Machine Learning

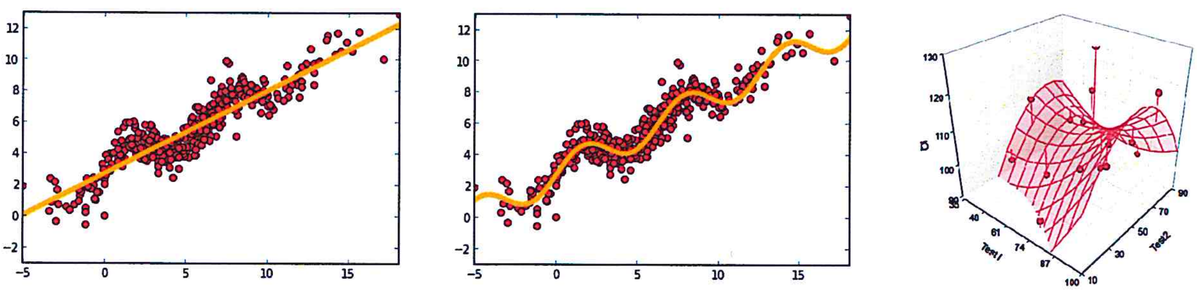
– winter term 2016/17 –

Chapter 04: Logistic Regression

Prof. Adrian Ulges
Masters "Computer Science"
DCSM Department
University of Applied Sciences RheinMain

Classification vs. Regression

- Given: training samples $\mathbf{x}_1, ..., \mathbf{x}_n \in \mathcal{X}$ with labels $y_1, ..., y_n$
- **Classification**
  - Labels indicate class membership
  - Learn a **classifier** function $\mathbb{R}^d \to \{1, ..., C\}$, assigning samples to classes.
- **Regression**
  - Labels are **real-valued**!
  - Learn a **regression** function $f : \mathbb{R}^d \to \mathbb{R}$, assigning samples to continuous values.
- **Regression Examples** *(incl. the "classic": linear regression)*



Logistic Regression: Approach

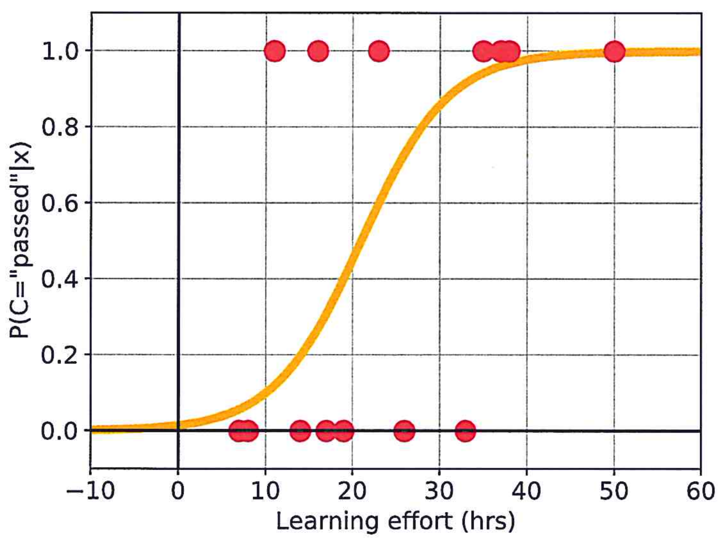
- **Logistic Regression** (aka. **Maximum Entropy**) is a common approach in statistical data analysis[1]
- Idea: Use a **regression model for classification**
  - Compute a **score** for each class using regression
  - This score should approximate the probability that the given object belongs to class c, given that its features are $\mathbf{x}$: $P(c|\mathbf{x})$
  - The classifier picks the class with **maximum score**

[1] Cox, DR (**1958**), "The regression analysis of binary sequences (with discussion)". J Roy Stat Soc B.

