Pattern Recognition
Chapter 11: Probabilistic Discriminative Classifiers

Prof. Dr.-Ing. Uwe Sörgel
soergel@ifp.uni-stuttgart.de

Contents

- Generative vs. discriminative classifiers
- Linear Discriminant Function
- Logistic Regression
- Generalized Linear Models
- Training
- Multi-class Problems

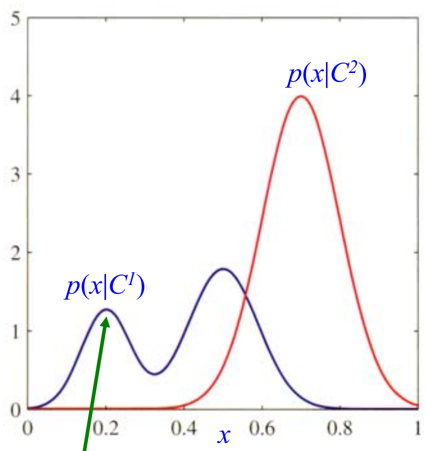
Generative vs. discriminative classifiers

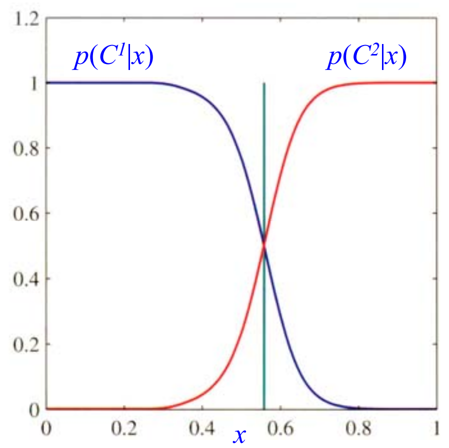
- Generative classifiers:
  - Determine the parameters of the likelihood $p(\mathbf{x}|C)$ (training).
  - Determine the prior $p(C)$.
  - Determine $p(C|\mathbf{x})$ with the help of the theorem of Bayes.
  - 'Generative': It is possible to generate synthetic data sets by sampling from the joint distribution $p(\mathbf{x},C) = p(\mathbf{x}|C) \cdot p(C)$.

- Discriminative classifiers:
  - Direct modelling of $p(C|\mathbf{x})$
  - Focus on the separating surfaces in feature space.
  - In general, this leads to simpler models and, therefore, requires fewer training samples

Generative and discriminative classifiers

- Example for the comparison of generative and discriminative models for two classes C^1 and C^2:





© Bishop, 2006

This part of $p(x|C^1)$ has very little influence on the result because $p(x|C^2)$ is very small.

The decision based on $p(C|x)$ considers only the threshold or the separating plane in case of higher dimension.

Discriminant function

- Often, one is less interested in a model of the probability density than in a sub-division of the feature space into regions which are assigned to the individual classes.
- Discriminant function: A function $g_i(\mathbf{x})$ that assigns $\mathbf{x}$ to the class C^i if $g_i(\mathbf{x}) > g_j(\mathbf{x})$ for all $i \neq j$.
- Examples:
  - $p(C^i|\mathbf{x})$
  - $p(\mathbf{x}|C^i) \cdot p(C^i)$
  - $\ln p(\mathbf{x}|C^i) + \ln p(C^i)$
- The discriminant function sub-divides the feature space into regions R_i, which are assigned to the class C^i.
- The boundaries of these regions are given by $g_i(\mathbf{x}) = g_j(\mathbf{x})$.

Discriminative methods: Overview

- Probabilistic discriminative classifiers:
  Discriminant function is based on $p(C_i|\mathbf{x})$
  - Logistic Regression: binary classification
  - Generalized linear models
- Non-probabilistic discriminative classifiers:
  The discriminant function cannot be interpreted as a probability.
  - Decision trees
  - Random forests
  - Boosting
  - Support vector machines
  - Artificial neural networks

Contents

Discriminant function: linear or non-linear function

- The simplest and most common model is a linear function of the input feature vectors $\mathbf{x}$ of dimension D:

$$C(\mathbf{w},\mathbf{x}) = w_0 + w_1 x_1 + \ldots + w_D x_D = w_0 + \sum_{i=1}^{D} w_i x_i = w_0 + \mathbf{w}^T \cdot \mathbf{x}$$

- However, by some modifications we can induce **non-linear** elements too (we will see later why this might be beneficial):
  - $M - 1$ non-linear basis functions Φ_i to transform input data $\mathbf{x}$:

$$C(\mathbf{w},\mathbf{x},\Phi) = w_0 + \sum_{i=1}^{M-1} w_i \Phi_i(\mathbf{x}) = w_0 + \mathbf{w}^T \cdot \mathbf{\Phi}(\mathbf{x})$$

  - And/or non-linear transform f of outputs:

$$C(\mathbf{w},\mathbf{x},\Phi,f) = f\left(w_0 + \sum_{i=1}^{M-1} w_i \Phi_i(\mathbf{x})\right) = f\left(w_0 + \mathbf{w}^T \cdot \mathbf{\Phi}(\mathbf{x})\right)$$

Contents

- Generative vs. discriminative classifiers
- Linear Discriminant function
- **Logistic Regression**
- Generalized Linear Models
- Training
- Multi-class Problems

Logistic sigmoid function

- Distinction of two classes C^1, C^2 (object, background)
- Theorem of Bayes:

$$p\left(C^1 \middle| \mathbf{x}\right) = \frac{p\left(\mathbf{x} \middle| C^1\right) \cdot p\left(C^1\right)}{p\left(\mathbf{x} \middle| C^1\right) \cdot p\left(C^1\right) + p\left(\mathbf{x} \middle| C^2\right) \cdot p\left(C^2\right)}$$

$$= \frac{1}{1 + \frac{p\left(\mathbf{x} \middle| C^2\right) \cdot p\left(C^2\right)}{p\left(\mathbf{x} \middle| C^1\right) \cdot p\left(C^1\right)}} = \frac{1}{1+e^{-a}} := \sigma(a)$$

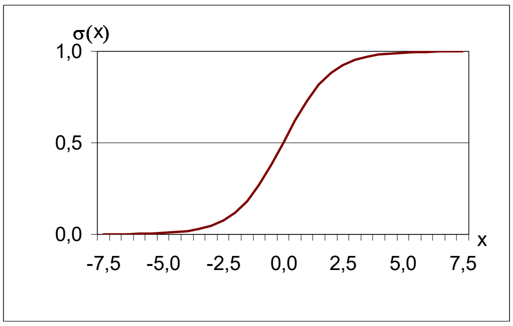
with $a(\mathbf{x}) = \ln \frac{p\left(\mathbf{x} \middle| C^1\right) \cdot p\left(C^1\right)}{p\left(\mathbf{x} \middle| C^2\right) \cdot p\left(C^2\right)} = \ln \frac{p\left(C^1 \middle| \mathbf{x}\right)}{p\left(C^2 \middle| \mathbf{x}\right)}$

Logistic sigmoid function $\quad \sigma(a) = \frac{1}{1+e^{-a}}$

Logistic sigmoid function I

- Originally, this is a generative model, because it is based on the theorem of Bayes.
- $a(\mathbf{x})$ is the negative logarithm of the ratio of the posterior probabilities.
- From now on: Consideration of $a(\mathbf{x})$ without Bayesian interpretation.
- Simple models for $a(\mathbf{x})$: linear or quadratic functions
- Logistic sigmoid ("S-shaped") function:

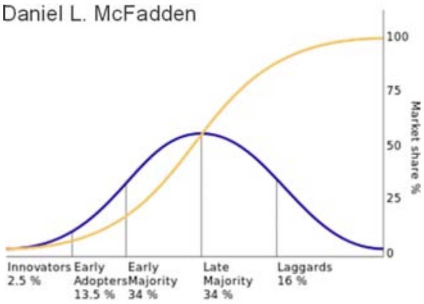
$$\sigma(a) = \frac{1}{1+e^{-a}}$$



Logistic sigmoid function II

- The sigmoid function is useful to model various kinds of processes
  - Growth of population, for example, bacteria:
    - Initially only few individuals, then exponential rise, later slow down and eventually convergence to carrying capacity (the latter defined as the environment's maximal load).
  - Economic process
    - Models diffusion of an innovation through its life cycle (railway, electric power, light bulbs, cars, air travel, transistor, PC, internet, GPS…)
    - Nobel Memorial Prize in Economic Sciences 2000 for Daniel L. McFadden
  - Pattern recognition
    - Neural networks: non-linear *activation function* of an "artificial neuron"

$$\sigma(a) = \frac{1}{1+e^{-a}}$$



https://en.wikipedia.org/wiki/Diffusion_of_innovations

Logistic regression

- (Unrealistic) assumption: The features of $\mathbf{x}$ are normally distributed with mean values $\boldsymbol{\mu}_1$ and $\boldsymbol{\mu}_2$ and identical covariance matrices $\boldsymbol{\Sigma}_1 = \boldsymbol{\Sigma}_2 = \boldsymbol{\Sigma}$.

