Separating Hyperplanes

Carlos Valle

Departamento de Informática
Universidad Técnica Federico Santa María

cvalle@inf.utfsm.cl

November 25, 2015

Overview

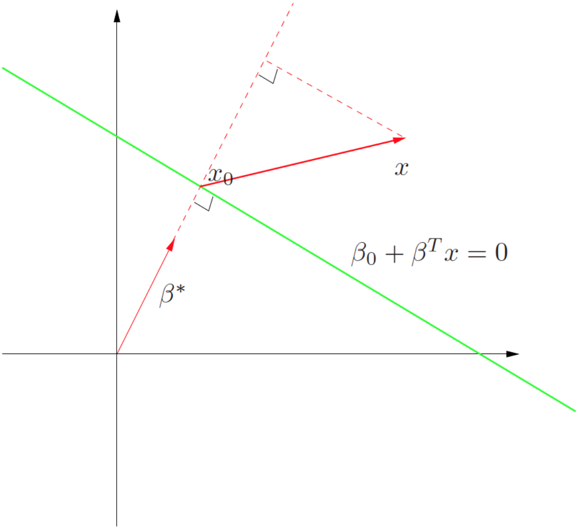
Separating Hyperplanes

- So far, we have seen learning algorithms which incidentally build a hyperplane to separate data.
- No we will see an algorithm that explicitly try to separate the data into different classes as well as possible.
- Finally, we will study an approach that try to construct an optimal margin classifier.
- Conveniently, we will define our targets $y_m \in \{-1, 1\}$
- Also, we will rearrange our notation for an hyperplane or affine set L as:

$$f(x) = w^T \mathbf{x} + b$$

- This means $b = \beta_0$, and $w = (\beta_1, \beta_2, \ldots, \beta_I)^T$

Separating Hyperplanes (2)



Separating Hyperplanes (3)

- For any two points $\mathbf{x}_1, \mathbf{x}_2$ in L we have $w^T\mathbf{x}_1 = -b$, $w^T\mathbf{x}_2 = -b$
- Then, $w^T(\mathbf{x}_1 - \mathbf{x}_2) = 0$
- Hence, $w^* = \frac{w}{||w||}$ is orthonormal to L
- The hyperplane L defines a half-space of the form

$$x : w^T x \geq b$$

- where $w^T(\mathbf{x} - \mathbf{x}_0) \geq 0$ means that the angle between $\mathbf{x} - \mathbf{x}_0$ is acute ($[-\pi/2; \pi/2]$).
- Thus, $w^{*T}(\mathbf{x} - \mathbf{x}_0) = \frac{w^T\mathbf{x}+b}{||w||} = \frac{f(\mathbf{x})}{||f'(\mathbf{x})||}$ is the signed distance from $\mathbf{x}$ to L
- Therefore, $f(\mathbf{x})$ is proportional to the signed distance from $\mathbf{x}$ to the hyperplane defined by $f(\mathbf{x}) = 0$.

Minimizing the functional margin

- As we have seen, the functional margin of a single point $(\mathbf{x}_m, y_m)$ is given by

$$\gamma_m = y_m f(\mathbf{x}_m) = y_m(w^T \mathbf{x}_m + b)$$

- Then, it seems natural that the hypothesis maximizes the margin of the training examples.
- This is equivalent to minimizes

$$J(\mathbf{w}, b) = -y_m(\mathbf{w}^T \mathbf{x}_m + b)$$

Gradient descent rule

- For a single example $(\mathbf{x}_m, y_m)$, the derivatives of $J(\mathbf{w}, b)$ respect to w and b are

$$\begin{aligned}\frac{\partial J(\mathbf{w}, b)}{\partial w} &= -y_m \mathbf{x}_m \\ \frac{\partial J(\mathbf{w}, b)}{\partial b} &= -y_m\end{aligned}$$

- Then, gradient descent rule is

$$\begin{aligned}\mathbf{w}^{p+1} &= \mathbf{w}^p + \eta y_m \mathbf{x}_m \\ b^{p+1} &= b^p + \eta y_m\end{aligned}$$

