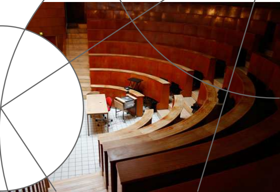
Faculty of Science

Linear Classification

Statistical Methods for Machine Learning

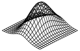
Christian Igel
Department of Computer Science



Recall I: Gaussian Distribution

Gaussian distribution of a single real-valued variable with mean $\mu \in \mathbb{R}$ and variance σ^2:

$$\mathsf{N}(x \,|\, \mu, \sigma^2) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left\{-\frac{1}{2\sigma^2}(x-\mu)^2\right\}$$



Multivariate Gaussian distribution of a d-dimensional real-valued random vector with mean $\boldsymbol{\mu} \in \mathbb{R}^d$ and covariance matrix $\boldsymbol{\Sigma} \in \mathbb{R}^{d\times d}$:

$$\mathsf{N}(\boldsymbol{x} \,|\, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{\sqrt{(2\pi)^d \det \boldsymbol{\Sigma}}} \exp\left\{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^{\mathsf{T}}\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right\}$$

Some reasons for Gaussians

Normal distributions play an important role in modeling for several reasons:

- Gaussians arise in the *central limit theorem*, which states that the probability distribution of a sum of n i.i.d. random variables with finite mean and variance approaches a Gaussian distribution with increasing n.

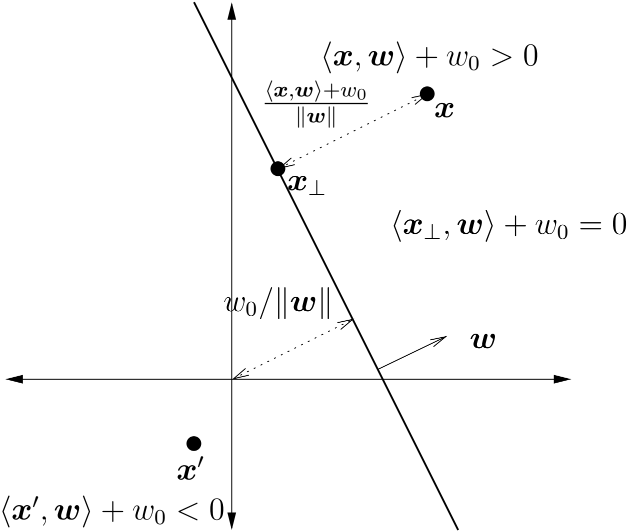
  Thus, if some outcome depends on several sources of randomness (and we assume that these sources add up) it may be well described by a Gaussian.

- Among all distributions having some given mean and variance, the Gaussian distribution has the highest entropy.

  This means, if we can or want to fix mean and variance and want to express maximum uncertainty about the outcomes, then we arrive at the Gaussian distribution.

Recall II: Linear functions

$$f(\boldsymbol{x}) = \langle \boldsymbol{x}, \boldsymbol{w} \rangle + w_0$$



Outline

Outline

Decision functions

- Classification means assigning an input $x \in \mathcal{X}$ to one class of a finite set of classes $\mathcal{Y} = \{\mathcal{C}_1, \dots, \mathcal{C}_m\}$, $2 \leq m < \infty$.

- One approach is to learn appropriate discrimination functions $\delta_k : \mathcal{X} \to \mathbb{R}$, $1 \leq k \leq m$, and assign a pattern x to class $\hat{y}$ using:

$$\hat{y} = h(x) = \mathrm{argmax}_k \, \delta_k(x)$$

Binary decision functions

- If we have only two classes, we can consider a single function

$$\delta(x) = \delta_1(x) - \delta_2(x)$$

and the hypothesis

$$h(x) = \begin{cases} \mathcal{C}_1 & \text{if } \delta(x) > 0 \\ \mathcal{C}_2 & \text{otherwise} \end{cases} .$$

- For $\mathcal{Y} = \{-1, 1\}$ this is equal to

$$h(x) = \operatorname{sgn}(\delta(x)) = \begin{cases} 1 & \text{if } \delta(x) > 0 \\ -1 & \text{otherwise} \end{cases} .$$

Decision functions and class posteriors

- If we know the class posteriors $\Pr(Y \mid X)$ we can perform optimal classification: a pattern x is assigned to class $\mathcal{C}_k$ with maximum $\Pr(Y = \mathcal{C}_k \mid X = x)$, i.e.,

$$\hat{y} = h(x) = \mathrm{argmax}_k \Pr(Y = \mathcal{C}_k \mid X = x)$$

  or in the binary case with $\mathcal{Y} = \{-1, 1\}$

$$\delta = \Pr(Y = \mathcal{C}_1 \mid X = x) - \Pr(Y = \mathcal{C}_2 \mid X = x)$$

  and $\hat{y} = h(x) = \mathrm{sgn}(\delta(x))$.