$$a(\mathbf{x}) = \ln \frac{p\left(\mathbf{x} \middle| C^1\right) \cdot p\left(C^1\right)}{p\left(\mathbf{x} \middle| C^2\right) \cdot p\left(C^2\right)} =$$

$$= -\tfrac{1}{2}\left(\mathbf{x}-\boldsymbol{\mu}_1\right)^T \cdot \boldsymbol{\Sigma}^{-1} \cdot \left(\mathbf{x}-\boldsymbol{\mu}_1\right) + \tfrac{1}{2}\left(\mathbf{x}-\boldsymbol{\mu}_2\right)^T \cdot \boldsymbol{\Sigma}^{-1} \cdot \left(\mathbf{x}-\boldsymbol{\mu}_2\right) + \ln p\left(C^1\right) - \ln p\left(C^2\right) =$$

$$= \underbrace{\left(\boldsymbol{\mu}_1-\boldsymbol{\mu}_2\right)^T \cdot \boldsymbol{\Sigma}^{-1} \cdot \mathbf{x}}_{\text{wT*x}} \underbrace{- \tfrac{1}{2}\boldsymbol{\mu}_1^T \cdot \boldsymbol{\Sigma}^{-1} \cdot \boldsymbol{\mu}_1 + \tfrac{1}{2}\boldsymbol{\mu}_2^T \cdot \boldsymbol{\Sigma}^{-1} \cdot \boldsymbol{\mu}_2 + \ln p\left(C^1\right) - \ln p\left(C^2\right)}_{\text{w0}} =$$

- Thus $a(\mathbf{x})$) is a linear function of the features!

$$p\left(C^1 \middle| \mathbf{x}\right) = \frac{1}{1+e^{-\left(\mathbf{w}^T \cdot \mathbf{x} + w_0\right)}} = \sigma\left(a\left(\mathbf{x}\right)\right) = \sigma\left(\mathbf{w}^T \cdot \mathbf{x} + w_0\right)$$

Logistic regression: Parameters

- In the binary case, we have

$$p\left(C^2 \middle| \mathbf{x}\right) = 1 - p\left(C^1 \middle| \mathbf{x}\right) \quad \text{und} \quad 1-\sigma\left(a\right) = \sigma\left(-a\right)$$

$$p\left(C^1 \middle| \mathbf{x}\right) = \frac{1}{1+e^{-\left(\mathbf{w}^T \cdot \mathbf{x} + w_0\right)}} \quad \text{und} \quad p\left(C^2 \middle| \mathbf{x}\right) = \frac{1}{1+e^{\left(\mathbf{w}^T \cdot \mathbf{x} + w_0\right)}}$$

- Parameters to be learned:
  - *Generative view*: $\boldsymbol{\Sigma},\ \boldsymbol{\mu}_1, \boldsymbol{\mu}_2, p(C^1), p(C^2)$

    → With D features: $D \cdot (D+1)/2 + 2 \cdot D + 2$ parameters

    → The number of parameters grows quadratically

  - *Discriminative view*: $\mathbf{w}, w_0$

    → With D features: $D + 1$ parameters

    → The number of parameters grows linearly

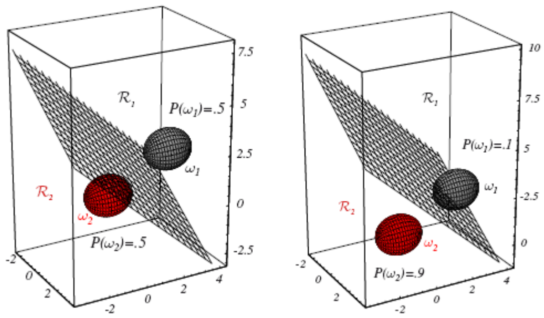
Logistic regression: Decision boundary

- Class boundary in feature space: $p(C^1 \mid \mathbf{x}) = p(C^2 \mid \mathbf{x})$

$$\rightarrow \frac{1}{1+e^{-(\mathbf{w}^T\cdot\mathbf{x}+w_0)}} = \frac{1}{1+e^{(\mathbf{w}^T\cdot\mathbf{x}+w_0)}}$$

$$\rightarrow -(\mathbf{w}^T\cdot\mathbf{x}+w_0) = \mathbf{w}^T\cdot\mathbf{x}+w_0$$

$$\rightarrow \mathbf{w}^T\cdot\mathbf{x}+w_0 = 0$$



© Duda, Hart, Stork, 2001

- The decision boundary between the classes is a hyperplane.

- Linear discriminative function → Logistic Regression

Logistic regression: Decision boundary

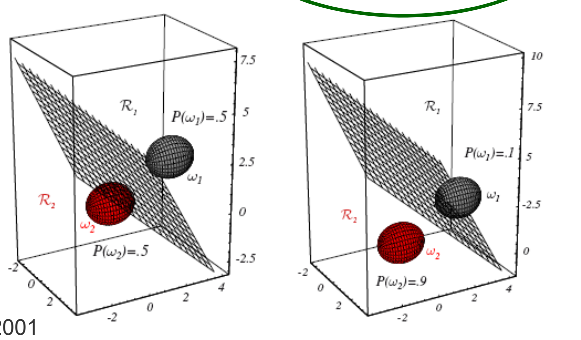
- Class boundary in feature space: $\mathbf{w}^T\cdot\mathbf{x}+w_0 = 0$

- Normal vector **w**: $\mathbf{w} = (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^T\cdot\boldsymbol{\Sigma}^{-1}$

  depends on the vector between the class centers, direction is also influenced by Σ.

- Offset w_0: $w_0 = \tfrac{1}{2}\boldsymbol{\mu}_1^T\cdot\boldsymbol{\Sigma}^{-1}\cdot\boldsymbol{\mu}_1 + \tfrac{1}{2}\boldsymbol{\mu}_2^T\cdot\boldsymbol{\Sigma}^{-1}\cdot\boldsymbol{\mu}_2 + \ln p(C^1) - \ln p(C^2)$

- Changes to the prior lead to a parallel shift of the decision boundary.



© Duda, Hart, Stork, 2001

Logistic regression: Geometrical interpretation

- Decision boundary in feature space: $\varepsilon : \mathbf{w}^T \cdot \mathbf{x} + w_0 = 0$

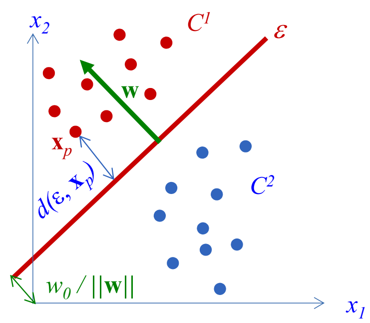
- For a point $\mathbf{x}_p$ that does not lie on the separating surface:

$$\mathbf{w}^T \cdot \mathbf{x}_p + w_0 = \|\mathbf{w}\| \cdot d\left(\varepsilon, \mathbf{x}_p\right)$$

→Interpretation of

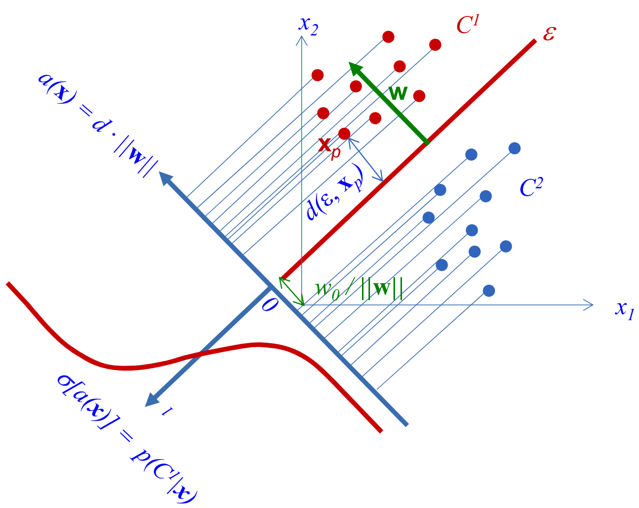
$$p\left(C^1 \mid \mathbf{x}\right) = \frac{1}{1 + e^{-\left(\mathbf{w}^T \cdot \mathbf{x} + w_0\right)}}$$

as a sigmoid function applied to the (scaled) distance from the separating surface that maps this distance into the interval *[0,1]*!



