*** Applied Machine Learning Fundamentals ***
Logistic Regression

Daniel Wehner

SAP SE

October 29, 2019

Find all slides on GitHub

Agenda October 29, 2019

Section:

Introduction

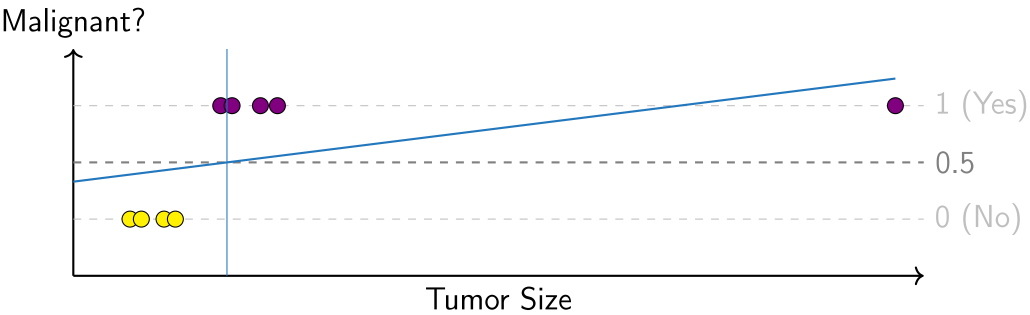
DHBW
Duale Hochschule
Baden-Württemberg

What is logistic Regression?

- Learning algorithm for **classification** *(despite the name...)*
- In its standard form it's applicable to **binary classification problems only**, but you can use techniques like:
  - One-vs-One (OVO)
  - One-vs-Rest (OVR)
- **Class labels:**
  - The 'positive class' is usually encoded as **1** / $\oplus$
  - The 'negative class' as **0** / $\ominus$
- **Probabilistic interpretation:** The output of the algorithm is between 0 and 1 *(probability of the instance belonging to the positive class)*

Why you should not use linear Regression...

Why you should not use linear Regression...



Why you should not use linear Regression... (Ctd.)

- Linear regression: $h_{\boldsymbol{\theta}}(x) = \boldsymbol{\theta}^\intercal \boldsymbol{x}$
- By putting a **threshold** at 0.5, we can turn linear regression into a classifier
  - If $h_{\boldsymbol{\theta}}(x) \geqslant 0.5$, predict $y = 1$
  - If $h_{\boldsymbol{\theta}}(x) < 0.5$, predict $y = 0$
- Outliers **affect the decision boundary**
- Furthermore, we only want $0 \leqslant h_{\boldsymbol{\theta}}(x) \leqslant 1$
- Linear regression can output $h_{\boldsymbol{\theta}}(x) \ll 0$ or $h_{\boldsymbol{\theta}}(x) \gg 1$
- **We need a better strategy!**

Section:

Model Architecture

Duale Hochschule
Baden-Württemberg

Logistic Regression Model

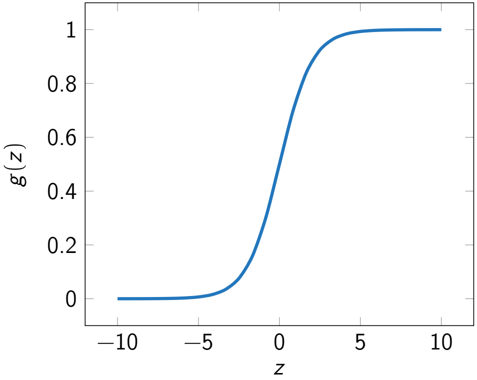
- Remember that we want: $0 \leqslant h_{\theta}(x) \leqslant 1$
- **Solution:** Logistic/Sigmoid function:

$$g(z) = \frac{1}{1 + e^{-z}} \tag{1}$$

- We plug $\boldsymbol{\theta}^{\intercal}\boldsymbol{x}$ into the sigmoid function:

$$h_{\theta}(x) = g(\boldsymbol{\theta}^{\intercal}\boldsymbol{x}) = \frac{1}{1 + e^{-(\boldsymbol{\theta}^{\intercal}\boldsymbol{x})}} \tag{2}$$

Logistic/Sigmoid Function



- $g(z)$ is symmetric around $z = 0$
- $0 \leqslant g(z) \leqslant 1$ holds true

Where does the Sigmoid come from?