- $\Pr(Y = \mathcal{C}_k \mid X = x)$ is proportional to the class-conditional density $p(X = x \mid Y = \mathcal{C}_k)$ times the class prior $\Pr(Y = \mathcal{C}_k)$:

$$\Pr(Y = \mathcal{C}_k \mid X = x) = \frac{p(X = x \mid Y = \mathcal{C}_k)\Pr(Y = \mathcal{C}_k)}{p(X = x)}$$

Gaussian class-conditionals

Let's consider

$$\ln \Pr(Y = \mathcal{C}_k \mid X = \boldsymbol{x}) = \ln p(X = \boldsymbol{x} \mid Y = \mathcal{C}_k) + \ln \Pr(Y = \mathcal{C}_k) + \text{const}$$

For $\mathcal{X} = \mathbb{R}^d$ and Gaussian class-conditionals

$$p(X = x \mid Y = \mathcal{C}_k) = \mathsf{N}(\boldsymbol{x} \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k, Y = \mathcal{C}_k)$$

we have:

$$\ln \mathsf{N}(\boldsymbol{x} \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k, Y = \mathcal{C}_k) =$$

$$\ln \left(\frac{1}{\sqrt{(2\pi)^d \det \boldsymbol{\Sigma}_k}} \exp\left\{ -\frac{1}{2} (\boldsymbol{x} - \boldsymbol{\mu}_k)^{\mathsf{T}} \boldsymbol{\Sigma}_k^{-1} (\boldsymbol{x} - \boldsymbol{\mu}_k) \right\} \right) =$$

$$-\frac{d}{2} \ln 2\pi - \frac{1}{2} \ln \det \boldsymbol{\Sigma}_k - \frac{1}{2} (\boldsymbol{x} - \boldsymbol{\mu}_k)^{\mathsf{T}} \boldsymbol{\Sigma}_k^{-1} (\boldsymbol{x} - \boldsymbol{\mu}_k) =$$

$$-\frac{d}{2} \ln 2\pi - \frac{1}{2} \ln \det \boldsymbol{\Sigma}_k - \frac{1}{2} \boldsymbol{x}^{\mathsf{T}} \boldsymbol{\Sigma}_k^{-1} \boldsymbol{x} - \frac{1}{2} \boldsymbol{\mu}_k^{\mathsf{T}} \boldsymbol{\Sigma}_k^{-1} \boldsymbol{\mu}_k + \boldsymbol{x}^{\mathsf{T}} \boldsymbol{\Sigma}_k^{-1} \boldsymbol{\mu}_k$$

Linear Discriminant Analysis (LDA)

Assume identical covariance matrix for all class-conditionals

$$\ln \Pr(Y = \mathcal{C}_k \mid X = \boldsymbol{x}) - \text{const} = \ln \Pr(Y = \mathcal{C}_k)$$

$$-\frac{d}{2}\ln 2\pi - \frac{1}{2}\ln\det\boldsymbol{\Sigma} - \frac{1}{2}\boldsymbol{x}^{\mathsf{T}}\boldsymbol{\Sigma}^{-1}\boldsymbol{x} - \frac{1}{2}\boldsymbol{\mu}_k^{\mathsf{T}}\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_k + \boldsymbol{x}^{\mathsf{T}}\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_k \Rightarrow$$

$$\delta_k(\boldsymbol{x}) = \boldsymbol{x}^{\mathsf{T}}\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_k - \frac{1}{2}\boldsymbol{\mu}_k^{\mathsf{T}}\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_k + \ln\Pr(Y = \mathcal{C}_k)$$

