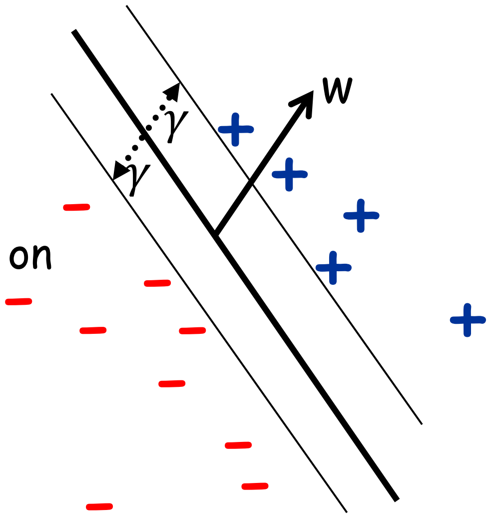
Support Vector Machines (SVMs). Kernelizing SVMs Margin Based Guarantees for SVMs

Maria-Florina Balcan

09/26/2018

Margin Important Theme in ML

- If large margin, # mistakes Peceptron makes is small (independent on the dim of the ambient space)!
- Large margin can help prevent overfitting.
    - If large margin γ and if alg. produces a large margin classifier, then amount of data needed depends only on R/γ e.g., [Bartlett & Shawe-Taylor '99].



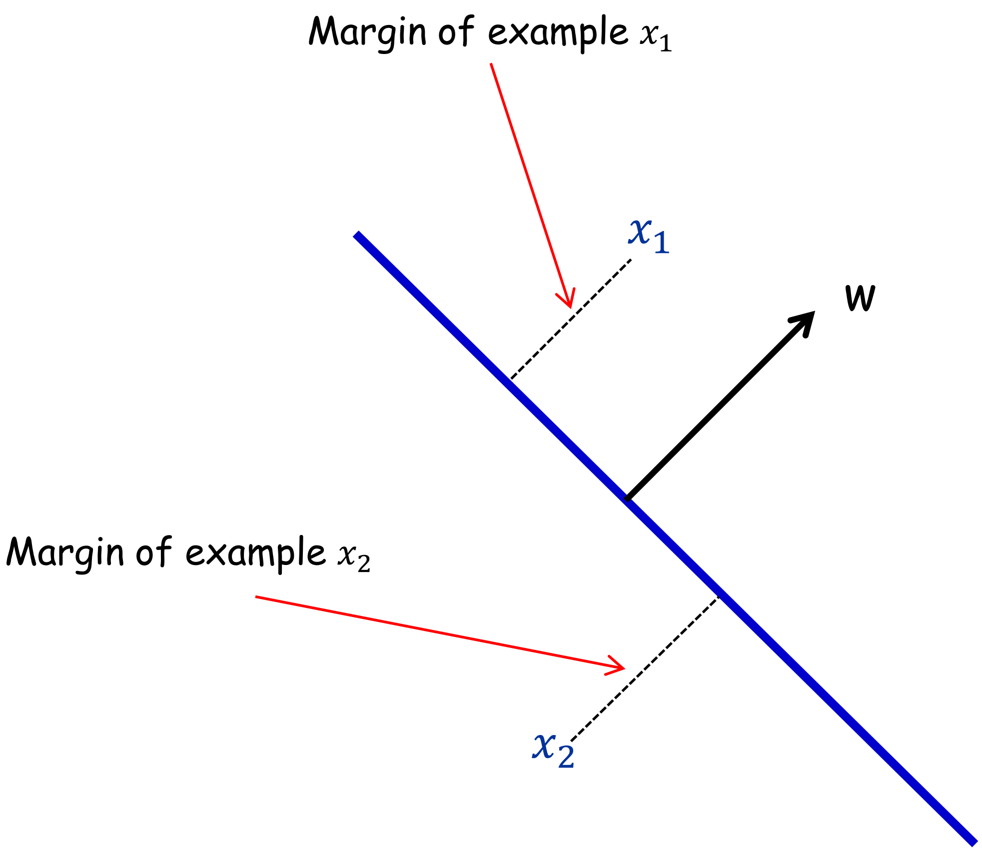
- Idea: Directly search for a large margin classifier!!!

Support Vector Machines (SVMs).

Geometric Margin

WLOG homogeneous linear separators $[w_0 = 0]$.

Definition: The geometric margin of example x w.r.t. a linear sep. w is the distance from x to the plane $w \cdot x = 0$.



If $||w|| = 1$, margin of x w.r.t. w is $|x \cdot w|$.