$$p(\mathcal{C}_1|\boldsymbol{x}) = \frac{p(\boldsymbol{x}|\mathcal{C}_1)p(\mathcal{C}_1)}{p(\boldsymbol{x})} = \frac{p(\boldsymbol{x}|\mathcal{C}_1)p(\mathcal{C}_1)}{\sum_j p(\boldsymbol{x}, \mathcal{C}_j)} = \frac{p(\boldsymbol{x}|\mathcal{C}_1)p(\mathcal{C}_1)}{\sum_j p(\boldsymbol{x}|\mathcal{C}_j)p(\mathcal{C}_j)} \tag{3}$$

$$= \frac{p(\boldsymbol{x}|\mathcal{C}_1)p(\mathcal{C}_1)}{p(\boldsymbol{x}|\mathcal{C}_1)p(\mathcal{C}_1) + p(\boldsymbol{x}|\mathcal{C}_2)p(\mathcal{C}_2)} \tag{4}$$

$$= \frac{1}{1 + p(\boldsymbol{x}|\mathcal{C}_2)p(\mathcal{C}_2)/(p(\boldsymbol{x}|\mathcal{C}_1)p(\mathcal{C}_1))} \tag{5}$$

$$= \frac{1}{1 + \exp\{-z\}} = g(z) \qquad \longrightarrow \text{logistic sigmoid} \tag{6}$$

$$z = \log \frac{p(\boldsymbol{x}|\mathcal{C}_1)p(\mathcal{C}_1)}{p(\boldsymbol{x}|\mathcal{C}_2)p(\mathcal{C}_2)} \qquad \longrightarrow \text{log odds} \tag{7}$$

Interpretation of Hypothesis Output

- $h_{\boldsymbol{\theta}}(x)$ is interpreted as the probability of instance x belonging to class $y = 1$
- **Example:**

$$\mathbf{x} = \begin{bmatrix} x_0 \\ x_1 \end{bmatrix} = \begin{bmatrix} 1 \\ tumorSize \end{bmatrix} \tag{8}$$

- If $h_{\boldsymbol{\theta}}(x) = 0.7$, we have to tell the patient that there is a **70 % chance** of the tumor being malignant $\Rightarrow p(y = 1|\mathbf{x}, \boldsymbol{\theta})$
- **Binary case:** $p(y = 0|\mathbf{x}, \boldsymbol{\theta}) = 1 - p(y = 1|\mathbf{x}, \boldsymbol{\theta})$

Training Setup

- We have a labeled training set ($\Rightarrow$ **supervised learning**):

$$\mathcal{D} = \left\{(\boldsymbol{x}^{(1)}, y^{(1)}), (\boldsymbol{x}^{(2)}, y^{(2)}), \dots, (\boldsymbol{x}^{(n)}, y^{(n)})\right\} = \left\{(\boldsymbol{x}^{(i)}, y^{(i)})\right\}_{i=1}^{n} \tag{9}$$

- Each $\boldsymbol{x}$ is a vector of features:

$$\boldsymbol{x} = \begin{bmatrix} x_0 \\ \vdots \\ x_m \end{bmatrix} \in \mathbb{R}^{m+1} \quad \text{and} \quad x_0 = 1 \quad \text{and} \quad y \in \{0, 1\} \tag{10}$$

- **How to choose the parameters θ?**

Logistic Regression Cost Function

- Gradient descent is performed in order to find the parameters $\boldsymbol{\theta}$
- To this end, a cost function is needed:

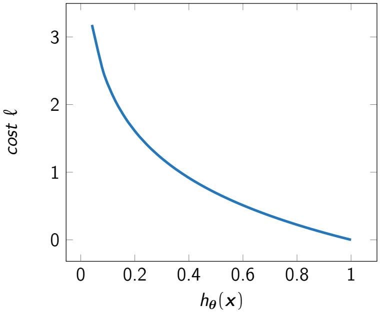
$$\mathcal{J}(\boldsymbol{\theta}) = \frac{1}{n} \sum_{i=1}^{n} \ell(h_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)}), y^{(i)}) \tag{11}$$

- The cost function $\ell(h_{\boldsymbol{\theta}}(\boldsymbol{x}), y)$ is defined as follows:
  *(square loss would be **non-convex...**)*

