Machine Learning

B. Supervised Learning: Nonlinear Models
B.4. Support Vector Machines (SVMs)

Lars Schmidt-Thieme

Information Systems and Machine Learning Lab (ISMLL)
Institute for Computer Science
University of Hildesheim, Germany

Syllabus

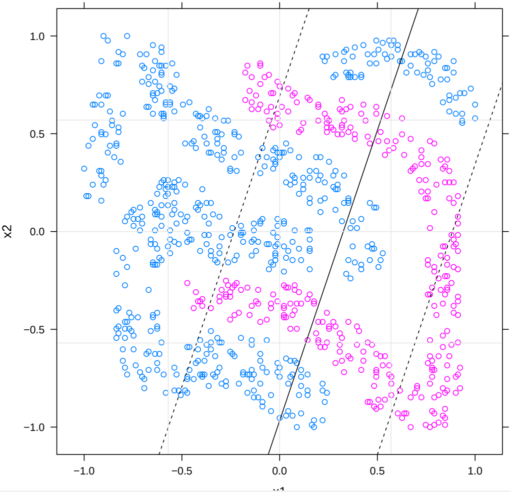
Fri. 26.10.	(1)	0. Introduction
		A. Supervised Learning: Linear Models & Fundamentals
Fri. 2.11.	(2)	A.1 Linear Regression
Fri. 9.11.	(3)	A.2 Linear Classification
Fri. 16.11.	(4)	A.3 Regularization
Fri. 23.11.	(5)	A.4 High-dimensional Data
		B. Supervised Learning: Nonlinear Models
Fri. 30.11.	(6)	B.1 Nearest-Neighbor Models
Fri. 7.12.	(7)	B.2 Neural Networks
Fri. 14.12.	(8)	B.3 Decision Trees
Fri. 21.12.	(9)	B.4 Support Vector Machines
		— *Christmas Break* —
Fri. 11.1.	(10)	B.5 A First Look at Bayesian and Markov Networks
		C. Unsupervised Learning
Fri. 18.1.	(11)	C.1 Clustering
Fri. 25.1.	(12)	C.2 Dimensionality Reduction
Fri. 1.2.	(13)	C.3 Frequent Pattern Mining
Fri. 8.2.	(14)	Q&A

Outline

Outline

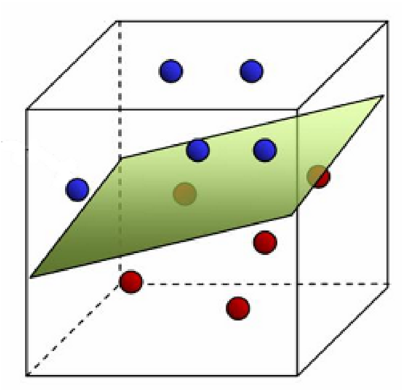
Separating Hyperplanes

Logistic Regression & Linear Discriminant Analysis (LDA):



Separating Hyperplanes

Logistic Regression & Linear Discriminant Analysis (LDA):



linear decision boundary = separating hyperplane

source: http://fouryears.eu/2009/

Hyperplanes

Hyperplanes can be modeled explicitly as

$$H_{\beta,\beta_0} := \{x \,|\, \langle \beta, x\rangle = -\beta_0\}, \quad \beta = \begin{pmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_M \end{pmatrix} \in \mathbb{R}^M, \beta_0 \in \mathbb{R}$$

We will write H_β shortly for H_{β,β_0} (although β_0 is very relevant!).

For any two points $x, x' \in H_\beta$ we have

$$\langle \beta, x - x'\rangle = \langle \beta, x\rangle - \langle \beta, x'\rangle = -\beta_0 + \beta_0 = 0$$

thus β is orthogonal to all translation vectors in H_β,
and thus $\beta/||\beta||$ is the **normal vector** of H_β.

Hyperplanes

The projection of a point $x \in \mathbb{R}^M$ onto H_β, i.e., the closest point on H_β to x is given by

$$\pi_{H_\beta}(x) := x - \frac{\langle \beta, x\rangle + \beta_0}{\langle \beta, \beta\rangle}\beta$$

Proof:
(i) $\pi x := \pi_{H_\beta}(x) \in H_\beta$:

$$\begin{aligned}\langle \beta, \pi_{H_\beta}(x)\rangle =& \langle \beta, x - \frac{\langle \beta, x\rangle + \beta_0}{\langle \beta, \beta\rangle}\beta\rangle \\ =& \langle \beta, x\rangle - \frac{\langle \beta, x\rangle + \beta_0}{\langle \beta, \beta\rangle}\langle \beta, \beta\rangle = -\beta_0\end{aligned}$$

(ii) $\pi_{H_\beta}(x)$ is the closest such point to x:
For any other point $x' \in H_\beta$:

$$\begin{aligned}||x - x'||^2 =& \langle x - x', x - x'\rangle = \langle x - \pi x + \pi x - x', x - \pi x + \pi x - x'\rangle \\ =& \langle x - \pi x, x - \pi x\rangle + 2\langle x - \pi x, \pi x - x'\rangle + \langle \pi x - x', \pi x - x'\rangle \\ =& ||x - \pi x||^2 + 0 + ||\pi x - x'||^2\end{aligned}$$

Hyperplanes

The **signed distance** of a point $x \in \mathbb{R}^M$ to H_β is given by

$$\frac{\langle \beta, x \rangle + \beta_0}{||\beta||}$$

Proof:

$$x - \pi x = \frac{\langle \beta, x \rangle - \beta_0}{\langle \beta, \beta \rangle} \beta$$