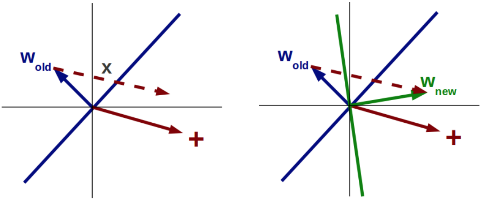
Primal Perceptron algorithm

Algorithm 1 Primal perceptron algorithm

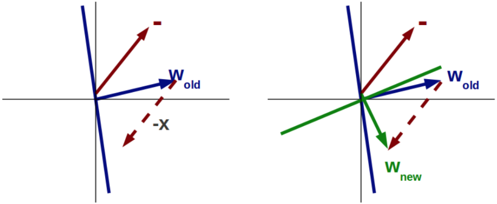
Given a training set S
$w_0 \leftarrow 0; b_0 \leftarrow 0; p \leftarrow 0$
$R \leftarrow \max_{1 \leq m \leq M} ||x_m||$
repeat
  for $m = 1$ to M **do**
    if $y_m(\mathbf{w}^T \mathbf{x}_m + b_p) \leq 0$ **then**
      $w_{p+1} \leftarrow w_p + \eta y_m x_m$
      $b_{p+1} \leftarrow b_p + \eta y_m R^2$
      $p \leftarrow p + 1$
    end if
  end for
until No mistakes are made within the loop
Output: $(\mathbf{w}_p, b_p)$

What does the weight update is doing?

$$\mathbf{w}_{new} = \mathbf{w}_{old} + \mathbf{x}$$



$$\mathbf{w}_{new} = \mathbf{w}_{old} - \mathbf{x}$$



Primal Perceptron algorithm (2)

- Note that the contribution of $\mathbf{x}_m$ to the weight update is $\alpha_m \eta y_m x_m$
- where α_m is the times that x_m is misclassified.
- Then, the number of errors is

$$k = \sum_{m=1}^{M} \alpha_m$$

- $$w = \sum_{m=1}^{M} \alpha_m y_m x_m$$

- The algorithm directly modifies w and b.
- If exists an hyperplane that correctly classifies the training data, this implies that data is linearly separable

Dual Perceptron algorithm

- Observe that we can define the boundary decision as

$$
\begin{aligned}
f(\mathbf{x}) &= \text{sign}(y(w^T\mathbf{x} + b)) &(1)\\
&= \text{sign}\left(y\left(\left(\sum_{m=1}^{M} \alpha_m y_m \mathbf{x}_m\right)^T \mathbf{x} + b\right)\right) &(2)\\
&= \text{sign}\left(y\left(\sum_{j=1}^{m} \alpha_m y_m (\mathbf{x}_m^T \mathbf{x}) + b\right)\right) &(3)
\end{aligned}
$$

- To compute the boundary decision we only need $\langle \mathbf{x}_m, \mathbf{x} \rangle$
- Note that the decision boundary does not change if we multiply $w^T\mathbf{x} + b$ for an scalar $c \neq 0$. Thus, we can choose the hyperplane $\left(\frac{\mathbf{w}}{||\mathbf{w}||}\right)^T \mathbf{x} + \frac{b}{||\mathbf{w}||}$

Dual Perceptron algorithm (2)

Algorithm 2 Dual perceptron algorithm

Given a training set S

$\alpha \leftarrow 0; b_0 \leftarrow 0$

$R \leftarrow \max_{1 \leq m \leq M} ||\mathbf{x}_m||$

repeat

  for $m = 1$ to M **do**

    if $\left(y_m \left(\sum_{l=1}^{L} \alpha_l y_l \langle \mathbf{x}_l, \mathbf{x}_m \rangle + b\right)\right) \leq 0$ **then**

      $\alpha_m \leftarrow \alpha_m + 1$

      $b \leftarrow b + \eta y_m R^2$

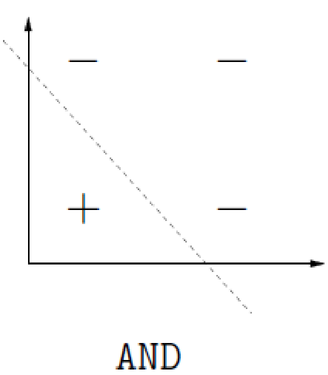
    end if

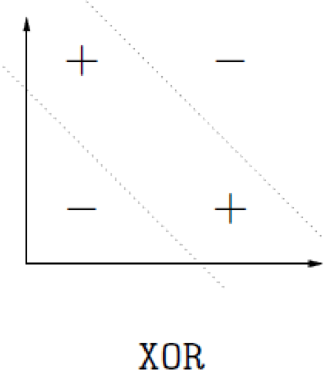
  end for

until No mistakes are made within the loop

Output: (α, b) to define $f(\mathbf{x})$ according to equation (1)

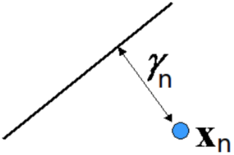
Perceptron problem





- Minsky,1969
- Not all problem are linearly separable

Geometric margin



- Let γ_m the Geometric margin of an example x_m be its distance from the hyperplane
- We have that $\mathbf{x}_m - \hat{\gamma}_m \cdot \frac{\mathbf{w}}{||\mathbf{w}||}$ is the nearest point of the plane along the direction of $\mathbf{w}$.

