Machine Learning – Lecture 6

Linear Discriminants II

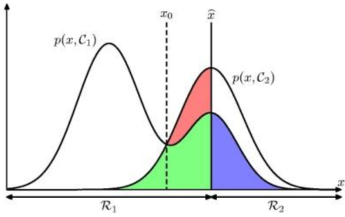
09.05.2016

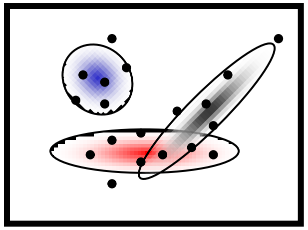
Bastian Leibe
RWTH Aachen
http://www.vision.rwth-aachen.de

leibe@vision.rwth-aachen.de

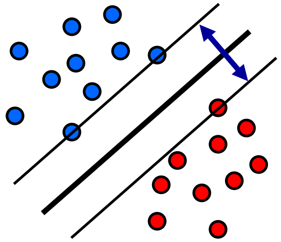
Course Outline

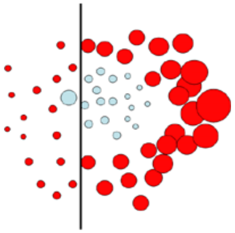
- **Fundamentals (2 weeks)**
  - Bayes Decision Theory
  - Probability Density Estimation



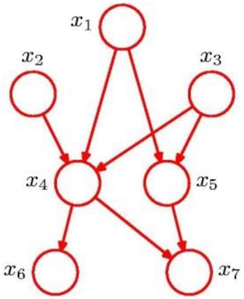


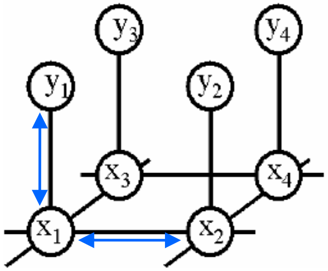
- **Discriminative Approaches (5 weeks)**
  - Linear Discriminant Functions
  - Support Vector Machines
  - Ensemble Methods & Boosting
  - Randomized Trees, Forests & Ferns





- **Generative Models (4 weeks)**
  - Bayesian Networks
  - Markov Random Fields





Recap: Linear Discriminant Functions

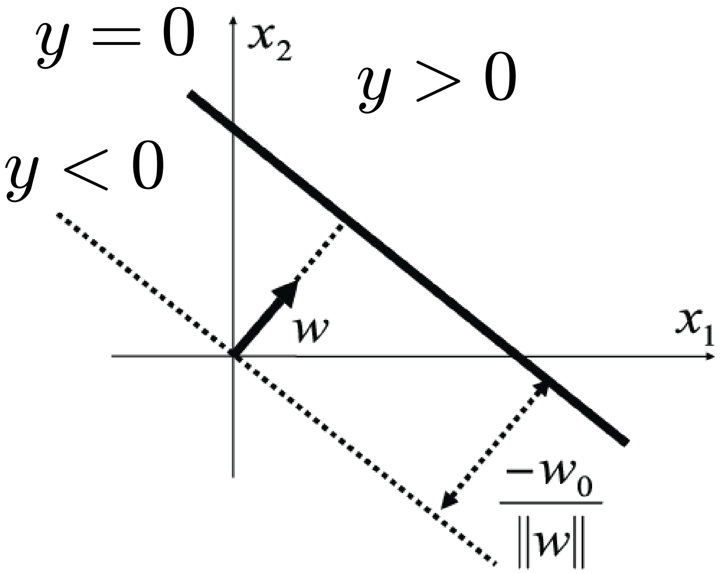
- Basic idea
  - Directly encode decision boundary
  - Minimize misclassification probability directly.

- Linear discriminant functions

$$y(\mathbf{x}) = \mathbf{w}^{\mathrm{T}}\mathbf{x} + w_0$$

weight vector

"bias"
(= threshold)



  - $\mathbf{w}$, w_0 define a hyperplane in $\mathbb{R}^D$.
  - If a data set can be perfectly classified by a linear discriminant, then we call it linearly separable.

Slide adapted from Bernt Schiele

Recap: Least-Squares Classification

- Simplest approach
  - Directly try to minimize the sum-of-squares error

$$E(\mathbf{w}) = \sum_{n=1}^{N} (y(\mathbf{x}_n; \mathbf{w}) - \mathbf{t}_n)^2$$

$$E_D(\widetilde{\mathbf{W}}) = \frac{1}{2}\mathrm{Tr}\left\{(\widetilde{\mathbf{X}}\widetilde{\mathbf{W}} - \mathbf{T})^{\mathrm{T}}(\widetilde{\mathbf{X}}\widetilde{\mathbf{W}} - \mathbf{T})\right\}$$

  - Setting the derivative to zero yields

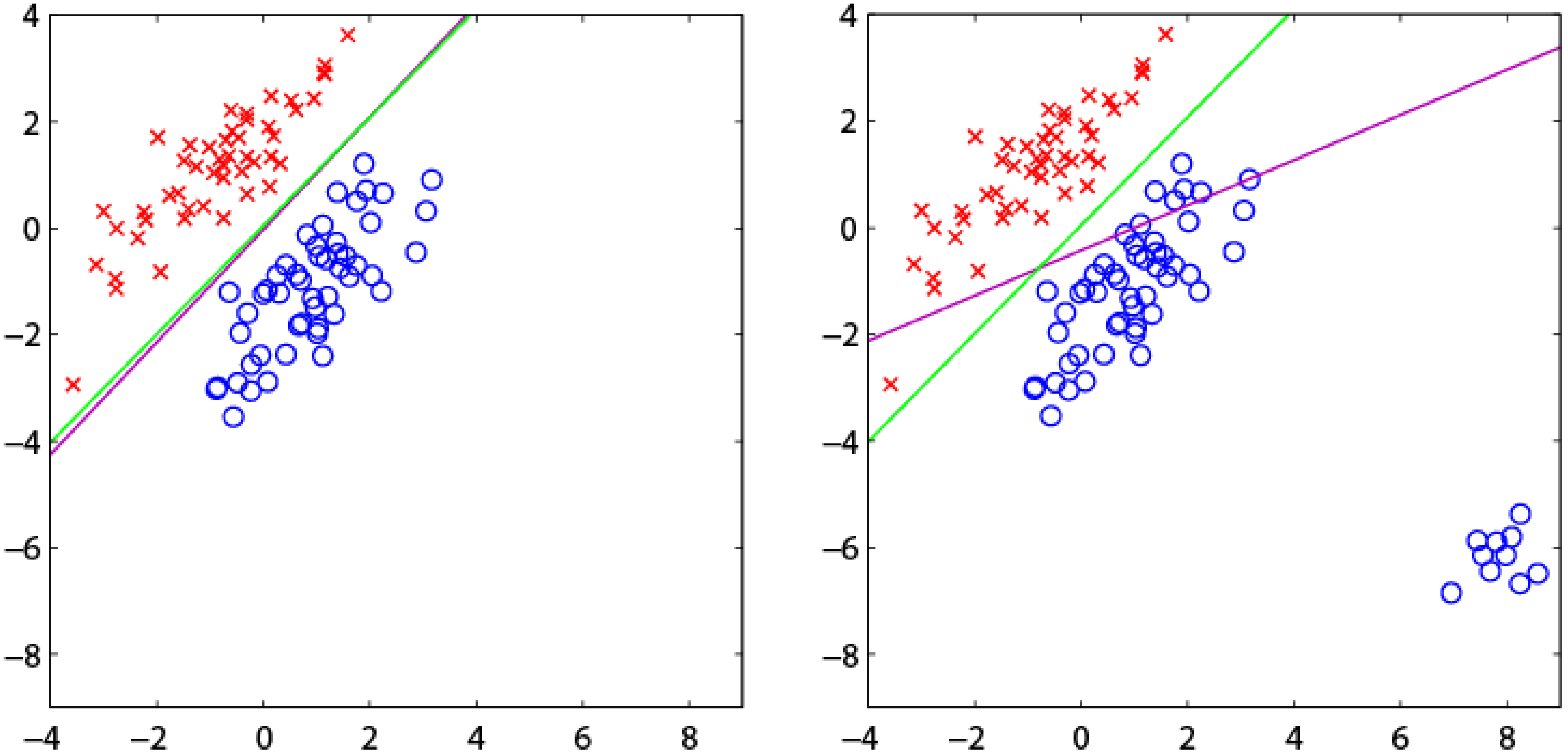
$$\widetilde{\mathbf{W}} \;=\; (\widetilde{\mathbf{X}}^{\mathrm{T}}\widetilde{\mathbf{X}})^{-1}\widetilde{\mathbf{X}}^{\mathrm{T}}\mathbf{T} = \widetilde{\mathbf{X}}^{\dagger}\mathbf{T} = (\widetilde{\mathbf{X}}^{\mathrm{T}}\widetilde{\mathbf{X}})^{-1}\widetilde{\mathbf{X}}^{\mathrm{T}}\mathbf{T}$$