Therefore

$$\begin{aligned} ||x - \pi x||^2 =& \langle \frac{\langle \beta, x \rangle + \beta_0}{\langle \beta, \beta \rangle} \beta, \frac{\langle \beta, x \rangle + \beta_0}{\langle \beta, \beta \rangle} \beta \rangle \\ =& (\frac{\langle \beta, x \rangle + \beta_0}{\langle \beta, \beta \rangle})^2 \langle \beta, \beta \rangle \\ ||x - \pi x|| =& |\frac{\langle \beta, x \rangle + \beta_0}{||\beta||}| \end{aligned}$$

Separating Hyperplanes

For given data

$$(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)$$

with a binary class label $Y \in \{-1, +1\}$
a hyperplane H_β is called **separating** if

$$\begin{aligned} h(x_n) &> 0, \quad \text{if } y_n = +1, \quad n = 1, \ldots, N \\ h(x_n) &< 0, \quad \text{if } y_n = -1 \end{aligned}$$

or briefer, but equivalently:

$$y_n h(x_n) > 0, \quad n = 1, \ldots, N, \quad \text{with } h(x) := \langle \beta, x \rangle + \beta_0$$

Linear Separable Data

- A data set is called **linear separable** if there exists such a separating hyperplane.
- In general, if there is one, there are many.
- If there is a choice, we need a criterion to narrow down which one we want / is the best.

Outline

Perceptron as Linear Model

Perceptron is another name for a linear binary classification model (Rosenblatt 1958):

$$Y(X) = \operatorname{sign} h(X), \quad \text{with } \operatorname{sign} x := \begin{cases} +1, & x > 0 \\ 0, & x = 0 \\ -1, & x < 0 \end{cases}$$

$$h(X) = \beta_0 + \langle \beta, X \rangle + \epsilon$$

that is very similar to the logistic regression model

$$Y(X) = \arg\max_y p(Y = y \mid X) = \operatorname{sign} p(Y = y \mid X) - 0.5$$

$$p(Y = +1 \mid X) = \operatorname{logistic}(\langle \beta, X \rangle)$$

$$p(Y = -1 \mid X) = 1 - p(Y = +1 \mid X)$$

as well as to linear discriminant analysis (LDA).

Perceptron as Linear Model

- The perceptron does just provide class labels $\hat{y}(x)$ and unscaled certainty factors $\hat{h}(x)$, but no class probabilities $\hat{p}(Y \mid X)$.
- Therefore, probabilistic fit/error criteria such as maximum likelihood cannot be applied.
- For perceptrons, the

  sum of the certainty factors of misclassified points

  is used as error criterion:

$$q(\beta, \beta_0) := \sum_{n=1:\hat{y}_n \neq y_n}^{N} |h_\beta(x_n)| = -\sum_{n=1:\hat{y}_n \neq y_n}^{N} y_n h_\beta(x_n)$$

Perceptron as Linear Model

- use gradient descent to learn the model:

$$\frac{\partial q(\beta, \beta_0)}{\partial \beta} = - \sum_{n=1:\hat{y}_n \neq y_n}^{N} y_n x_n$$

$$\frac{\partial q(\beta, \beta_0)}{\partial \beta_0} = - \sum_{n=1:\hat{y}_n \neq y_n}^{N} y_n$$

- Instead of looking at all points at the same time, stochastic gradient descent is applied where all points are looked at sequentially (in random order).

- The update for a single point (x_n, y_n) then is

$$\hat{\beta}^{(k+1)} := \hat{\beta}^{(k)} + \mu y_n x_n$$

$$\hat{\beta}_0^{(k+1)} := \hat{\beta}_0^{(k)} + \mu y_n$$

with a step length μ (often called **learning rate**).

Perceptron Learning Algorithm

```
learn-perceptron(training data D^train, step length μ):
  β̂ := a random vector
  β̂_0 := a random value
  errors := 1
  while errors > 0:
    errors := 0
    for (x, y) ∈ D^train (in random order):
      if y(β̂_0 + ⟨β̂, x⟩) ≤ 0:
        errors := errors + 1
        β̂ := β̂ + μyx
        β̂_0 := β̂_0 + μy
  return (β̂, β̂_0)
```

Perceptron Learning Algorithm: Properties

For linear separable data the perceptron learning algorithm can be shown to converge: it finds a separating hyperplane in a finite number of steps.

But there are many problems with this simple algorithm:

- If there are several separating hyperplanes, there is no control about which one is found (it depends on the starting values).
- If the gap between the classes is narrow, it may take many steps until convergence.
- If the data are not separable, the learning algorithm does not converge at all.

Outline

Maximum Margin Separating Hyperplanes

Many of the problems of perceptrons can be overcome by designing a better fit/error criterion.

Maximum Margin Separating Hyperplanes use the width of the margin, i.e., the distance of the closest points to the hyperplane as criterion:

$$\begin{aligned} &\text{maximize } C \\ \text{w.r.t. } y_n \frac{\beta_0 + \langle \beta, x_n \rangle}{||\beta||} &\geq C, \quad n = 1, \ldots, N \\ \beta &\in \mathbb{R}^M \\ \beta_0 &\in \mathbb{R} \end{aligned}$$

Maximum Margin Separating Hyperplanes

As for any solutions β, β_0 also all positive scalar multiples fullfil the equations, we can arbitrarily set

$$||\beta|| = \frac{1}{C}$$

Then the problem can be reformulated as

$$\begin{aligned} \text{minimize } & \frac{1}{2}||\beta||^2 \\ \text{w.r.t. } y_n(\beta_0 + \langle \beta, x_n \rangle) \geq & 1, \quad n = 1, \ldots, N \\ \beta \in & \mathbb{R}^M \\ \beta_0 \in & \mathbb{R} \end{aligned}$$

This problem is a convex optimization problem
(quadratic target function with linear inequality constraints).