Geometric margin (2)

- Moreover, this point belongs to the hyperplane which implies

$$
\begin{aligned}
\mathbf{w}^T\left(\mathbf{x}_m - \hat{\gamma}_m \cdot \frac{\mathbf{w}}{||\mathbf{w}||}\right) + b &= 0 \\
\hat{\gamma}_m \cdot \frac{\mathbf{w}^T\mathbf{w}}{||\mathbf{w}||} &= \mathbf{w}^T\mathbf{x}_m + b \\
\hat{\gamma}_m \cdot \frac{||\mathbf{w}||^2}{||\mathbf{w}||} &= \mathbf{w}^T\mathbf{x}_m + b \\
\hat{\gamma}_m &= \left(\frac{\mathbf{w}}{||\mathbf{w}||}\right)^T \mathbf{x}_m + \frac{b}{||\mathbf{w}||} \\
\hat{\gamma}_m &= \frac{\gamma}{||\mathbf{w}||}
\end{aligned}
$$

- This implies that the functional margin is equal to the geometric marginal when $||\mathbf{w}|| = 1$

Optimally Separating Hyperplanes

- We would like to find a function $f_{\mathbf{w},b}(\mathbf{x})$ to be the maximum of the geometric margins of the individual training examples

$$\begin{aligned} \max_{\gamma,\mathbf{w},b} \quad & \gamma \\ \text{s.t.} \quad & y_m(\mathbf{w}^T x_m + b) \geq \gamma,\ m = 1, \ldots, M \\ & ||\mathbf{w}|| = 1 \end{aligned}$$

- However, the $||\mathbf{w}|| = 1$ constraint makes this problem not easy to solve (non convex)
- This is equivalent to

$$\begin{aligned} \max_{\gamma,\mathbf{w},b} \quad & \frac{\gamma}{||\mathbf{w}||} \\ \text{s.t.} \quad & y_m(\mathbf{w}^T x_m + b) \geq ||\mathbf{w}||\gamma,\ m = 1, \ldots, M \end{aligned}$$

Optimally Separating Hyperplanes (2)

- We can arbitrary set $\gamma = 1/||\mathbf{w}||$ Thus, the optimization problem above is equivalent to

$$\begin{array}{ll} \min\limits_{\mathbf{w},b} & \frac{1}{2}||\mathbf{w}||^2 \\ \text{s.t.} & y_m(\mathbf{w}^T x_m + b) \geq 1,\ m = 1, \ldots, M \end{array}$$

- This is a convex quadratic optimization problem (quadratic criterion with linear inequality constraints)

Lagrange duality

- Consider the following optimization problem:

$$\begin{aligned} \min_{\mathbf{w}} \quad & f(\mathbf{w}) \\ \text{s.t.} \quad & h_j(\mathbf{w}) = 0,\ j = 1, \ldots, J \end{aligned}$$

- We define the Lagrangian to be

$$\mathcal{L}(\mathbf{w}, \lambda) = f(\mathbf{w}) + \sum_{j=1}^{J} \lambda_j h_j(\mathbf{w}),$$

- Where λ_j are the Lagrange Multipliers
- Stationary points are those points where the partial derivatives of $\mathcal{L}$ are zero:

$$\frac{\delta \mathcal{L}}{\delta w_I} = 0, \frac{\delta \mathcal{L}}{\partial \lambda_j} = 0$$

Primal optimization problem

- Consider the following optimization problem:

$$
\begin{aligned}
\min_{\mathbf{w}} \quad & f(\mathbf{w}) \\
\text{s.t.} \quad & g_k(\mathbf{w}) \leq 0,\; k = 1, \ldots, K \\
& h_j(\mathbf{w}) = 0,\; j = 1, \ldots, J
\end{aligned}
$$

- We define the generalized Lagrangian

$$\mathcal{L}(\mathbf{w}, \lambda, \alpha) = f(\mathbf{w}) + \sum_{j=1}^{J} \lambda_j h_j(\mathbf{w}) + \sum_{k=1}^{K} \alpha_k g_k(\mathbf{w}), \tag{4}$$

- $\lambda_j, j = 1, \ldots, J$ and $\alpha_k, k = 1, \ldots, K$ are the Lagrange multipliers.