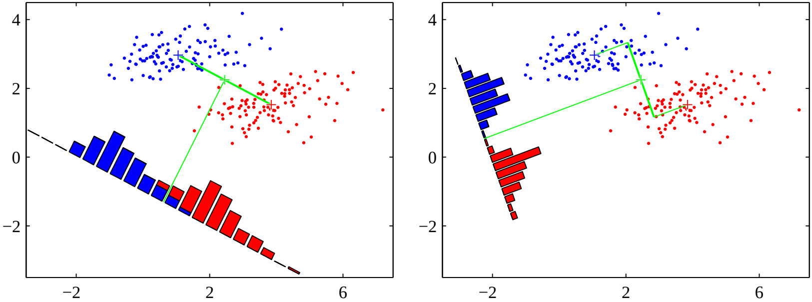
With $S_k = \{(\boldsymbol{x}, y) \in S \mid y = \mathcal{C}_k\}$ and $\ell_k = |S_k|$ we estimate:

$$\hat{\Pr}(Y = \mathcal{C}_k) = \ell_k/\ell$$

$$\hat{\boldsymbol{\mu}}_k = \frac{1}{\ell_k}\sum_{(\boldsymbol{x},y)\in S_k}\boldsymbol{x}$$

$$\hat{\boldsymbol{\Sigma}} = \frac{1}{\ell - m}\sum_{k=1}^{m}\sum_{(\boldsymbol{x},y)\in S_k}(\boldsymbol{x} - \hat{\boldsymbol{\mu}}_k)(\boldsymbol{x} - \hat{\boldsymbol{\mu}}_k)^{\mathsf{T}}$$

Effect of learning the covariance



C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer-Verlag, 2006

Linear and Quadratic Discriminant Analysis

- In LDA the decision boundaries $\{\boldsymbol{x} \mid \delta_i(\boldsymbol{x}) = \delta_j(\boldsymbol{x})\}$ between two classes i and j are linear, and the hypotheses are linear functions $\mathbb{R}^d \to \mathbb{R}$.

- Modeling independent covariance matrices for the class-conditionals leads to *quadratic discriminant analysis* (QDA) with quadratic decision functions:

$$\delta_k(\boldsymbol{x}) = -\frac{1}{2}\ln\det\boldsymbol{\Sigma}_k - \frac{1}{2}\boldsymbol{x}^{\mathsf{T}}\boldsymbol{\Sigma}_k^{-1}\boldsymbol{x} - \frac{1}{2}\boldsymbol{\mu}_k^{\mathsf{T}}\boldsymbol{\Sigma}_k^{-1}\boldsymbol{\mu}_k + \boldsymbol{x}^{\mathsf{T}}\boldsymbol{\Sigma}_k^{-1}\boldsymbol{\mu}_k + \ln\Pr(Y = \mathcal{C}_k)$$

Outline

General linear binary classification

- LDA is a linear classification method
- Given training examples

$$S = \{(\boldsymbol{x}_1, y_1), \dots, (\boldsymbol{x}_\ell, y_\ell)\} \subseteq (\mathbb{R}^n \times \{-1, 1\})^\ell$$

a binary linear classifier assigns $\boldsymbol{x} \in \mathbb{R}^n$ to one of two classes $\{-1, 1\}$ by an affine linear decision function identified by $(\boldsymbol{w}, w_0)$:

$$\delta(\boldsymbol{x}) = \langle \boldsymbol{w}, \boldsymbol{x} \rangle + w_0 = \boldsymbol{w}^\mathsf{T} \boldsymbol{x} + w_0 = \sum_{i=1}^{n} w_i x_i + w_0$$

$\boldsymbol{x}$ belongs to the first class if $\delta(\boldsymbol{x}) \geq 0$, otherwise to the second, i.e., the resulting hypothesis is:

$$h(\boldsymbol{x}) = \operatorname{sgn}(\delta(\boldsymbol{x}))$$

Margins I

The functional margin of an example $(\boldsymbol{x}_i, y_i)$ with respect to a hyperplane $(\boldsymbol{w}, w_0)$ is