Quadratic Optimization

- To get rid of the linear inequality constraints, one usually applies Lagrange multipliers.
- The Lagrange (primal) function of this problem is

$$L := \frac{1}{2}||\beta||^2 - \sum_{n=1}^{N} \alpha_n (y_n(\beta_0 + \langle \beta, x_n \rangle) - 1)$$

w.r.t. $\alpha_n \geq 0$

- For an extremum it is required that

$$\frac{\partial L}{\partial \beta} = \beta - \sum_{n=1}^{N} \alpha_n y_n x_n \stackrel{!}{=} 0 \quad \rightsquigarrow \quad \beta = \sum_{n=1}^{N} \alpha_n y_n x_n$$

and

$$\frac{\partial L}{\partial \beta_0} = -\sum_{n=1}^{N} \alpha_n y_n \stackrel{!}{=} 0$$

Quadratic Optimization

Input

$$\beta = \sum_{n=1}^{N} \alpha_n y_n x_n, \quad \sum_{n=1}^{N} \alpha_n y_n = 0$$

into

$$L := \frac{1}{2}||\beta||^2 - \sum_{n=1}^{N} \alpha_n (y_n(\beta_0 + \langle \beta, x_n \rangle) - 1)$$

yields the **dual problem**

$$\begin{aligned}
L =& \frac{1}{2}\langle \sum_{n=1}^{N} \alpha_n y_n x_n, \sum_{m=1}^{N} \alpha_m y_m x_m \rangle - \sum_{n=1}^{N} \alpha_n (y_n(\beta_0 + \langle \sum_{m=1}^{N} \alpha_m y_m x_m, x_n \rangle) - 1) \\
=& \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} \alpha_n \alpha_m y_n y_m \langle x_n, x_m \rangle + \sum_{n=1}^{N} \alpha_n - \sum_{n=1}^{N} \alpha_n y_n \beta_0 - \sum_{n=1}^{N} \sum_{m=1}^{N} \alpha_n \alpha_m y_n y_m \\
=& -\frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} \alpha_n \alpha_m y_n y_m \langle x_n, x_m \rangle + \sum_{n=1}^{N} \alpha_n
\end{aligned}$$

Quadratic Optimization

The dual problem is

$$
\begin{aligned}
\text{maximize } L(\alpha) = & -\frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N}\alpha_n\alpha_m y_n y_m \langle x_n, x_m\rangle + \sum_{n=1}^{N}\alpha_n \\
\text{w.r.t. } \sum_{n=1}^{N}\alpha_n y_n = & 0 \\
\alpha_n \geq & 0
\end{aligned}
$$

with much simpler constraints.

Outline

Minimize a Quadratic Function with Linear Equality and Inequality Constraints

$$
\begin{aligned}
\text{minimize } & f(x) := x^T C x - c^T x \\
\text{w.r.t. } & Ax - a = 0 \\
& Bx - b \leq 0 \\
& x \in \mathbb{R}^N
\end{aligned}
$$

where

$C \in \mathbb{R}^{N \times N}, c \in \mathbb{R}^N$	
$A \in \mathbb{R}^{M \times N}, a \in \mathbb{R}^M$	M equality constraints
$B \in \mathbb{R}^{K \times N}, b \in \mathbb{R}^K$	K inequality constraints

Note: Each equality constraint can also be represented as two inequality constraints.

Submanifold Minimization Algorithm

min-ineq-cstr-submanifold($C \in \mathbb{R}^{N\times N}, c \in \mathbb{R}^N, A \in \mathbb{R}^{M\times N}, a \in \mathbb{R}^M,$
$B \in \mathbb{R}^{K\times N}, b \in \mathbb{R}^K, x \in \mathbb{R}^N)$:

$K_0 := \{k \in \{1, \ldots, K\} \mid (Bx - b)_k = 0\}$

while true :

    while true :

$$\tilde{A} := \begin{pmatrix} A \\ (B_{k,.})_{k\in K_0} \end{pmatrix}, \tilde{a} := \begin{pmatrix} a \\ (b_{k,.})_{k\in K_0} \end{pmatrix}$$

$$(x^*, \nu^*) := \text{solve}(\begin{pmatrix} C & \tilde{A}^T \\ \tilde{A} & 0 \end{pmatrix} \begin{pmatrix} x \\ \nu \end{pmatrix} = \begin{pmatrix} c \\ \tilde{a} \end{pmatrix})$$

        if $f(x^*) \geq f(x)$: break

        $\mu := \max\{\mu \in [0,1] \mid B(x + \mu(x^* - x) - b \leq 0\}$

        $x := x + \mu(x^* - x)$

        $K_0 := \{k \in \{1, \ldots, K\} \mid (Bx - b)_k = 0\}$

    if $v^* \geq 0$: break

    choose $k \in K_0 : v^*_k < 0$

    $K_0 := K_0 \setminus \{k\}$

return x

The dual problem for the maximum margin separating hyperplane is such a constrained quadratic optimization problem:

$$\begin{aligned}
\text{maximize } L = & -\frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N}\alpha_n\alpha_m y_n y_m\langle x_n, x_m\rangle + \sum_{n=1}^{N}\alpha_n \\
\text{w.r.t. } \sum_{n=1}^{N}\alpha_n y_n = & 0 \\
\alpha_n \geq & 0
\end{aligned}$$

$$\begin{aligned}
\text{Set } f := & -L \\
C_{n,m} := & y_n y_m \langle x_n, x_m\rangle \\
c_n := & 1 \\
x_n := & \alpha_n \\
A := & (y_1, y_2, \ldots, y_N), \quad a := (0) \\
B_{n,.} := & (0, 0, \ldots, 0, -1, 0, \ldots, 0) \quad \text{(with the -1 at column } n), n = 1, \ldots, N \\
b_n := & 0
\end{aligned}$$