Primal optimization problem (2)

- Consider the quantity

$$\theta_{\mathcal{P}}(\mathbf{w}) = \max_{\lambda,\alpha:\alpha_k \geq 0} \mathcal{L}(\mathbf{w}, \lambda, \alpha)$$

- Note that $\theta_{\mathcal{P}}(\mathbf{w}) = \infty$ if w violates any constraint. Otherwise, $\theta_{\mathcal{P}}(\mathbf{w}) = f(\mathbf{w})$.
- Hence, the original primal problem defined in equation (4) is equivalent to

$$p^* = \min_{\mathbf{w}} \theta_{\mathcal{P}}(\mathbf{w}) = \min_{\mathbf{w}} \max_{\lambda,\alpha:\alpha_k \geq 0} \mathcal{L}(\mathbf{w}, \lambda, \alpha)$$

Dual optimization problem

- Now, consider the quantity

$$\theta_{\mathcal{D}}(\lambda, \alpha) = \min_{\mathbf{w}} (\mathbf{w}, \lambda, \alpha)$$

- Now we define the dual problem

$$d^* = \max_{\lambda, \alpha : \alpha_k \geq 0} \theta_{\mathcal{D}}(\lambda, \alpha) = \max_{\lambda, \alpha : \alpha_k \geq 0} \min_{\mathbf{w}} \mathcal{L}(\mathbf{w}, \lambda, \alpha)$$

- We obtain the Lagrange dual problem by solving for some primal variable values that minimize the Lagrangian.
- This solution gives the primal variables as functions of the Lagrange multipliers, which are called dual variables.

Dual optimization problem (2)

- So that the new problem is to maximize the objective function with respect to the dual variables under the derived constraints on the dual variables (including at least the nonnegativity).
- However, we must recall the maxmin: for any function $f : Z \times W \mapsto \mathbb{R}$

$$\sup_{\mathbf{z} \in Z} \inf_{\mathbf{w} \in W} f(\mathbf{z}, \mathbf{w}) \leq \inf_{w \in W} \sup_{\mathbf{z} \in Z} f(\mathbf{z}, \mathbf{w}).$$

Thus, in general

$$d^* \leq p^*$$

When $d^* = p^*$?

- If f and $g_k, k = 1, \ldots, K$ are convex and $h_j, j = 1, \ldots, J$ are affine $(h_j(w) = a_j^T w + b_i)$
- Then, there must exist $\mathbf{w}^*$,λ^*,α^* so that $\mathbf{w}^*$ is the solution to the primal problem and λ^*,α^* are solution to the dual problem. And moreover $d^* = p^* = \mathcal{L}(\mathbf{w}^*, \lambda^*, \alpha^*)$
- Moreover, $\mathbf{w}^*$, λ^* and α^* satisfy the Karush-Kuhn-Tucker (KKT) conditions:

$$
\begin{aligned}
\frac{\delta}{\delta w_i}\mathcal{L}(\mathbf{w}^*, \lambda^*, \alpha^*) &= 0,\ i = 1, \ldots, I && (5)\\
\frac{\delta}{\delta \lambda_j}\mathcal{L}(\mathbf{w}^*, \lambda^*, \alpha^*) &= 0,\ j = 1, \ldots, J && (6)\\
\alpha_k^* g_k(\mathbf{w}^*) &= 0,\ k = 1, \ldots, K && (7)\\
g_k(\mathbf{w}^*) &\leq 0,\ k = 1, \ldots, K && (8)\\
\alpha_k^* &\geq 0,\ k = 1, \ldots, K && (9)
\end{aligned}
$$

- Equation (8) is called KKT dual complementary. It implies that if $\alpha_k^* > 0$ means that $g_k(\mathbf{w}^*) = 0$.

Optimal margin classifiers

- Recall our optimization problem for finding the optimal margin classifier:

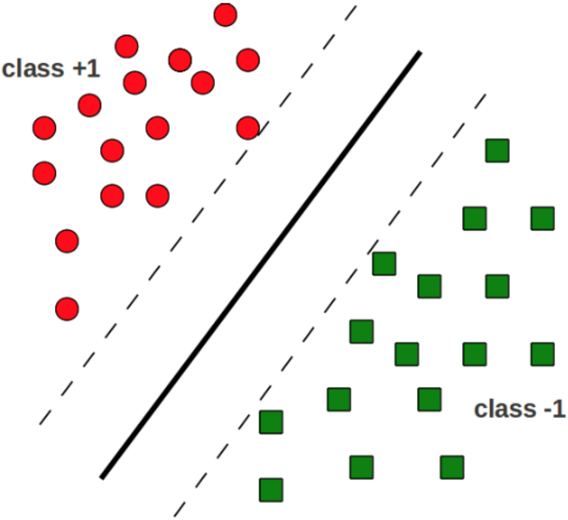
$$\begin{aligned} \min_{\mathbf{w},b} \quad & \frac{1}{2}||\mathbf{w}||^2 \\ \text{s.t.} \quad & y_m(\mathbf{w}^T x_m + b) \geq 1,\ m = 1, \ldots, M \end{aligned}$$