Support Vector Machines (SVMs)

Directly optimize for the maximum margin separator: SVMs

First, the case where the data is truly linearly separable by margin γ

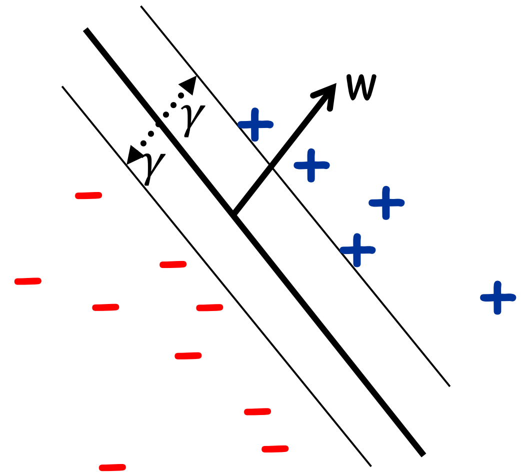
If we know a lower bound on the margin γ

<u>Input</u>: γ, S={(x_1, y_1), …,(x_m, y_m)};

<u>Find</u>: some w where:

- $||w||^2 = 1$
- For all i, $y_i w \cdot x_i \geq \gamma$

<u>Output</u>: w, a separator of margin γ over S



Support Vector Machines (SVMs)

Directly optimize for the maximum margin separator: SVMs

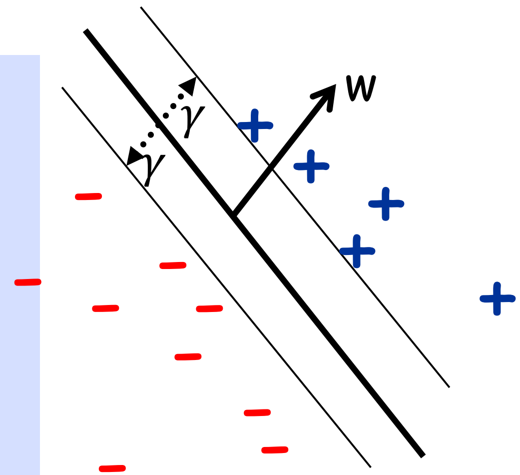
If we know a lower bound on the margin γ, also search for the best possible γ

<u>Input</u>: S=$\{(x_1, y_1), \ldots, (x_m, y_m)\}$;

<u>Find</u>: some w and maximum γ where:

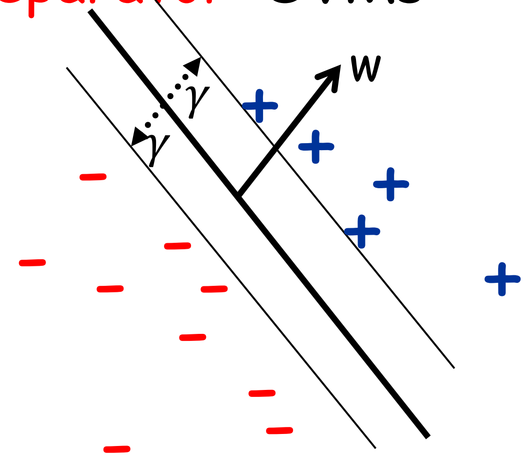
- $||w||^2 = 1$
- For all i, $y_i w \cdot x_i \geq \gamma$

<u>Output</u>: maximum margin separator over S



Support Vector Machines (SVMs)

Directly optimize for the maximum margin separator: SVMs

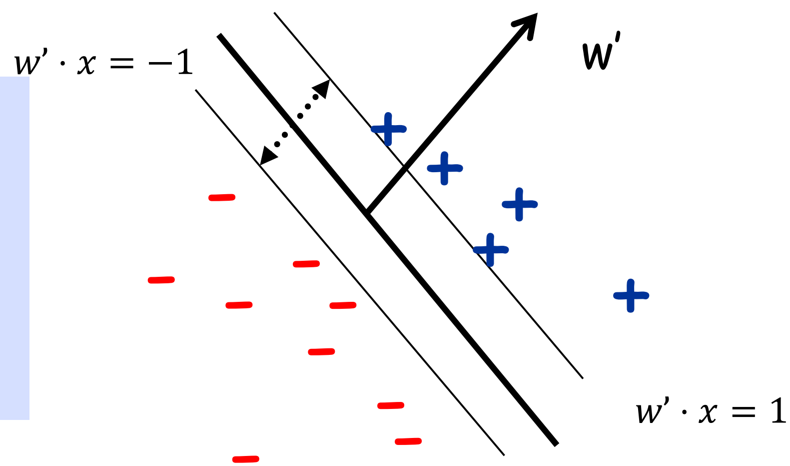


Input: S={(x_1, y_1), …,(x_m, y_m)};

Maximize γ under the constraint:

- $||w||^2 = 1$
- For all i, $y_i w \cdot x_i \geq \gamma$

$w' = w/\gamma$, then max γ is equiv. to minimizing $||w'||^2$ (since $||w'||^2 = 1/\gamma^2$).
So, dividing both sides by γ and writing in terms of w′ we get:



Input: S={(x_1, y_1), …,(x_m, y_m)};

Minimize $||w'||^2$ under the constraint:

- For all i, $y_i w' \cdot x_i \geq 1$

Support Vector Machines (SVMs)

Directly optimize for the maximum margin separator: SVMs

Input: S={$(x_1, y_1), \ldots, (x_m, y_m)$};

$\text{argmin}_w \, ||w||^2$ s.t.:

- For all i, $y_i w \cdot x_i \geq 1$

This is a **constrained convex optimization** problem.