Learning SVMs by Submanifold Minimization

procedure $\textsc{Learn-SVM}(\mathcal{D}^{\text{train}} := \{(x_1, y_1), \ldots, (x_N, y_N)\})$

$C := \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} \alpha_n \alpha_m y_n y_m \langle x_n, x_m \rangle$

$c := (1, 1, \ldots, 1)$

$A := (y_1, y_2, \ldots, y_N), a := (0)$

$B := -I, b := (0, 0, \ldots, 0)$

$N^+ := |\{n \in \{1, \ldots, N\} \mid y_n = +1\}|, N^- := N - N^+$

$\alpha_n := \begin{cases} \frac{1}{N^+} & \text{, if } y_n = +1 \\ \frac{1}{N^-} & \text{, else} \end{cases}$

$\alpha := \textsc{minimize-cstr-ineq}(C, c, A, a, B, b, \alpha)$

$\beta := \sum_{n=1}^{N} \alpha_n y_n x_n$

$N^S := |\{n \in \{1, \ldots, N\} \mid \alpha_n > 0\}|$

$\beta_0 := \frac{1}{N^S} \sum_{n=1, \alpha_n > 0}^{N} y_n - \beta^T x_n$

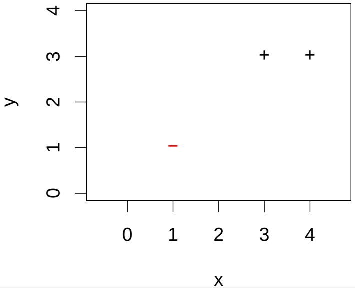
return (β_0, β)

Note: I is the identity matrix.

Example

Find a maximum margin separating hyperplane for the following data:

x_1	x_2	y
1	1	-1
3	3	$+1$
4	3	$+1$



Example

$$C = (y_n y_m \langle x_n, x_y \rangle)_{n,m} = \begin{pmatrix} 2 & -6 & -7 \\ -6 & 18 & 21 \\ -7 & 21 & 25 \end{pmatrix}, \quad c = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix},$$

$$A = (-1\ 1\ 1), \quad a = (0)$$

$$B = \begin{pmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{pmatrix}, \quad b = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

As the equality constraint $A\alpha = a$ always needs to be met, it can be added to C:

$$C' = \begin{pmatrix} C & y \\ y^T & 0 \end{pmatrix} = \begin{pmatrix} 2 & -6 & -7 & -1 \\ -6 & 18 & 21 & 1 \\ -7 & 21 & 25 & 1 \\ -1 & 1 & 1 & 0 \end{pmatrix}$$

Example

Let us start with a random

$$x = \begin{pmatrix} 2 \\ 1 \\ 1 \end{pmatrix}$$

that meets both constraints:

$$Ax - a = \langle y, x \rangle = -2 + 1 + 1 = 0$$

$$Bx - b = \begin{pmatrix} -2 \\ -1 \\ -1 \end{pmatrix} \leq 0$$

As none of the inequality constaints is active: $I_0(x) = \emptyset$.
Step 1: We have to solve

$$C' \begin{pmatrix} x \\ \mu \end{pmatrix} = \begin{pmatrix} c \\ 0 \end{pmatrix}$$

Example

This yields

$$x^* = \begin{pmatrix} 0.5 \\ 1.5 \\ -1.0 \end{pmatrix}$$

which does not fulfill the (inactive) inequality constraint $x_3 \geq 0$.
So we look for

$$x + \mu(x^* - x) = \begin{pmatrix} 2 \\ 1 \\ 1 \end{pmatrix} + \mu \begin{pmatrix} -1.5 \\ 0.5 \\ -2 \end{pmatrix} \geq 0$$

that fulfills all inequality constraints and has large step size μ.
Obviously, $\mu = 0.5$ is best and yields

$$x := x + \mu(x^* - x) = \begin{pmatrix} 1.25 \\ 1.25 \\ 0 \end{pmatrix}$$

Example

Step 2: Now the third inequality constraint is active: $I_0(x) = \{3\}$.

$$C'' = \begin{pmatrix} C' & y & -e_3 \\ y^T & 0 & 0 \\ -e_3^T & 0 & 0 \end{pmatrix}, \quad = \begin{pmatrix} 2 & -6 & -7 & -1 & 0 \\ -6 & 18 & 21 & 1 & 0 \\ -7 & 21 & 25 & 1 & -1 \\ -1 & 1 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 \end{pmatrix},$$

and we have to solve

$$C'' \begin{pmatrix} x \\ \mu \\ \nu^* \end{pmatrix} = \begin{pmatrix} c \\ 0 \\ 0 \end{pmatrix}$$

which yields

$$x^* = \begin{pmatrix} 0.25 \\ 0.25 \\ 0 \end{pmatrix}, \quad \nu^* = 0.5$$

Example

As x^* fulfills all constraints, it becomes the next x (step size $\mu = 1$):

$$x := x^*$$

As the lagrange multiplier $\nu^* \geq 0$, the algorithm stops:
x is optimal.