- We can write the constraints as $g_k(\mathbf{w}) = -y_m(\mathbf{w}^T x_m + b) + 1 \leq 0$
- Thus, the Lagrangian for the optimization problem above is given by

$$\mathcal{L}(\mathbf{w}, \lambda, \alpha) = \frac{1}{2}||\mathbf{w}||^2 - \sum_{m=1}^{M} \alpha_m \left[y_m(\mathbf{w}^T x_m + b) - 1 \right]. \tag{10}$$

- Note that $\alpha_m > 0$ implies that the margin of the m-th example is equal to one. (geometrical margin $\frac{1}{||\mathbf{w}||}$).
- These points are called support vectors.

Optimal margin classifiers (2)



Lagrangian primal derivatives

- In order to obtain the dual form of the problem
- Let's set the Lagrangian derivative to zero:

$$\nabla_{\mathbf{w}}\mathcal{L}(\mathbf{w},\lambda,\alpha) = \mathbf{w} - \sum_{m=1}^{M} \alpha_m y_m \mathbf{x}_m = 0$$

- This implies that

$$\mathbf{w} = \sum_{m=1}^{M} \alpha_m y_m \mathbf{x}_m$$

- And,

$$\frac{\partial}{\partial b}\mathcal{L}(\mathbf{w},\lambda,\alpha) = \sum_{m=1}^{M} \alpha_m y_m = 0$$

Wolfe dual

- Substituting these in (10) we have the so-called Wolfe dual
- $$\mathcal{L}(\mathbf{w}, \lambda, \alpha) = \sum_{m=1}^{M} \alpha_m - \frac{1}{2} \sum_{m=1}^{M} \sum_{\ell=1}^{M} \alpha_m \alpha_\ell y_m y_\ell x_m^T x_\ell$$
- To satisfy the KKT condition we add the constraints $\alpha_m^* \geq 0$, $m = 1, \ldots, M$ and the constraint $\sum_{m=1}^{M} \alpha_m y_m = 0$.
- Thus, we get the dual optimization problem

$$\begin{array}{ll} \max\limits_{\alpha} & \displaystyle\sum_{m=1}^{M} \alpha_m - \frac{1}{2} \sum_{m=1}^{M} \sum_{\ell=1}^{M} \alpha_m \alpha_\ell y_m y_\ell \langle x_m, x_\ell \rangle \\ \text{s.t.} & \alpha_m^* \geq 0,\ m = 1, \ldots, M \\ & \displaystyle\sum_{m=1}^{M} \alpha_m y_m = 0 \end{array}$$

Remarks

- After getting the α_m's which maximizes the objective function of the dual, we can obtain $\mathbf{w}^*$.
- Then we can compute b as

$$b^* = -\frac{\max_{m:y_m=-1} \mathbf{w}^{*T} x_m + \min_{m:y_m=1} \mathbf{w}^{*T} x_m}{2}$$

- After obtaining $\mathbf{w}^*$ and b^*, our classifier is given by

$$\begin{aligned}
\hat{y} = f_{\mathbf{w}^*, b^*}(\mathbf{x}) &= \text{sign}\left(\mathbf{w}^{*T}\mathbf{x} + b^*\right) \\
&= \text{sign}\left(\left(\sum_{m=1}^{M} \alpha_m y_m \mathbf{x}_m\right)^T \mathbf{x} + b^*\right) \\
&= \text{sign}\left(\sum_{m=1}^{M} \alpha_m y_m \langle \mathbf{x}_m, \mathbf{x} \rangle + b^*\right)
\end{aligned}$$

- Again, we only need the inner products

Implicit map with kernels

- We can map our input features to feature space $\mathcal{F}$ where the problem is linearly separable.
- We use the feature mapping ϕ:

$$\phi : \mathbf{x} \in \mathcal{X} \mapsto \phi(\mathbf{x}) \in \mathcal{F}$$

- Then, we can replace x for $\phi(\mathbf{x})$ in our optimization problem.
- However, we saw that for computing $f_{\mathbf{w}^*,b^*}(\mathbf{x})$ we only need

$$\langle \phi(\mathbf{x}), \phi(\mathbf{z}) \rangle$$

Kernel function

- Fortunately, we can replace our inner products with a Kernel function to be

$$K(\mathbf{x},\mathbf{z}) = \langle \phi(\mathbf{x}), \phi(\mathbf{z})\rangle, \forall \mathbf{x},\mathbf{z} \in \mathcal{X}$$