  - We then obtain the discriminant function as

$$\mathbf{y}(\mathbf{x}) = \widetilde{\mathbf{W}}^{\mathrm{T}}\widetilde{\mathbf{x}} = \mathbf{T}^{\mathrm{T}}\left(\widetilde{\mathbf{X}}^{\dagger}\right)^{\mathrm{T}}\widetilde{\mathbf{x}}$$

⇒ Exact, closed-form solution for the discriminant function parameters.

Recap: Problems with Least Squares



- **Least-squares is very sensitive to outliers!**
  - **The error function penalizes predictions that are "too correct".**

Image source: C.M. Bishop, 2006

Recap: Generalized Linear Models

- Generalized linear model

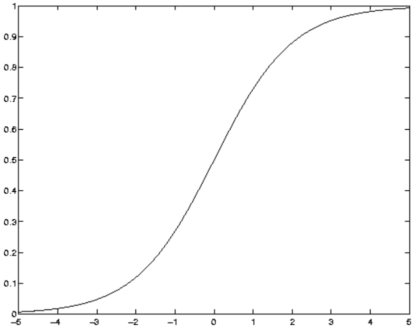
$$y(\mathbf{x}) = g(\mathbf{w}^{\mathrm{T}}\mathbf{x} + w_0)$$

  - $g(\,\cdot\,)$ is called an activation function and may be nonlinear.
  - The decision surfaces correspond to

$$y(\mathbf{x}) = const. \quad \Leftrightarrow \quad \mathbf{w}^{\mathrm{T}}\mathbf{x} + w_0 = const.$$

  - If g is monotonous (which is typically the case), the resulting decision boundaries are still linear functions of $\mathbf{x}$.

- Advantages of the non-linearity
  - Can be used to bound the influence of outliers and "too correct" data points.
  - When using a sigmoid for $g(\cdot)$, we can interpret the $y(\mathbf{x})$ as posterior probabilities.

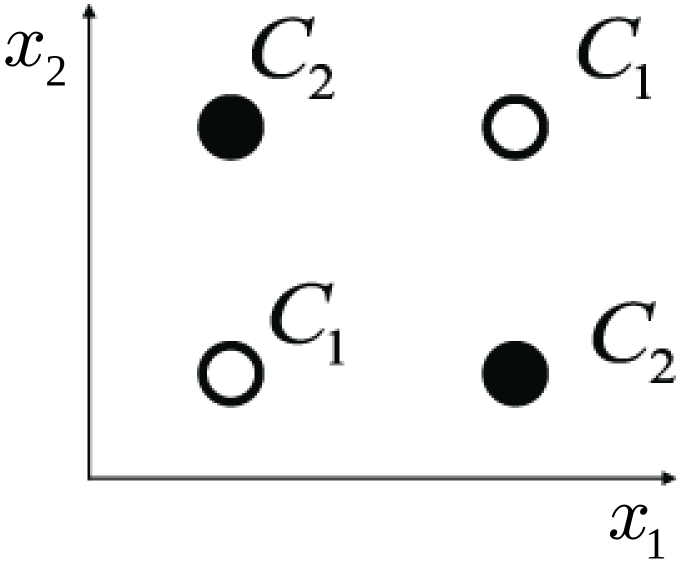


$$g(a) \equiv \frac{1}{1+\exp(-a)}$$

Recap: Linear Separability

- Up to now: restrictive assumption
  - Only consider linear decision boundaries

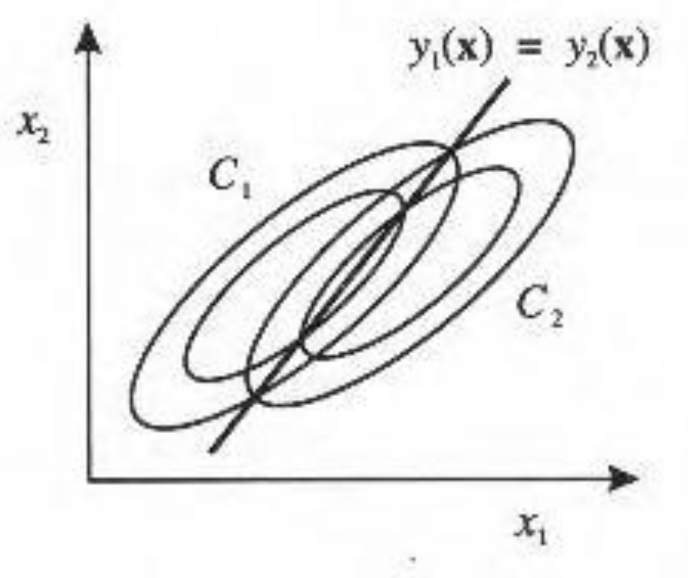
- Classical counterexample: XOR



Slide credit: Bernt Schiele

Linear Separability

- Even if the data is not linearly separable, a linear decision boundary may still be "optimal".
  - Generalization
  - E.g. in the case of Normal distributed data (with equal covariance matrices)



- Choice of the right discriminant function is important and should be based on
  - Prior knowledge (of the general functional form)
  - Empirical comparison of alternative models
  - Linear discriminants are often used as benchmark.

Slide credit: Bernt Schiele

Generalized Linear Discriminants

- Generalization
  - Transform vector $\mathbf{x}$ with M nonlinear basis functions $\phi_j(\mathbf{x})$:

$$y_k(\mathbf{x}) = \sum_{j=1}^{M} w_{kj}\phi_j(\mathbf{x}) + w_{k0}$$

  - Purpose of $\phi_j(\mathbf{x})$: basis functions
  - Allow non-linear decision boundaries.
  - By choosing the right ϕ_j, every continuous function can (in principle) be approximated with arbitrary accuracy.
- Notation

$$y_k(\mathbf{x}) = \sum_{j=0}^{M} w_{kj}\phi_j(\mathbf{x}) \qquad \text{with } \phi_0(\mathbf{x}) = 1$$

Slide credit: Bernt Schiele

Generalized Linear Discriminants

- **Model**

$$y_k(\mathbf{x}) = \sum_{j=0}^{M} w_{kj}\phi_j(\mathbf{x}) = y_k(\mathbf{x};\mathbf{w})$$

  - K **functions (outputs)** $y_k(\mathbf{x};\mathbf{w})$

- **Learning in Neural Networks**
  - **Single-layer networks:** ϕ_j **are fixed, only weights** $\mathbf{w}$ **are learned.**
  - **Multi-layer networks: both the** $\mathbf{w}$ **and the** ϕ_j **are learned.**

  - **In the following, we will not go into details about neural networks in particular, but consider generalized linear discriminants in general…**

Slide credit: Bernt Schiele

Gradient Descent

- **Learning the weights $\mathbf{w}$:**
  - N **training data points:** $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$
  - K **outputs of decision functions:** $y_k(\mathbf{x}_n;\mathbf{w})$
  - **Target vector for each data point:** $\mathbf{T} = \{\mathbf{t}_1, \ldots, \mathbf{t}_N\}$
  - **Error function (least-squares error) of linear model**

$$E(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\sum_{k=1}^{K}\left(y_k(\mathbf{x}_n;\mathbf{w}) - t_{kn}\right)^2$$

$$= \frac{1}{2}\sum_{n=1}^{N}\sum_{k=1}^{K}\left(\sum_{j=1}^{M} w_{kj}\phi_j(\mathbf{x}_n) - t_{kn}\right)^2$$

Slide credit: Bernt Schiele

Gradient Descent

- Problem
  - The error function can in general no longer be minimized in closed form.