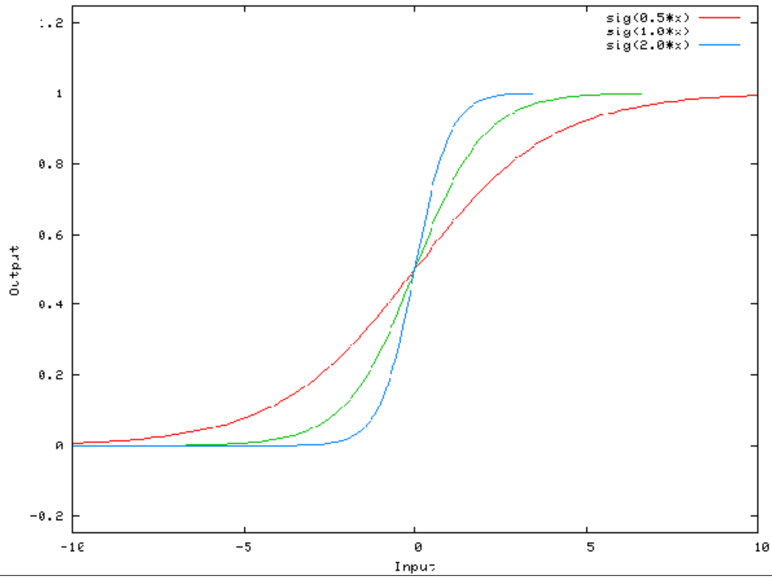
Logistic regression: Geometrical interpretation

- Interpretation of || w ||: The larger || w ||, the steeper the sigmoid function.



Logistic regression: Geometrical interpretation

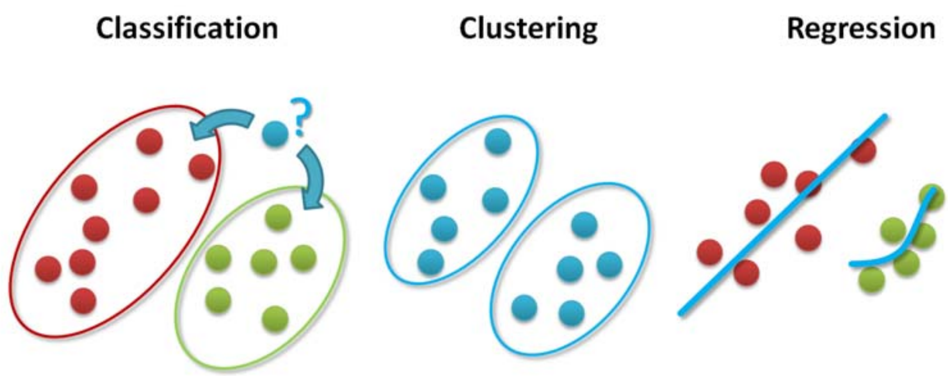
- Interpretation of || w ||: The larger || w ||, the steeper the sigmoid function.



Universität Stuttgart ifp

Notion: “Logistic Regression“ I

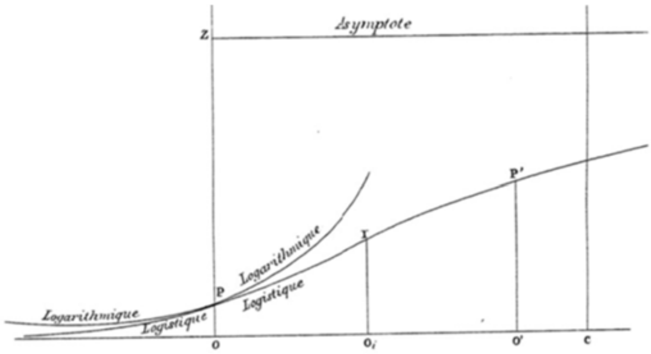
- „Regression“: The name refers to the initial purpose namely statistical regression. However, it is rather a classification method since we search for an optimal linear separating surface in feature space.
- Different aims of classification, (unsupervised) clustering and regression:



Universität Stuttgart ifp

Notion: "Logistic Regression" II

- „Logistic": The notion was coined about 1835 from Belgium mathematician Adolphe Quetelet, who desired to discriminate the initiale phase of population growth of Belgium from a "logarithmic" (today: exponential) curve:



- The principle that the sigmoid function is applied to a scaled distance to get a probability is often used in other contexts.

- What happens with data that are not linearly separable?

Contents

- Generative vs. discriminative classifiers
- Linear Discriminant function
- Logistic Regression
- Generalized Linear Models
- Training
- Multi-class Problems

Generative Model: Normal distribution with different covariance matrices

- If the covariance matrices are not identical, the quadratic term in the exponent does not disappear:

$$p\left(C^1 \mid \mathbf{x}\right) = \frac{1}{1 + e^{-\left(\mathbf{x}^T \mathbf{W} \cdot \mathbf{x} + \mathbf{w}^T \cdot \mathbf{x} + w_0\right)}}$$

with

$$\mathbf{W} = 1/2 \cdot \left(\Sigma_2^{-1} - \Sigma_1^{-1}\right)$$

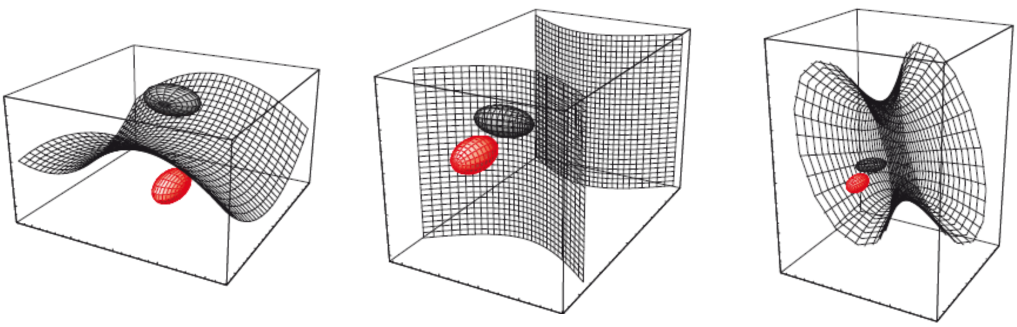
$$\mathbf{w} = \Sigma_1^{-1} \cdot \boldsymbol{\mu}_1 - \Sigma_2^{-1} \cdot \boldsymbol{\mu}_2$$

$$w_0 = 1/2 \cdot \boldsymbol{\mu}_1^T \cdot \Sigma_1^{-1} \cdot \boldsymbol{\mu}_1 + 1/2 \cdot \boldsymbol{\mu}_2^T \cdot \Sigma_2^{-1} \cdot \boldsymbol{\mu}_2 + 1/2 \cdot \ln\left\|\Sigma_2^{-1}\right\| - 1/2 \cdot \ln\left\|\Sigma_1^{-1}\right\| + \ln p\left(C^1\right) - \ln p\left(C^2\right)$$

- In general, the class boundary is not a hyperplane but a hyperquadric.

