

# Pattern Recognition (PR)

Prof. Dr.-Ing. Andreas Maier  
Pattern Recognition Lab (CS 5), Friedrich-Alexander-Universität Erlangen-Nürnberg  
Winter Term 2020/21



This is a printable version of the slides of the lecture

**Pattern Recognition (PR)**  
*Winter term 2020/21*  
*Friedrich-Alexander University of Erlangen-Nuremberg.*

These slides are released under Creative Commons License Attribution CC BY 4.0.

Please feel free to reuse any of the figures and slides, as long as you keep a reference to the source of these slides at <https://lme.tf.fau.de/teaching/> acknowledging the authors Niemann, Hornegger, Hahn, Steidl, Nöth, Seitz, Rodriguez, Das and Maier.

Erlangen, January 8, 2021  
Prof. Dr.-Ing. Andreas Maier

# Logistic Regression II



## Parameterization

- Until now,  $F(\mathbf{x})$  was some arbitrary function in  $\mathbf{x}$   
**Example:**  $F(\mathbf{x}) = \mathbf{x}^T \mathbf{A} \mathbf{x} + \boldsymbol{\alpha}^T \mathbf{x} + \alpha_0$  with components defined by Gaussian distributions

- We can express a nonlinear  $F(\mathbf{x})$  as a scalar product by lifting  $\mathbf{x}$  into a higher dimensional space:

Given

$$\mathbf{x} = (x_1, x_2)^T \in \mathbb{R}^2, \mathbf{A} = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}, \boldsymbol{\alpha} = (\alpha_1, \alpha_2)^T, \alpha_0,$$

then

$$F(\mathbf{x}) = a_{11}x_1^2 + (a_{12} + a_{21})x_1x_2 + a_{22}x_2^2 + \alpha_1x_1 + \alpha_2x_2 + \alpha_0.$$

- Rewrite  $F(\mathbf{x}) = \boldsymbol{\theta}^T \mathbf{x}'$  with  $\boldsymbol{\theta}, \mathbf{x}' \in \mathbb{R}^6$ :  
 $\boldsymbol{\theta} = (a_{11}, a_{12} + a_{21}, a_{22}, \alpha_1, \alpha_2, \alpha_0)^T$   
 $\mathbf{x}' = (x_1^2, x_1x_2, x_2^2, x_1, x_2, 1)^T$

## Parameterization (cont.)

### Definition

We write the parameterized logistic function in the following:

$$g(\theta^T \mathbf{x}) = \frac{1}{1 + e^{-\theta^T \mathbf{x}}}$$

where  $\theta, \mathbf{x}$  are the lifted parameters of the original decision function  $F$  (if it was not already linear).

## Log-Likelihood Function

- Let us assume the posteriors are given by

$$\begin{aligned} p(y = 0 | \mathbf{x}) &= 1 - g(\theta^T \mathbf{x}) \\ p(y = 1 | \mathbf{x}) &= g(\theta^T \mathbf{x}) \end{aligned}$$

where  $g(\theta^T \mathbf{x})$  is the sigmoid function parameterized in  $\theta$ .

- The parameter vector  $\theta$  has to be estimated from a set  $S$  of  $m$  training samples:

$$S = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), (\mathbf{x}_3, y_3), \dots, (\mathbf{x}_m, y_m)\} \quad .$$

- Method of choice: Maximum Likelihood Estimation

## Log-Likelihood Function (cont.)

Before we work on the formulas of the ML-estimator, we rewrite the posteriors using Bernoulli probability:

$$p(y|\mathbf{x}) = g(\boldsymbol{\theta}^T \mathbf{x})^y (1 - g(\boldsymbol{\theta}^T \mathbf{x}))^{1-y}$$

which shows the great benefit of the chosen notation for class numbers.

## Log-Likelihood Function (cont.)

Now we can compute the log-likelihood function (assuming that the training samples are mutually independent):

$$\begin{aligned} \mathcal{L}(\boldsymbol{\theta}) &= \log \left( \prod_{i=1}^m p(y_i | \mathbf{x}_i) \right) \\ &= \sum_{i=1}^m \log \left( g(\boldsymbol{\theta}^T \mathbf{x}_i)^{y_i} (1 - g(\boldsymbol{\theta}^T \mathbf{x}_i))^{1-y_i} \right) \\ &= \sum_{i=1}^m (y_i \log g(\boldsymbol{\theta}^T \mathbf{x}_i) + (1 - y_i) \log (1 - g(\boldsymbol{\theta}^T \mathbf{x}_i))) \end{aligned}$$