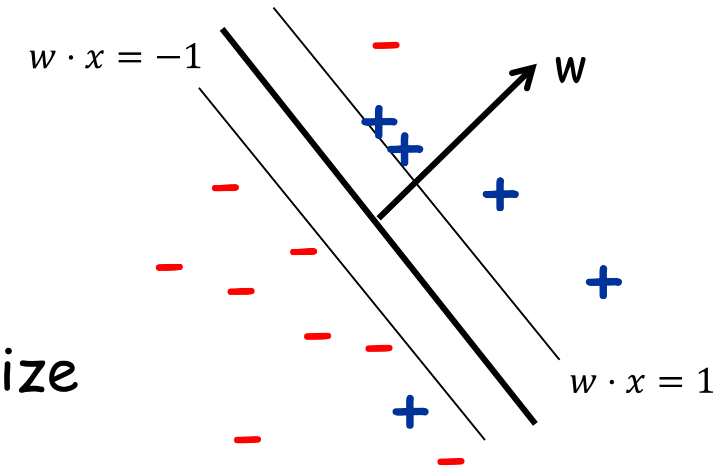
- The objective is convex (quadratic)
- All constraints are linear
- Can solve efficiently (in poly time) using standard **quadratic programing** (QP) software

Support Vector Machines (SVMs)

Question: what if data isn't perfectly linearly separable?

Issue 1: now have two objectives

- maximize margin
- minimize # of misclassifications.



Ans 1: Let's optimize their sum: minimize

$$||w||^2 + C(\#\ \text{misclassifications})$$

where C is some tradeoff constant.

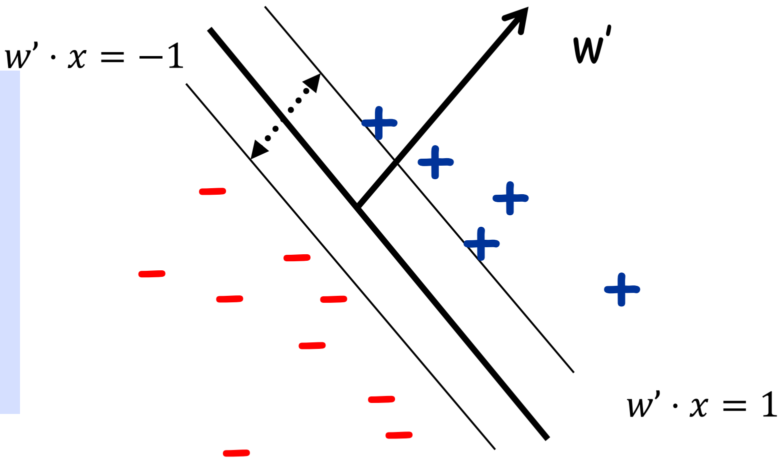
Issue 2: This is computationally very hard (NP-hard).

[even if didn't care about margin and minimized # mistakes]

Support Vector Machines (SVMs)

Question: what if data isn't perfectly linearly separable?

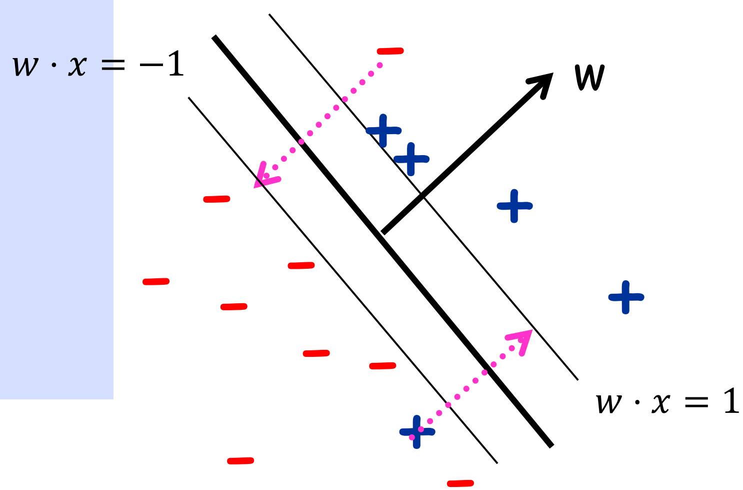
Replace "# mistakes" with upper bound called "hinge loss"



Input: S={(x_1, y_1), …,(x_m, y_m)};

Minimize $||w'||^2$ under the constraint:

- For all i, $y_i w' \cdot x_i \geq 1$



Input: S={(x_1, y_1), …,(x_m, y_m)};

Find $\mathrm{argmin}_{w,\xi_1,\ldots,\xi_m} ||w||^2 + C \sum_i \xi_i$ s.t.:

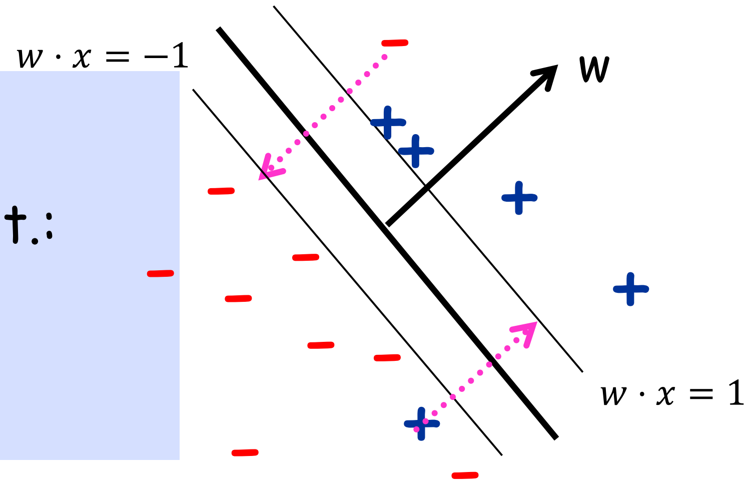
- For all i, $y_i w \cdot x_i \geq 1 - \xi_i$

  $\xi_i \geq 0$

ξ_i are "slack variables"

Support Vector Machines (SVMs)

Question: what if data isn't perfectly linearly separable?
Replace "# mistakes" with upper bound called "hinge loss"



Input: S={(x_1, y_1), …,(x_m, y_m)};

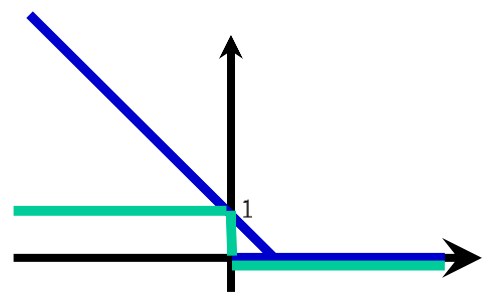
Find $\text{argmin}_{w,\xi_1,\ldots,\xi_m} ||w||^2 + C \sum_i \xi_i$ s.t.:

- For all i, $y_i w \cdot x_i \geq 1 - \xi_i$

  $\xi_i \geq 0$

ξ_i are "slack variables"

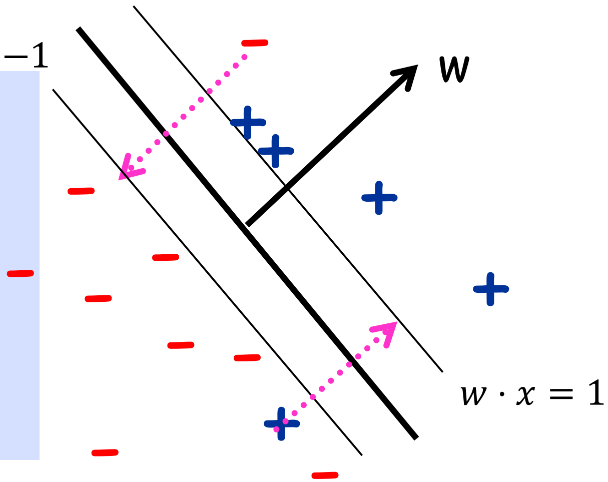
C controls the relative weighting between the twin goals of making the $||w||^2$ small (margin is large) and ensuring that most examples have functional margin ≥ 1.



$l(w, x, y) = \max(0, 1 - y\, w \cdot x)$

Support Vector Machines (SVMs)

Question: what if data isn't perfectly linearly separable?
Replace "# mistakes" with upper bound called "hinge loss"

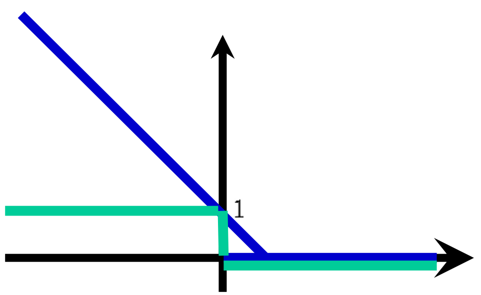


Input: S={(x_1, y_1), …,(x_m, y_m)};

Find $\mathrm{argmin}_{w,\xi_1,\ldots,\xi_m} ||w||^2 + C \sum_i \xi_i$ s.t.:

- For all i, $y_i w \cdot x_i \geq 1 - \xi_i$
  $\xi_i \geq 0$

Total amount have to move the points to get them on the correct side of the lines $w \cdot x = +1/-1$, where the distance between the lines $w \cdot x = 0$ and $w \cdot x = 1$ counts as "1 unit".



$l(w, x, y) = \max(0, 1 - y\, w \cdot x)$

Support Vector Machines (SVMs)

<u>Input</u>: S={(x_1, y_1), …,(x_m, y_m)};

<u>Find</u> $\mathrm{argmin}_{w,\xi_1,\ldots,\xi_m} ||w||^2 + C \sum_i \xi_i$ s.t.:

- For all i, $y_i w \cdot x_i \geq 1 - \xi_i$

$$\xi_i \geq 0$$

Primal form

Which is equivalent to:

Can be kernelized!!!