- Idea (Gradient Descent)
  - Iterative minimization
  - Start with an initial guess for the parameter values $w_{kj}^{(0)}$.
  - Move towards a (local) minimum by following the gradient.

$$w_{kj}^{(\tau+1)} = w_{kj}^{(\tau)} - \eta \left. \frac{\partial E(\mathbf{w})}{\partial w_{kj}} \right|_{\mathbf{w}^{(\tau)}}$$

η: Learning rate

  - This simple scheme corresponds to a 1st-order Taylor expansion (There are more complex procedures available).

Gradient Descent - Basic Strategies

- "Batch learning"

$$w_{kj}^{(\tau+1)} = w_{kj}^{(\tau)} - \eta \left. \frac{\partial E(\mathbf{w})}{\partial w_{kj}} \right|_{\mathbf{w}^{(\tau)}}$$

η: Learning rate

  - Compute the gradient based on all training data:

$$\frac{\partial E(\mathbf{w})}{\partial w_{kj}}$$

Slide credit: Bernt Schiele

Gradient Descent - Basic Strategies

- "Sequential updating"

$$E(\mathbf{w}) = \sum_{n=1}^{N} E_n(\mathbf{w})$$

$$w_{kj}^{(\tau+1)} = w_{kj}^{(\tau)} - \eta \left. \frac{\partial E_n(\mathbf{w})}{\partial w_{kj}} \right|_{\mathbf{w}^{(\tau)}}$$

η: Learning rate

  - Compute the gradient based on a single data point at a time:

$$\frac{\partial E_n(\mathbf{w})}{\partial w_{kj}}$$

Slide credit: Bernt Schiele

Gradient Descent

- **Error function**

$$E(\mathbf{w}) = \sum_{n=1}^{N} E_n(\mathbf{w}) \;=\; \frac{1}{2}\sum_{n=1}^{N}\sum_{k=1}^{K}\left(\sum_{j=1}^{M} w_{kj}\phi_j(\mathbf{x}_n) - t_{kn}\right)^2$$

$$E_n(\mathbf{w}) \;=\; \frac{1}{2}\sum_{k=1}^{K}\left(\sum_{j=1}^{M} w_{kj}\phi_j(\mathbf{x}_n) - t_{kn}\right)^2$$

$$\frac{\partial E_n(\mathbf{w})}{\partial w_{kj}} \;=\; \left(\sum_{\tilde{j}=1}^{M} w_{k\tilde{j}}\phi_{\tilde{j}}(\mathbf{x}_n) - t_{kn}\right)\phi_j(\mathbf{x}_n)$$

$$=\; \left(y_k(\mathbf{x}_n;\mathbf{w}) - t_{kn}\right)\phi_j(\mathbf{x}_n)$$

Slide credit: Bernt Schiele

Gradient Descent

- **Delta rule (=LMS rule)**

$$w_{kj}^{(\tau+1)} = w_{kj}^{(\tau)} - \eta \left(y_k(\mathbf{x}_n; \mathbf{w}) - t_{kn} \right) \phi_j(\mathbf{x}_n)$$

$$= w_{kj}^{(\tau)} - \eta \delta_{kn} \phi_j(\mathbf{x}_n)$$

  - where

$$\delta_{kn} = y_k(\mathbf{x}_n; \mathbf{w}) - t_{kn}$$

⇒ Simply feed back the input data point, weighted by the classification error.

Slide credit: Bernt Schiele

Gradient Descent

- Cases with differentiable, non-linear activation function

$$y_k(\mathbf{x}) = g(a_k) = g\left(\sum_{j=0}^{M} w_{ki}\phi_j(\mathbf{x}_n)\right)$$

- Gradient descent

$$\frac{\partial E_n(\mathbf{w})}{\partial w_{kj}} = \frac{\partial g(a_k)}{\partial w_{kj}} \left(y_k(\mathbf{x}_n;\mathbf{w}) - t_{kn}\right)\phi_j(\mathbf{x}_n)$$

$$w_{kj}^{(\tau+1)} = w_{kj}^{(\tau)} - \eta\delta_{kn}\phi_j(\mathbf{x}_n)$$

$$\delta_{kn} = \frac{\partial g(a_k)}{\partial w_{kj}} \left(y_k(\mathbf{x}_n;\mathbf{w}) - t_{kn}\right)$$

Slide credit: Bernt Schiele

Summary: Generalized Linear Discriminants

- Properties
  - General class of decision functions.
  - Nonlinearity $g(\cdot)$ and basis functions ϕ_j allow us to address linearly non-separable problems.
  - Shown simple sequential learning approach for parameter estimation using gradient descent.
  - Better 2^{nd} order gradient descent approaches available (e.g. Newton-Raphson).
- Limitations / Caveats
  - Flexibility of model is limited by curse of dimensionality
    - $g(\cdot)$ and ϕ_j often introduce additional parameters.
    - Models are either limited to lower-dimensional input space or need to share parameters.
  - Linearly separable case often leads to overfitting.
    - Several possible parameter choices minimize training error.

Topics of This Lecture

- Fisher's linear discriminant (FLD)
  - Classification as dimensionality reduction
  - Linear discriminant analysis
  - Multiple discriminant analysis
  - Applications

- Logistic Regression
  - Probabilistic discriminative models
  - Logistic sigmoid (logit function)
  - Cross-entropy error
  - Gradient descent
  - Iteratively Reweighted Least Squares

- Note on Error Functions

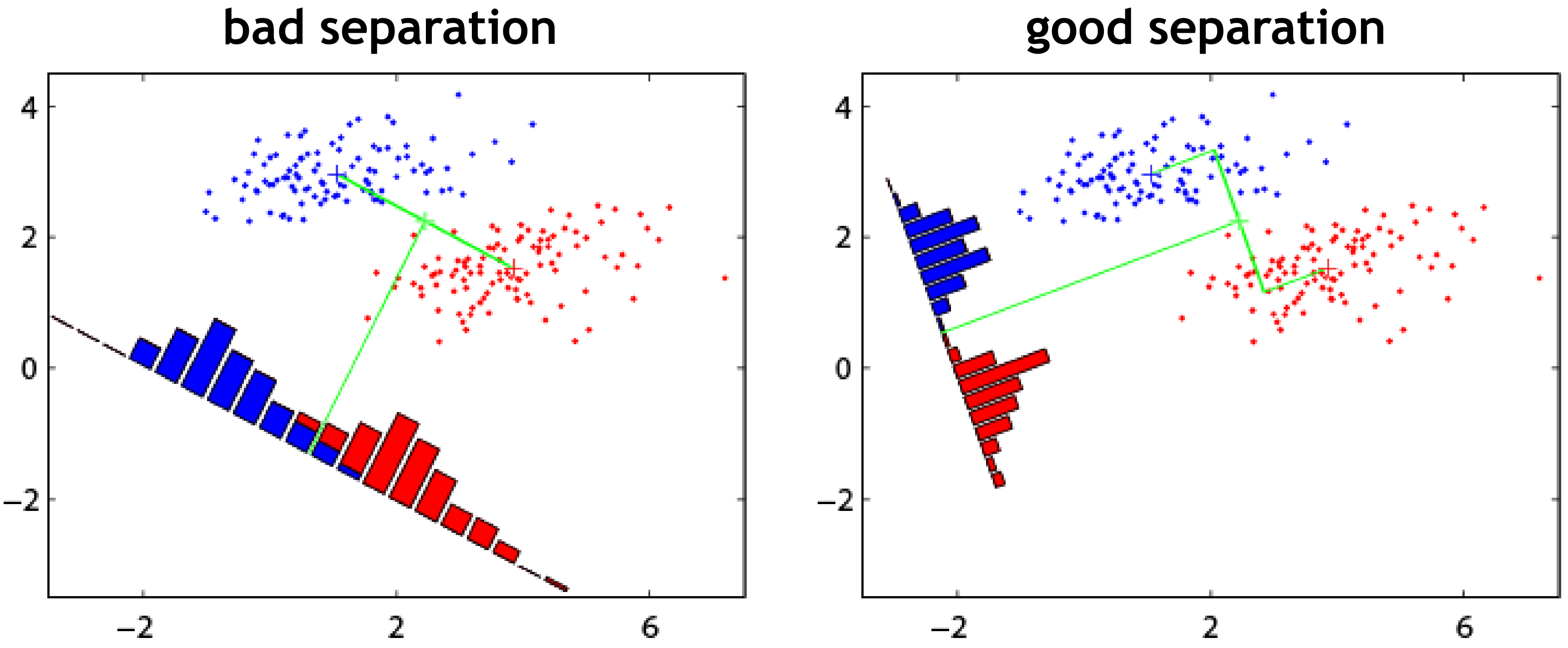
Classification as Dimensionality Reduction