Logistic Regression: Approach

- Assumption: 2 classes only (0 vs. 1) *(success/failure, well/sick, ...)*
- **Given**: a test sample $\mathbf{x}$
- **Goal**: estimate $P(C = 1|\mathbf{x})$

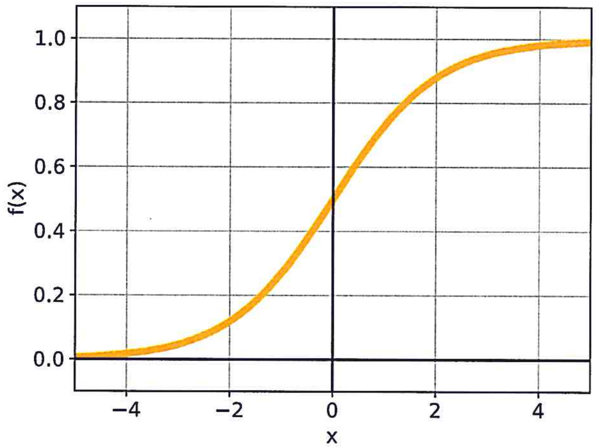
Example (math exam)



Logistic Regression: Model

- As a base model, we use the so-called **Sigmoid** function

$$P(C=1|x) \approx f(x) = \frac{1}{1+e^{-x}}$$



- **Property A**: $lim_{x \to -\infty} f(x) = 0$ and $lim_{x \to \infty} f(x) = 1$
- **Property B**: $P(C=1|x=0) = f(0) = 0.5$
  $\rightarrow$ We choose class 1 iff. $x \geq 0$.

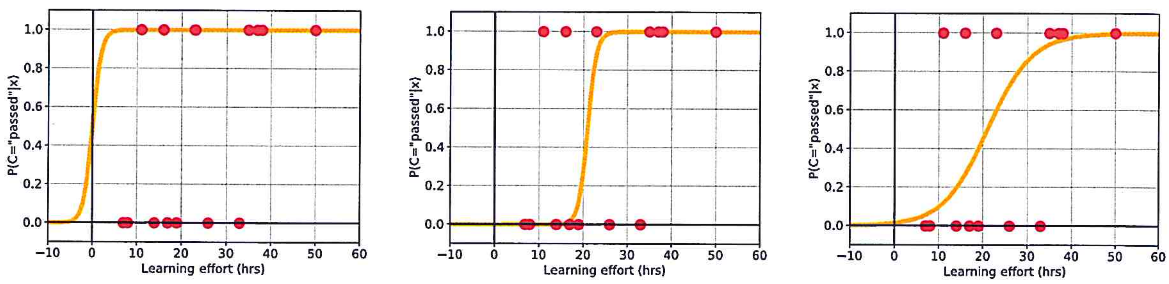
Logistic Regression: Model

Extension

- We allow a shift and scaling of the sigmoid:

$$f(x; w_0, w) = \frac{1}{1+e^{-(w_0 + w \cdot x)}}$$

- The parameters w_0, w are estimated via learning (soon...)



Logistic Regression: Remarks

Why this Model?

- **simplicity**, intuition
- The model is correct for **normally distributed** classes with identical variance
- **tradition**
- **few parameters** to fit → little overfitting, even in case of few training samples

Why not use **linear** regression?

Multi-variate Logistic Regression

- We apply logistic regression in case of multiple features $\mathbf{x} \in \mathbb{R}^d$?
- We extend the sigmoid function:

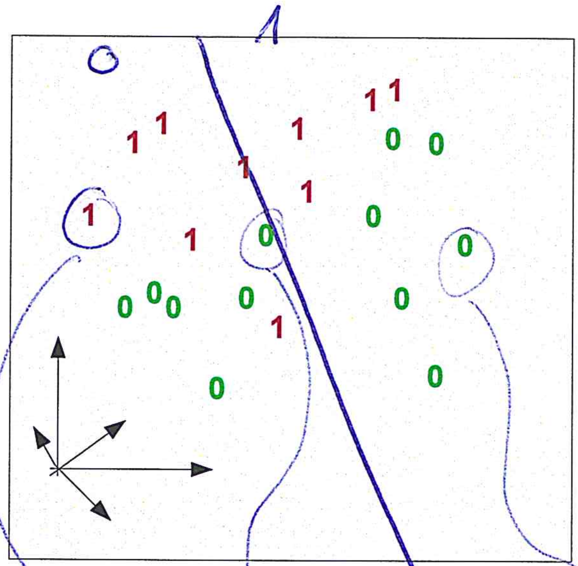
$$f(\mathbf{x}; w_0, w_1, w_2..., w_d) = \frac{1}{1 + e^{-(w_0 + w_1 \cdot x_1 + w_2 \cdot x_2 + ... + w_d \cdot x_d)}}$$