- Note that the kernel function does not depend on the dimension of the feature space $\mathcal{F}$.
- Consider two points $\mathbf{x} = (x_1, x_2)$ y $\mathbf{z} = (z_1, z_2)$ in a two-dimensional space and the function $K(\mathbf{x},\mathbf{z}) = \langle \mathbf{x},\mathbf{z}\rangle^2$

$$\begin{aligned}
\langle \mathbf{x},\mathbf{z}\rangle^2 &= \langle (x_1, x_2), (z_1, z_2)\rangle^2 \\
&= (x_1 z_1 + x_2 z_2)^2 \\
&= x_1^2 z_1^2 + x_2^2 z_2^2 + 2x_1 x_2 z_1 z_2 \\
&= \langle (x_1^2, x_2^2, \sqrt{2}x_1 x_2), (z_1^2, z_2^2, \sqrt{2} z_1 z_2)\rangle
\end{aligned}$$

Kernel function (2)

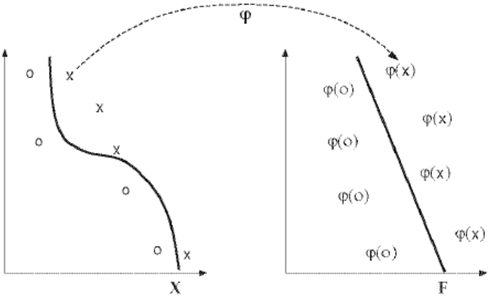
- The inner product above corresponds to the feature mapping

$$(x_1, x_2) \mapsto \phi(x_1, x_2) = (x_1^2, x_2^2, \sqrt{2}x_1x_2)$$

- Hence, $K(\mathbf{x}, \mathbf{z}) = \langle \mathbf{x}, \mathbf{z} \rangle^2 = \langle \phi(\mathbf{x}), \phi(\mathbf{z}) \rangle$
- It is easy to compute $K(\mathbf{x}, \mathbf{z}) = \langle \mathbf{x}, \mathbf{z} \rangle^d$. Since it induces a future space of $\binom{n+d-1}{d}$ dimensions, to compute $\phi(\mathbf{x})$ becomes computational unfeasible.

Kernel trick

- Fortunately we can use kernels without learning the mapping ϕ (kernel trick):



- Then, we need a square $M \times M$ matrix K, where $K_{ij} = K(\mathbf{x}_i, \mathbf{x}_j)$.
- This matrix is called the Kernel matrix.
- K measures how similar are $\phi(\mathbf{x}_i)$ and $\phi(\mathbf{x}_j)$.

Does any matrix K works?

- If K is a valid kernel, then $K_{ij} = K(\mathbf{x}_i, \mathbf{x}_j) = \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle = \langle \phi(\mathbf{x}_j), \phi(\mathbf{x}_i) \rangle = K(\mathbf{x}_j, \mathbf{x}_i) = K_{ji}$
- Therefore K must be symmetric.
- Moreover, let $\phi_k(\mathbf{x})$ be the k-th coordinate of the vector $\phi(\mathbf{x})$.
- For any vector $\mathbf{z}$ we have

$$
\begin{aligned}
\mathbf{z}^T K z &= \sum_i \sum_j z_i K_{ij} z_j \\
&= \sum_i \sum_j z_i \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle z_j \\
&= \sum_i \sum_j z_i \sum_k \phi_k(\mathbf{x}_i) \phi_k(\mathbf{x}_j) z_j \\
&= \sum_k \sum_i \sum_j z_i \phi_k(\mathbf{x}_i) \phi_k(\mathbf{x}_j) z_j \\
&= \sum_k \left(\sum_i z_i \phi_k(\mathbf{x}_i) \right)^2 \\
&\geq 0
\end{aligned}
$$

Mercer's theorem (simple version)

- Let $K : \mathbb{R}^n \times \mathbb{R}^n \mapsto \mathbb{R}$ be a Kernel function. Then, for K to be a valid (Mercer) kernel it is necessary and sufficient that for any $\{\mathbf{x}_1, \ldots, \mathbf{x}_M\} (m < \infty)$ the corresponding kernel matrix is symmetric and positive semi-definite.