- Classification as dimensionality reduction
  - We can interpret the linear classification model as a projection onto a lower-dimensional space.
  - E.g., take the D-dimensional input vector $\mathbf{x}$ and project it down to one dimension by applying the function
$$y = \mathbf{w}^{\mathrm{T}}\mathbf{x}$$
  - If we now place a threshold at $y \geq -w_{\mathrm{o}}$, we obtain our standard two-class linear classifier.
  - The classifier will have a lower error the better this projection separates the two classes.

⇒ New interpretation of the learning problem

  - Try to find the projection vector $\mathbf{w}$ that maximizes the class separation.

Classification as Dimensionality Reduction



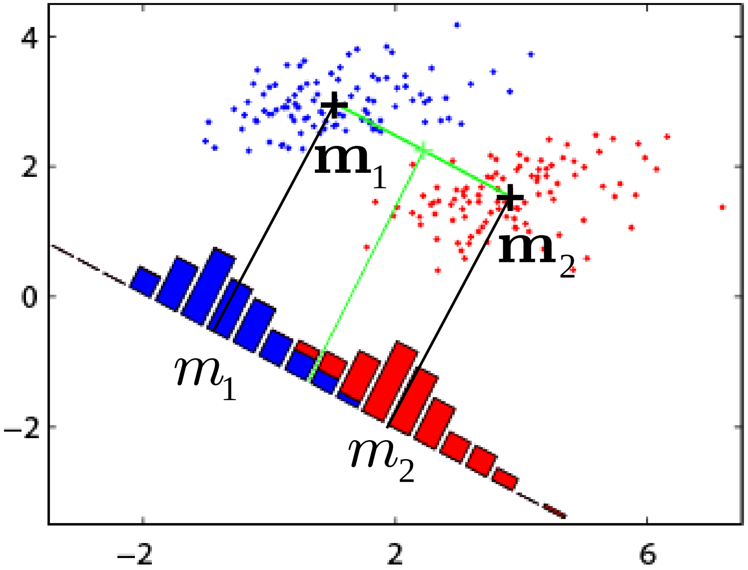
- **Two questions**
  - How to measure class separation?
  - How to find the best projection (with maximal class separation)?

Image source: C.M. Bishop, 2006

Classification as Dimensionality Reduction

- **Measuring class separation**
  - We could simply measure the separation of the class means.
  - ⇒ Choose $\mathbf{w}$ so as to maximize

$$(m_2 - m_1) = \mathbf{w}^{\mathrm{T}}(\mathbf{m}_2 - \mathbf{m}_1)$$



- **Problems with this approach**
  1. This expression can be made arbitrarily large by increasing $\|\mathbf{w}\|$.
     ⇒ Need to enforce additional constraint $\|\mathbf{w}\| = 1$.
     ⇒ This constrained minimization results in $\mathbf{w} \propto (\mathbf{m}_2 - \mathbf{m}_1)$
  2. The criterion may result in bad separation if the clusters have elongated shapes.

Image source: C.M. Bishop, 2006

Fisher's Linear Discriminant Analysis (FLD)

- **Better idea:**
  - Find a projection that maximizes the ratio of the between-class variance to the within-class variance:

$$J(\mathbf{w}) = \frac{(m_2 - m_1)^2}{s_1^2 + s_2^2} \quad \text{with} \quad s_k^2 = \sum_{n \in \mathcal{C}_k} (y_n - m_k)^2$$

  - Usually, this is written as

$$J(\mathbf{w}) = \frac{\mathbf{w}^{\mathrm{T}} \mathbf{S}_B \mathbf{w}}{\mathbf{w}^{\mathrm{T}} \mathbf{S}_W \mathbf{w}}$$

  - where

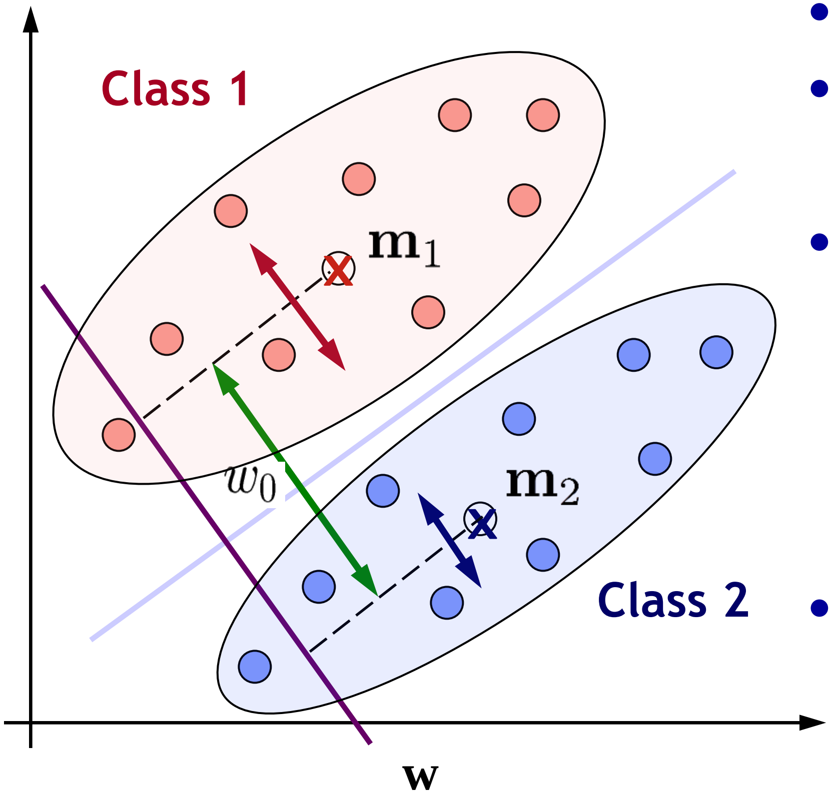
$$\mathbf{S}_B = (\mathbf{m}_2 - \mathbf{m}_1)(\mathbf{m}_2 - \mathbf{m}_1)^{\mathrm{T}}$$

between-class scatter matrix

$$\mathbf{S}_W = \sum_{k=1}^{2} \sum_{n \in \mathcal{C}_k} (\mathbf{x}_n - \mathbf{m}_k)(\mathbf{x}_n - \mathbf{m}_k)^{\mathrm{T}}$$

within-class scatter matrix

Fisher's Linear Discriminant Analysis (FLD)



- Maximize distance between classes
- Minimize distance within a class
- Criterion: $J(\mathbf{w}) = \frac{\mathbf{w}^{\mathrm{T}}\mathbf{S}_B\mathbf{w}}{\mathbf{w}^{\mathrm{T}}\mathbf{S}_W\mathbf{w}}$