<u>Input</u>: S={(x_1, y_1), …,(x_m, y_m)};

<u>Find</u> $\mathrm{argmin}_{\alpha} \frac{1}{2} \sum_i \sum_j y_i y_j \alpha_i \alpha_j x_i \cdot x_j - \sum_i \alpha_i$ s.t.:

- For all i, $0 \leq \alpha_i \leq C_i$

$$\sum_i y_i \alpha_i = 0$$

Lagrangian Dual

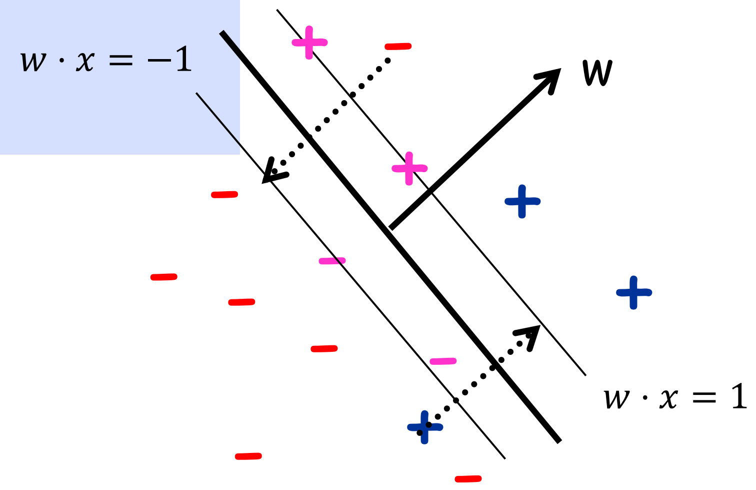
SVMs (Lagrangian Dual)

Input: $S=\{(x_1, y_1), \ldots, (x_m, y_m)\}$;

Find $\mathrm{argmin}_\alpha \frac{1}{2} \sum_i \sum_j y_i y_j \alpha_i \alpha_j x_i \cdot x_j - \sum_i \alpha_i$ s.t.:

- For all i, $0 \le \alpha_i \le C_i$

$$\sum_i y_i \alpha_i = 0$$



- Final classifier is: $w = \sum_i \alpha_i y_i x_i$
- The points x_i for which $\alpha_i \neq 0$ are called the "support vectors"

Margin Based Guarantees for SVMs

VC-based bounds of linear separators

- Learning guarantees: for linear separators in N-dimensional space, with probability at least $1-\delta$,

$$R(h) \leq \widehat{R}(h) + \sqrt{\frac{2(N+1)\log\frac{em}{N+1}}{m}} + \sqrt{\frac{\log\left(\frac{1}{\delta}\right)}{2m}}$$

- Bound is uninformative for $N \gg m$.
- But SVMs have been remarkably successful in high dimensions.
- Can provide a theoretical justification via margin based bounds.

Rademacher Complexity of Linear Hypotheses

Theorem: Let $S \subseteq \{x\colon \|x\| \leq R\}$ be a sample of size m and let $H = \{x \to w \cdot x\colon \|w\| \leq \Lambda\}$. Then,

$$\widehat{\mathfrak{R}}_S(H) \leq \sqrt{\frac{R^2\Lambda^2}{m}}.$$

Proof:

$$\widehat{\mathfrak{R}}_S(H) = \frac{1}{m}\mathrm{E}_\sigma\left[\sup_{\|w\|\leq\Lambda}\sum_{i=1}^{m}\sigma_i w \cdot x_i\right] = \frac{1}{m}\mathrm{E}_\sigma\left[\sup_{\|w\|\leq\Lambda} w \cdot \sum_{i=1}^{m}\sigma_i x_i\right]$$

$$\leq \frac{\Lambda}{m}\mathrm{E}_\sigma\left[\|\sum_{i=1}^{m}\sigma_i x_i\|\right] \leq \frac{\Lambda}{m}\left[\mathrm{E}_\sigma\left[\|\sum_{i=1}^{m}\sigma_i x_i\|^2\right]\right]^{1/2}$$

$$\frac{\Lambda}{m}\left[\mathrm{E}_\sigma\left[\sum_{i=1}^{m}\|x_i\|^2\right]\right]^{1/2} \leq \frac{\Lambda\sqrt{mR^2}}{m} \leq \sqrt{\frac{R^2\Lambda^2}{m}}$$

Confidence Margin

Definition: the confidence functional margin of a real-valued function h at $(x, y) \in X \times Y$ is $\rho_h(x, y) = yh(x)$.