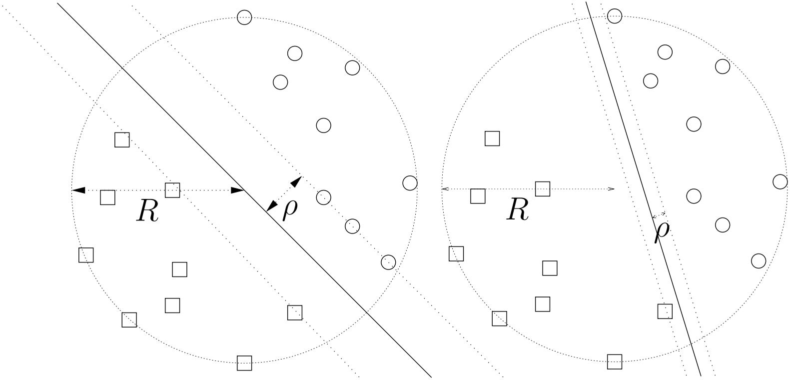
$$\gamma_i := y_i(\langle \boldsymbol{w}, \boldsymbol{x}_i \rangle + w_0) \ .$$

The geometric margin of an example $(\boldsymbol{x}_i, y_i)$ with respect to a hyperplane $(\boldsymbol{w}, w_0)$ is

$$\rho_i := y_i(\langle \boldsymbol{w}, \boldsymbol{x}_i \rangle + w_0)/\|\boldsymbol{w}\| = \gamma_i/\|\boldsymbol{w}\| \ .$$

A positive margin implies correct classification.
The margin of a hyperplane $(\boldsymbol{w}, w_0)$ with respect to a training set S is $\min_i \rho_i$. The margin of a training set S is the maximum geometric margin over all hyperplanes. A hyperplane realizing this margin is called maximum margin hyperplane.

Margins II



Outline

Analyzing the Perceptron

Why should we look at the Perceptron?

- Linear classifiers such as perceptrons are the basis of technical neurocomputing
- Support Vector Machines are basically linear classifiers
- Basic concepts of learning theory can be explained easily:
  - Margins
  - Dual representation
  - Bounds involving margins and the radius of the ball containing the data

Perceptron learning algorithm (primal form)

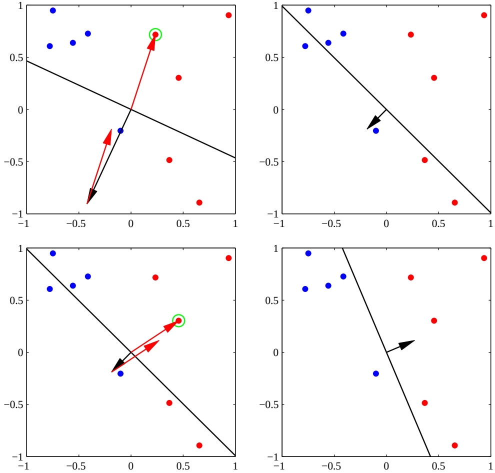
For simplicity, consider hyperplanes with no bias ($w_0 = \mathbf{0}$), i.e., $\mathcal{H} = \{h(\boldsymbol{x}) = \operatorname{sgn}(\langle \boldsymbol{w}, \boldsymbol{x} \rangle) \,|\, \boldsymbol{w} \in \mathbb{R}^n\}$.

Algorithm 1: Perceptron

Input: separable data $\{(\boldsymbol{x}_1, y_1), \dots\} \subseteq (\mathbb{R}^n \times \{-1, 1\})^\ell$
Output: hypothesis $h(\boldsymbol{x}) = \operatorname{sgn}(\langle \boldsymbol{w}_k, \boldsymbol{x} \rangle)$

```
w_0 ← 0; k ← 0
repeat
    for i = 1, ..., ℓ do
        if y_i ⟨w_k, x_i⟩ ≤ 0 then
            w_{k+1} ← w_k + y_i x_i
            k ← k + 1
until no mistake made within for loop
```

Perceptron learning in pictures



C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer-Verlag, 2006

Dual representation

- Weight vector of hyperplane computed by online perceptron algorithm can be written as

$$\boldsymbol{w} = \sum_{i=1}^{\ell} \alpha_i y_i \boldsymbol{x}_i$$

- Function $h(\boldsymbol{x}) = \operatorname{sgn}(\delta(\boldsymbol{x}))$ can be written in dual coordinates

$$\begin{aligned}\delta(\boldsymbol{x}) &= \langle \boldsymbol{w}, \boldsymbol{x} \rangle \\ &= \left\langle \sum_{i=1}^{\ell} \alpha_i y_i \boldsymbol{x}_i, \boldsymbol{x} \right\rangle \\ &= \sum_{i=1}^{\ell} \alpha_i y_i \langle \boldsymbol{x}_i, \boldsymbol{x} \rangle\end{aligned}$$