$\mathbf{S}_B$ … between-class scatter matrix
$\mathbf{S}_W$ … within-class scatter matrix

- The optimal solution for $\mathbf{w}$ can be obtained as:

$$\mathbf{w} \propto \mathbf{S}_W^{-1}(\mathbf{m}_2 - \mathbf{m}_1)$$

- Classification function:

$$y(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + w_0 \overset{\text{Class 1}}{\underset{\text{Class 2}}{\gtrless}} 0$$

Slide adapted from Ales Leonardis

Multiple Discriminant Analysis

- Generalization to K classes

$$J(\mathbf{W}) = \frac{|\mathbf{W}^{\mathrm{T}}\mathbf{S}_B\mathbf{W}|}{|\mathbf{W}^{\mathrm{T}}\mathbf{S}_W\mathbf{W}|}$$

  - where

$$\mathbf{W} = [\mathbf{w}_1, \ldots, \mathbf{w}_K] \qquad \mathbf{m} = \frac{1}{N}\sum_{n=1}^{N}\mathbf{x}_n = \frac{1}{N}\sum_{k=1}^{K}N_k\mathbf{m}_k$$

$$\mathbf{S}_B = \sum_{k=1}^{K}N_k(\mathbf{m}_k - \mathbf{m})(\mathbf{m}_k - \mathbf{m})^{\mathrm{T}}$$

$$\mathbf{S}_W = \sum_{k=1}^{K}\sum_{n\in\mathcal{C}_k}(\mathbf{x}_n - \mathbf{m}_k)(\mathbf{x}_n - \mathbf{m}_k)^{\mathrm{T}}$$

Maximizing J(W)

- "Rayleigh quotient" ⇒ Generalized eigenvalue problem

$$J(\mathbf{W}) = \frac{|\mathbf{W}^{\mathrm{T}}\mathbf{S}_B\mathbf{W}|}{|\mathbf{W}^{\mathrm{T}}\mathbf{S}_W\mathbf{W}|}$$

  - The columns of the optimal $\mathbf{W}$ are the eigenvectors corresponding to the largest eigenvalues of

$$\mathbf{S}_B\mathbf{w}_i = \lambda_i\mathbf{S}_W\mathbf{w}_i$$

  - Defining $\mathbf{v} = \mathbf{S}_B^{\frac{1}{2}}\mathbf{w}$, we get

$$\mathbf{S}_B^{\frac{1}{2}}\mathbf{S}_W^{-1}\mathbf{S}_B^{\frac{1}{2}}\mathbf{v} = \lambda\mathbf{v}$$

    which is a regular eigenvalue problem.

  ⇒ Solve to get eigenvectors of $\mathbf{v}$, then from that of $\mathbf{w}$.

- For the K-class case we obtain (at most) K-1 projections.
  - (i.e. eigenvectors corresponding to non-zero eigenvalues.)

What Does It Mean?

- **What does it mean to apply a linear classifier?**

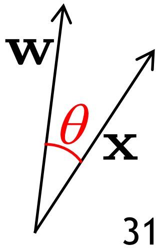
$$y(\mathbf{x}) = \underbrace{\widetilde{\mathbf{w}}}_{\text{Weight vector}}{}^{\mathrm{T}}\underbrace{\widetilde{\mathbf{x}}}_{\text{Input vector}}$$

- **Classifier interpretation**
  - **The weight vector has the same dimensionality as $\mathbf{x}$.**
  - **Positive contributions where $\mathrm{sign}(x_i) = \mathrm{sign}(w_i)$.**

$\Rightarrow$ **The weight vector identifies which input dimensions are important for positive or negative classification (large $|w_i|$) and which ones are irrelevant (near-zero w_i).**

$\Rightarrow$ **If the inputs $\mathbf{x}$ are normalized, we can interpret $\mathbf{w}$ as a "template" vector that the classifier tries to match.**

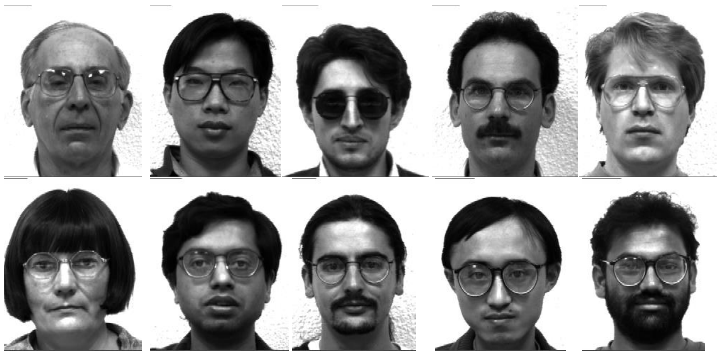
$$\mathbf{w}^{\mathrm{T}}\mathbf{x} = \|\mathbf{w}\|\,\|\mathbf{x}\|\cos\theta$$



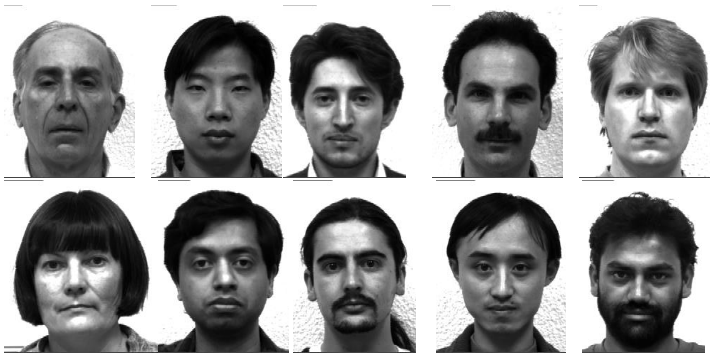
Example Application: Fisherfaces

- Visual discrimination task
  - Training data:

C_1: Subjects with glasses

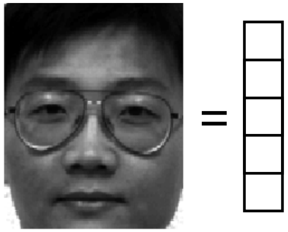


C_2: Subjects without glasses



  - Test:

– glasses?



Take each image as a vector of pixel values and apply FLD…

Image source: Yale Face Database

Fisherfaces: Interpretability

- Resulting weight vector for “Glasses/NoGlasses“

$\mathbf{x}$

$\mathbf{w}$

Slide credit: Peter Belhumeur

[Belhumeur et.al. 1997]

Summary: Fisher’s Linear Discriminant

- Properties
  - Simple method for dimensionality reduction, preserves class discriminability.
  - Can use parametric methods in reduced-dim. space that might not be feasible in original higher-dim. space.
  - Widely used in practical applications.

- Restrictions / Caveats
  - Not possible to get more than K-1 projections.
  - FLD reduces the computation to class means and covariances.

  ⇒ Implicit assumption that class distributions are unimodal and well-approximated by a Gaussian/hyperellipsoid.

Topics of This Lecture

- Fisher's linear discriminant (FLD)
  - Classification as dimensionality reduction
  - Linear discriminant analysis
  - Multiple discriminant analysis
  - Applications

- Logistic Regression
  - Probabilistic discriminative models
  - Logistic sigmoid (logit function)
  - Cross-entropy error
  - Gradient descent
  - Iteratively Reweighted Least Squares

- Note on Error Functions

Probabilistic Discriminative Models

- We have seen that we can write

$$\begin{aligned} p(\mathcal{C}_1|\mathbf{x}) &= \sigma(a) \\ &= \frac{1}{1+\exp(-a)} \end{aligned}$$

logistic sigmoid function

- We can obtain the familiar probabilistic model by setting

$$a = \ln \frac{p(\mathbf{x}|\mathcal{C}_1)p(\mathcal{C}_1)}{p(\mathbf{x}|\mathcal{C}_2)p(\mathcal{C}_2)}$$

- Or we can use generalized linear discriminant models

$$a = \mathbf{w}^T\mathbf{x}$$

or $$a = \mathbf{w}^T\phi(\mathbf{x})$$

Probabilistic Discriminative Models

- In the following, we will consider models of the form

$$p(\mathcal{C}_1|\phi) = y(\phi) = \sigma(\mathrm{w}^T\phi)$$

with
$$p(\mathcal{C}_2|\phi) = 1 - p(\mathcal{C}_1|\phi)$$

- This model is called logistic regression.

