Machine Learning

7. Support Vector Machines (SVMs)

Lars Schmidt-Thieme

Information Systems and Machine Learning Lab (ISMLL)
Institute for Business Economics and Information Systems
& Institute for Computer Science
University of Hildesheim
http://www.ismll.uni-hildesheim.de

Separating Hyperplanes

Logistic Regression:

Linear Discriminant Analysis (LDA):

Hyperplanes

Hyperplanes can be modeled explicitly as

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

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 $x \in \mathbb{R}^p$ 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 $x' \in H_\beta$:

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

as $x - \pi x$ is proportional to β and πx and x' are on H_β.

Machine Learning / 1. Separating Hyperplanes

Hyperplanes

The **signed distance** of a point $x \in \mathbb{R}^p$ 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

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

Linear Separable Data

The data 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.

Perceptron as Linear Model

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

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

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

that is very similar to the logisitic regression model

$$Y(X) = \text{argmax}_y\, p(Y = y \mid X)$$

$$p(Y = +1 \mid X) = \text{logistic}(\langle X, \beta \rangle) + \epsilon = \frac{e^{\sum_{i=1}^n \beta_i X_i}}{1 + e^{\sum_{i=1}^n \beta_i X_i}} + \epsilon$$

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

as well as to linear discriminant analysis (LDA).

The perceptron does just provide class labels $\hat{y}(x)$ and unscaled certainty factors $\hat{h}(x)$, but no class probabilities $\hat{p}(Y \mid X)$.

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:

$$q(\beta, \beta_0) := \sum_{i=1:\hat{y}_i \neq y_i}^{n} |h_\beta(x_i)| = -\sum_{i=1:\hat{y}_i \neq y_i}^{n} y_i h_\beta(x_i)$$

Perceptron as Linear Model

For learning, gradient descent is used:

$$\frac{\partial q(\beta, \beta_0)}{\partial \beta} = -\sum_{i=1:\hat{y}_i \neq y_i}^{n} y_i x_i$$

$$\frac{\partial q(\beta, \beta_0)}{\partial \beta_0} = -\sum_{i=1:\hat{y}_i \neq y_i}^{n} y_i$$

Instead of looking at all points at the same time,
stochastic gradient descent is applied where all points are looked at sequentially (in a random sequence).
The update for a single point (x_i, y_i) then is

$$\hat{\beta}^{(k+1)} := \hat{\beta}^{(k)} + \alpha y_i x_i$$

$$\hat{\beta}_0^{(k+1)} := \hat{\beta}_0^{(k)} + \alpha y_i$$

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

Perceptron Learning Algorithm

learn-perceptron(training data X, step length α) :
$\hat{\beta} :=$ a random vector
$\hat{\beta}_0 :=$ a random value
do
    errors $:= 0$
    for $(x, y) \in X$ (in random order) **do**
        if $y(\hat{\beta}_0 + \langle \hat{\beta}, x \rangle) \leq 0$
            errors $:=$ *errors* $+ 1$
            $\hat{\beta} := \hat{\beta} + \alpha y x$
            $\hat{\beta}_0 := \hat{\beta}_0 + \alpha y$
        fi
    od
while *errors* > 0
return $(\hat{\beta}, \hat{\beta}_0)$

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.

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:

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

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

$$\begin{aligned} \text{minimize } & \frac{1}{2}||\beta||^2 \\ \text{w.r.t. } y_i(\beta_0 + \langle \beta, x_i \rangle) & \geq 1, \quad i = 1, \dots, n \\ \beta & \in \mathbb{R}^p \\ \beta_0 & \in \mathbb{R} \end{aligned}$$

This problem is a convex optimization problem (quadaratic 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

$$L := \frac{1}{2}||\beta||^2 - \sum_{i=1}^{n} \alpha_i (y_i(\beta_0 + \langle \beta, x_i \rangle) - 1)$$

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

For an extremum it is required that

$$\frac{\partial L}{\partial \beta} = \beta - \sum_{i=1}^{n} \alpha_i y_i x_i \stackrel{!}{=} 0$$

$$\Rightarrow \beta = \sum_{i=1}^{n} \alpha_i y_i x_i$$

and

$$\frac{\partial L}{\partial \beta_0} = -\sum_{i=1}^{n} \alpha_i y_i \stackrel{!}{=} 0$$