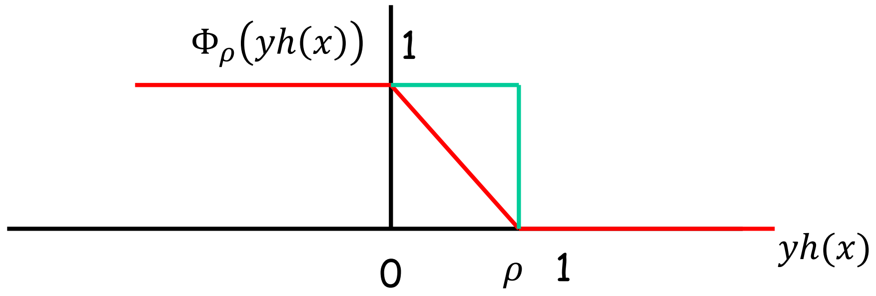
- Interpreted as the confidence of h in its prediction.
- If correctly classified, coincides with $|h(x)|$.

Relationship with geometric margin for linear functions $h: x \to w \cdot x$, for x in the sample,

$$|\rho_h(x, y)| = \rho_{geom} \, \|w\|$$

Confidence Margin Loss

Definition: for any confidence margin parameter $\rho > 0$, the ρ-margin loss function Φ_ρ is defined by



For a sample $S = (x_1, \ldots, x_m)$ and real-valued hypothesis h, the empirical margin loss is

$$\widehat{R}_\rho(h) = \frac{1}{m} \sum_{i=1}^{m} \Phi_\rho\big(y_i h(x_i)\big) \leq \boxed{\frac{1}{m} \sum_{i=1}^{m} 1_{y_i h(x_i) < \rho}}$$

General Margin Bound

- **Theorem:** Let H be a set of real-valued functions. Fix $\rho > 0$. For any $\delta > 0$, with probability at least $1 - \delta$, the following holds for all $h \in H$:

$$R(h) \leq \widehat{R}_\rho(h) + \frac{2}{\rho}\mathfrak{R}_m(H) + \sqrt{\frac{\log\frac{1}{\delta}}{2m}}$$

$$R(h) \leq \widehat{R}_\rho(h) + \frac{2}{\rho}\widehat{\mathfrak{R}}_S(H) + 3\sqrt{\frac{\log\frac{2}{\delta}}{2m}}$$

- **Proof:** Let $\widetilde{H} = \{z = (x, y) \to yh(x): h \in H\}$. Consider the family of functions taking values in $[0,1]$:

$$\widetilde{\boldsymbol{H}} = \{\Phi_\rho \circ f: f \in \widetilde{H}\}$$

- By the theorem of Lecture 3, with probability at least $1-\delta$, for all $g \in \widetilde{\boldsymbol{H}}$,

$$E[g(z)] \leq \frac{1}{m}\sum_{i=1}^{m} g(z_i) + 2\mathfrak{R}_m(\widetilde{\boldsymbol{H}}) + \sqrt{\frac{\log\frac{1}{\delta}}{2m}}$$

- Thus,

$$E\big[\Phi_\rho\big(yh(x)\big)\big] \leq \hat{R}_\rho(h) + 2\mathfrak{R}_m\big(\Phi_\rho \circ \widetilde{H}\big) + \sqrt{\frac{\log\frac{1}{\delta}}{2m}}$$

- Since Φ_ρ is $\frac{1}{\rho}$ - Lipschitz, by Talagrand's lemma,

$$\mathfrak{R}_m\big(\Phi_\rho \circ \widetilde{H}\big) \leq \frac{1}{\rho}\mathfrak{R}_m\big(\widetilde{H}\big) = \frac{1}{\rho m} E_{\sigma,S}\Big[\sup_{h\in H} \textstyle\sum_{i=1}^{m} \sigma_i y_i h(x_i)\Big] = \frac{1}{\rho}\,\mathfrak{R}_m(H)$$

- Since $1_{yh(x)<0} \leq \Phi_\rho(yh(x))$, this shows the first statement, and similarly the second one.

Margin Bound – Linear Classifiers

- **Corollary:** Let $\rho > 0$ and $H = \{x \to w \cdot x : \|w\| \leq \Lambda\}$. Assume that $X \subseteq \{x : \|x\| \leq R\}$. Then, for any $\delta > 0$, with probability at least $1 - \delta$, for any $h \in H$:

$$R(h) \leq \hat{R}_\rho(h) + 2\sqrt{\frac{R^2\Lambda^2/\rho^2}{m}} + 3\sqrt{\frac{\log\frac{2}{\delta}}{2m}}$$

- **Proof:** Follows directly the general margin bound and the bound on $\widehat{\mathfrak{R}}_S(H)$ for linear classifiers.

High-Dimensional Feature Space

- **Observations:**
  - Generalization bound does not depend on the dimension but only on the margin.
  - This suggests seeking a large-margin separating hyperplane in a higher-dimensional feature space.
- **Computational problems:**
  - Taking dot products in a high-dimensional feature space can be very costly.
  - Solution based on kernels.

What you should know

- The importance of margins in machine learning.
- The SVM algorithm. Primal and Dual Form.
- Kernelizing SVM.
- Margin Based Bounds for SVM.