## Log-Likelihood Function (cont.)

Simplification of the log-likelihood function:

$$\begin{aligned}
 \mathcal{L}(\theta) &= \sum_{i=1}^m (y_i \log g(\theta^T \mathbf{x}_i) + (1 - y_i) \log (1 - g(\theta^T \mathbf{x}_i))) \\
 &= \sum_{i=1}^m \left( y_i \log \frac{e^{\theta^T \mathbf{x}_i}}{1 + e^{\theta^T \mathbf{x}_i}} + (1 - y_i) \log \frac{1}{1 + e^{\theta^T \mathbf{x}_i}} \right) \\
 &= \sum_{i=1}^m \left( y_i \theta^T \mathbf{x}_i + \log \frac{1}{1 + e^{\theta^T \mathbf{x}_i}} \right) \\
 &= \sum_{i=1}^m (y_i \theta^T \mathbf{x}_i + \log (1 - g(\theta^T \mathbf{x}_i)))
 \end{aligned}$$

## Log-Likelihood Function (cont.)

Notes for the expert:


- The negative of the log-likelihood function is the cross entropy of  $y$  and  $g(\theta^T \mathbf{x})$ .
- The negative of the log-likelihood function is a convex function.



# Next Time in Pattern Recognition



## Maximization of the log-likelihood function

- The log-likelihood function is concave.
- We use the  Newton-Raphson algorithm to solve the unconstrained optimization problem:

For the  $(k + 1)$ -st iteration step, we get:

$$\boldsymbol{\theta}^{(k+1)} = \boldsymbol{\theta}^{(k)} - \left( \frac{\partial^2}{\partial \boldsymbol{\theta} \partial \boldsymbol{\theta}^T} \mathcal{L}(\boldsymbol{\theta}^{(k)}) \right)^{-1} \frac{\partial}{\partial \boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}^{(k)})$$

**Note:** If you write the Newton-Raphson iteration in matrix form, you will end up with a weighted least squares iteration scheme.

## Newton-Raphson Iteration

Taylor's Theorem:

Approximation of a  $k$ -times differentiable function  $f(x)$  around a given point  $x_0$ :

$$f(x_0 + h) = f(x_0) + f'(x_0)h + \frac{f''(x_0)}{2!}h^2 + \dots + \frac{f^{(k)}(x_0)}{k!}h^k + r_k(x_0 + h)h^k, \quad \lim_{h \rightarrow 0} r_k(x_0 + h) = 0$$

Second order Taylor polynomial:

$$f(x_0 + h) \approx f(x_0) + f'(x_0)h + \frac{1}{2}f''(x_0)h^2$$

## Newton-Raphson Iteration (cont.)

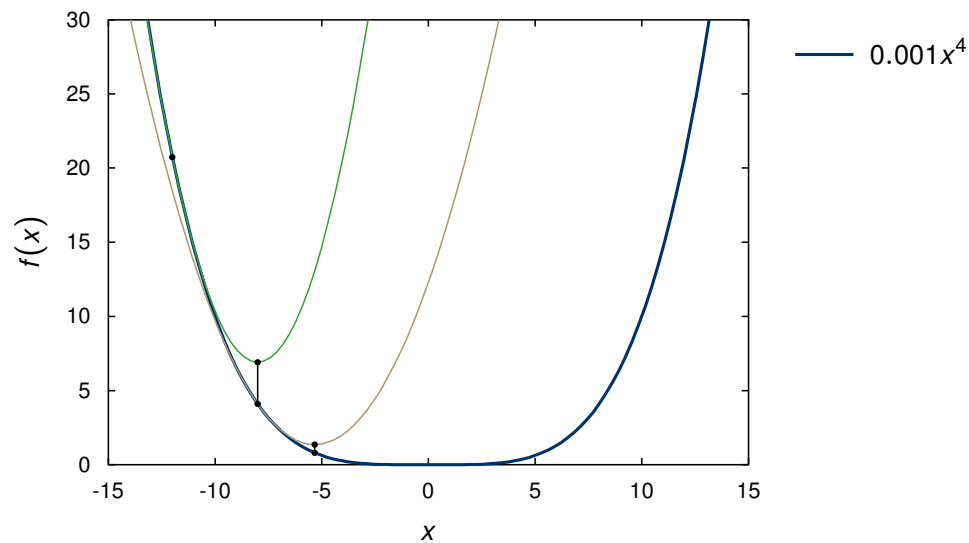
Extremum:

$$f'(x_0 + h) = f'(x_0) + f''(x_0)h \stackrel{!}{=} 0$$

$$\hat{h} = -\frac{f'(x_0)}{f''(x_0)}$$

$$x_1 = x_0 + \hat{h} = x_0 - \frac{f'(x_0)}{f''(x_0)}$$

## Newton-Raphson Iteration (cont.)



## Gradient of the Log-Likelihood Function

The gradient:

$$\begin{aligned}\frac{\partial}{\partial \theta_j} \mathcal{L}(\theta) &= \frac{\partial}{\partial \theta_j} \left( \sum_{i=1}^m (y_i \theta^T \mathbf{x}_i + \log(1 - g(\theta^T \mathbf{x}_i))) \right) \\ &= \sum_{i=1}^m \left( y_i x_{i,j} - \frac{1}{1 - g(\theta^T \mathbf{x}_i)} \frac{\partial}{\partial \theta_j} g(\theta^T \mathbf{x}_i) \right)\end{aligned}$$