Quadratic Optimization

Input

$$\beta = \sum_{i=1}^{n} \alpha_i y_i x_i, \quad \sum_{i=1}^{n} \alpha_i y_i = 0$$

into

$$L := \frac{1}{2}||\beta||^2 - \sum_{i=1}^{n} \alpha_i (y_i(\beta_0 + \langle \beta, x_i \rangle) - 1)$$

yields the **dual problem**

$$\begin{aligned}
L =& \frac{1}{2}\langle \sum_{i=1}^{n} \alpha_i y_i x_i, \sum_{j=1}^{n} \alpha_j y_j x_j \rangle - \sum_{i=1}^{n} \alpha_i (y_i(\beta_0 + \langle \sum_{j=1}^{n} \alpha_j y_j x_j, x_i \rangle) - 1) \\
=& \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \langle x_i, x_j \rangle + \sum_{i=1}^{n} \alpha_i - \sum_{i=1}^{n} \alpha_i y_i \beta_0 - \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \langle x_i, x_j \rangle \\
=& -\frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \langle x_i, x_j \rangle + \sum_{i=1}^{n} \alpha_i
\end{aligned}$$

Quadratic Optimization

The dual problem is

$$\text{maximize } L = -\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j \langle x_i, x_j\rangle + \sum_{i=1}^{n}\alpha_i$$

$$\begin{aligned}\text{w.r.t. } \sum_{i=1}^{n}\alpha_i y_i &= 0\\ x_i\rangle) - 1) &= 0\\ \alpha_i &\geq 0\end{aligned}$$

with much simpler constraints.

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 equality constraints** is

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

(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 1. 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 $h : \mathbb{R}^n \to \mathbb{R}^p$.

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**.

The **dual problem** is defined as

$$\begin{aligned} \text{maximize } & \bar{f}(\lambda, \nu) := \inf_x L(x, \lambda, \nu) \\ \text{subject to } & \lambda \geq 0 \\ & \lambda \in \mathbb{R}^m, \quad \nu \in \mathbb{R}^p \end{aligned}$$

Lower Bounds Lemma

Lemma 1. *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 1 (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:

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

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

Karush-Kuhn-Tucker Conditions

Proof: "$\Rightarrow$"

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

and therefore equality holds, thus

$$\langle \lambda, g(x) \rangle = \sum_{i=1}^{m} \lambda_i g_i(x) = 0$$

and as all terms are non-positive: $\lambda_i g_i(x) = 0$.
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 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 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_i with $g_i(x) = 0$ and
inactive constraints g_i with $g_i(x) < 0$.
Active set:

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

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

$$h_i(x) := g_i(x), \quad i \in I_0$$

.

Inequality Constraints

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

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

then x^* with

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

fullfils the KKT conditions of the original problem:

$$\lambda_i^* g_i(x^*) = \begin{cases} \nu_i^* h_i(x^*) = 0, & i \in I_0 \\ 0 g_i(x^*) = 0, & \text{else} \end{cases}$$

and

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

Inequality Constraints

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

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

then f decreases along $h_i := g_i$, 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
    $x^* := \operatorname{argmin}_x f(x)$ subject to $g_i(x) = 0, i \in I_0$
    while $f(x^*) < f(x)$ **do**
        $\alpha := \max\{\alpha \in [0,1] \,|\, g(x + \alpha(x^* - x)) \leq 0\}$
        $x := x + \alpha(x^* - x)$
        $I_0 := I_0(x)$
        $x^* := \operatorname{argmin}_x f(x)$ 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

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

$$\text{maximize } L = -\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j \langle x_i, x_j\rangle + \sum_{i=1}^{n}\alpha_i$$

