1}^{m} \alpha_i y^{(i)} = 0$. As a result, the smallest # of α's which can be simultaneously optimized without violating the constraints is 2. This is what SMO does: it optimizes pairs of α's until convergence. The interesting part is that because the objective function is quadratic in α's, this can be done analytically. This was John Platt's key insight, which made SMO much faster than other algorithms which used numerical QP solvers. Let's work out the optimization for any two α's, starting with the optimization problem itself:

$$\max_{\alpha} W(\alpha_1, \ldots, \alpha_m) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} y^{(i)} y^{(j)} K_{ij} \alpha_i \alpha_j$$

$$\text{s.t.} \quad 0 \leq \alpha_i \leq C \quad \forall \; i = 1, \ldots, m$$

$$\sum_{i=1}^{m} \alpha_i y^{(i)} = 0$$

where $K_{ij} \equiv K(x^{(i)}, x^{(j)})$

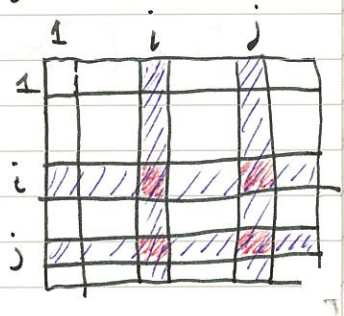
Let's say we have the values $\alpha_1^*, \alpha_2^*, \dots, \alpha_m^*$ from the previous optimization step, & they satisfy both constraints. We now want to optimize w.r.t α_i & α_j, keeping all other α's the same:

$$W(\alpha_1^*, \dots, \alpha_{i-1}^*, \alpha_i, \alpha_{i+1}^*, \dots, \alpha_{j-1}^*, \alpha_j, \alpha_{j+1}^*, \dots, \alpha_m^*) \equiv \Psi(\alpha_i, \alpha_j)$$

$$\Psi(\alpha_i, \alpha_j) = \alpha_i + \alpha_j - \frac{1}{2}\alpha_i y^{(i)}\left[\alpha_i y^{(i)} K_{ii} + \alpha_j y^{(j)} K_{ij} + 2\sum_{\substack{k=1 \\ k\neq i,j}}^{m} \alpha_k^* y^{(k)} K_{ik}\right]$$

$$- \frac{1}{2}\alpha_j y^{(j)}\left[\alpha_i y^{(i)} K_{ji} + \alpha_j y^{(j)} K_{jj} + 2\sum_{\substack{k=1 \\ k\neq i,j}}^{m} \alpha_k^* y^{(k)} K_{jk}\right]$$

$+ C$ ← constant with no dependency on α_i & α_j



Let $v_i \equiv \sum_{\substack{k=1 \\ k\neq i,j}}^{m} \alpha_k^* y^{(k)} K_{ik}$ $\quad$ $s \equiv y^{(i)} y^{(j)}$

$$\Psi(\alpha_i, \alpha_j) = -\frac{1}{2} K_{ii}\alpha_i^2 - \frac{1}{2} K_{jj}\alpha_j^2 - s K_{ij}\alpha_i\alpha_j + (1 - y^{(i)} v_i)\alpha_i + (1 - y^{(j)} v_j)\alpha_j + C$$

What does the equality constraint tell us?

From previous step: $\sum_{\substack{k=1 \\ k\neq i,j}}^{m} \alpha_k^* y^{(k)} + y^{(i)}\alpha_i^* + y^{(j)}\alpha_j^* = 0$

Current step: $\sum_{\substack{k=1 \\ k\neq i,j}}^{m} y^{(k)}\alpha_k^* + y^{(i)}\alpha_i + y^{(j)}\alpha_j = 0$

It follows that: $y^{(i)}\alpha_i + y^{(j)}\alpha_j = y^{(i)}\alpha_i^* + y^{(j)}\alpha_j^*$

$$\Rightarrow y^{(j)}\alpha_j = y^{(j)}\alpha_j^* - y^{(i)}(\alpha_i - \alpha_i^*)$$

multiply both sides by $y^{(j)}$

$$\Rightarrow \boxed{\alpha_j = \alpha_j^* - s(\alpha_i - \alpha_i^*)} \quad \oplus$$