Additional Slides

Lagrange duality

SVM Dual

Lagrange duality

Consider the following "primal" optimization problem:

$$\min_w f(w)$$

subject to $g_i(w) \leq 0$ for $i = 1,2,\ldots,k$

To solve it, we define the Lagrangian:

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

where the $\alpha_i \geq 0$ are called **Lagrange multiplers**.

(Conceptually, think of α_i as penalties for violating the $g_i(w) \leq 0$ constraints)

Now consider (P is for "primal"): $\Theta_P(w) = \max_{\alpha:\alpha_i \geq 0} L(w,\alpha)$

Note that if w violates any $g_i(w) \leq 0$ constraints then $\Theta_P(w) = \infty$ (set α_i to ∞). Else, $\Theta_P(w) = f(w)$ (set all α to 0).

Lagrange duality

Consider the following "primal" optimization problem:

$$\min_{w} f(w)$$

subject to $g_i(w) \leq 0$ for $i = 1,2, \dots, k$

$\Theta_P(w) = \max_{\alpha:\alpha_i \geq 0} L(w, \alpha)$ where $L(w, \alpha) = f(w) + \sum_{i=1}^{k} \alpha_i g_i(w)$

Summarizing:

$$\Theta_P(w) = \begin{cases} f(w) & \textit{if all } g_i(w) \leq 0 \textit{ satisfied} \\ \infty & \textit{if any } g_i(w) \leq 0 \textit{ violated} \end{cases}$$

So, our original problem is equivalent to:

$$\min_{\mathrm{w}} \Theta_P(w) = \min_{w} \max_{\alpha:\alpha_i \geq 0} L(w, \alpha)$$

Lagrange duality

Consider the following "primal" optimization problem:

$$\min_{w} f(w)$$

subject to $g_i(w) \leq 0$ for $i = 1,2, \dots, k$

$$\Theta_P(w) = \max_{\alpha:\alpha_i \geq 0} L(w, \alpha) \quad \text{where} \quad L(w, \alpha) = f(w) + \sum_{i=1}^{k} \alpha_i g_i(w)$$

Our original pb equivalent to: $\min_{\mathrm{w}} \Theta_P(w) = \min_{w} \max_{\alpha:\alpha_i \geq 0} L(w, \alpha)$

Lagrange duality

Consider the following "primal" optimization problem:

$$\min_{w} f(w)$$

subject to $g_i(w) \leq 0$ for $i = 1,2, \dots, k$

$$\Theta_P(w) = \max_{\alpha:\alpha_i \geq 0} L(w, \alpha) \quad \text{where} \quad L(w, \alpha) = f(w) + \sum_{i=1}^{k} \alpha_i g_i(w)$$

Our original pb equivalent to: $\min_{\mathrm{w}} \Theta_P(w) = \min_{w} \max_{\alpha:\alpha_i \geq 0} L(w, \alpha)$

Consider a different function (D is for "dual"):

$$\Theta_D(\alpha) = \min_{w} L(w, \alpha)$$

We can now pose the dual optimization problem:

$$\max_{\alpha:\alpha_i \geq 0} \Theta_D(\alpha) = \max_{\alpha:\alpha_i \geq 0} \min_{w} L(w, \alpha)$$

[the order of "max" and "min" has been swapped]

Relation between primal and dual

Consider the following "primal" optimization problem:

$$\min_w f(w)$$

subject to $g_i(w) \leq 0$ for $i = 1,2, \dots, k$

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

Primal

$$\min_{\mathrm{w}} \Theta_P(w) = \min_w \max_{\alpha:\alpha_i \geq 0} L(w,\alpha)$$

p^*optimal primal value:

$$p^* = \min_w \Theta_P(w) = \min_w \max_{\alpha:\alpha_i \geq 0} L(w,\alpha)$$

Dual

$$\max_{\alpha:\alpha_i \geq 0} \Theta_D(\alpha) = \max_{\alpha:\alpha_i \geq 0} \min_w L(w,\alpha)$$

d^* optimal dual value:

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

Simple to show $d^* \leq p^*$ (max min ≤ min max)

Under appropriate conditions (e.g., f and g_i are convex functions), $d^* = p^*$.

So, can solve dual instead of primal.

Sufficient conditions for $d^* = p^*$

Suppose f and g_i are convex functions.

Suppose $\exists w$ s.t. $g_i(w) < 0$ for all i. (constraints strictly feasible)

Then there exist w^*, α^* such that w^* is solution to primal, α^* is solution to dual, and $d^* = p^* = L(w^*, \alpha^*)$.

Furthermore, w^*, α^* satisfy Karush-Kuhn-Tucker (KKT) conditions:

- $\frac{\partial}{\partial w_i} L(w^*, \alpha^*) = 0$ for all i.
- $\alpha_i^* g_i(w^*) = 0$ for all i.
- $g_i(w^*) \leq 0$ for all i.
- $\alpha_i^* \geq 0$ for all i.

And, any solution to KKT conditions is optimal for primal & dual.

Sufficient conditions for $d^* = p^*$

Suppose f and g_i are convex functions.