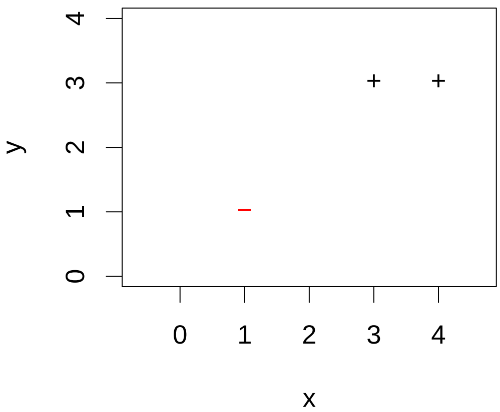
$$\begin{aligned}\text{w.r.t. } \sum_{i=1}^{n}\alpha_i y_i =& 0\\ \alpha_i \geq& 0\end{aligned}$$

$$\begin{aligned}\text{Set } f :=& -L\\ C_{i,j} :=& y_i y_j \langle x_i, x_j\rangle\\ c_i :=& 1\\ x_i :=& \alpha_i\\ A_i :=& (0,0,\ldots,0,-1,0,\ldots,0) \quad \text{(with the -1 at column } i), i = 1,\ldots,n\\ b_i :=& 0\\ h(x) :=& \sum_{i=1}^{n}\alpha_i y_i\end{aligned}$$

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

$$C = (y_i y_j \langle x_i, x_y \rangle)_{i,j} = \begin{pmatrix} 2 & -6 & -7 \\ -6 & 18 & 21 \\ -7 & 21 & 25 \end{pmatrix}, \quad c = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix},$$

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

$$h(\alpha) = \langle \alpha, y \rangle = -\alpha_1 + \alpha_2 + \alpha_3$$

As the equality constraint h always needs to be met, it can be added to C:

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

Example

Let us start with a random

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

that meets both constraints:

$$g(x) = Ax - b = \begin{pmatrix} -2 \\ -1 \\ -1 \end{pmatrix} \leq 0$$
$$h(x) = \langle y, x \rangle = -2 + 1 + 1 = 0$$

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

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

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

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

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 $\alpha = 1$):

$$x := x^*$$

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

So we found the optimal

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

and can compute

$$\beta = \sum_{i=1}^{n} \alpha_i y_i x_i = 0.25 \cdot (-1) \cdot \begin{pmatrix} 1 \\ 1 \end{pmatrix} + 0.25 \cdot (+1) \cdot \begin{pmatrix} 3 \\ 3 \end{pmatrix} = \begin{pmatrix} 0.5 \\ 0.5 \end{pmatrix}$$

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

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

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_i).

$$\text{minimize } \frac{1}{2}||\beta||^2 + \gamma \sum_{i=1}^{n} \xi_i$$

$$\begin{aligned}
\text{w.r.t. } y_i(\beta_0 + \langle \beta, x_i \rangle) &\geq 1 - \xi_i, \quad i = 1, \ldots, n \\
\xi &\geq 0 \\
\beta &\in \mathbb{R}^p \\
\beta_0 &\in \mathbb{R}
\end{aligned}$$

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

Dual Problem

Compute again the dual problem:

$$L := \frac{1}{2}||\beta||^2 + \gamma \sum_{i=1}^{n} \xi_i - \sum_{i=1}^{n} \alpha_i (y_i(\beta_0 + \langle \beta, x_i \rangle) - (1 - \xi_i)) - \sum_{i=1}^{n} \mu_i \xi_i$$

w.r.t. $\alpha_i \geq 0$
$\mu_i \geq 0$

For an extremum it is required that

$$\frac{\partial L}{\partial \beta} = \beta - \sum_{i=1}^{n} \alpha_i y_i x_i \stackrel{!}{=} 0 \qquad \Rightarrow \beta = \sum_{i=1}^{n} \alpha_i y_i x_i$$

and

$$\frac{\partial L}{\partial \beta_0} = -\sum_{i=1}^{n} \alpha_i y_i \stackrel{!}{=} 0$$

and

$$\frac{\partial L}{\partial \xi_i} = \gamma - \alpha_i - \mu_i \stackrel{!}{=} 0 \qquad \Rightarrow \alpha_i = \gamma - \mu_i$$