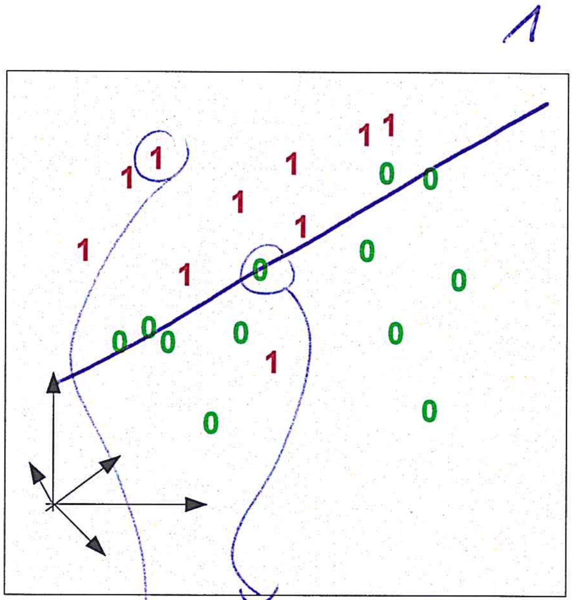
or short (with vector $\boldsymbol{w} := (w_1, ..., w_d)$):

$$f(\mathbf{x}; w_0, \boldsymbol{w}) = \frac{1}{1 + e^{-(w_0 + \mathbf{x} \cdot \boldsymbol{w})}}$$

- The boundary between the two classes (or **decision boundary**) of this model is at $\boldsymbol{x} \cdot \boldsymbol{w} + w_0 = 0$.
  This is a **hyperplane** (in normal form)!

Logistic Regression: Example





P(C=0|x) ≈ 50% (ok)

P(C=1|x) ≈ 0% (bad)

P(C=0|x) ≈ 0%

P(C=0|x) ≈ 50% (ok!)

P(C=1|x) ≈ 100% (nice!)

Logistic Regression: Formalization

Maximum-Likelihood Estimation

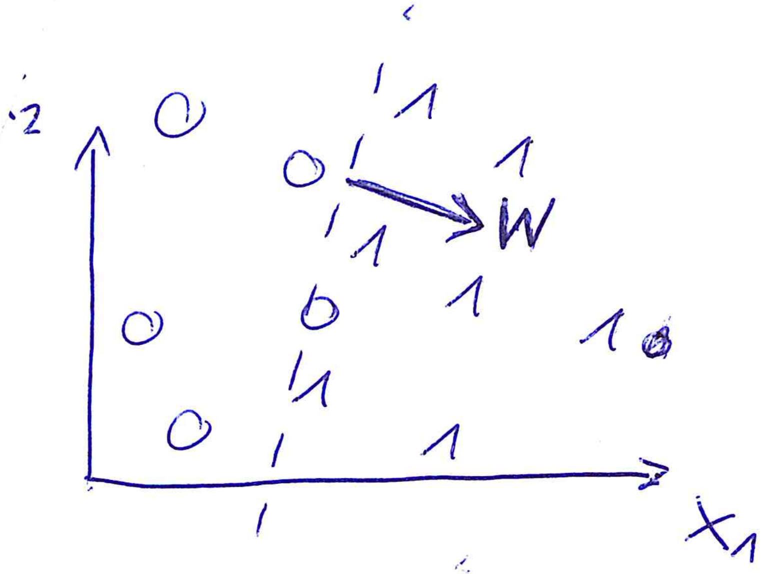
We define a likelihood function and formulate an optimization problem:

P(C=1|x_i)