We need to find the maximum of $\Psi(\alpha_i, \alpha_j)$ along the line defined by $\oplus$:

$$\Psi(\alpha_i) \equiv \Psi(\alpha_i, \alpha_j^* - s(\alpha_i - \alpha_i^*))$$

$$= -\frac{1}{2}K_{ii}\alpha_i^2 - \frac{1}{2}K_{jj}[\alpha_j^* - s(\alpha_i - \alpha_i^*)]^2$$

$$- sK_{ij}\alpha_i[\alpha_j^* - s(\alpha_i - \alpha_i^*)]$$

$$+ (1 - y^{(i)}v_i)\alpha_i + (1 - y^{(j)}v_j)[\alpha_j^* - s(\alpha_i - \alpha_i^*)] + C$$

$$= \alpha_i^2\left[-\frac{1}{2}K_{ii} - \frac{1}{2}K_{jj} + K_{ij}\right]$$

$$+ \alpha_i[K_{jj}\alpha_i^* + K_{jj}s\alpha_j^* - sK_{ij}\alpha_j^* - K_{ij}\alpha_i^*$$

$$+ 1 - y^{(i)}v_i - s(1 - y^{(j)}v_j)]$$

$$+ C \leftarrow \text{constant with no dependency on } \alpha_i$$

Using the fact that $sy^{(j)} = y^{(i)}$, we can further simplify:

$$\Psi(\alpha_i) = -\frac{1}{2}(K_{ii} + K_{jj} - 2K_{ij})\alpha_i^2 + \Lambda\alpha_i + C$$

where $\Lambda \equiv K_{jj}\alpha_i^* + sK_{jj}\alpha_j^* - sK_{ij}\alpha_j^* - K_{ij}\alpha_i^* + 1 - s + y^{(i)}(v_j - v_i)$

We can further simplify Λ.

Let $\boxed{z_i \equiv \sum_{k=1}^{m} \alpha_k^* y^{(k)} K_{ki} + b^*}$ be the previous step's prediction for the i^{th} example.

$$\begin{cases} v_i = z_i - b^* - \alpha_i^* y^{(i)} K_{ii} - \alpha_j^* y^{(j)} K_{ji} \\ v_j = z_j - b^* - \alpha_j^* y^{(j)} K_{jj} - \alpha_i^* y^{(i)} K_{ij} \end{cases}$$

$$v_j - v_i = z_j - z_i - \alpha_j^* y^{(j)} K_{jj} - \alpha_i^* y^{(i)} K_{ij} + \alpha_i^* y^{(i)} K_{ii} + \alpha_j^* y^{(j)} K_{ji}$$

$$1 - s = y^{(i)} y^{(i)} - y^{(i)} y^{(j)} = y^{(i)} (y^{(i)} - y^{(j)})$$

$$\Delta = K_{jj}\alpha_i^* + s K_{jj}\alpha_j^* - s K_{ij}\alpha_j^* - K_{ij}\alpha_i^* + y^{(i)}(y^{(i)} - y^{(j)})$$
$$+ y^{(i)}(z_j - z_i) - s\alpha_j^* K_{jj} - \alpha_i^* K_{ij} + \alpha_i^* K_{ii} + s\alpha_j^* K_{ji}$$
$$= \alpha_i^* (K_{jj} + K_{ii} - 2K_{ij}) + y^{(i)} [(z_j - y^{(j)}) - (z_i - y^{(i)})]$$

Let $\boxed{\eta \equiv K_{ii} + K_{jj} - 2K_{ij}}$

$\boxed{E_i \equiv z_i - y^{(i)}}$

↳ can think of this as the error of prediction of the previous step. for the i^{th} example.

$$\boxed{\Psi(\alpha_i) = -\frac{1}{2}\eta\alpha_i^2 + [\eta\alpha_i^* + y^{(i)}(E_j - E_i)]\alpha_i + C}$$