So we found the optimal

$$\alpha = \begin{pmatrix} 0.25 \\ 0.25 \\ 0 \end{pmatrix} \qquad \text{(called } x \text{ in the optimization algorithm!)}$$

and can compute

$$\beta = \sum_{n=1}^{N} \alpha_n y_n x_n = 0.25 \cdot (-1) \cdot \begin{pmatrix} 1 \\ 1 \end{pmatrix} + 0.25 \cdot (+1) \cdot \begin{pmatrix} 3 \\ 3 \end{pmatrix} = \begin{pmatrix} 0.5 \\ 0.5 \end{pmatrix}$$

β_0 can be computed from the original constraints of the points with $\alpha_n > 0$ which have to be sharp, i.e.,

$$y_1(\beta_0 + \langle \beta, x_1 \rangle) = 1 \quad \Rightarrow \quad \beta_0 = y_1 - \langle \beta, x_1 \rangle = -1 - \langle \begin{pmatrix} 0.5 \\ 0.5 \end{pmatrix}, \begin{pmatrix} 1 \\ 1 \end{pmatrix} \rangle = -2$$

Outline

Optimal Hyperplane

Inseparable problems can be modeled by allowing some points to be on the wrong side of the hyperplane.

Hyperplanes are better if

(i) the fewer points are on the wrong side and

(ii) the closer these points are to the hyperplane

(modeled by **slack variables** ξ_n).

$$\begin{aligned} \text{minimize } & \frac{1}{2}||\beta||^2 + \gamma \sum_{n=1}^{N} \xi_n \\ \text{w.r.t. } y_n(\beta_0 + \langle \beta, x_n \rangle) \geq & 1 - \xi_n, \quad n = 1, \ldots, N \\ \xi \geq & 0 \\ \beta \in & \mathbb{R}^M \\ \beta_0 \in & \mathbb{R} \end{aligned}$$

This problem also is a convex optimization problem

(quadratic target function with linear inequality constraints).

Dual Problem

Compute again the dual problem:

$$L := \frac{1}{2}||\beta||^2 + \gamma \sum_{n=1}^{N} \xi_n - \sum_{n=1}^{N} \alpha_n (y_n(\beta_0 + \langle \beta, x_n \rangle) - (1 - \xi_n)) - \sum_{n=1}^{N} \mu_n \xi_n$$

w.r.t. $\alpha_n \geq 0$

$\mu_n \geq 0$

For an extremum it is required that

$$\frac{\partial L}{\partial \beta} = \beta - \sum_{n=1}^{N} \alpha_n y_n x_n \stackrel{!}{=} 0 \qquad \rightsquigarrow \beta = \sum_{n=1}^{N} \alpha_n y_n x_n$$

and

$$\frac{\partial L}{\partial \beta_0} = -\sum_{n=1}^{N} \alpha_n y_n \stackrel{!}{=} 0$$

and

$$\frac{\partial L}{\partial \xi_n} = \gamma - \alpha_n - \mu_n \stackrel{!}{=} 0 \qquad \Rightarrow \alpha_n = \gamma - \mu_n$$

Dual Problem

Input

$$\beta = \sum_{n=1}^{N} \alpha_n y_n x_n, \quad \sum_{n=1}^{N} \alpha_n y_n = 0, \quad \alpha_n = \gamma - \mu_n$$

into

$$L := \frac{1}{2}||\beta||^2 + \gamma \sum_{n=1}^{N} \xi_n - \sum_{n=1}^{N} \alpha_n (y_n(\beta_0 + \langle \beta, x_n \rangle) - (1 - \xi)) - \sum_{n=1}^{N} \mu_n \xi_n$$

yields the **dual problem**

$$\begin{aligned} L = & \frac{1}{2} \langle \sum_{n=1}^{N} \alpha_n y_n x_n, \sum_{m=1}^{N} \alpha_m y_m x_m \rangle \\ & - \sum_{n=1}^{N} \alpha_n (y_n(\beta_0 + \langle \sum_{m=1}^{N} \alpha_m y_m x_m, x_n \rangle) - (1 - \xi_n)) \\ & + \gamma \sum_{n=1}^{N} \xi_n - \sum_{n=1}^{N} \mu_n \xi_n \end{aligned}$$

Dual Problem

$$
\begin{aligned}
L =& \frac{1}{2}\langle \sum_{n=1}^{N} \alpha_n y_n x_n, \sum_{m=1}^{N} \alpha_m y_m x_m \rangle \\
& - \sum_{n=1}^{N} \alpha_n (y_n(\beta_0 + \langle \sum_{m=1}^{N} \alpha_m y_m x_m, x_n \rangle) - (1 - \xi_n)) \\
& + \gamma \sum_{n=1}^{N} \xi_n - \sum_{n=1}^{N} \mu_n \xi_n \\
=& \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} \alpha_n \alpha_m y_n y_m \langle x_n, x_m \rangle + \sum_{n=1}^{N} \alpha_n - \sum_{n=1}^{N} \alpha_n y_n \beta_0 \\
& - \sum_{n=1}^{N} \sum_{m=1}^{N} \alpha_n \alpha_m y_n y_m \langle x_n, x_m \rangle - \sum_{n=1}^{N} \alpha_n \xi_n + \sum_{n=1}^{N} \alpha_n \xi_n \\
=& - \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} \alpha_n \alpha_m y_n y_m \langle x_n, x_m \rangle + \sum_{n=1}^{N} \alpha_n
\end{aligned}
$$

Dual Problem

The dual problem is

$$\text{maximize } L = -\frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N}\alpha_n\alpha_m y_n y_m\langle x_n, x_m\rangle + \sum_{n=1}^{N}\alpha_n$$

$$\begin{aligned}\text{w.r.t. } \sum_{n=1}^{N}\alpha_n y_n &= 0\\ \alpha_n &\leq \gamma\\ \alpha_n &\geq 0\end{aligned}$$

with much simpler constraints.

Outline

Support Vectors / Separable Case

For points on the right side of the hyperplane (i.e., if a constraint holds),

$$y_n(\beta_0 + \langle \beta, x_n \rangle) > 1$$

then L is maximized by $\alpha_n = 0$: x_n is irrelevant.