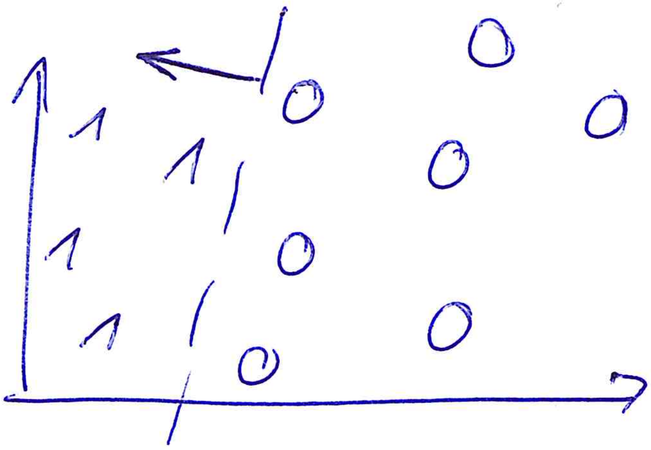
$$w_0^*, \mathbf{w}^* = \arg\max_{w_0,\mathbf{w}} \underbrace{\prod_{i:y_1=1} f(\mathbf{x}_i) \cdot \prod_{i:y_i=0} (1 - f(\mathbf{x}_i))}_{\text{"Likelihood function" } L(w_0,\mathbf{w})}$$

We rewrite the optimization problem:

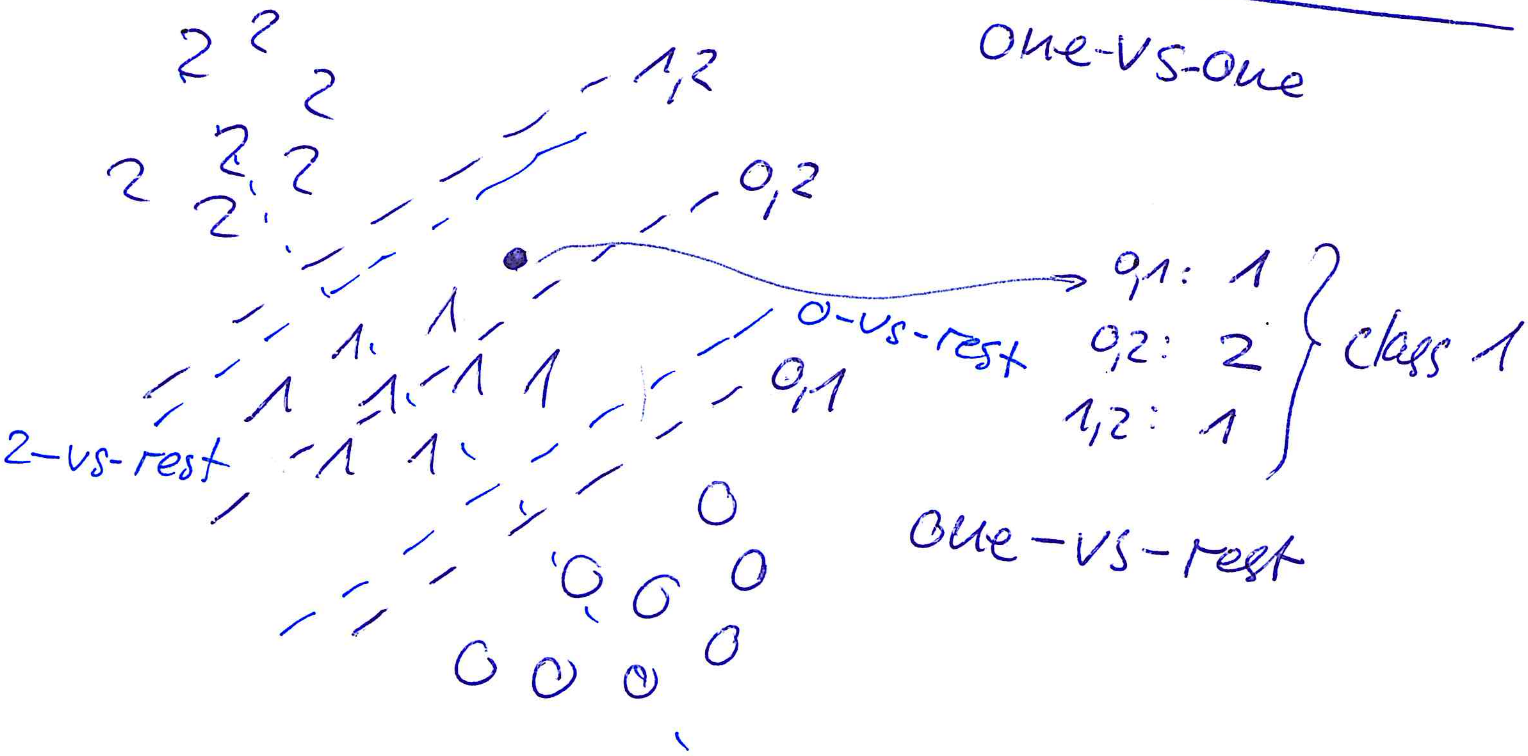
$$\begin{aligned} w_0^*, \mathbf{w}^* &= \arg\max_{w_0,\mathbf{w}} \prod_{i:y_1=1} f(\mathbf{x}_i) \cdot \prod_{i:y_i=0} (1 - f(\mathbf{x}_i)) \\ &= \arg\max_{w_0,\mathbf{w}} \prod_{i} f(\mathbf{x}_i)^{y_i} \cdot (1 - f(\mathbf{x}_i))^{1-y_i} \qquad \text{// log} \\ &= \arg\max_{w_0,\mathbf{w}} \sum_{i} y_i \cdot log(f(\mathbf{x}_i)) + (1 - y_i) \cdot log(1 - f(\mathbf{x}_i)) \end{aligned}$$