- Why should we do this? What advantage does such a model have compared to modeling the probabilities?

$$p(\mathcal{C}_1|\phi) = \frac{p(\phi|\mathcal{C}_1)p(\mathcal{C}_1)}{p(\phi|\mathcal{C}_1)p(\mathcal{C}_1) + p(\phi|\mathcal{C}_2)p(\mathcal{C}_2)}$$

- Any ideas?

Comparison

- **Let's look at the number of parameters…**
  - Assume we have an M-dimensional feature space ϕ.
  - And assume we represent $p(\phi|\mathcal{C}_k)$ and $p(\mathcal{C}_k)$ by Gaussians.
  - How many parameters do we need?
    - For the means: $2M$
    - For the covariances: $M(M+1)/2$
    - Together with the class priors, this gives $M(M+5)/2+1$ parameters!
  - How many parameters do we need for logistic regression?

$$p(\mathcal{C}_1|\phi) \;=\; y(\phi) = \sigma(\mathrm{w}^T \phi)$$

    - Just the values of w ⇒ M parameters.

⇒ ***For large M, logistic regression has clear advantages!***

Logistic Sigmoid

- Properties
  - Definition: $$\sigma(a) = \frac{1}{1+\exp(-a)}$$
  - Inverse: $$a = \ln\left(\frac{\sigma}{1-\sigma}\right)$$ "logit" function
  - Symmetry property: $$\sigma(-a) = 1 - \sigma(a)$$
  - Derivative: $$\frac{d\sigma}{da} = \sigma(1-\sigma)$$

Logistic Regression

- **Let's consider a data set $\{\phi_n, t_n\}$ with $n = 1, \ldots, N$, where $\phi_n = \phi(\mathbf{x}_n)$ and $t_n \in \{0, 1\}$, $\mathbf{t} = (t_1, \ldots, t_N)^T$.**

- **With $y_n = p(\mathcal{C}_1|\phi_n)$, we can write the likelihood as**

$$p(\mathbf{t}|\mathbf{w}) = \prod_{n=1}^{N} y_n^{t_n} \left\{1 - y_n\right\}^{1-t_n}$$

- **Define the error function as the negative log-likelihood**

$$\begin{aligned} E(\mathbf{w}) &= -\ln p(\mathbf{t}|\mathbf{w}) \\ &= -\sum_{n=1}^{N} \{t_n \ln y_n + (1 - t_n) \ln(1 - y_n)\} \end{aligned}$$

  - **This is the so-called cross-entropy error function.**

Gradient of the Error Function

$$\begin{aligned} y_n &= \sigma(\mathbf{w}^T\phi_n) \\ \frac{dy_n}{d\mathbf{w}} &= y_n(1-y_n)\phi_n \end{aligned}$$

- **Error function**

$$E(\mathbf{w}) = -\sum_{n=1}^{N}\{t_n \ln y_n + (1-t_n)\ln(1-y_n)\}$$

- **Gradient**

$$\begin{aligned} \nabla E(\mathbf{w}) &= -\sum_{n=1}^{N}\left\{t_n \frac{\frac{d}{d\mathbf{w}}y_n}{y_n} + (1-t_n)\frac{\frac{d}{d\mathbf{w}}(1-y_n)}{(1-y_n)}\right\} \\ &= -\sum_{n=1}^{N}\left\{t_n \frac{\cancel{y_n}(1-y_n)}{\cancel{y_n}}\phi_n - (1-t_n)\frac{y_n\cancel{(1-y_n)}}{\cancel{(1-y_n)}}\phi_n\right\} \\ &= -\sum_{n=1}^{N}\{(t_n - \cancel{t_n y_n} - y_n + \cancel{t_n y_n})\phi_n\} \\ &= \sum_{n=1}^{N}(y_n - t_n)\phi_n \end{aligned}$$

Gradient of the Error Function

- Gradient for logistic regression

$$\nabla E(\mathbf{w}) = \sum_{n=1}^{N} (y_n - t_n)\boldsymbol{\phi}_n$$

- Does this look familiar to you?
- This is the same result as for the Delta (=LMS) rule

$$w_{kj}^{(\tau+1)} = w_{kj}^{(\tau)} - \eta(y_k(\mathbf{x}_n; \mathbf{w}) - t_{kn})\phi_j(\mathbf{x}_n)$$

- We can use this to derive a sequential estimation algorithm.
  - However, this will be quite slow…

A More Efficient Iterative Method…

- Second-order Newton-Raphson gradient descent scheme

$$\mathbf{w}^{(\tau+1)} = \mathbf{w}^{(\tau)} - \mathbf{H}^{-1}\nabla E(\mathbf{w})$$

where $\mathbf{H} = \nabla\nabla E(\mathbf{w})$ is the Hessian matrix, i.e. the matrix of second derivatives.

- Properties
  - Local quadratic approximation to the log-likelihood.
  - Faster convergence.

Newton-Raphson for Least-Squares Estimation

- **Let's first apply Newton-Raphson to the least-squares error function:**

$$
\begin{aligned}
E(\mathbf{w}) &= \frac{1}{2}\sum_{n=1}^{N}\left(\mathbf{w}^T\phi_n - t_n\right)^2 \\
\nabla E(\mathbf{w}) &= \sum_{n=1}^{N}\left(\mathbf{w}^T\phi_n - t_n\right)\phi_n = \mathbf{\Phi}^T\mathbf{\Phi}\mathbf{w} - \mathbf{\Phi}^T\mathbf{t} \\
\mathbf{H} = \nabla\nabla E(\mathbf{w}) &= \sum_{n=1}^{N}\phi_n\phi_n^T = \mathbf{\Phi}^T\mathbf{\Phi}
\end{aligned}
$$

where $\mathbf{\Phi} = \begin{bmatrix} \phi_1^T \\ \vdots \\ \phi_N^T \end{bmatrix}$

- **Resulting update scheme:**

$$
\begin{aligned}
\mathbf{w}^{(\tau+1)} &= \mathbf{w}^{(\tau)} - (\mathbf{\Phi}^T\mathbf{\Phi})^{-1}(\mathbf{\Phi}^T\mathbf{\Phi}\mathbf{w}^{(\tau)} - \mathbf{\Phi}^T\mathbf{t}) \\
&= (\mathbf{\Phi}^T\mathbf{\Phi})^{-1}\mathbf{\Phi}^T\mathbf{t}
\end{aligned}
$$

Closed-form solution!

Newton-Raphson for Logistic Regression

- Now, let's try Newton-Raphson on the cross-entropy error function:

$$E(\mathbf{w}) = -\sum_{n=1}^{N} \{t_n \ln y_n + (1-t_n)\ln(1-y_n)\}$$

$$\boxed{\frac{dy_n}{d\mathbf{w}} = y_n(1-y_n)\phi_n}$$

$$\nabla E(\mathbf{w}) = \sum_{n=1}^{N} (y_n - t_n)\phi_n = \mathbf{\Phi}^T(\mathbf{y}-\mathbf{t})$$

$$\mathbf{H} = \nabla\nabla E(\mathbf{w}) = \sum_{n=1}^{N} y_n(1-y_n)\phi_n\phi_n^T = \mathbf{\Phi}^T\mathbf{R}\mathbf{\Phi}$$

where $\mathbf{R}$ is an $N \times N$ diagonal matrix with $R_{nn} = y_n(1-y_n)$.

⇒ The Hessian is no longer constant, but depends on $\mathbf{w}$ through the weighting matrix $\mathbf{R}$.