Now we use the derivative of the sigmoid function and get

$$\begin{aligned}\frac{\partial}{\partial \theta_j} \mathcal{L}(\theta) &= \sum_{i=1}^m \left( y_i x_{i,j} - \frac{1}{1 - g(\theta^T \mathbf{x}_i)} g(\theta^T \mathbf{x}_i) (1 - g(\theta^T \mathbf{x}_i)) x_{i,j} \right) \\ &= \sum_{i=1}^m (y_i - g(\theta^T \mathbf{x}_i)) x_{i,j}\end{aligned}$$

where  $x_{i,j}$  is the  $j$ -th component of the  $i$ -th training feature vector.



## Gradient of the Log-Likelihood Function (cont.)

Finally, we have a quite simple gradient:

$$\frac{\partial}{\partial \theta_j} \mathcal{L}(\boldsymbol{\theta}) = \sum_{i=1}^m (y_i - g(\boldsymbol{\theta}^T \mathbf{x}_i)) x_{i,j}$$

where  $x_{i,j}$  is the  $j$ -th component of the  $i$ -th training feature vector.

Or in vector notation:

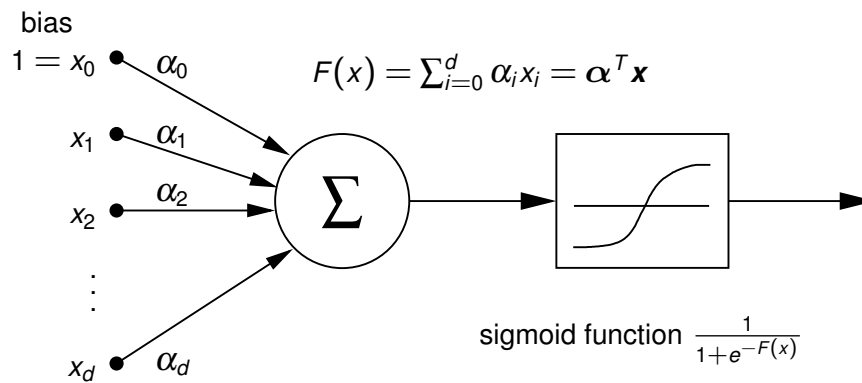
$$\frac{\partial}{\partial \boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}) = \sum_{i=1}^m (y_i - g(\boldsymbol{\theta}^T \mathbf{x}_i)) \mathbf{x}_i$$

## Hessian of the Log-Likelihood Function

- The Newton-Raphson algorithm requires the Hessian matrix.
- Remember the derivative of the sigmoid function!

$$\frac{\partial^2}{\partial \boldsymbol{\theta} \partial \boldsymbol{\theta}^T} \mathcal{L}(\boldsymbol{\theta}) = - \sum_{i=1}^m g(\boldsymbol{\theta}^T \mathbf{x}_i) (1 - g(\boldsymbol{\theta}^T \mathbf{x}_i)) \mathbf{x}_i \mathbf{x}_i^T$$

## Perceptron and Logistic Regression



## Lessons Learned

- Posteriors can be rewritten in terms of a logistic function.
- Given the decision boundary  $F(\mathbf{x}) = 0$ , we can write down the posterior  $p(y|\mathbf{x})$  right away.
- Decision boundary for normally distributed feature vectors for each class is a quadratic function.
- If Gaussians share the same covariances, the decision boundary is a linear function.



# Next Time in Pattern Recognition



## Further Readings

- T. Hastie, R. Tibshirani, and J. Friedman:  
The Elements of Statistical Learning –  
Data Mining, Inference, and Prediction,  
2nd edition, Springer, New York, 2009.
- David W. Hosmer, Stanley Lemeshow:  
Applied Logistic Regression, 2nd Edition,  
John Wiley & Sons, Hoboken, 2000.

## Comprehensive Questions (cont.)

- How can a nonlinear function be written as a scalar product?
- What is the objective function for the ML-estimation of the logistic regression parameters?
- What is the difference between a gradient descent and Newton-Raphson numerical optimization scheme?
- What is the parameter update rule for the logistic regression parameters using the Newton-Raphson scheme?