For points on the wrong side of the hyperplane (i.e., if a constraint is violated),

$$y_n(\beta_0 + \langle \beta, x_n \rangle) < 1$$

then L is maximized for $\alpha_n \to \infty$.
For separable data, β and β_0 have to be changed to make the constraint hold.

For points on the margin, i.e.,

$$y_n(\beta_0 + \langle \beta, x_n \rangle) = 1$$

α_n is some finite value.

Support Vectors / Inseparable Case

For points on the right side of the hyperplane,

$$y_n(\beta_0 + \langle \beta, x_n \rangle) > 1, \quad \xi_n = 0$$

then L is maximized by $\alpha_n = 0$: x_n is irrelevant.

For points in the margin as well as on the wrong side of the hyperplane,

$$y_n(\beta_0 + \langle \beta, x_n \rangle) = 1 - \xi_n, \quad \xi_n > 0$$

α_n is some finite value.

For points on the margin, i.e.,

$$y_n(\beta_0 + \langle \beta, x_n \rangle) = 1, \quad \xi_n = 0$$

α_n is some finite value.

The data points x_n with $\alpha_n > 0$ are called **support vectors**.

Decision Function

Due to

$$\hat{\beta} = \sum_{n=1}^{N} \hat{\alpha}_n y_n x_n,$$

the decision function

$$\hat{y}(x) = \operatorname{sign} \hat{\beta}_0 + \langle \hat{\beta}, x \rangle$$

can be expressed using the training data:

$$\hat{y}(x) = \operatorname{sign} \hat{\beta}_0 + \sum_{n=1}^{N} \hat{\alpha}_n y_n \langle x_n, x \rangle$$

Only support vectors are required, as only for them $\hat{\alpha}_n \neq 0$.

Both, the learning problem and the decision function can be expressed using an inner product / a similarity measure / a kernel $\langle x, x' \rangle$.

High-Dimensional Embeddings / The "kernel trick"

Example:

we map points from $\mathbb{R}^2$ into the higher dimensional space $\mathbb{R}^6$ via

$$h : \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \mapsto \begin{pmatrix} 1 \\ \sqrt{2}x_1 \\ \sqrt{2}x_2 \\ x_1^2 \\ x_2^2 \\ \sqrt{2}x_1 x_2 \end{pmatrix}$$

Then the inner product

$$\langle h(\begin{pmatrix} x_1 \\ x_2 \end{pmatrix}), h(\begin{pmatrix} x_1' \\ x_2' \end{pmatrix})\rangle = 1 + 2x_1 x_1' + 2x_2 x_2' + x_1^2 {x_1'}^2 + x_2^2 {x_2'}^2 + 2x_1 x_2 x_1' x_2'$$
$$= (1 + x_1 x_1' + x_2 x_2')^2$$

can be computed without having to compute the embedding h explicitely !

Popular Kernels

Some popular kernels are:

linear kernel:

$$K(x, x') := \langle x, x' \rangle := \sum_{n=1}^{N} x_n x'_n$$

polynomial kernel of degree d:

$$K(x, x') := (1 + \langle x, x' \rangle)^d$$

radial basis kernel / gaussian kernel :

$$K(x, x') := e^{-\frac{||x-x'||^2}{c}}$$

neural network kernel / sigmoid kernel :

$$K(x, x') := \tanh(a\langle x, x' \rangle + b)$$

Learning SVMs by Submanifold Minimization

procedure $\textsc{Learn-SVM}(\mathcal{D}^{\text{train}} := \{(x_1, y_1), \ldots, (x_N, y_N)\}, \gamma \in \mathbb{R}^+, K)$

$\quad C := \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} \alpha_n \alpha_m y_n y_m K(x_n, x_m)$

$\quad c := (1, 1, \ldots, 1)$

$\quad A := (y_1, y_2, \ldots, y_N), a := (0)$

$\quad B := \begin{pmatrix} -I \\ I \end{pmatrix}, b := \begin{pmatrix} 0 \\ \gamma \end{pmatrix}$

$\quad N^+ := |\{n \in \{1, \ldots, N\} \mid y_n = +1\}|, N^- := N - N^+$

$\quad \alpha_n := \gamma \cdot \begin{cases} \frac{1}{N^+} & \text{, if } y_n = +1 \\ \frac{1}{N^-} & \text{, else} \end{cases}$

$\quad \alpha := \textsc{minimize-cstr-ineq}(C, c, A, a, B, b, \alpha)$

$\quad N^S := |\{n \in \{1, \ldots, N\} \mid \alpha_n > 0\}|$

$\quad \beta_0 := \frac{1}{N^S} \sum_{n=1, \alpha_n > 0}^{N} y_n - \sum_{m=1, \alpha_m > 0}^{N} \alpha_m y_m K(x_m, x_n)$

$\quad$ **return** (β_0, α)

Note: I is the identity matrix.

Predicting with SVMs

procedure $\text{PREDICT-SVM}(\alpha \in (\mathbb{R}_0^+)^N, \beta_0 \in \mathbb{R}, \mathcal{D}^{\text{train}} := \{(x_1, y_1), \ldots, (x_N, y_N)\}, K)$
  $\hat{y} := \beta_0$
  for $n := 1, \ldots, N$ with $\alpha_n \neq 0$ **do**
    $\hat{y} := \hat{y} + \alpha_n y_n K(x_n, x)$
  return $\hat{y}$

Note: $\hat{y}$ yields the score/certainty factor, $\operatorname{sign} \hat{y}$ the predicted class.
From $\mathcal{D}^{\text{train}}$, only the support vectors (x_n, y_n) (having $\alpha_n > 0$) are required.