Iteratively Reweighted Least Squares

- Update equations

$$\begin{aligned}\mathbf{w}^{(\tau+1)} &= \mathbf{w}^{(\tau)} - (\mathbf{\Phi}^T\mathbf{R}\mathbf{\Phi})^{-1}\mathbf{\Phi}^T(\mathbf{y}-\mathbf{t}) \\ &= (\mathbf{\Phi}^T\mathbf{R}\mathbf{\Phi})^{-1}\left\{\mathbf{\Phi}^T\mathbf{R}\mathbf{\Phi}\mathbf{w}^{(\tau)} - \mathbf{\Phi}^T(\mathbf{y}-\mathbf{t})\right\} \\ &= (\mathbf{\Phi}^T\mathbf{R}\mathbf{\Phi})^{-1}\mathbf{\Phi}^T\mathbf{R}\mathbf{z}\end{aligned}$$

with $\mathbf{z} = \mathbf{\Phi}\mathbf{w}^{(\tau)} - \mathbf{R}^{-1}(\mathbf{y}-\mathbf{t})$

- Again very similar form (normal equations)
  - But now with non-constant weighing matrix $\mathbf{R}$ (depends on $\mathbf{w}$).
  - Need to apply normal equations iteratively.

⇒ Iteratively Reweighted Least-Squares (IRLS)

Summary: Logistic Regression

- Properties
  - Directly represent posterior distribution $p(\phi|\mathcal{C}_k)$
  - Requires fewer parameters than modeling the likelihood + prior.
  - Very often used in statistics.
  - It can be shown that the cross-entropy error function is concave
    - Optimization leads to unique minimum
    - But no closed-form solution exists
    - Iterative optimization (IRLS)
  - Both online and batch optimizations exist
  - There is a multi-class version described in (Bishop Ch.4.3.4).
- Caveat
  - Logistic regression tends to systematically overestimate odds ratios when the sample size is less than ~500.

Topics of This Lecture

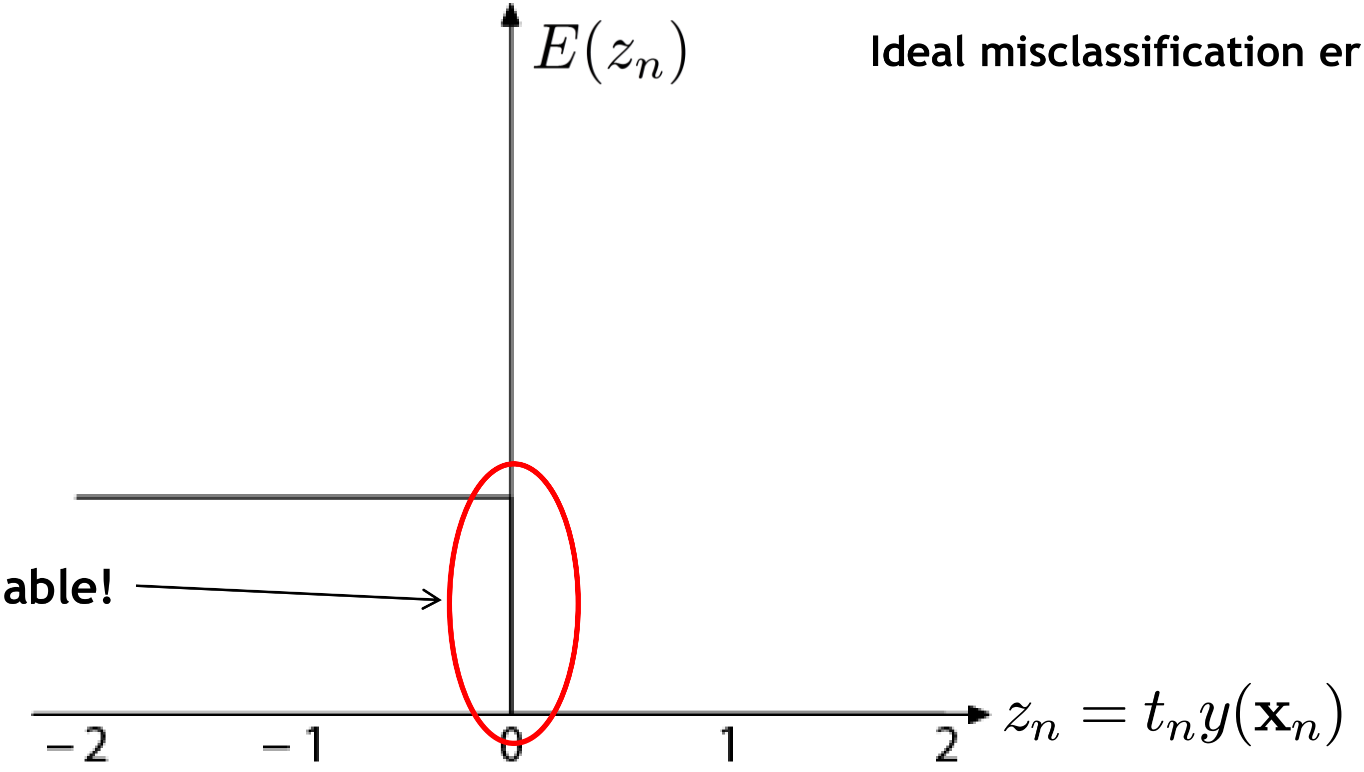
- Fisher's linear discriminant (FLD)
  - Classification as dimensionality reduction
  - Linear discriminant analysis
  - Multiple discriminant analysis
  - Applications

- Logistic Regression
  - Probabilistic discriminative models
  - Logistic sigmoid (logit function)
  - Cross-entropy error
  - Gradient descent
  - Iteratively Reweighted Least Squares