Popular kernel functions

- Gaussian kernel (radial basis)

$$K(\mathbf{x},\mathbf{z}) = \exp\left(-\frac{||\mathbf{x}-\mathbf{z}||^2}{2\sigma^2}\right)$$

- Polinomial kernel of grade d

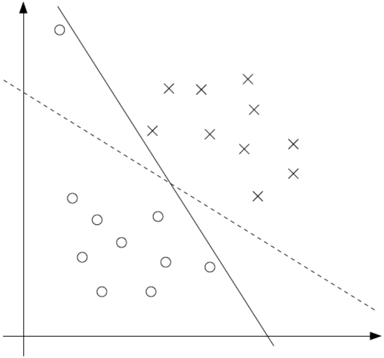
$$K(\mathbf{x},\mathbf{z}) = (\langle \mathbf{x},\mathbf{z}\rangle + 1)^d$$

- Neural Network

$$K(\mathbf{x},\mathbf{z}) = \tanh\left(k_1\langle \mathbf{x},\mathbf{z}\rangle + k_2\right)$$

Non-separable case

- So far we assumed that the data is linearly separable.
- Even in these cases the hyperplane constructed by the learning algorithm is susceptible to outliers.



Reformulating the optimization problem (Soft Margin)

- We introduce slack variables in the optimization problem to relax the linearly separable assumption:

$$\begin{aligned} \min_{\mathbf{w},b} \quad & \frac{1}{2}||\mathbf{w}||^2 + C\sum_{m=1}^{M}\xi_m \\ \text{s.t.} \quad & y_m(\mathbf{w}^T x_m + b) \geq 1 - \xi_m, m = 1, \ldots, M \\ & \xi_m \geq 0, m = 1, \ldots, M \end{aligned} \tag{11}$$

- The parameter C controls the trade-off between the classification errors and the margin size.
- Now, the Lagrangian is given by

$$\mathcal{L}(\mathbf{w}, b, \xi, \alpha, r) = \frac{1}{2}||\mathbf{w}||^2 + C\sum_{m=1}^{M}\xi_m - \sum_{m=1}^{M}\alpha_m\left[y_m(\mathbf{w}^T x_m + b) - 1 + \xi_m\right] - \sum_{m=1}^{M} r_m\xi_r$$

Dual form (norm-1)

- where $\alpha_m \geq 0$ y $r_m \geq 0$
- As before, to find the dual form we have to calculate the derivatives w.r.t. $\mathbf{w}, \xi$ y b

$$\begin{aligned}
\frac{\partial \mathcal{L}(\mathbf{w}, b, \xi, \alpha, r)}{\partial \mathbf{w}} &= \mathbf{w} - \sum_{m=1}^{M} y_m \alpha_m \mathbf{x}_m = 0 \\
\frac{\partial \mathcal{L}(\mathbf{w}, b, \xi, \alpha, r)}{\partial \xi} &= C - \alpha_m - r_m = 0 \\
\frac{\partial \mathcal{L}(\mathbf{w}, b, \xi, \alpha, r)}{\partial b} &= \sum_{i=1}^{m} y_m \alpha_m = 0
\end{aligned}$$

- Replacing these constraints into the primal we obtain

$$L(w, b, \xi, \alpha, r) = \sum_{m=1}^{M} \alpha_i - \frac{1}{2} \sum_{m,\ell} y_m y_\ell \alpha_i \alpha_j \langle \mathbf{x}_m, \mathbf{x}_\ell \rangle$$

Dual form (norm-1) (2)

- Observe that the optimization function is the same as the one without adding the slack variables.
- We only add the restrictions $C - \alpha_m - r_m = 0$ and $r_m \geq 0$. Thus, we have $\alpha_m \leq C$
- And, we obtain the dual optimization problem:

$$\max_{\alpha} \quad \sum_{m=1}^{M} \alpha_m - \frac{1}{2} \sum_{m=1}^{M} \sum_{\ell=1}^{M} \alpha_m \alpha_\ell y_m y_\ell \langle x_m, x_\ell \rangle \tag{12}$$

$$\text{s.t.} \quad 0 \leq \alpha_m^* \leq C, \; m = 1, \ldots, M \tag{13}$$

$$\sum_{m=1}^{M} \alpha_m y_m = 0 \tag{14}$$

KKT conditions

- The KKT conditions for this problem are

$$\begin{aligned} \alpha_m[y_m(\langle x_m, \mathbf{w}\rangle + b) - 1 + \xi_m] = 0, m = 1, \ldots, M \\ \xi_m(\alpha_m - C) = 0, m = 1, \ldots, M \end{aligned}$$

- Observe that $\xi_m \neq 0 \Rightarrow \alpha_m = C$
- The points with $\xi_m \neq 0$ have a margin lower than $1/||\mathbf{w}||$.