Dual Problem

Input

$$\beta = \sum_{i=1}^{n} \alpha_i y_i x_i, \quad \sum_{i=1}^{n} \alpha_i y_i = 0, \quad \alpha_i = \gamma - \mu_i$$

into

$$L := \frac{1}{2}||\beta||^2 + \gamma \sum_{i=1}^{n} \xi_i - \sum_{i=1}^{n} \alpha_i (y_i(\beta_0 + \langle \beta, x_i \rangle) - (1 - \xi)) - \sum_{i=1}^{n} \mu_i \xi_i$$

yields the **dual problem**

$$L = \frac{1}{2}\langle \sum_{i=1}^{n} \alpha_i y_i x_i, \sum_{j=1}^{n} \alpha_j y_j x_j \rangle - \sum_{i=1}^{n} \alpha_i (y_i(\beta_0 + \langle \sum_{j=1}^{n} \alpha_j y_j x_j, x_i \rangle) - (1 - \xi_i)) + \gamma \sum_{i=1}^{n} \xi_i - \sum_{i=1}^{n} \mu_i \xi_i$$

Dual Problem

$$
\begin{aligned}
L =&\frac{1}{2}\langle\sum_{i=1}^{n}\alpha_i y_i x_i, \sum_{j=1}^{n}\alpha_j y_j x_j\rangle - \sum_{i=1}^{n}\alpha_i (y_i(\beta_0 + \langle\sum_{j=1}^{n}\alpha_j y_j x_j, x_i\rangle) - (1-\xi_i))\\
&+\gamma\sum_{i=1}^{n}\xi_i - \sum_{i=1}^{n}\mu_i\xi_i\\
=&\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j\langle x_i, x_j\rangle + \sum_{i=1}^{n}\alpha_i - \sum_{i=1}^{n}\alpha_i y_i\beta_0 - \sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j\langle x_i, x_j\rangle\\
&-\sum_{i=1}^{n}\alpha_i\xi_i + \sum_{i=1}^{n}\alpha_i\xi_i\\
=&-\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j\langle x_i, x_j\rangle + \sum_{i=1}^{n}\alpha_i
\end{aligned}
$$

Dual Problem

The dual problem is

$$
\begin{aligned}
\text{maximize } L =& -\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j\langle x_i, x_j\rangle + \sum_{i=1}^{n}\alpha_i\\
\text{w.r.t. } \sum_{i=1}^{n}\alpha_i y_i =& 0\\
\alpha_i \leq& \gamma\\
\alpha_i \geq& 0
\end{aligned}
$$

with much simpler constraints.

Support Vectors / Separable Case

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

$$y_i(\beta_0 + \langle \beta, x_i \rangle) > 1$$

then L is maximized by $\alpha_i = 0$: x_i is irrelevant.

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

$$y_i(\beta_0 + \langle \beta, x_i \rangle) < 1$$

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

For points on the margin, i.e.,

$$y_i(\beta_0 + \langle \beta, x_i \rangle) = 1$$

α_i is some finite value.

Support Vectors / Inseparable Case

For points on the right side of the hyperplane,

$$y_i(\beta_0 + \langle \beta, x_i \rangle) > 1, \quad \xi_i = 0$$

then L is maximized by $\alpha_i = 0$: x_i is irrelevant.

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

$$y_i(\beta_0 + \langle \beta, x_i \rangle) = 1 - \xi_i, \quad \xi_i > 0$$

α_i is some finite value.

For points on the margin, i.e.,

$$y_i(\beta_0 + \langle \beta, x_i \rangle) = 1, \quad \xi_i = 0$$

α_i is some finite value.

The data points x_i with $\alpha_i > 0$ are called **support vectors**.