Generative Model: Normal distribution with different covariance matrices

Examples for decision boundaries (3D feature vectors)



© Duda, Hart, Stork, 2001

Feature space transformations and generalized linear models

- General formula: $$p\left(C^1 \mid \mathbf{x}\right) = \sigma\left(a\left(\mathbf{x}\right)\right) = \frac{1}{1+e^{-a(\mathbf{x})}}$$
- For identical covariance matrices, $a(\mathbf{x})$ was a linear function of the features $\mathbf{x}$:

$$a\left(\mathbf{x}\right) = \mathbf{w}^T \cdot \mathbf{x} + w_0$$

- With increasing complexity of the models for the probability densities, the complexity of $a(\mathbf{x})$ is increased.
  - For example, a quadratic form for normal distributions
- In order still to be able to work with linear functions, one can move on to another feature space:
  - Transformation of the feature space (Feature space mapping)
  - Generalized linear models

Feature space transformations and generalized linear models

- Feature space mapping $$\Phi\left(\mathbf{x}\right) = \left[\Phi_1\left(\mathbf{x}\right), \Phi_2\left(\mathbf{x}\right), \ldots, \Phi_N\left(\mathbf{x}\right)\right]^T$$
  - $\Phi_i(\mathbf{x})$: (in principle) arbitrary functions: frequently, polynomials
  - N: Dimension of the transformed feature vector (usually greater than the dimension of $\mathbf{x}$)
  - Frequent choice $\Phi_1(\mathbf{x}) = 1$
  - Example for 2D feature space, i.e. $\mathbf{x} = \left[x_1, x_2\right]^T$

$$\Phi\left(\mathbf{x}\right) = \left[1, x_1, x_2, x_1 \cdot x_2, x_1^2, x_2^2\right]^T$$

  - Instead of using a complex model for $a(\mathbf{x})$:
    Transition into a higher dimensional feature space in which $a(\Phi(\mathbf{x}))$ is linear.
    $\Rightarrow$ Generalized linear models

Feature space transformations and generalized linear models

- Generalized Linear Models: $p\left(C^1 \mid \mathbf{x}\right) = \sigma\left(a\left(\mathbf{x}\right)\right) = \dfrac{1}{1+e^{-a(\mathbf{x})}}$

  with $a\left(\mathbf{x}\right) = \mathbf{w}^T \cdot \Phi\left(\mathbf{x}\right)$

  and $\Phi\left(\mathbf{x}\right) = \left[\Phi_1\left(\mathbf{x}\right), \Phi_2\left(\mathbf{x}\right), \ldots, \Phi_N\left(\mathbf{x}\right)\right]^T$.

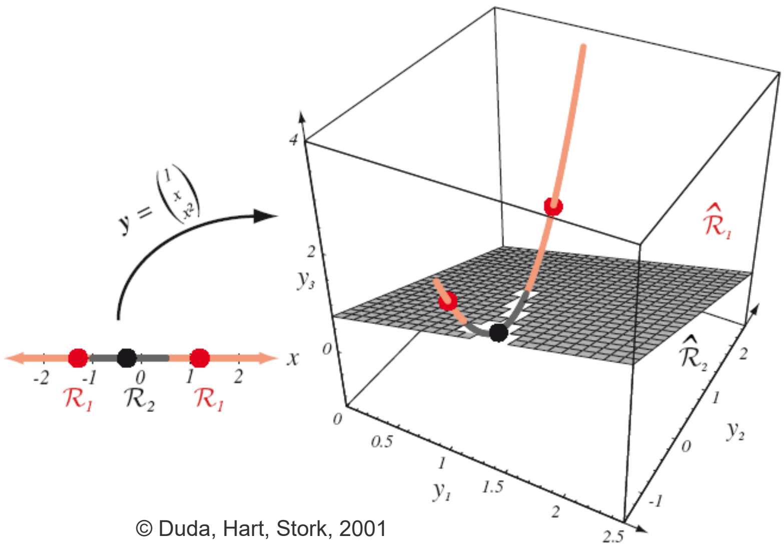
- Note: Due to $\Phi_1(\mathbf{x}) = 1$, w_0 becomes the first component of $\mathbf{w}$.

- The example of $\Phi(\mathbf{x}) = (1, x_1, x_2, x_1 \cdot x_2, x_1^2, x_2^2)^T$ leads to a quadratic form for $a(\mathbf{x})$ similar to the normal distribution!

- Assumptions about the distribution of the features are dropped in favor of a choice of a feature space mapping.

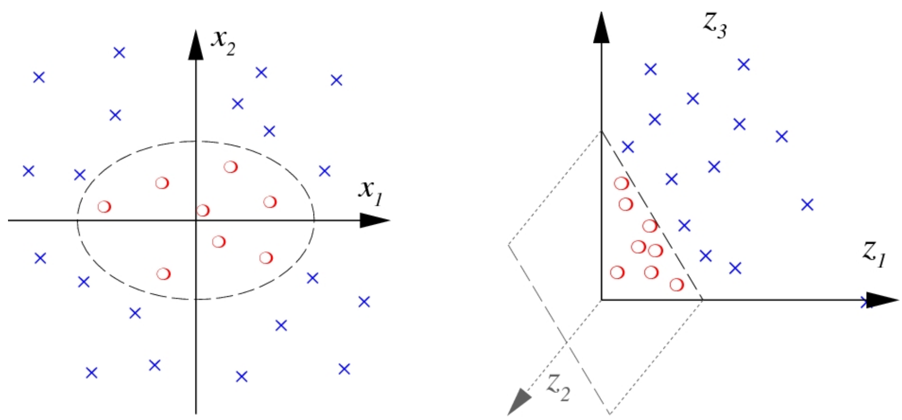
Examples of feature space mappings I

- In higher dimensional feature space the data can be better separated.



Examples of feature space mappings II

- In higher dimensional feature space the data can be better separated.



Examples of feature space mappings III

- Examples of feature space transformations with $\mathbf{x} = [x_1, x_2]^T$

  - Quadratic expansion: $\Phi(\mathbf{x}) = \left[1, x_1, x_2, x_1 \cdot x_2, x_1^2, x_2^2\right]^T$

  - Cubic expansion : $\Phi(\mathbf{x}) = \left[1, x_1, x_2, x_1 \cdot x_2, x_1^2, x_2^2, x_1^2 \cdot x_2, x_1 \cdot x_2^2, x_1^3, x_2^3\right]^T$

  - Kernel function ("Kernel logistic regression"):

$$\Phi(\mathbf{x}) = \left[k(\mathbf{x}, \mathbf{x}_1), k(\mathbf{x}, \mathbf{x}_2), \ldots, k(\mathbf{x}, \mathbf{x}_N)\right]^T$$

with kernel function

$$k(\mathbf{x}_n, \mathbf{x}_m) = e^{-\frac{\|\mathbf{x}_n - \mathbf{x}_m\|^2}{2 \cdot \sigma^2}}$$

as a measure for the "distance" of a point from the training points $\mathbf{x}_n$

→ very flexible
→ needs hyper-parameter σ.

Feature space mapping

- Using a feature space mapping, linear models can also be applied to problems where the classes are not linearly separable.

- **Disadvantage**: Increase of the number N of parameters:

  - Polynomial expansion: with D features (incl. $\Phi_1(\mathbf{x}) = 1$), order G:

$$N = \binom{D+G-1}{G} = \frac{(D+G-1)!}{(D-1)! \cdot G!}$$

    - $G = 2 \rightarrow N = D \cdot (D+1) / 2$
    - $G = 3 \rightarrow N = D \cdot (D+1) \cdot (D+2) / 6$

  - Kernel function: N is equal to the number of training points (we center a kernel at each training point and need to determine a weight for each).

- Could be problematic for feature spaces with $D > 10$.

Contents

- Generative vs. discriminative classifiers
- Linear Discriminant function
- Logistic Regression
- Generalized Linear Models
- Training
- Multi-class Problems

Logistic regression: Training

- Given:
  - Functional model of feature space mapping
  - N points $\mathbf{x}_i$ with known binary class indicator $t_i \in \{0,1\}$
  - t_i: **Indicator variable** that shows whether $\mathbf{x}_i$ belongs to C^1 ($t_i = 1$) or not ($t_i = 0$)
  - All the indicator variables t_i can be collected in a vector $\mathbf{t}$.
- Wanted:
  - Parameter vector $\mathbf{w}$ of the generalized linear model

$$p\left(C^1 \mid \mathbf{x}\right) = \frac{1}{1+e^{-\left[\mathbf{w}^T \cdot \boldsymbol{\Phi}(\mathbf{x})\right]}}$$

Logistic regression: Maximum likelihood training I

- Determine $\mathbf{w}$ such that $p\left(\mathbf{t} \mid \mathbf{w}, \mathbf{x}_1, \ldots, \mathbf{x}_N\right) \to \max$ with

$$y_n = p\left(C^1 \mid \mathbf{x}_n\right) = \frac{1}{1+e^{-\left[\mathbf{w}^T \cdot \boldsymbol{\Phi}(\mathbf{x}_n)\right]}} \quad und \quad p\left(C^2 \mid \mathbf{x}_n\right) = 1 - y_n$$