Summary (1/2)

- Binary classification problems with linear decision boundaries can be rephrased as finding a **separating hyperplane**.
- In the **linear separable case**, there are simple algorithms like **perceptron** learning to find such a separating hyperplane.
- If one requires the additional property that the hyperplane should have **maximal margin**, i.e., maximal distance to the closest points of both classes, then a quadratic optimization problem with inequality constraints arises.
- Quadratic optimization problems without constraints as well as with equality constraints can be solved by linear systems of equations. **Quadratic optimization problems with inequality constraints** require some more complex methods such as **submanifold optimization** (a sequence of linear systems of equations).

Summary (2/2)

- Optimal hyperplanes can also be formulated for the **linear inseparable case** by allowing some points to be on the wrong side of the margin, but penalize for their distance from the margin. This also can be formulated as a quadratic optimization problem with inequality constraints.
- The final decision function can be computed in terms of inner products of the query points with some of the data points (called **support vectors**), which allows to bypass the explicit computation of high dimensional embeddings (**kernel trick**).

Further Readings

- [Hastie et al., 2005, chapter 12.1–3], [Murphy, 2012, chapter 14.1+2+5], [James et al., 2013, chapter 9].

References

Trevor Hastie, Robert Tibshirani, Jerome Friedman, and James Franklin. *The Elements of Statistical Learning: Data Mining, Inference and Prediction*, volume 27. Springer, 2005.

Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. *An Introduction to Statistical Learning*. Springer, 2013.

Kevin P. Murphy. *Machine Learning: A Probabilistic Perspective*. The MIT Press, 2012.

Unconstrained Problem

The **unconstrained quadratic optimization problem** is

$$\text{minimize } f(x) := \frac{1}{2}\langle x, Cx\rangle - \langle c, x\rangle$$
$$\text{w.r.t. } x \in \mathbb{R}^N$$

(with $C \in \mathbb{R}^{N\times N}$ symmetric and positive definite, $c \in \mathbb{R}^N$).

The solution of the unconstrained quadratic optimization problem coincides with the solution of the linear systems of equations

$$Cx = c$$

that can be solved by Gaussian Elimination, Cholesky decomposition, QR decomposition etc.

Proof:

$$\frac{\partial f(x)}{\partial x} = x^T C - c^T \stackrel{!}{=} 0 \Leftrightarrow Cx = c$$

Equality Constraints

The **quadratic optimization problem with linear equality constraints** is

$$\begin{aligned} \text{minimize } f(x) :=& \frac{1}{2}\langle x, Cx\rangle - \langle c, x\rangle \\ \text{w.r.t. } h(x) :=& Ax - b = 0 \\ x \in& \mathbb{R}^N \end{aligned}$$

(with $C \in \mathbb{R}^{N\times N}$ symmetric and positive definite, $c \in \mathbb{R}^N$, $A \in \mathbb{R}^{M\times N}$, $b \in \mathbb{R}^M$).

Lagrange Function

Definition

Consider the optimization problem

$$\begin{aligned} \text{minimize } & f(x) \\ \text{subject to } & g(x) \leq 0 \\ & h(x) = 0 \\ & x \in \mathbb{R}^N \end{aligned}$$

with $f : \mathbb{R}^N \to \mathbb{R}$, $g : \mathbb{R}^N \to \mathbb{R}^M$ and $h : \mathbb{R}^N \to \mathbb{R}^M$.

The **Lagrange function** of this problem is defined as

$$L(x, \lambda, \nu) := f(x) + \langle \lambda, g(x) \rangle + \langle \nu, h(x) \rangle$$

λ and ν are called **Lagrange multipliers**.

Lower Bounds Lemma

Lemma

Die dual function yields lower bounds for the optimal value of the problem, i.e.,

$$\bar{f}(\lambda, \nu) \leq f(x^*), \quad \forall \lambda \geq 0, \nu$$

Proof:

For feasible x, i.e., $g(x) \leq 0$ and $h(x) = 0$:

$$L(x, \lambda, \nu) = f(x) + \langle \lambda, g(x) \rangle + \langle \nu, h(x) \rangle \leq f(x)$$

Hence

$$\bar{f}(\lambda, \nu) = \inf_x L(x, \lambda, \nu) \leq f(x)$$

and especially for $x = x^*$.

Karush-Kuhn-Tucker Conditions

Theorem (Karush-Kuhn-Tucker Conditions)

If
(i) x *is optimal for the problem,*
(ii) λ, ν *are optimal for the dual problem and*
(iii) $f(x) = \bar{f}(\lambda, \nu)$*,*
then the following conditions hold:

$$\begin{aligned} g(x) &\leq 0 \\ h(x) &= 0 \\ \lambda &\geq 0 \\ \lambda_n g_n(x) &= 0 \\ \frac{\partial f(x)}{\partial x} + \langle \lambda, \frac{\partial g(x)}{\partial x} \rangle + \langle \nu, \frac{\partial h(x)}{\partial x} \rangle &= 0 \end{aligned}$$

If f *is convex and* h *is affine, then the KKT conditions are also sufficient.*

Karush-Kuhn-Tucker Conditions

Proof: "$\Rightarrow$"

$$f(x) = \bar{f}(\lambda, \nu) = \inf_{x'} f(x') + \langle \lambda, g(x') \rangle + \langle \nu, h(x') \rangle$$
$$\leq f(x) + \langle \lambda, g(x) \rangle + \langle \nu, h(x) \rangle \leq f(x)$$

and therefore equality holds, thus

$$\langle \lambda, g(x) \rangle = \sum_{n=1}^{M} \lambda_n g_n(x) = 0$$

and as all terms are non-positive: $\lambda_n g_n(x) = 0$.
Since x minimizes $L(x', \lambda, \nu)$ over x', the derivative must vanish:

$$\frac{\partial L(x, \lambda, \nu)}{\partial x} = \frac{\partial f(x)}{\partial x} + \langle \lambda, \frac{\partial g(x)}{\partial x} \rangle + \langle \nu, \frac{\partial h(x)}{\partial x} \rangle = 0$$

Karush-Kuhn-Tucker Conditions

Proof (ctd.): "$\Leftarrow$"
Now let f be convex. Since $\lambda \geq 0$, $L(x', \lambda, \nu)$ is convex in x'.
As its first derivative vanishes at x, x minimizes $L(x', \lambda, \nu)$ over x', and thus:

$$\bar{f}(\lambda, \nu) = L(x, \lambda, \nu) = f(x) + \langle \lambda, g(x) \rangle + \langle \nu, h(x) \rangle = f(x)$$

Therefore is x optimal for the problem and λ, ν optimal for the dual problem.