- **Note on Error Functions**

Note on Error Functions

$t_n \in \{-1, 1\}$

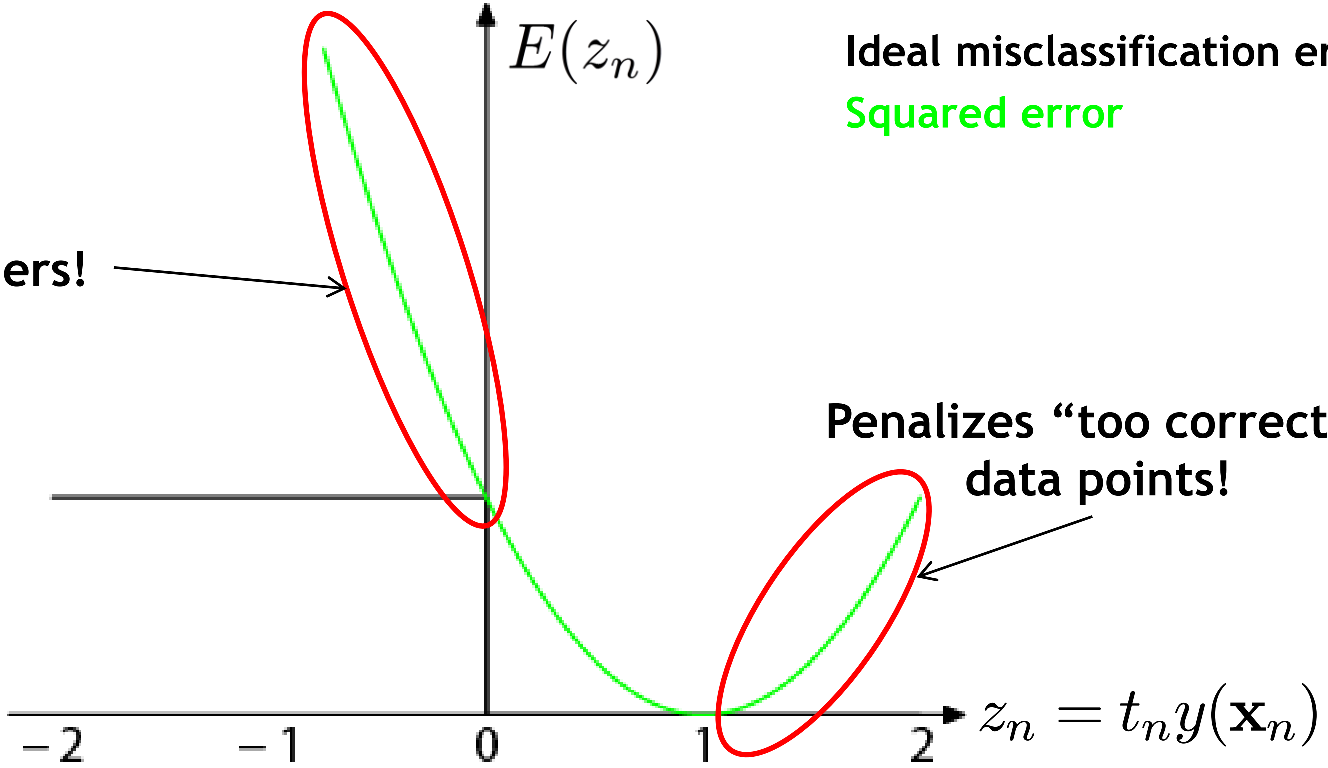


- Ideal misclassification error function (black)
  - This is what we want to approximate,
  - Unfortunately, it is not differentiable.
  - The gradient is zero for misclassified points.
  - $\Rightarrow$ We cannot minimize it by gradient descent.

Image source: Bishop, 2006

Note on Error Functions

$$t_n \in \{-1, 1\}$$



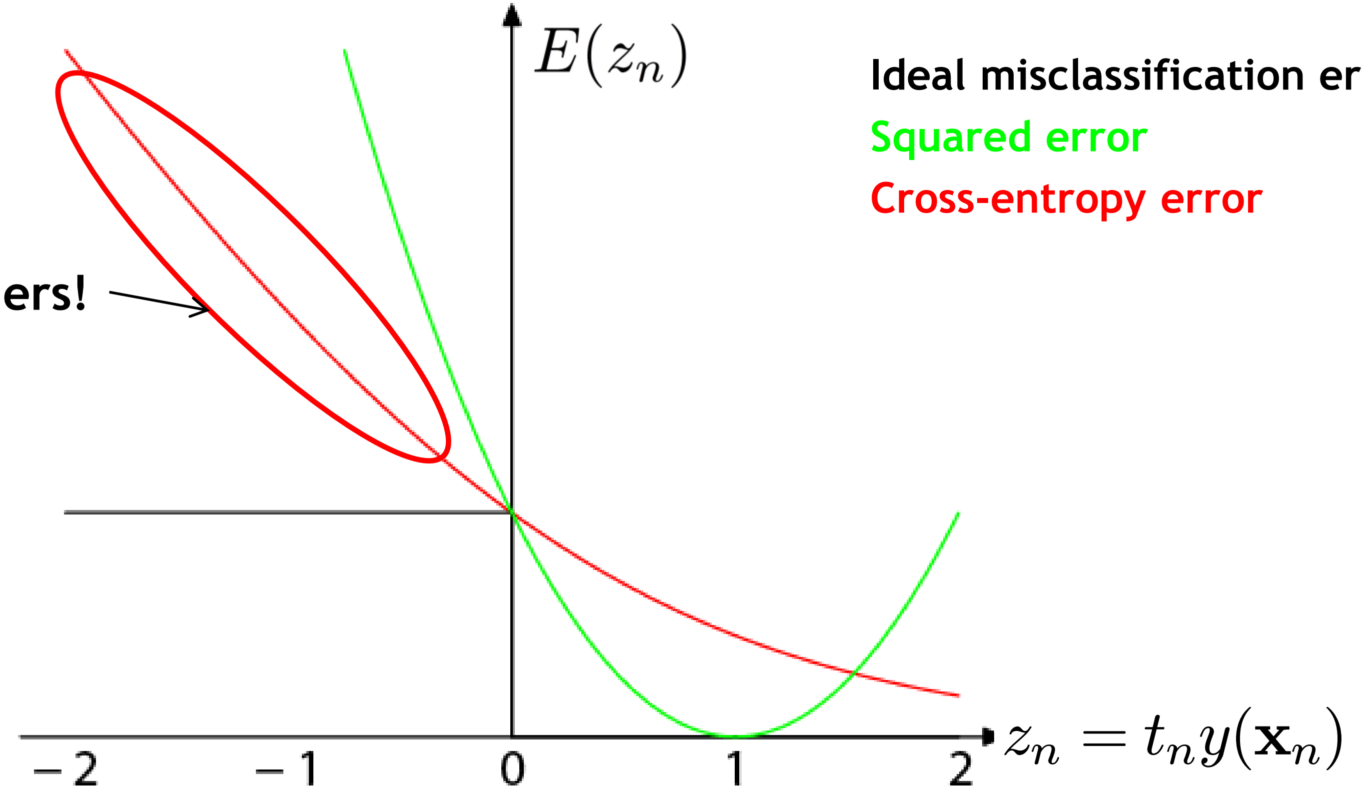
- Squared error used in Least-Squares Classification
  - Very popular, leads to closed-form solutions.
  - However, sensitive to outliers due to squared penalty.
  - Penalizes "too correct" data points

$\Rightarrow$ Generally does not lead to good classifiers.

Image source: Bishop, 2006

Comparing Error Functions (Loss Functions)

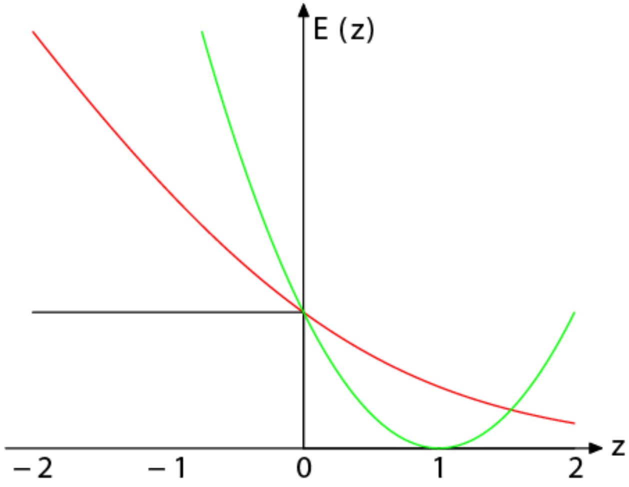
$t_n \in \{-1, 1\}$



- **Cross-Entropy Error**
  - Minimizer of this error is given by posterior class probabilities.
  - Concave error function, unique minimum exists.
  - Robust to outliers, error increases only roughly linearly
  - But no closed-form solution, requires iterative estimation.

Image source: Bishop, 2006

Overview: Error Functions

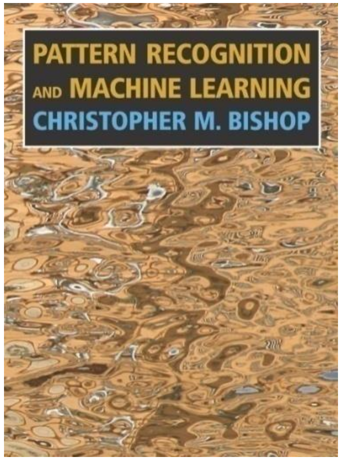


- Ideal Misclassification Error
  - This is what we would like to optimize.
  - But cannot compute gradients here.
- Quadratic Error
  - Easy to optimize, closed-form solutions exist.
  - But not robust to outliers.
- Cross-Entropy Error
  - Minimizer of this error is given by posterior class probabilities.
  - Concave error function, unique minimum exists.
  - But no closed-form solution, requires iterative estimation.

⇒ *Analysis tool to compare classification approaches*

References and Further Reading

- **More information on Linear Discriminant Functions can be found in Chapter 4 of Bishop's book (in particular Chapter 4.1 - 4.3).**



Christopher M. Bishop
Pattern Recognition and Machine Learning
Springer, 2006