Decision Function

Due to

$$\hat{\beta} = \sum_{i=1}^{n} \hat{\alpha}_i y_i x_i,$$

the decision function

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

can be expressed using the training data:

$$\hat{y}(x) = \text{sign}\, \hat{\beta}_0 + \sum_{i=1}^{n} \hat{\alpha}_i y_i \langle x_i, x \rangle$$

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

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 R^2 into the higher dimensional space $\mathbb{R}^6$ via

$$h : \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \mapsto \begin{pmatrix} 1 \\ \sqrt{2}x_1 \\ \sqrt{2}x_2 \\ x_1^2 \\ x_2^2 \\ \sqrt{2}x_1x_2 \end{pmatrix}$$

Then the inner product

$$\begin{aligned} \langle h(\begin{pmatrix} x_1 \\ x_2 \end{pmatrix}), h(\begin{pmatrix} x'_1 \\ x'_2 \end{pmatrix})\rangle &= 1 + 2x_1x'_1 + 2x_2x'_2 + x_1^2{x'_1}^2 + x_2^2{x'_2}^2 + 2x_1x_2x'_1x'_2 \\ &= (1 + x_1x'_1 + x_2x'_2)^2 \end{aligned}$$

can be computed without having to compute h explicitely !

Popular Kernels

Some popular kernels are:

linear kernel:

$$K(x, x') := \langle x, x' \rangle := \sum_{i=1}^{n} x_i x'_i$$

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 :

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

Optimal Hyperplanes as Regularization

Optimal separating hyperplanes

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

can also be understood as regularization “error + complexity”:

$$\begin{aligned} \text{minimize } & \gamma \sum_{i=1}^{n} [1 - y_i(\beta_0 + \langle \beta, x_i \rangle)]_+ + \frac{1}{2}||\beta||^2 \\ \text{w.r.t. } \beta \in & \mathbb{R}^p \\ \beta_0 \in & \mathbb{R} \end{aligned}$$

where the postitive part is defined as

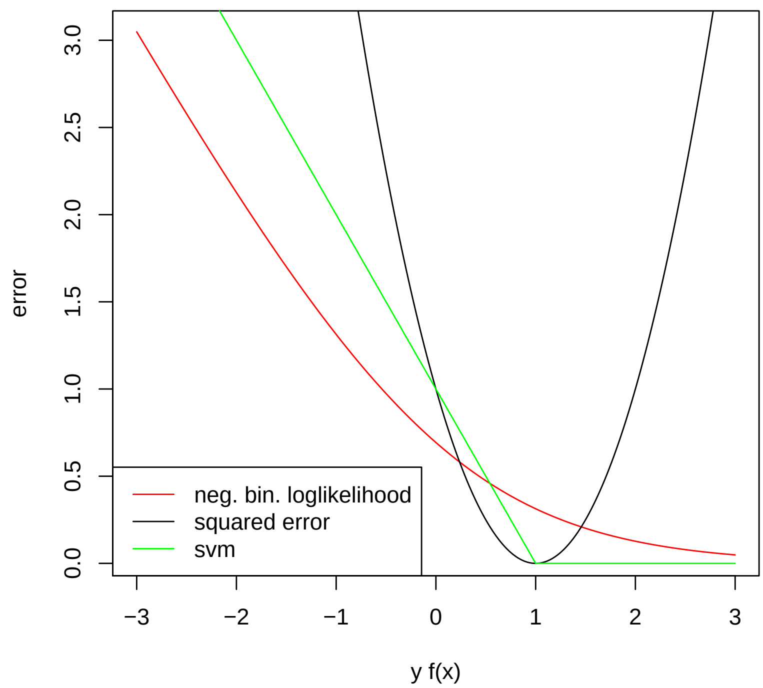
$$[x]_+ := \begin{cases} x, & \text{if } x \geq 0 \\ 0, & \text{else} \end{cases}$$

Optimal Hyperplanes as Regularization / Error functions

Specific for the SVM model then is the error function (also often called **loss function**).

model	error function		minimizing function
logistic regression	negative binomial loglikelihood	$\log(1+e^{-yf(x)})$	$f(x) = \log \frac{p(y=+1\|x)}{p(y=-1\|x)}$
LDA	squared error	$(y-f(x))^2$	$f(x) = p(y=+1\|x) - p(y=-1\|x)$
SVM		$[1-yf(x)]_+$	$f(x) = \begin{cases} 1, & \text{if } p(y=+1\|x) \geq \frac{1}{2} \\ 0, & \text{else} \end{cases}$