- Result: $$p\left(\mathbf{t} \mid \mathbf{w}, \mathbf{x}_1, \ldots, \mathbf{x}_N\right) = \prod_{n=1}^{N} y_n^{t_n} \cdot \left(1-y_n\right)^{(1-t_n)}$$

because $p(C^1|\mathbf{x}_n) > p(C^2|\mathbf{x}_n) \rightarrow y_n > (1 - y_n)$ for $t_n = 1$ (i.e., y_n contributes) and
$p(C^2|\mathbf{x}_n) > p(C^1|\mathbf{x}_n) \rightarrow (1 - y_n) > y_n$ for $t_n = 0$ (i.e., $(1 - y_n)$ contributes)

- Instead of the maximization of $p(\mathbf{t} \mid \mathbf{w}, \mathbf{x}_1, \ldots \mathbf{x}_N)$:
  - Minimization of the negative log-likelihood:

$$E\left(\mathbf{w}\right) = -\ln p\left(\mathbf{t} \mid \mathbf{w}, \mathbf{x}_1, \ldots, \mathbf{x}_N\right) \to \min$$

Logistic regression: Maximum likelihood training II

- Negative log-likelihood $E(\mathbf{w})$:

$$E(\mathbf{w}) = -\sum_{n=1}^{N} \left[t_n \cdot \ln(y_n) + (1-t_n) \cdot \ln(1-y_n) \right] \rightarrow \min$$

- As y_n depends on $\mathbf{w}$, $E(\mathbf{w})$ is a non-linear function of $\mathbf{w}$.
- Therefore, the minimum of $E(\mathbf{w})$ can only be determined iteratively.
- Initial values w^0: e.g. random numbers
- $E(\mathbf{w})$ is concave and has a single minimum
- Determination of the minimum: **gradient** $\nabla E(\mathbf{w}) = 0$

Logistic regression: Maximum likelihood training III

- Newton-Raphson method: using the initial values $\mathbf{w}^{t-1}$:

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

- Gradient $\nabla E(\mathbf{w})$: $\quad \nabla E(\mathbf{w}) = \sum_{n=1}^{N} (y_n - t_n) \cdot \mathbf{\Phi}(\mathbf{x}_n)$

  - Interpretation: $(y_n - t_n)$ can be considered as classification error for the training point:

    → If $t_n = 1$ → $C = C^1$ → $y_n = p(C^1 \mid \mathbf{x}_n)$ should be close to 1

    → If $t_n = 0$ → $C = C^2$ → y_n should be close to 0

  - $\nabla E(\mathbf{w})$: Sum of the feature vectors weighted by $(y_n - t_n)$

- Hesse-Matrix $\quad \mathbf{H} = \nabla\nabla E(\mathbf{w}) = \sum_{n=1}^{N} y_n \cdot (1-y_n) \cdot \mathbf{\Phi}(\mathbf{x}_n) \cdot \mathbf{\Phi}(\mathbf{x}_n)^T$

Maximum likelihood training: Discussion

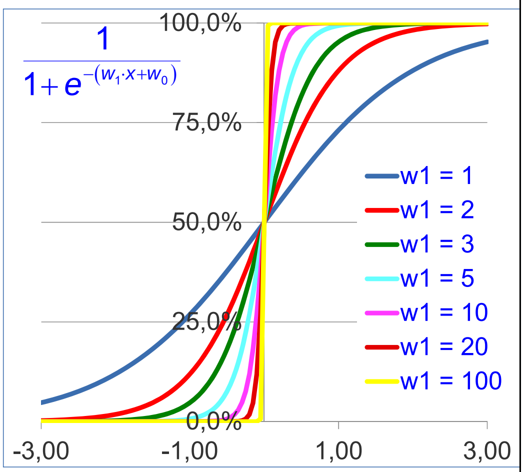
- Hesse-Matrix is positive definite → inverse exists

- In order to avoid numerical problems:
  - Scaling of the features:
    - Determination of mean value μ and standard deviation σ of all features from the training data.
    - Shift by μ, scaling with 1 / σ
      → Features all have the same range of values
    - The same scaling has to be applied for training and classification!

- ML has the tendency to **overfit** the classifier to the training data:
  - For example, $|\mathbf{w}|$ might become very large → sigmoid approximates step function!
  - ML provides no means to enforce certain desired behavior (i.e., apply knowledge!)
    → Baysian method, which here **is equivalent to regularization** with prior for $\mathbf{w}$.

Logistic regression: Training with regularization I

- MAP: Maximization of $p(\mathbf{w} \mid \mathbf{t}, \mathbf{x}_1, \ldots, \mathbf{x}_N) = p(\mathbf{t} \mid \mathbf{w}, \mathbf{x}_1, \ldots, \mathbf{x}_N) \cdot p(\mathbf{w})$
- $p(\mathbf{t} \mid \mathbf{w}, \mathbf{x}_1, \ldots, \mathbf{x}_N)$ corresponds to the Likelihood (as with ML)
- Prior $p(\mathbf{w})$:
  - Sigmoid slope depends on the size of the numerical values of the coefficients w_i in $\mathbf{w}$:
    - The larger $|w_i|$, the steeper the sigmoid function.
    - For w_i → ∞ the sigmoid function becomes a step function.



Logistic regression: Training with regularization II

$$p(\mathbf{w} \mid \mathbf{t}, \mathbf{x}_1, \ldots, \mathbf{x}_N) = p(\mathbf{t} \mid \mathbf{w}, \mathbf{x}_1, \ldots, \mathbf{x}_N) \cdot p(\mathbf{w}) \rightarrow \max$$

- To keep the numerical values of $\mathbf{w}$ small:
- Prior $p(\mathbf{w})$: Normal distribution with expectation value $\mathbf{0}$ and covariance matrix $\sigma^2 \cdot \mathbf{I}$
- Corresponds to regularization in adjustment theory.
- Requires hyper-parameter σ, which is either fixed by the user or determined via a procedure such as cross-validation.
- Negative logarithm (excluding constant terms):

$$E(\mathbf{w}) = -\sum_{n=1}^{N} \left[t_n \cdot \ln(y_n) + (1-t_n) \cdot \ln(1-y_n) \right] + \frac{\mathbf{w}^T \cdot \mathbf{w}}{2 \cdot \sigma^2} \rightarrow \min$$

Logistic regression: Training with regularization III

- Minimization of

$$E(\mathbf{w}) = -\sum_{n=1}^{N} \left[t_n \cdot \ln(y_n) + (1-t_n) \cdot \ln(1-y_n) \right] + \frac{\mathbf{w}^T \cdot \mathbf{w}}{2 \cdot \sigma^2} \rightarrow \min$$

leads to the numerical values of $\mathbf{w}$ that are as small as possible.

- The gradient has to be extended compared to the ML method :

$$\nabla E(\mathbf{w}) = \sum_{n=1}^{N} (y_n - t_n) \cdot \mathbf{\Phi}(\mathbf{x}_n) + \frac{1}{\sigma^2} \cdot \mathbf{w}$$

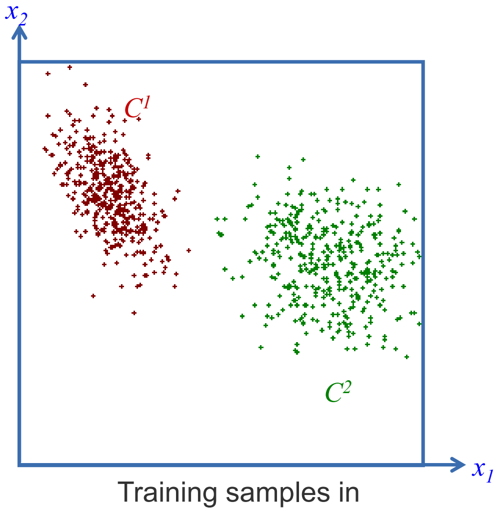
- This is also true for the Hesse Matrix:

$$\mathbf{H} = \nabla\nabla E(\mathbf{w}) = \sum_{n=1}^{N} \left[y_n \cdot (1-y_n) \cdot \mathbf{\Phi}(\mathbf{x}_n) \cdot \mathbf{\Phi}(\mathbf{x}_n)^T \right] + \frac{1}{\sigma^2} \cdot \mathbf{I}$$

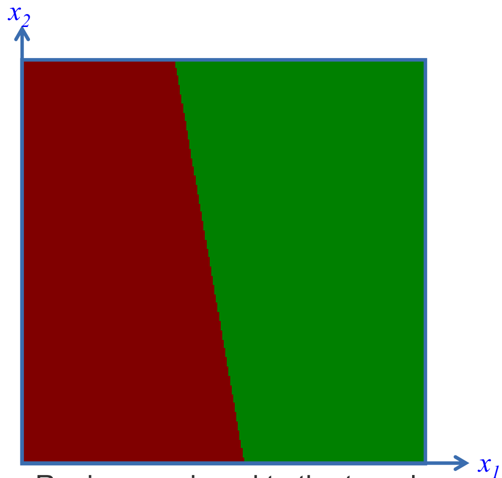
This means, in the main diagonal, the weights of the direct observations for $\mathbf{w}$ are added (as in the case of regularization in adjustment).