Equality Constraints

The **quadratic optimization problem with linear equality constraints** is

$$
\begin{aligned}
\text{minimize } f(x) :=& \frac{1}{2}\langle x, Cx\rangle - \langle c, x\rangle \\
\text{w.r.t. } h(x) :=& Ax - b = 0 \\
x \in& \mathbb{R}^N
\end{aligned}
$$

(with $C \in \mathbb{R}^{N\times N}$ symmetric and positive definite, $c \in \mathbb{R}^N$, $A \in \mathbb{R}^{M\times N}$, $b \in \mathbb{R}^M$).

The KKT conditons for the optimal solution x^*, ν^* are:

$$
\begin{aligned}
h(x^*) &= Ax^* - b = 0 \\
\frac{\partial f(x^*)}{\partial x} + \langle \nu^*, \frac{\partial h(x^*)}{\partial x}\rangle &= Cx^* - c + A^T\nu^* = 0
\end{aligned}
$$

which can be written as a single system of linear equations

$$
\begin{pmatrix} C & A^T \\ A & 0 \end{pmatrix} \begin{pmatrix} x^* \\ \nu^* \end{pmatrix} = \begin{pmatrix} c \\ b \end{pmatrix}
$$

Inequality Constraints

The **quadratic optimization problem with linear inequality constraints** is

$$\begin{aligned} \text{minimize } f(x) :=& \frac{1}{2}\langle x, Cx\rangle - \langle c, x\rangle \\ \text{w.r.t. } g(x) :=& Ax - b \leq 0 \\ x \in& \mathbb{R}^N \end{aligned}$$

(with $C \in \mathbb{R}^{N\times N}$ symmetric and positive definite, $c \in \mathbb{R}^N$, $A \in \mathbb{R}^{M\times N}$, $b \in \mathbb{R}^M$).

Inequality constraints are more complex to solve.
But they can be reduced to a sequence of equality constraints.

Inequality Constraints

At each point $x \in \mathbb{R}^N$ one distinguishes between
active constraints g_n with $g_n(x) = 0$ and
inactive constraints g_n with $g_n(x) < 0$.

Active set:

$$I_0(x) := \{i \in \{1, \ldots, m\} \,|\, g_n(x) = 0\}$$

Inactive constraints stay inactive in a neighborhood of x and can be neglected there.
Active constraints are equality constraints that identify points at the border of the feasible area.
We can restrict our attention to just the points at the actual border, i.e., use the equality constraints

$$h_n(x) := g_n(x), \quad i \in I_0$$

.

Inequality Constraints

If there is an optimal point x^* found with optimal lagrange multiplier $\nu^* \geq 0$:

$$\frac{\partial f(x^*)}{\partial x} + \sum_{i \in I_0} \nu_n^* \frac{\partial h_n(x^*)}{\partial x} = 0$$

then x^* with

$$\lambda_n^* := \begin{cases} \nu_n^*, & i \in I_0 \\ 0, & \text{else} \end{cases}$$

fullfils the KKT conditions of the original problem:

$$\lambda_n^* g_n(x^*) = \begin{cases} \nu_n^* h_n(x^*) = 0, & i \in I_0 \\ 0 g_n(x^*) = 0, & \text{else} \end{cases}$$

and

$$\frac{\partial f(x^*)}{\partial x} + \langle \lambda^*, \frac{\partial h(x^*)}{\partial x} \rangle = \frac{\partial f(x^*)}{\partial x} + \sum_{i \in I_0} \nu_n^* \frac{\partial h_n(x^*)}{\partial x} = 0$$

Inequality Constraints

If the optimal point x^* on the border has an optimal lagrange multiplier ν^* with $\nu_n^* < 0$ for some $i \in I_0$,

$$\frac{\partial f(x^*)}{\partial x} + \sum_{i \in I_0} \nu_n^* \frac{\partial h_n(x^*)}{\partial x} = 0$$

then f decreases along $h_n := g_n$, thus we can decrease f by moving away from the border by dropping the constraint i.

Inequality Constraints

minimize-submanifold(target function f, inequality constraint function g) :
$x :=$ a random vector with $g(x) \leq 0$
$I_0 := I_0(x) := \{i \,|\, g_i(x) = 0\}$
do
    $x^* := \text{argmin}_x\, f(x)$ subject to $g_i(x) = 0, i \in I_0$
    while $f(x^*) < f(x)$ **do**
        $\alpha := \max\{\alpha \in [0,1] \,|\, g(x + \alpha(x^* - x)) \leq 0\}$
        $x := x + \alpha(x^* - x)$
        $I_0 := I_0(x)$
        $x^* := \text{argmin}_x\, f(x)$ subject to $g_i(x) = 0, i \in I_0$
    od
    Let ν^* be the optimal Lagrange multiplier for x^*
    if $\nu^* \geq 0$ break **fi**
    choose $i \in I_0 : \nu_i^* < 0$
    $I_0 := I_0 \setminus \{i\}$
while true
return x