Suppose $\exists w$ s.t. $g_i(w) < 0$ for all i. (constraints strictly feasible)

Then there exist w^*, α^* such that w^* is solution to primal, α^* is solution to dual, and $d^* = p^* = L(w^*, \alpha^*)$.

Furthermore, w^*, α^* satisfy Karush-Kuhn-Tucker (KKT) conditions:

- $\frac{\partial}{\partial w_i} L(w^*, \alpha^*) = 0$ for all i.
- $\boxed{\alpha_i^* g_i(w^*) = 0 \text{ for all } i.}$
- $g_i(w^*) \leq 0$ for all i.
- $\alpha_i^* \geq 0$ for all i.

KKT dual complementarity

If $\alpha_i^* > 0$ then $g_i(w^*) = 0$, i.e., this constraint is "tight".

And, any solution to KKT conditions is optimal for primal & dual.

Support Vector Machines (SVMs)

Primal optimization:

$$\min \frac{1}{2}||w||^2$$

subject to $y_i(w \cdot x_i) \geq 1$ for all i.

Rewrite constraints as: $g_i(w) = 1 - y_i(w \cdot x_i) \leq 0$ for all i.

So, the Lagrangian is: $L(w, \alpha) = \frac{1}{2}||w||^2 + \sum_i \alpha_i (1 - y_i(w \cdot x_i))$

Let's now solve for the dual: $\Theta_D(\alpha) = \min_w L(w, \alpha)$

To do this, we set $\nabla_w L(w, \alpha) = 0$:

$w - \sum_i \alpha_i y_i x_i = 0$ which means $w = \sum_i \alpha_i y_i x_i$

Support Vector Machines (SVMs)

Plugging our solution $w = \sum_i \alpha_i y_i \, x_i$ back into the Lagrangian equation:

$$L(w, \alpha) = \frac{1}{2}||w||^2 + \sum_i \alpha_i (1 - y_i (w \cdot x_i))$$

and simplifying, we get:

$$L(w, \alpha) = \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} y_i y_j \alpha_i \alpha_j (x_i \cdot x_j)$$

<u>Find</u> $\mathrm{argmin}_\alpha \frac{1}{2} \sum_i \sum_j y_i y_j \, \alpha_i \alpha_j x_i \cdot x_j - \sum_i \alpha_i$ s.t.:

- For all i, $\alpha_i \geq 0$

Lagrangian Dual

Support Vector Machines (SVMs)

Input: S={(x_1, y_1), …,(x_m, y_m)};

Find $\operatorname{argmin}_{w,\xi_1,\ldots,\xi_m} ||w||^2 + C\sum_i \xi_i$ s.t.:

- For all i, $y_i w \cdot x_i \geq 1 - \xi_i$

  $\xi_i \geq 0$

Primal form

Which is equivalent to:

Can be kernelized!!!

Input: S={(x_1, y_1), …,(x_m, y_m)};

Find $\operatorname{argmin}_{\alpha} \frac{1}{2}\sum_i \sum_j y_i y_j \alpha_i \alpha_j x_i \cdot x_j - \sum_i \alpha_i$ s.t.:

- For all i, $0 \leq \alpha_i \leq C_i$

  $$\sum_i y_i \alpha_i = 0$$

Lagrangian Dual

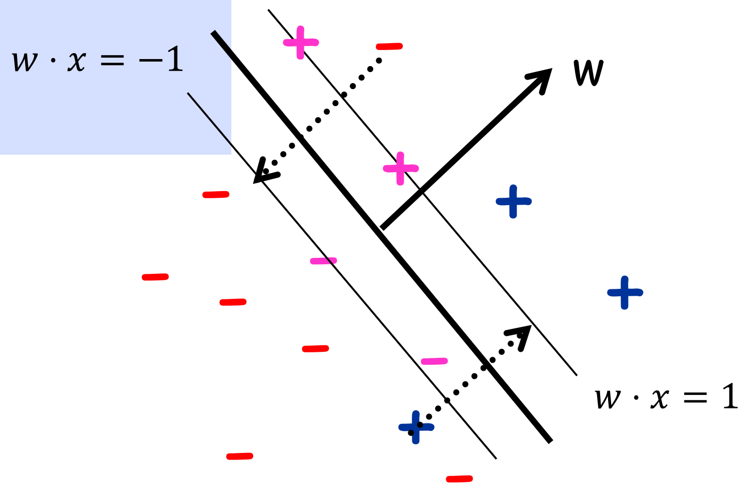
SVMs (Lagrangian Dual)

Input: S={(x_1, y_1), …,(x_m, y_m)};

Find $\operatorname{argmin}_{\alpha} \frac{1}{2} \sum_i \sum_j y_i y_j \alpha_i \alpha_j x_i \cdot x_j - \sum_i \alpha_i$ s.t.:

- For all i, $0 \leq \alpha_i \leq C_i$

$$\sum_i y_i \alpha_i = 0$$



- Final classifier is: $w = \sum_i \alpha_i y_i x_i$
- The points x_i for which $\alpha_i \neq 0$ are called the "support vectors"