Logistic Regression: Example (ML-Training)

Two classes, two features: linearly separable case



Training samples in feature space (800)



Regions assigned to the two classes in feature space

Logistic regression: Example

Two classes, two features: linearly separable case
posterior probabilities

$p(C^1|x_1,x_2)$

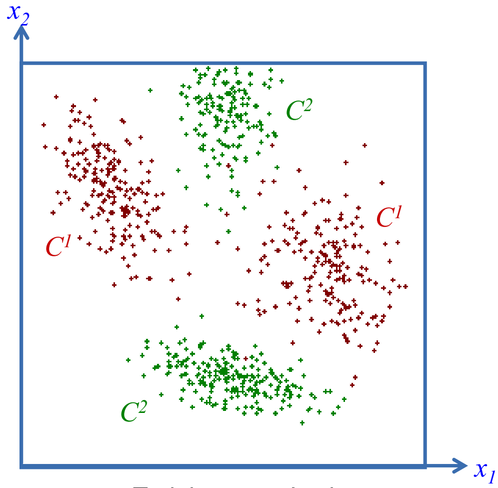
$p(C^2|x_1,x_2)$

white ... high probability, black ... low probability

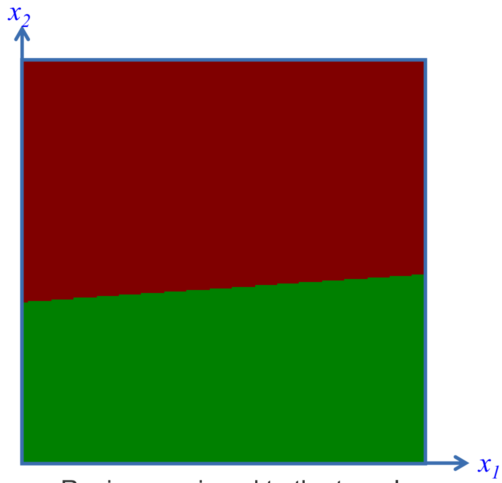
The sharp boundaries suggest a higher security of the classification in the border region than is actually achieved → Overfitting!

Logistic regression: Example

Two classes, two features: non linearly separable case
→ The classifier



Training samples in feature space (800)



Regions assigned to the two classes in feature space

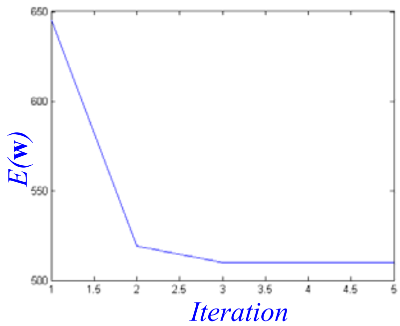
Logistic regression: Example

Two classes, two features: non-linearly separable case

$p(C^1|x_1,x_2)$

white ... high probability,
black ... low probability

$p(C^2|x_1,x_2)$

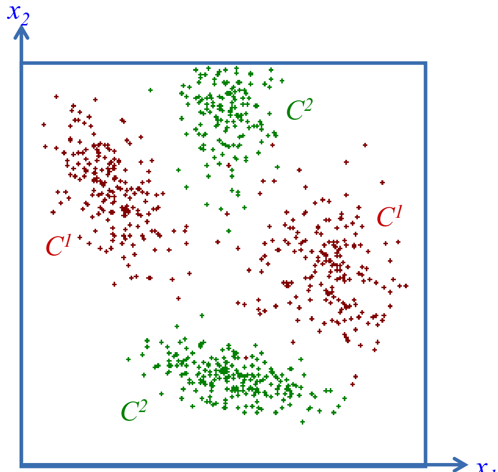


log-likelihood as a function of the iteration count in training

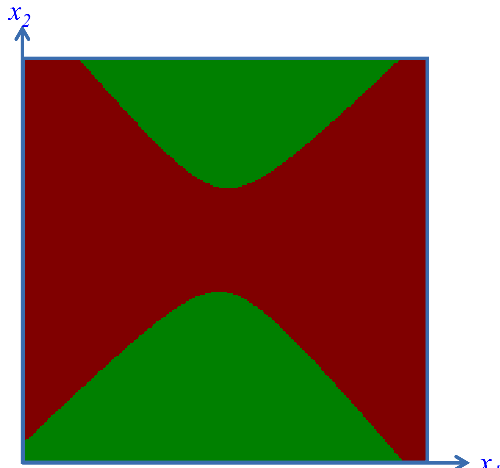
- Small differences in the posterior probabilities
- Relatively large value for $E(\mathbf{w})$

Logistic regression: Example

Two classes, two features: non-linearly separable case
Feature space transformation: the classes can be separated!



Training samples in feature space (800)



Regions assigned to the two classes in feature space

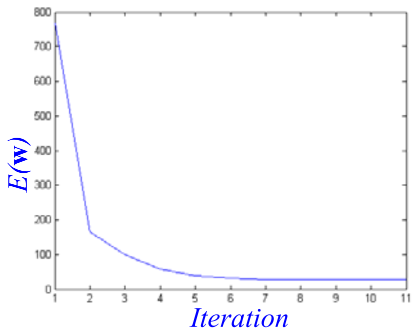
Logistic regression: Example

Two classes, two features: non-linear separated case
with feature space transform

$p(C^1|x_1,x_2)$

white ... high probability,
black ... low probability

$p(C^2|x_1,x_2)$



log-likelihood as a function of the iteration count in training

- Significant differences in the posterior probabilities
- Low value for $E(\mathbf{w})$ is reached

Contents

Transition to multi-class problems

- The posterior probability $p(C^k|\mathbf{x})$ for each class C^k can be modelled using the

$$p\left(C^k|\mathbf{x}\right) = \frac{e^{\left(a_k(\mathbf{x})\right)}}{\sum_j e^{\left(a_j(\mathbf{x})\right)}}$$

$$\text{with} \quad a_k(\mathbf{x}) = \ln p\left(\mathbf{x}|C^k\right) + \ln p\left(C^k\right)$$

- Assumptions about $p(\mathbf{x}|C^k)$ and $p(C^k)$ lead to models for $a_k(\mathbf{x})$
- Again, feature space mapping can help to obtain linear models:

$$a_k(\mathbf{x}) = a_k\left(\Phi(\mathbf{x})\right) = \mathbf{w}_k^T \cdot \Phi(\mathbf{x})$$

- In training, one parameter vector $\mathbf{w}_k$ per class has to be determined.