$$W = \begin{pmatrix} 3 \\ -0{,}1 \end{pmatrix}$$



$$W = \begin{pmatrix} -3 \\ +0{,}1 \end{pmatrix}$$



Logistic Regression: Formalization

$$w_0^*, \mathbf{w}^* = \arg\max_{w_0,\mathbf{w}} \underbrace{\prod_{i:y_1=1} f(\mathbf{x}_i) \cdot \prod_{i:y_i=0} (1 - f(\mathbf{x}_i))}_{\text{"Likelihood-Funktion"} L(w_0,\mathbf{w})}$$

$$= \arg\max_{w_0,\mathbf{w}} \prod_i f(\mathbf{x}_i)^{y_i} \cdot (1 - f(\mathbf{x}_i))^{1-y_i} \qquad // \log$$

$$= \arg\max_{w_0,\mathbf{w}} \sum_i y_i \cdot log(f(\mathbf{x}_i)) + (1 - y_i) \cdot log(1 - f(\mathbf{x}_i))$$

$$= \arg\max_{w_0,\mathbf{w}} \sum_i log(1 - f(\mathbf{x}_i)) + y_i \cdot log\Big(\frac{f(\mathbf{x}_i)}{1 - f(\mathbf{x}_i)}\Big)$$

$$= \arg\max_{w_0,\mathbf{w}} \sum_i log\Big(\frac{\cancel{1} + exp(-(w_0 + \mathbf{x}_i\mathbf{w})) - \cancel{1}}{1 + exp(-(w_0 + \mathbf{x}_i\mathbf{w}))}\Big) + y_i \cdot log\left(\frac{\frac{1}{(1+exp(\cancel{-(w_0+\mathbf{x}_i\mathbf{w})}))}}{\frac{exp(-(w_0+\mathbf{x}_i\mathbf{w}))}{(1+exp(\cancel{-(w_0+\mathbf{x}_i\mathbf{w})}))}}\right)$$

$$= \arg\max_{w_0,\mathbf{w}} \sum_i -log\Big(\frac{1 + exp(-(w_0 + \mathbf{x}_i\mathbf{w}))}{exp(-(w_0 + \mathbf{x}_i\mathbf{w}))}\Big) - y_i \cdot \cancel{log}\Big(\cancel{exp}(-(w_0 + \mathbf{x}_i\mathbf{w}))\Big)$$

$$= \arg\max_{w_0,\mathbf{w}} \sum_i -log\Big(e^{w_0+\mathbf{x}_i\mathbf{w}} + 1\Big) + \sum_i y_i \cdot (w_0 + \mathbf{x}_i\mathbf{w})$$