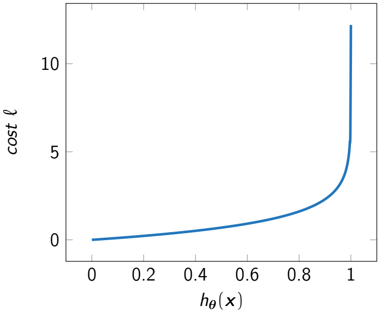
$$\ell(h_{\boldsymbol{\theta}}(\boldsymbol{x}), y) = \begin{cases} -\log(h_{\boldsymbol{\theta}}(\boldsymbol{x})) & \text{if } y = 1 \\ -\log(1 - h_{\boldsymbol{\theta}}(\boldsymbol{x})) & \text{if } y = 0 \end{cases} \tag{12}$$

Logistic Regression Cost Function (Ctd.)

$y = 1$:



$y = 0$:



Logistic Regression Cost Function (Ctd.)

- $\ell(h_{\boldsymbol{\theta}}(\boldsymbol{x}), y)$ can be written in a more compact form:

$$\ell(h_{\boldsymbol{\theta}}(\boldsymbol{x}), y) = -y \log(h_{\boldsymbol{\theta}}(\boldsymbol{x})) - (1-y)\log(1 - h_{\boldsymbol{\theta}}(\boldsymbol{x})) \tag{13}$$

  - If $y = 1$, we get: $-\log(h_{\boldsymbol{\theta}}(\boldsymbol{x}))$
  - If $y = 0$, we get: $-\log(1 - h_{\boldsymbol{\theta}}(\boldsymbol{x}))$
- This gives the **cross entropy** cost function $\mathcal{J}(\boldsymbol{\theta})$:

$$\mathcal{J}(\boldsymbol{\theta}) = \frac{1}{n}\sum_{i=1}^{n}\left[-y^{(i)}\log(h_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)})) - (1-y^{(i)})\log(1 - h_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)}))\right] \tag{14}$$

Derivation of Cross Entropy

- The likelihood function can be written in the form:

$$\mathcal{L}(\boldsymbol{\theta}) = \prod_{i=1}^{n} h_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)})^{y^{(i)}} \cdot (1 - h_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)}))^{1-y^{(i)}} \tag{15}$$

- The cost function is then given by the **negative log-likelihood**:

$$\mathcal{J}(\boldsymbol{\theta}) = -\log \mathcal{L}(\boldsymbol{\theta}) \tag{16}$$

Gradient Descent

- The goal is to minimize $\mathcal{J}(\boldsymbol{\theta})$: $\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \mathcal{J}(\boldsymbol{\theta})$
- `Repeat until convergence {`

  $\boldsymbol{\theta}^{(t+1)} \longleftarrow \boldsymbol{\theta}^{(t)} - \alpha \nabla_{\boldsymbol{\theta}} \mathcal{J}(\boldsymbol{\theta}^{(t)})$ *// simultaneously update all* θ_j

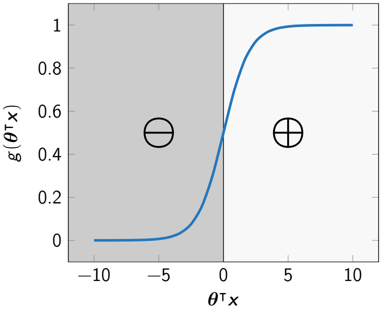
  `}`
- The partial derivative $\nabla_{\boldsymbol{\theta}} \mathcal{J}(\boldsymbol{\theta})$ is given by:

$$\nabla_{\boldsymbol{\theta}} \mathcal{J}(\boldsymbol{\theta}) = \frac{1}{n} \sum_{i=1}^{n} \left(h_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)}) - y^{(i)} \right) x_j^{(i)} \tag{17}$$

Algorithm looks identical to linear regression, but $h_{\theta}(x)$ is different!

Decision Boundary

- **For classification we have to set a threshold**
- Suppose we predict $y = 1$ if $h_{\boldsymbol{\theta}}(x) \geqslant 0.5$
  - This means $g(z) \geqslant 0.5$
  - This is equivalent to $z \geqslant 0$ and $\boldsymbol{\theta}^\intercal \boldsymbol{x} \geqslant 0$
- Suppose we predict $y = 0$ if $h_{\boldsymbol{\theta}}(x) < 0.5 \Rightarrow \boldsymbol{\theta}^\intercal \boldsymbol{x} < 0$



Decision Boundary (Ctd.)