Multi-class logistic regression: Training

- Softmax function: $p\left(C^k \mid \mathbf{x}_n\right) = \dfrac{\exp\left[\mathbf{w}_k^T \cdot \mathbf{\Phi}\left(\mathbf{x}_n\right)\right]}{\sum_{j=1}^{M} \exp\left[\mathbf{w}_j^T \cdot \mathbf{\Phi}\left(\mathbf{x}_n\right)\right]} = y_{nk}$

- Training: Class label C_n is given for each training point $\mathbf{x}_n$

- Maximum Likelihood training is similar to the two-class case: the negative log-likelihood has to be minimized:

$$E\left(\mathbf{w}_1, \ldots \mathbf{w}_M\right) = -\sum_{n=1}^{N} \sum_{k=1}^{M} t_{nk} \cdot \ln\left(y_{nk}\right) \rightarrow \min$$

with the binary indicator variables $t_{nk} = \begin{cases} 1 & \text{if} \quad C_n = C^k \\ 0 & \text{otherwise} \end{cases}$

M… Number of classes

Multi-class logistic regression: Maximum likelihood training

- Again, the the Newton-Raphson can be applies: Using the current values from the previous iteration, the weights are updated according to $\mathbf{w}^{\tau-1}$ from the previous iteration, the weights are updated according to

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

- The parameter vectors are not independent

→ **One parameter vector must be declared to be constant**, e.g. $\mathbf{w}_1^T = (0, \ldots 0)^T$

- $\mathbf{w}_1$ is not changed in the optimization procedure
→ The parameter vector $\mathbf{w}$ to be determined if M classes are to be discerned becomes: $\mathbf{w} = (\mathbf{w}_2^T, \ldots, \mathbf{w}_M^T)^T$

Multi-class logistic regression: Maximum likelihood training

- Gradient of the negative log-likelihood

  (Derivative of E by the weight vector of the class j):

$$\nabla_{\mathbf{w}_j} E(\mathbf{w}_1, \ldots \mathbf{w}_M) = \sum_{n=1}^{N} (y_{nj} - t_{nj}) \cdot \mathbf{\Phi}(\mathbf{x}_n)$$

- Total gradient vector:

$$\nabla E(\mathbf{w}_1, \ldots \mathbf{w}_M) = \left[\nabla_{\mathbf{w}_2} E(\mathbf{w}_1, \ldots \mathbf{w}_M)^T, \ldots, \nabla_{\mathbf{w}_M} E(\mathbf{w}_1, \ldots \mathbf{w}_M)^T \right]^T$$

- Again, the gradient can be interpreted as the sum of the (transformed) feature vectors weighted by the "classification error" $(y_{nj} - t_{nj})$.

Multi-class logistic regression: Maximum likelihood training

- Hesse matrix $\boldsymbol{H}$ also consists of several components:

$$\mathbf{H} = \begin{pmatrix} \mathbf{H}_{22} & \mathbf{H}_{23} & \cdots & \mathbf{H}_{2M} \\ \mathbf{H}_{23}^T & \mathbf{H}_{33} & \cdots & \mathbf{H}_{3M} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{H}_{2M}^T & \mathbf{H}_{3M}^T & \cdots & \mathbf{H}_{MM} \end{pmatrix}$$

with

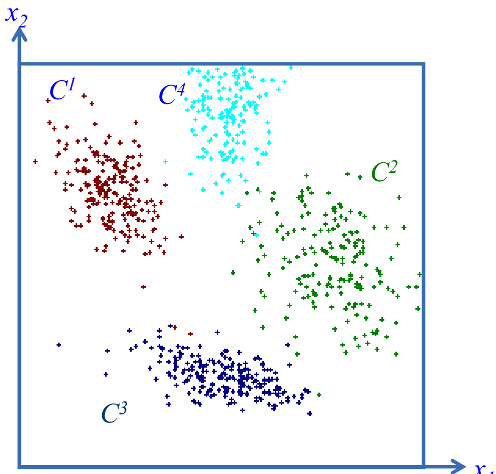
$$\mathbf{H}_{jk} = \nabla_{\mathbf{w}_j} \nabla_{\mathbf{w}_k} E(\mathbf{w}) = \sum_{n=1}^{N} y_{nk} \cdot (\mathbf{I}_{nk} - y_{nj}) \cdot \mathbf{\Phi}(\mathbf{x}_n) \cdot \mathbf{\Phi}(\mathbf{x}_n)^T$$

$\mathbf{I}_{nk}$ … Elements of a unit matrix

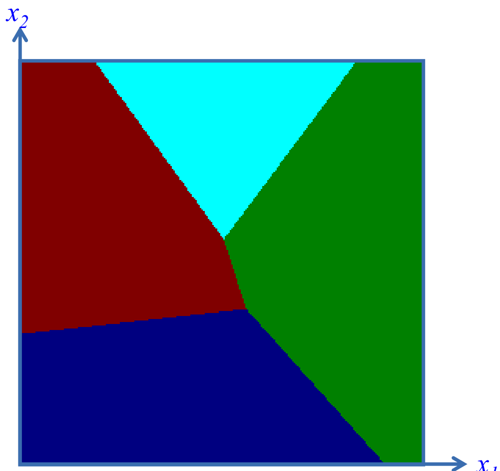
- Regularisation: As in the binary case (Gaussian prior with expectation **0** and Covariance $\sigma \cdot \mathbf{I}$

Multi-class logistic regression: Example (ML-Training)

Four classes, two features



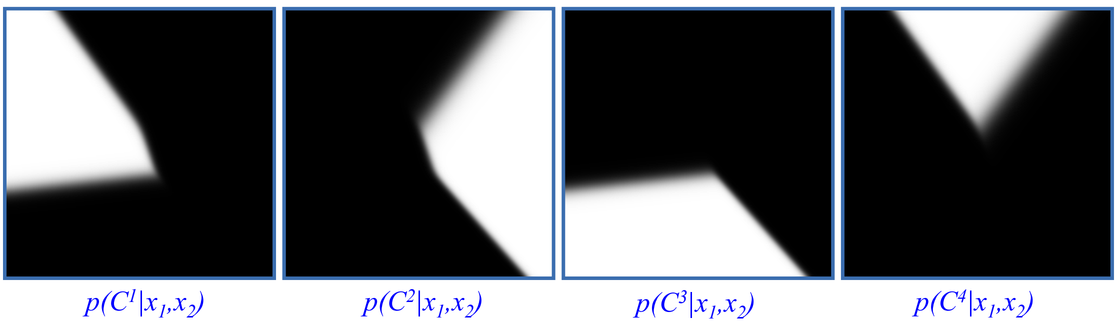
Training samples in feature space (800)



Regions assigned to the two classes in feature space

Multi-class logistic regression: Example (ML-Training)

Four classes, two features: posterior probabilities

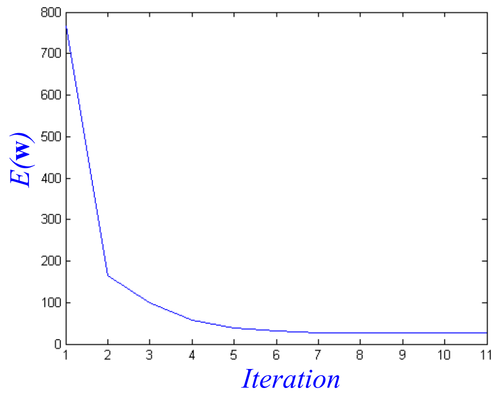


white ... high probability, black ... low probability

In the areas where the feature distributions overlap, the boundaries are slightly blurred

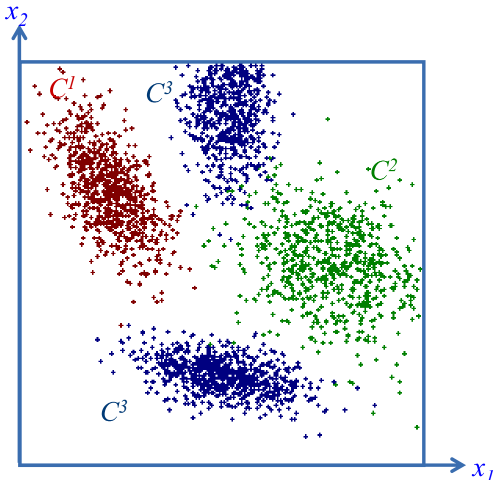
Multi-class logistic regression: Example (ML-Training)

Four classes, two features:
Development of the log-likelihood during training

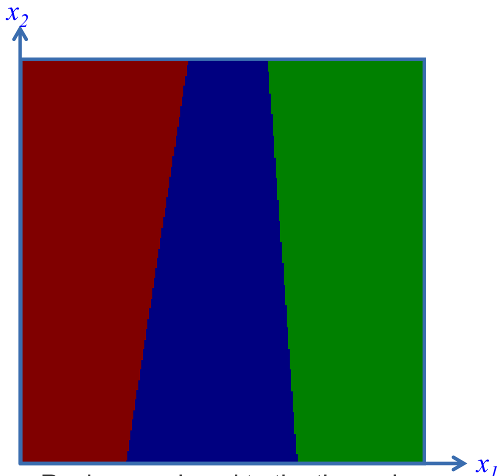


Multi-class case: Example (ML-Training)

