Machine Learning

A. Supervised Learning
A.7. Support Vector Machines (SVMs)

Lars Schmidt-Thieme, Nicolas Schilling

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

Outline

1. Separating Hyperplanes
2. Perceptron
3. Maximum Margin Separating Hyperplanes

Outline

Hyperplanes

Hyperplanes H are subsets of $\mathbb{R}^p$ with dimensionality $p-1$ and can be modeled explicitly as

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

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

- H_β is a point for $p = 1$
- H_β is a line for $p = 2$
- H_β is a plane for $p = 3$
- H_β is a hyperplane for higher dimensions

Example in two dimensions

Recall that a line in $\mathbb{R}^2$ is usually written as set of points (x_1, x_2) that fulfill:

$$x_2 = mx_1 + b$$

for some slope and intercept $m, b \in \mathbb{R}$

Rearranging the equation we get:

$$-b = mx_1 - x_2 = \langle \beta, x \rangle$$

for $\beta = (m, -1)^\top$ and $\beta_0 = b$, which is identical to the formulation before.

Example in three dimensions

For two dimensional planes, one usually writes:

$$ax_1 + bx_2 + cx_3 = -d$$

Which, again, is the same for $\beta = (a, b, c)^\top$ and $\beta_0 = d$.

β **is orthogonal to the plane**, as:

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

for any two points $x, x' \in H_\beta$, thus β is orthogonal to any translation vector within the plane and therefore is orthogonal to the plane. If we normalize β, then

$$n = \frac{\beta}{||\beta||}$$

is a normal vector to 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) First we show that the projected point is element of the hyperplane, i.e. $\pi x := \pi_{H_\beta}(x) \in H_\beta$:

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

Thus, $\pi_{H_\beta}(x)$ fulfills the criterion for a point to be located on 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

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

(ii) We show that $\pi_{H_\beta}(x)$ 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.
Thus $||x - x'||^2 \geq ||x - \pi x||^2$ and equality holds for $x' = \pi x$!

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

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

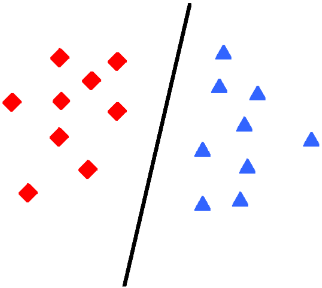
Separating Hyperplanes

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

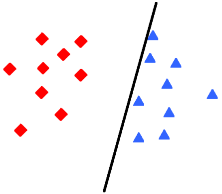
$$y_i h(x_i) > 0, \quad i = 1, \ldots, n, \quad \text{with } h(x) := \langle \beta, x \rangle + \beta_0$$

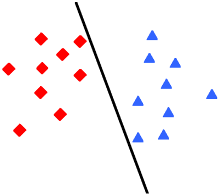


Linear Separable Data

The data is called **linear separable** if there exists such a separating hyperplane.

In general, if there is one, there are many, for example:





$\Rightarrow$ If there is a choice, we need a criterion to narrow down which one we want / is the best.

Outline

1. Separating Hyperplanes

2. Perceptron

3. Maximum Margin Separating Hyperplanes

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

$$Y(X) = \arg\max_y p(Y = y \mid X)$$

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

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

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: Example

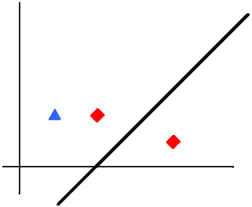
Let us have the data:

$$X = \begin{pmatrix} 1 & 2 \\ 4 & 1 \\ 2 & 2 \end{pmatrix} \qquad y = \begin{pmatrix} -1 \\ 1 \\ 1 \end{pmatrix}$$

We start with the initial hyperplane defined through

$$\beta = (1, -1)^\top \qquad \beta_0 = -2$$

which looks like this:



Perceptron: Example

We sequentially check all instances in a random order for misclassification

$$\langle \beta, x_1 \rangle + \beta_0 = (1, -1) \begin{pmatrix} 1 \\ 2 \end{pmatrix} - 2 = -3$$

$$\langle \beta, x_2 \rangle + \beta_0 = (1, -1) \begin{pmatrix} 4 \\ 1 \end{pmatrix} - 2 = 1$$

$$\langle \beta, x_3 \rangle + \beta_0 = (1, -1) \begin{pmatrix} 2 \\ 2 \end{pmatrix} - 2 = -2$$

and update the parameters as soon as an error is detected (in this case at x_3). Let us use a **learning rate** of $\alpha = 1/4$, then:

$$\beta^{\text{new}} = \begin{pmatrix} 1 \\ -1 \end{pmatrix} + 1/4 \cdot x_2 = \begin{pmatrix} 1 \\ -1 \end{pmatrix} + 1/4 \cdot \begin{pmatrix} 2 \\ 2 \end{pmatrix} = \begin{pmatrix} 1.5 \\ -0.5 \end{pmatrix}$$

$$\beta_0{}^{\text{new}} = \beta_0 + 1/4 = -2 + 1/4 = -1.75$$

Perceptron: Example

Now let us check the new hyperplane:

$$\langle \beta^{\text{new}}, x_1 \rangle + {\beta_0}^{\text{new}} = (1.5, -0.5) \begin{pmatrix} 1 \\ 2 \end{pmatrix} - 1.75 = -1.25$$

$$\langle \beta^{\text{new}}, x_2 \rangle + {\beta_0}^{\text{new}} = (1.5, -0.5) \begin{pmatrix} 4 \\ 1 \end{pmatrix} - 1.75 = 3.75$$

$$\langle \beta^{\text{new}}, x_3 \rangle + {\beta_0}^{\text{new}} = (1.5, -0.5) \begin{pmatrix} 2 \\ 2 \end{pmatrix} - 1.75 = 0.25$$

And all instances are classified correctly, algorithm stops.

The correct setting of the **learning rate** α cannot be determined beforehand and thus α is a hyperparameter of the method.

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.

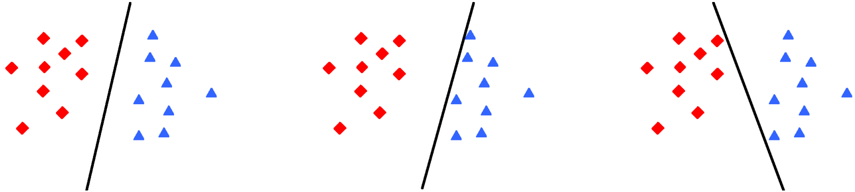
But there are many problems with this simple algorithm:

- 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.



$\Rightarrow$ We would probably choose the leftmost hyperplane, as it seems most general.

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

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

This problem is a convex optimization problem that can be solved using Lagrange Multipliers.

Merry Christmas and a happy new year!