This is now something we can optimize, more specifically maximize:

Assume $\eta > 0$ & $\frac{d\Psi}{d\alpha_i} = -\eta\alpha_i + \eta\alpha_i^* + y^{(i)}(E_j - E_i) = 0$

$$\Rightarrow \boxed{\alpha_i = \alpha_i^* + \frac{y^{(i)}(E_j - E_i)}{\eta}} \text{ (\#)}$$

$\eta = 0$ can happen if the i^{th} & j^{th} examples have the same input vectors basically when there's duplication in the training set

In this case $\Psi(\alpha_i) = y^{(i)}(E_i - E_j)\alpha_i + C$, but since $i \neq j$ have the same input vector (from which it follows they should have the same label, otherwise the same input vector is labled with both +1 & -1, which makes no sense), $E_i = E_j \Rightarrow \Psi(\alpha_i) = C$. Therefore, if $\eta = 0$, we cannot make progress and need to pick another pair for optimization.

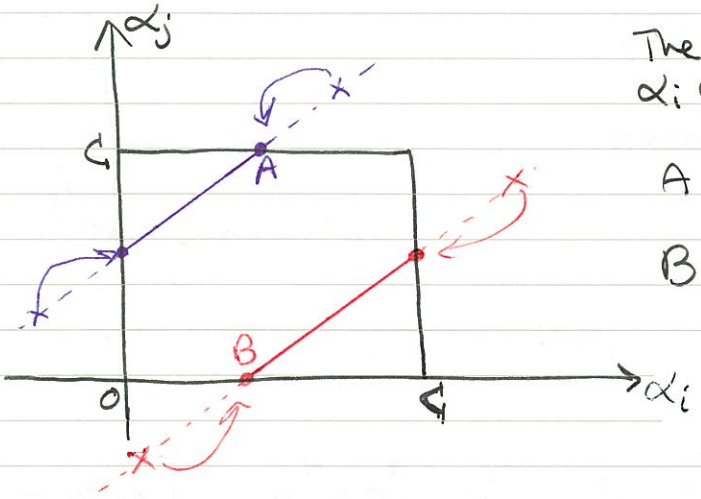
If $\eta < 0$, it means the Kernel is not semi-definite:

$$\begin{bmatrix} 1 & -1 \end{bmatrix} \begin{bmatrix} K_{ii} & K_{ij} \\ K_{ij} & K_{jj} \end{bmatrix} \begin{bmatrix} 1 \\ -1 \end{bmatrix} = K_{ii} + K_{jj} - 2K_{ij} < 0.$$

Since we imposed this as a requirement on the Kernel, we will not treat $\eta < 0$ here & assume our Kernel is semi-definite. (John Platt's paper actually does cover this case.)

So are we done? Nope, we completely forgot about the inequality constraint: $0 \leq \alpha_i, \alpha_j \leq C$. What if the optimal α_i computed by ⊕ on the previous page doesn't satisfy this inequality? Lets remind ourselves what problem we're actually solving here: maximize $\Psi(\alpha_i, \alpha_j)$ on the line segment $\alpha_j = \alpha_j^* - s(\alpha_i - \alpha_i^*)$ where the line is limited to $0 \leq \alpha_i, \alpha_j \leq C$.

Case I: $s = -1$ $\qquad \alpha_j = \alpha_i + (\alpha_j^* - \alpha_i^*)$



The crosses denote the optimal values of α_i & α_j without the box constraint.