Three classes, two features, not linearly separable
no feature space mapping



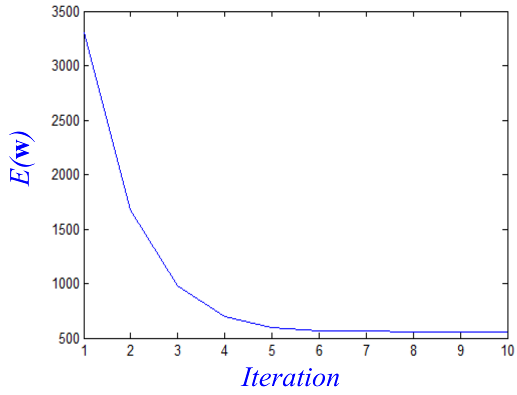
Training samples in feature space (800)



Regions assigned to the three classes in feature space

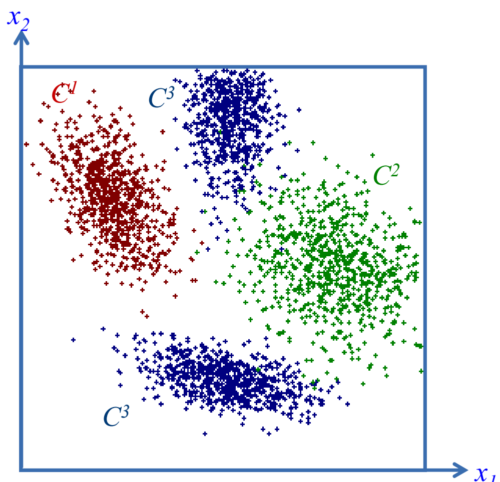
Multi-class case: Example (ML-Training)

Three classes, two features, not linearly separable, no feature space mapping: development of log-likelihood during training

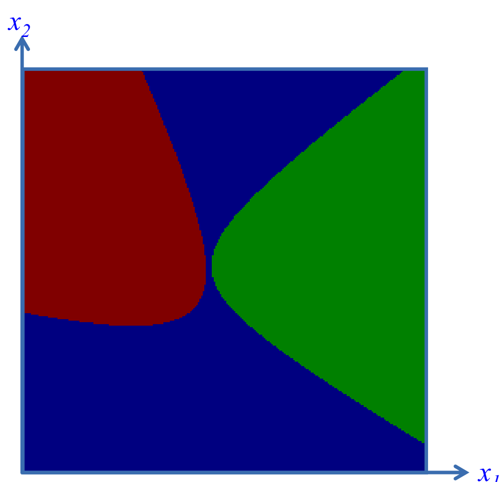


Multi-class case: Example (ML-Training)

Three classes, two features, not linearly separable – quadratic expansion



Training samples in feature space (800)



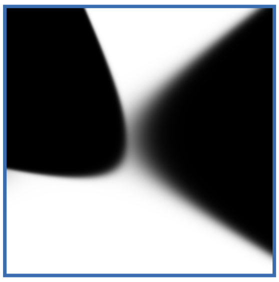
Regions assigned to the three classes in feature space

Multi-class case: Example (ML-Training)

Three classes, two features, not linearly separable –
quadratic expansion: posterior probabilities

$p(C^1|x_1,x_2)$

$p(C^2|x_1,x_2)$



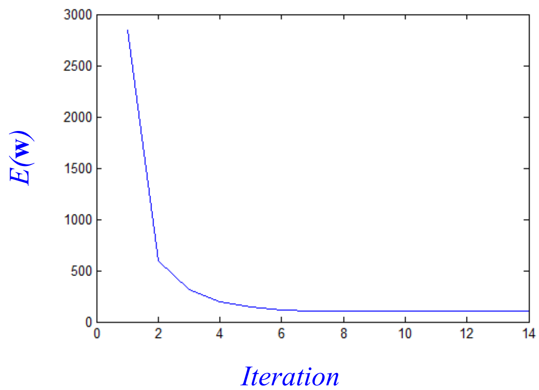
$p(C^3|x_1,x_2)$

white ... high probabilitiy, black ... low probability

- In the areas where the feature distributions overlap, the boundaries are slightly blurred.
- However, in general there is a very clear distinction → Overfitting

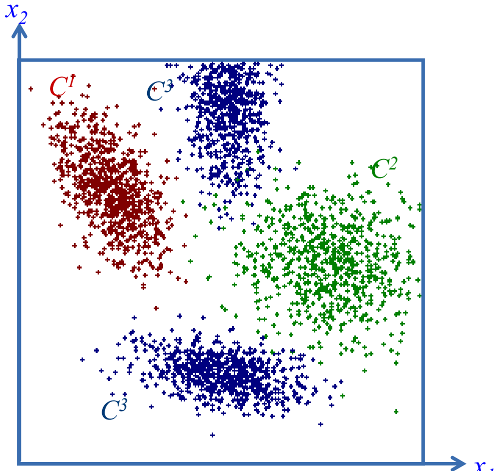
Multi-class case: Example (ML-Training)

Three classes, two features, not linearly separable –
quadratic expansion: development of log-likelihood during training

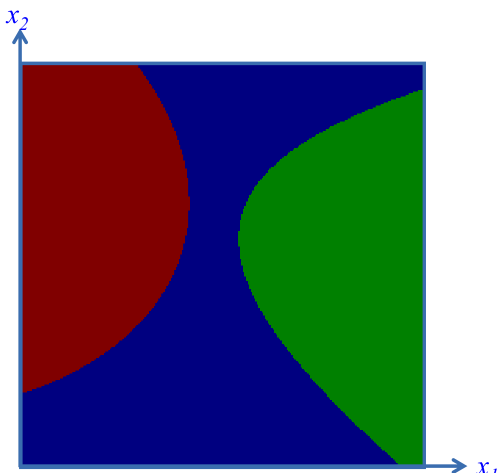


Multi-class case: Example (ML-Training)

Three classes, two features, not linearly separable – quadratic expansion, training with relatively strong regularization ($\sigma = 2$)



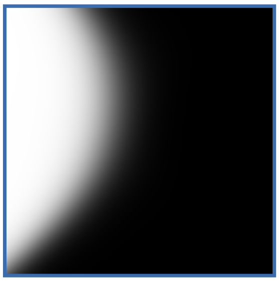
Training samples in feature space (800)



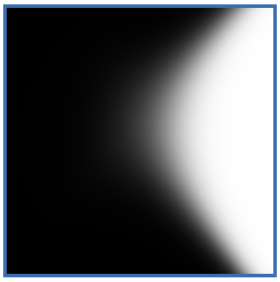
Regions assigned to the three classes in feature space

Multi-class case: Example (ML-Training)

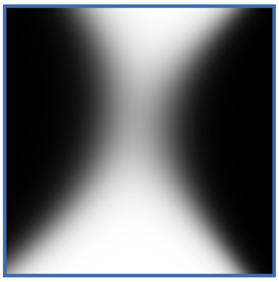
Three classes, two features, not linearly separable – quadratic expansion, training with regularization: posterior probabilities



$p(C^1|x_1,x_2)$



$p(C^2|x_1,x_2)$



$p(C^3|x_1,x_2)$

white ... high probability, black ... low probability

- Much smoother transitions, uncertainty of the classification is better represented
- Class boundaries may be regularized too strongly

Discussion

- Discriminative probabilistic methods directly model the posterior probability

  - No assumption about the distribution of data required
  - Basically, boundaries between classes are learned
  - Linear Models with / without feature space transformation

    - Fewer parameters to be determined
    - Fewer training data is required

  - Can be expanded to multi-class problems
  - Efficient learning / classification
  - Probabilistic output simplifies further processing.

Discussion

- Despite feature space transformation, the functional model can not fit properly to the distribution of the data

  → Transition to non-probabilistic methods

- High-dimensional feature vectors can lead to a large number of parameters to be learned.
- Numerical problems → scaling of the features in training and during the classification

- ML-Learning: Problem of overfitting → Regularisation

  → Requires prior for the parameter vector $\mathbf{w}$
    - Hyper-parameter σ (cross validation)

Literatur

- Bishop, C. : Pattern Recognition and Machine Learning. 1st edition, Springer, New York, USA, 2006.
- Duda, R. O., Hart, P. E., Stork, D. G.: Pattern Classification. 2nd edition, Wiley & Sons, New York, USA, 2001.
- Rottensteiner, Franz, 2014: Skript Bildanalyse II, IPI, Leibniz Universität Hannover