Logistic Regression: Formalization

$$\arg\max_{w_0,\mathbf{w}} \underbrace{\sum_i -log\Big(e^{w_0+\mathbf{x}_i\mathbf{w}} + 1\Big) + \sum_i y_i \cdot (w_0 + \mathbf{x}_i\mathbf{w})}_{\text{"Log-Likelihood Function"} L(w_0,\mathbf{w})}$$

Remarks

- There is no analytical solution for maximizing the Log-Likelihood function L.
- We solve the problem **numerically**: For example, by finding roots of the gradient using **Newton's method**.
- The weights $\mathbf{w}$ indicate the **importance** of the single features for the classification problem.

Logistic Regression: Regularization

- **Observation**: Even though logistic regression is fairly robust, the model tends to **overfit** when ...
  - ... single features get a very extreme weight
  - ... many unimportant weights get a weight $\neq 0$.
- To avoid this, we **regularize** the problem, such that the entries in **w** tend to be small (or even 0).
- We define the **norm** of the weight vector **w**

$$||\mathbf{w}||_1 := |w_1| + |w_2| + ... + |w_d| \qquad \text{L1 norm}$$

$$||\mathbf{w}||_2 := \sqrt{w_1^2 + w_2^2 + ... + w_d^2} \qquad \text{L2 norm}$$

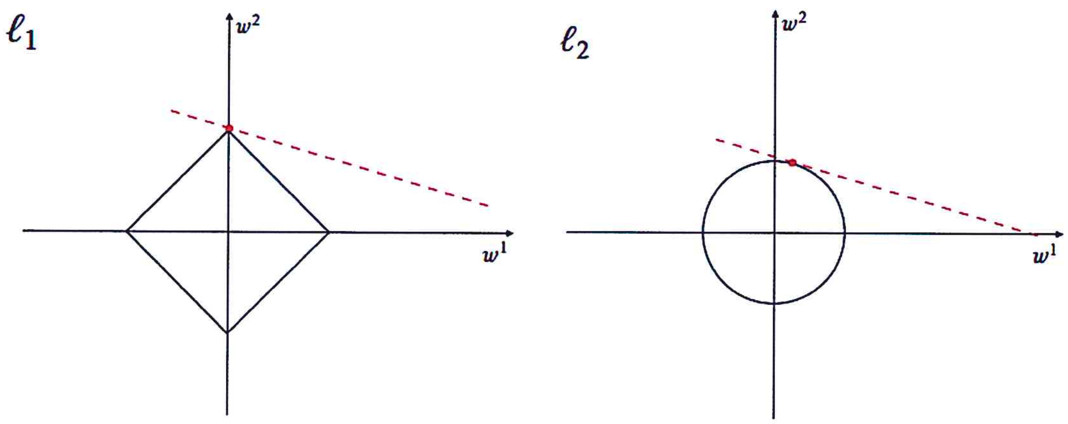
- We adapt the optimization problem such that high weights in **w** sanctioned (with $C \in \mathbb{R}$):

$$\arg\max_{w_0,\mathbf{w}} \; L(w_0, \mathbf{w}) - C \cdot ||\mathbf{w}||_1 \qquad // \text{ L1-Regularisierung}$$

$$\arg\max_{w_0,\mathbf{w}} \; L(w_0, \mathbf{w}) - C \cdot ||\mathbf{w}||_2 \qquad // \text{ L2-Regularisierung}$$

What Difference does L1 vs. L2 make?

Example: Optimizing a Linear Function (regularized)



Left: $\mathbf{w} = (0, 1)$ (= L1 solution). **Right**: $\mathbf{w} = (0.15, 0.99)$ (=L2 solution).

- **L1-Regularization** enforces the weights of uninformative features to be 0 (the weight vector is **sparse**). Put differently: The classifier conducts a built-in **feature selection**.
- **L2-Regularization** reduces outliers (= extreme weights)

Logistic Regression: Code Sample

- Bag-of-Words Features
- Logistic Regression
- Inspect term weights