Perceptron learning algorithm (dual form)

Algorithm 2: Perceptron (dual form)

Input: separable data $\{(\boldsymbol{x}_1, y_1), \dots\} \subseteq (\mathbb{R}^n \times \{-1, 1\})^\ell$

Output: hypothesis $h(\boldsymbol{x}) = \operatorname{sgn}\left(\sum_{i=1}^{\ell} \alpha_i y_i \langle \boldsymbol{x}_i, \boldsymbol{x} \rangle\right)$

```
α ← 0
repeat
    for i = 1, ..., ℓ do
        if y_i Σ_{j=1}^{ℓ} α_j y_j ⟨x_j, x_i⟩ ≤ 0 then
            α_i ← α_i + 1
until no mistake made within for loop
```

$\boldsymbol{\alpha} \leftarrow \mathbf{0}$

repeat

  for $i = 1, \dots, \ell$ **do**

    if $y_i \sum_{j=1}^{\ell} \alpha_j y_j \langle \boldsymbol{x}_j, \boldsymbol{x}_i \rangle \leq 0$ **then**

      $\alpha_i \leftarrow \alpha_i + 1$

until *no mistake made within* **for** *loop*

Outline

Novikoff

Theorem (Novikoff)

Let $S = \{(\boldsymbol{x}_1, y_1), \dots, (\boldsymbol{x}_\ell, y_\ell)\}$ *be a non-trivial training set (i.e., containing patterns of both classes),* $\boldsymbol{w}_0 = \boldsymbol{0} = \sum_{i=1}^{m} 0\boldsymbol{x}_i$ *and let*

$$R \leftarrow \max_{1 \leq i \leq \ell} \|\boldsymbol{x}_i\| \enspace .$$

Suppose that there exists $\boldsymbol{w}_{opt}$ *and* $\rho > 0$ *such that* $\|\boldsymbol{w}_{opt}\| = 1$ *and*

$$y_i \left\langle \boldsymbol{w}_{opt}, \boldsymbol{x}_i \right\rangle \geq \rho > 0$$

for $1 \leq i \leq \ell$. *Then the number of updates* k *made by the online perceptron algorithm on* S *is at most*

$$\left(\frac{R}{\rho}\right)^2 \enspace .$$

Novikoff, sketch of proof I

Let i be the index of the example in update k

$$\begin{aligned}
\|\boldsymbol{w}_{k+1}\|^2 &= \langle \boldsymbol{w}_k + y_i \boldsymbol{x}_i, \boldsymbol{w}_k + y_i \boldsymbol{x}_i \rangle \\
&= \|\boldsymbol{w}_k\|^2 + 2 y_i \langle \boldsymbol{w}_k, \boldsymbol{x}_i \rangle + \|\boldsymbol{x}_i\|^2 \\
&\leq \|\boldsymbol{w}_k\|^2 + R^2 \\
&\leq (k+1) R^2
\end{aligned}$$

Novikoff, sketch of proof II

$$\begin{aligned}\langle \boldsymbol{w}_{\mathsf{opt}}, \boldsymbol{w}_{k+1} \rangle &= \langle \boldsymbol{w}_{\mathsf{opt}}, \boldsymbol{w}_k \rangle + y_i \langle \boldsymbol{w}_{\mathsf{opt}}, \boldsymbol{x}_i \rangle \\ &\geq \langle \boldsymbol{w}_{\mathsf{opt}}, \boldsymbol{w}_k \rangle + \rho \\ &\geq (k+1)\rho\end{aligned}$$

$$k^2\rho^2 \leq \langle \boldsymbol{w}_{\mathsf{opt}}, \boldsymbol{w}_k \rangle^2 \leq \|\boldsymbol{w}_{\mathsf{opt}}\|^2 \|\boldsymbol{w}_k\|^2 \leq kR^2$$

$$k \leq \frac{R^2}{\rho^2}$$

Outline

Summary

- Linear discriminant analysis (LDA)
  - gives good results in practice,
  - is easy to use, because it has no hyperparameters,
  - usually does not overfit, but may not capture essential non-linearities,
  - it is highly recommended as a baseline method.
- Hey, we also now know about
  - perceptron learning,
  - margins,
  - dual representation,
  - bounds involving margins and the radius of the ball containing the data.

References:

J. Shawe-Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004

C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer-Verlag, 2006