Optimal Hyperplanes as Regularization / Error functions



SVM Regression / Error functions

In regression, squared error sometimes is dominated by outliers, i.e., points with large residuum, due to the quadratic dependency:

$$\mathrm{err}(y, \hat{y}) := (y - \hat{y})^2$$

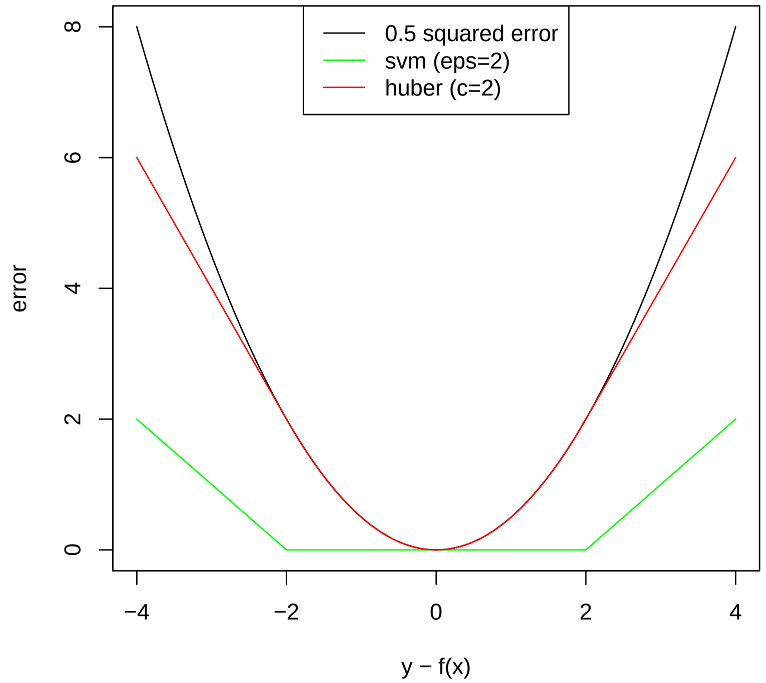
Therefore, robust error functions such as the **Huber error** have been developed that keep the quadratic form near zero, but are linear for larger values:

$$\mathrm{err}_c(y, \hat{y}) := \begin{cases} \frac{(y-\hat{y})^2}{2}, & \text{if } |y - \hat{y}| < c \\ c|y - \hat{y}| - \frac{c^2}{2}, & \text{else} \end{cases}$$

SVM regression uses the **ϵ-insensitive error**:

$$\mathrm{err}_c(y, \hat{y}) := \begin{cases} 0, & \text{if } |y - \hat{y}| < \epsilon \\ |y - \hat{y}| - \epsilon, & \text{else} \end{cases}$$

SVM Regression / Error functions



SVM Regression

Any of these error functions can be used to find optimal parameters for the linear regression model

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

by solving the optimization problem

$$\text{min.} \sum_{i=1}^{n} \text{err}(y_i, \hat{\beta}_0 + \langle \hat{\beta}, x_i \rangle) + \frac{\lambda}{2} ||\hat{\beta}||^2$$

SVM Regression

For the ϵ-insensitive error, the solution can be shown to have the form

$$\hat{\beta} = \sum_{i=1}^{n} (\hat{\alpha}_i^* - \hat{\alpha}_i) x_i$$

$$\hat{f}(x) = \beta_0 + \sum_{i=1}^{n} (\hat{\alpha}_i^* - \hat{\alpha}_i) \langle x_i, x \rangle$$

where $\hat{\alpha}_i^*$ and $\hat{\alpha}_i$ are the solutions of the quadratic problem

$$\min \epsilon \sum_{i=1}^{n} (\alpha_i^* - \alpha_i) - \sum_{i=1}^{n} y_i (\alpha_i^* - \alpha_i) + \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} (\alpha_i^* - \alpha_i)(\alpha_j^* - \alpha_j) \langle x_i, x_j \rangle$$

s.t. $\alpha_i \geq 0$

$$\alpha_i^* \leq \frac{1}{\lambda}$$

$$\sum_{i=1}^{n} (\alpha_i^* - \alpha_i) = 0$$

$$\alpha_i^* \alpha_i = 0$$

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 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.