SVM as a penalization method

- Note that the optimization problem in (11) has the same solutions than

$$\min_{\mathbf{w},b} \frac{\lambda}{2}||\mathbf{w}||^2 + \sum_{m=1}^{M} (1 - y_m f(\mathbf{x}_m))_+$$

- where $\lambda = \frac{1}{C}$
- Then the soft-margin SVM is a convex program for which the objective function is the hinge loss.

ν-SVM

- A problem is to determine the value of C
- This problem is equivalent to find $0 \leq \nu \leq 1$ in the following optimization problem:

$$
\begin{aligned}
\max\ W(\alpha) &= -\frac{1}{2}\sum_{m=1}^{M}\sum_{\ell=1}^{M}\alpha_m\alpha_\ell y_m y_\ell \\
\text{s.t. :} \quad & \sum_{m=1}^{M} y_m\alpha_m = 0 \\
& \sum_{m=1}^{M}\alpha_m \geq \nu \\
& 1/M \geq \alpha_m \geq 0, m = 1, \ldots, M
\end{aligned}
$$

- It can be obtained that ν is an upper bound on the fraction of margin errors (and hence also on the fraction of training errors).
- Also, ν is a lower bound on the fraction of SVs.

How to solve the optimization problem?

- How to efficiently solve the above optimization problem?
- We have already seen gradient ascent and Newton's method.
- We will study the SMO algorithm (sequential minimal optimization)
- Before of that, we will visit the coordinate ascent algorithm which is also used later in this course.

Coordinate ascent

- Consider the unconstrained optimization problem

$$\max_{\alpha} W(\alpha_1, \alpha_2, \dots, \alpha_p)$$

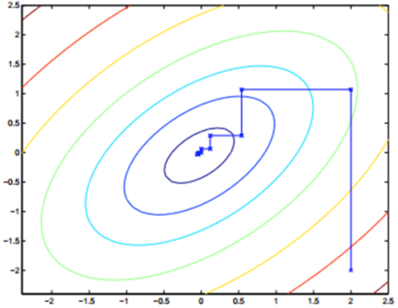
- Coordinate ascent algorithm solve this problem as

Algorithm 3 Coordinate ascent algorithm

1: **repeat**
2: **for** $m = 1$ to M **do**
3: $\alpha_m = \text{argmax}_{\hat{\alpha}_m} W(\alpha_1, \alpha_2, \dots, \alpha_{m-1}, \hat{\alpha}_m, \alpha_{m+1} \dots, \alpha_p)$
4: **end for**
5: **until** convergence

- The order of the variables α_ms can be changed. (choose the one with the largest increase)

Coordinate ascent (2)



- Note that on each step, the algorithm take a step that is parallel to one of the axes (only one variable is being optimized at the time)

SMO

- If we use gradient ascent to optimize the dual problem, according to equation (14) we would have this:

$$\alpha_1 = -y_1 \sum_{m=2}^{M} \alpha_m y_m$$

- We considered the fact that $y_1 \in \{-1, 1\}$.
- Thus, we can't make any change without violating the constraint (14)

Algorithm 4 SMO algorithm

1: **repeat**
2: Select some pair α_i and α_j to update next (using a heuristic that tries to pick the two multipliers that will make the biggest progress towards the maximum)
3: Reoptimize $W(\alpha)$ with respect to α_i and α_j while the rest multipliers are fixed.
4: **until** convergence

SMO (2)

- Let's reoptimize $W(\alpha)$ respect to α_1 and α_2.
- From (14) we need that

$$\alpha_1 y_1 + \alpha_2 y_2 = -\sum_{m=3}^{M} \alpha_m y_m = \zeta$$

- Thus, $\alpha_1 = (\zeta - \alpha_2 y_2) y_1$
- There is a lower bound L and a upper bound H for α_2 that ensures that $\alpha_1, \alpha_2 \in [0, C]$
- Now we can optimize $W(\alpha_1, \alpha_2, \ldots, \alpha_m) = W((\zeta - \alpha_2 y_2) y_1, \alpha_2, \ldots, \alpha_m)$
- Treating $\alpha_3, \ldots, \alpha_m$ as a constants we obtain a quadratic function in α_2

SMO (3)

- Solving it we obtain α_2^{raw}. To ensure that $L \leq \alpha_2 \leq H$:

$$\alpha_2^{new} = \begin{cases} H & \text{if } \alpha_2^{raw} > H \\ \alpha_2^{raw} & \text{if } \leq \alpha_2^{raw} \leq H \\ L & \text{if } \alpha_2^{raw} < L \end{cases}$$

- Now we can compute α_1^{new}
- How to pick α_i, α_j?
- How to compute b^*?

Any questions?