- Suppose we have the following hypothesis:

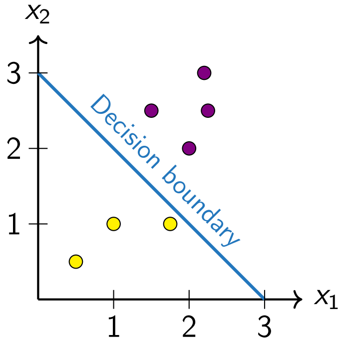
$$h_{\boldsymbol{\theta}}(\boldsymbol{x}) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$$

- Using gradient descent we obtained the following coefficients:

$$\theta_0 = -3 \qquad \theta_1 = 1 \qquad \theta_2 = 1$$

- Predict $y = 1$ if $-3 + x_1 + x_2 \geqslant 0$

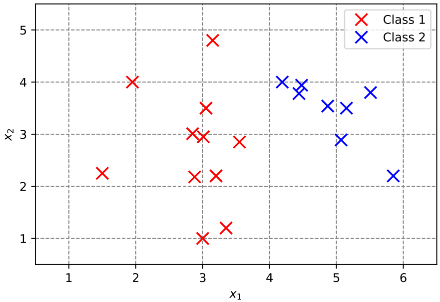
Decision Boundary (Ctd.)

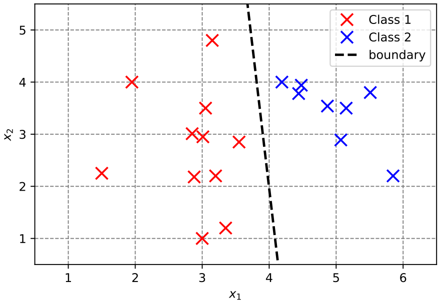


- Predict $y = 1$, if $-3 + x_1 + x_2 \geqslant 0$
- The decision boundary satisfies $-3 + x_1 + x_2 = 0$
- If $x_2 = 0$, then $x_1 = 3$ and vice versa

Logistic regression is not a maximum-margin classifier (although the cost function can be adjusted to get that $\Rightarrow$ Hinge loss)

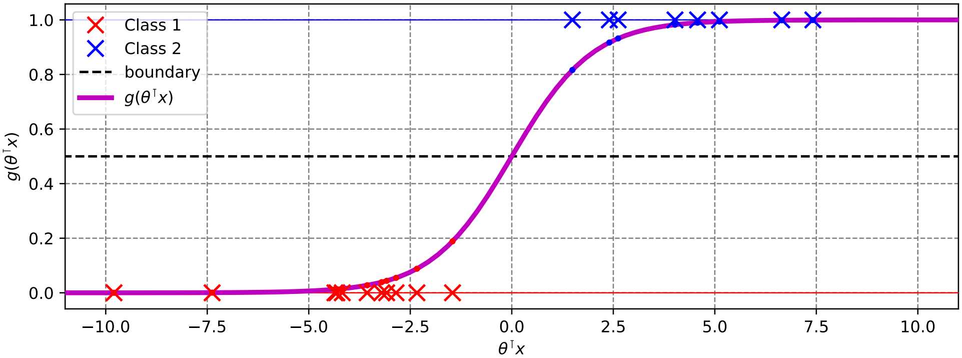
Example: Decision Boundary





Where is the sigmoid function?

Example: Logistic Function



Section:

Non-linear Data

DHBW
Duale Hochschule
Baden-Württemberg

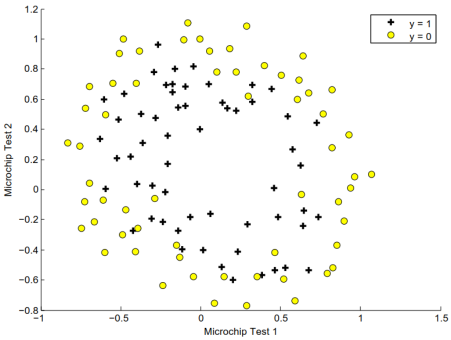
Non-Linear Decision Boundaries

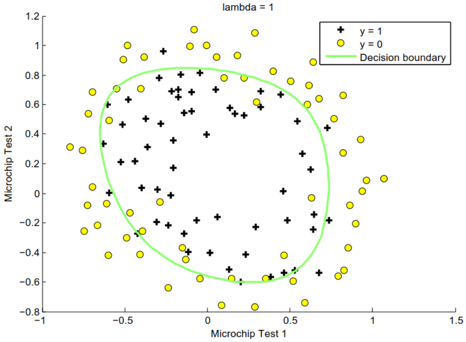
- **Feature mapping** can be used to obtain non-linear decision boundaries
- **Example:**
  - Imagine a circular data set
  - Using the features...