A has coordinates: $\langle C - \alpha_j^* + \alpha_i^*, C \rangle$

B has coordinates: $\langle -\alpha_j^* + \alpha_i^*, 0 \rangle$

Consider two sub-cases:

(i) $\alpha_j^* - \alpha_i^* \geq 0$ (blue): $\Rightarrow \begin{cases} \alpha_i^{\text{unclipped}} < 0 \Rightarrow \alpha_i^{\text{clipped}} = 0 \\ \alpha_i^{\text{unclipped}} > C - \alpha_j^* + \alpha_i^* \Rightarrow \alpha_i^{\text{clipped}} = C - \alpha_j^* + \alpha_i^* \end{cases}$

(ii) $\alpha_j^* - \alpha_i^* \leq 0$ (red): $\begin{cases} \alpha_i^{\text{unclipped}} < -\alpha_j^* + \alpha_i^* \Rightarrow \alpha_i^{\text{clipped}} = -\alpha_j^* + \alpha_i^* \\ \alpha_i^{\text{unclipped}} > C \Rightarrow \alpha_i^{\text{clipped}} = C \end{cases}$

let $L \equiv \max(0, \alpha_i^* - \alpha_j^*)$ & $H = \min(C, C + \alpha_i^* - \alpha_j^*)$

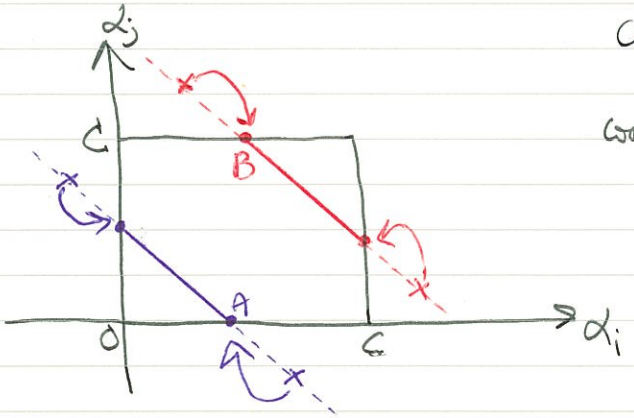
We can then summarize our results as follows:

$$\begin{cases} \alpha_i^{\text{unclipped}} < L \Rightarrow \alpha_i^{\text{clipped}} = L \\ \alpha_i^{\text{unclipped}} > H \Rightarrow \alpha_i^{\text{clipped}} = H \end{cases}$$

Lets now work out the case with $s = +1$.

Case II: $s = +1$

$$\alpha_j = -\alpha_i + (\alpha_i^* + \alpha_j^*)$$



Coordinates of A: $\langle \alpha_i^* + \alpha_j^*, 0 \rangle$

Coordinates of B: $\langle \alpha_i^* + \alpha_j^* - C, C \rangle$

$$L = \max(0, \alpha_i^* + \alpha_j^* - C) \quad \& \quad H = \min(C, \alpha_i^* + \alpha_j^*)$$

So:
$$\begin{cases} \alpha_i^{\text{unclipped}} < L \Rightarrow \alpha_i^{\text{clipped}} = L \\ \alpha_i^{\text{unclipped}} > H \Rightarrow \alpha_i^{\text{clipped}} = H \end{cases}$$

Let's summarize the results of jointly optimizing α_i & α_j:

$$\alpha_i = \begin{cases} L & \text{if } \alpha_i^{\text{unclipped}} < L \\ \alpha_i^{\text{unclipped}} & \text{if } L \leq \alpha_i^{\text{unclipped}} \leq H \\ H & \text{if } \alpha_i^{\text{unclipped}} > H \end{cases}$$

where
$$L = \begin{cases} \max(0, \alpha_i^* - \alpha_j^*) & \text{if } s = -1 \\ \max(0, \alpha_i^* + \alpha_j^* - C) & \text{if } s = 1 \end{cases}$$

$$H = \begin{cases} \min(C, C + \alpha_i^* - \alpha_j^*) & \text{if } s = -1 \\ \min(C, \alpha_i^* + \alpha_j^*) & \text{if } s = -1 \end{cases}$$

$$\alpha_i^{\text{unclipped}} = \alpha_i^* + \frac{y^{(i)}(E_j - E_i)}{\eta} \quad \text{(assuming } \eta > 0\text{)}$$

and $\alpha_j = \alpha_j^* - s(\alpha_i - \alpha_i^*)$