$$h_{\boldsymbol{\theta}}(\boldsymbol{x}) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_2^2)$$

  - ...the algorithm could e. g. choose: $\boldsymbol{\theta} = \begin{bmatrix}-1, 0, 0, 1, 1\end{bmatrix}^\intercal$
  - So we would get: $x_1^2 + x_2^2 = 1 \Rightarrow$ **equation of a unit circle**

Example: Non-Linear Decision Boundary





Logistic Regression Cost Function (Ctd.)

- We should apply regularization for non-linear decision boundaries:

$$\frac{1}{n}\sum_{i=1}^{n}\left[-y^{(i)}\log(h_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)})) - (1-y^{(i)})\log(1-h_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)}))\right] + \frac{\lambda}{2m}\sum_{j=1}^{m}\theta_j^2 \tag{18}$$

- The last term prevents the parameters θ_j from becoming too large
- $\lambda \geqslant 0$ controls the degree of regularization
- This leads to smoother decision boundaries

Section:

Multi-Class Classification

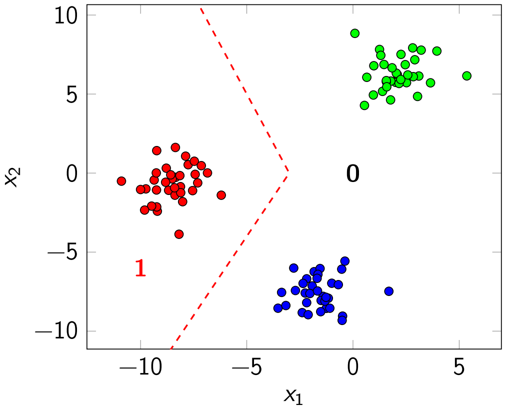
Duale Hochschule
Baden-Württemberg

Multi-Class Classification

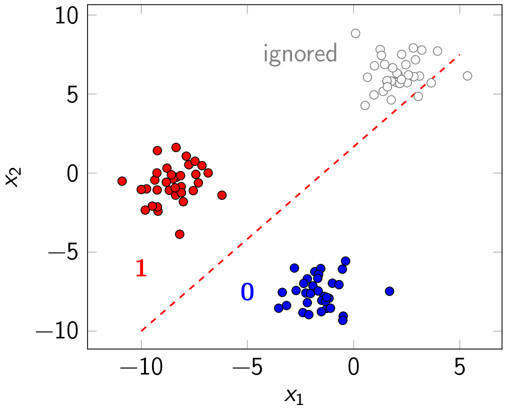
- Logistic regression can handle two classes only, namely **0** and **1**
- **What if there are more than two classes?**
- Two common techniques:
  - One-vs-Rest (OVR) $\Rightarrow$ One-against-All
  - One-vs-One (OVO) $\Rightarrow$ Pairwise classification
- Several classifiers are trained
- During prediction the classifiers **vote for the correct class**
- Such techniques can be used for all binary classifiers

Multi-Class Classification: One-vs-Rest (OVR)

- **Train one classifier per class** (expert for that class)
- We get $|\mathcal{C}|$ classifiers
- The k-th classifier learns to distinguish the k-th class from all the others
- Set the labels of examples from class k to **1**, all the others to **0**



Multi-Class Classification: One-vs-One (OVO)



- **Train one classifier for each pair of classes**
- We get $\binom{\mathcal{C}}{2}$ classifiers
- Ignore all other examples that do not belong to either of the two classes
- **Voting**: Count how often each class wins; the class with highest count is predicted

Section:

Wrap-Up

Summary

Lecture Overview

Unit I: Machine Learning Introduction

Self-Test Questions

Recommended Literature and further Reading

Thank you very much for the attention!

Topic: *** Applied Machine Learning Fundamentals *** Logistic Regression
Date: October 29, 2019

Contact:
Daniel Wehner (D062271)
SAP SE
daniel.wehner@sap.com

Do you have any questions?