Deep Learning

Summer Semester 2024

Monday, April 8, 2024

Prof. Dr.-Ing. Christian Bergler | OTH Amberg-Weiden

Training Parametric Models
Overview

Topics From Last Time: Mathematical Foundations

- Functions of one and several variables
- Continuous functions
- Derivative and gradient
- Convexity and optimization
- Local and global extrema of real functions
- Matrix calculation
- Norms

Topics of Today: Training of Parametric Models

- Principle of model training for parameterized models
- Vectorization
- Linear regression and logistic regression

Source: OTH-AW, Electrical Engineering, Media and Computer Science, Fabian Brunner – Vorlesung Deep Learning, Training parametrisierter ML-Modelle

Training Parametric Models

General Principle

Ingredients

- Training data $(\mathbf{x}^{(i)}, y^{(i)})_{i=1,\ldots,m}$
- Weights to be determined θ
- Model function that depends on the weights: $f(\mathbf{x}) = f_\theta(\mathbf{x})$
- Loss function L measures model error w.r.t the training data and parametric set θ:

$$L = L(\theta)$$

Model Training

Solve the minimisation problem

$$\min_\theta L(\theta)$$

Source: OTH-AW, Electrical Engineering, Media and Computer Science, Fabian Brunner – Vorlesung Deep Learning, Training parametrisierter ML-Modelle

Training Parametric Models

Approach for Linear Regression

Model Function for Linear Regression

Model function for multivariate linear regression with p features:

$$\hat{y} = f_\theta(\mathbf{x}) = \theta_0 + \theta_1 x_1 + \ldots + \theta_p x_p$$

where $\theta_0, \ldots, \theta_p$ denote the model parameters.

Remarks:

- If the features and weights are summarized in vectors $\mathbf{x} = (x_0, \ldots, x_p)^T$ and $\theta = (\theta_0, \ldots, \theta_p)^T$ (using the convention $x_0 = 1$), then the model function is in vectorised form
$$\hat{y} = f_\theta(\mathbf{x}) = \theta^T \cdot \mathbf{x}$$
- The approach is called *linear* regression, as the model function depends linearly on the parameters θ (often referred to as weights $\mathbf{w}$)

Source: OTH-AW, Electrical Engineering, Media and Computer Science, Fabian Brunner – Vorlesung Deep Learning, Training parametrisierter ML-Modelle

Training Parametric Models

Model Training Linear Regression

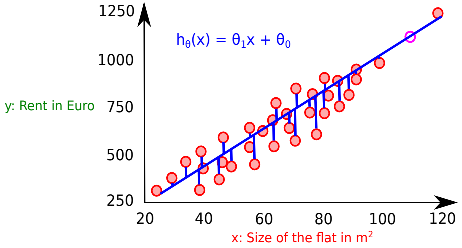
Given: Dataset with examples $(\mathbf{x}^{(1)}, y^{(1)}), \dots, (\mathbf{x}^{(m)}, y^{(m)})$ where $\mathbf{x}^{(i)} \in \mathbb{R}^p$ contains the feature values and $y^{(i)} \in \mathbb{R}$ contains the labels (supervised)

Model Training

Determine the weights so that the least squares error functional

$$L(\theta) = \frac{1}{2m} \sum_{i=1}^{m} (f_\theta(\mathbf{x}^{(i)}) - y^{(i)})^2$$

is minimized.



Source: OTH-AW, Electrical Engineering, Media and Computer Science, Fabian Brunner – Vorlesung Deep Learning, Training parametrisierter ML-Modelle

OTH Lecture – Prof. Dr.-Ing. Christian Bergler

Additional Notes

OTH Lecture – Prof. Dr.-Ing. Christian Bergler

Additional Notes

Training Parametric Models
Vectorization

Question: How can the model function be evaluated simultaneously for several inputs?

Evaluation Single Sample $\mathbf{x} \in \mathbb{R}^{p+1}$:

$$\hat{y} = \theta^T \mathbf{x} \ , \quad \hat{y} \in \mathbb{R} \ .$$

Evaluation Entire Batch $X \in \mathbb{R}^{m \times p+1}$

$$\hat{\mathbf{y}} = X\theta \ , \quad \hat{\mathbf{y}} \in \mathbb{R}^m$$

Comments:

- By executing calculations in vectorised form, an increase in efficiency can often be achieved compared to componentised execution
- A „middle way“ between the size of the batches and the required memory space can be selected

Source: OTH-AW, Electrical Engineering, Media and Computer Science, Fabian Brunner – Vorlesung Deep Learning, Training parametrisierter ML-Modelle

Training Parametric Models

Vectorization

Exercise: Display the least squares error functional for linear regression in vectorised form:

$$L(\theta) = \frac{1}{2m} \sum_{i=1}^{m} (f_\theta(\mathbf{x}^{(i)}) - y^{(i)})^2, \text{ with } \hat{y} = f_\theta(\mathbf{x}) = \theta^T \cdot \mathbf{x} = \sum_{j=0}^{p} \theta_j x_j^{(i)}$$

Training Parametric Models

Vectorization

Exercise: Display the least squares error functional for linear regression in vectorised form:

$$L(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(f_\theta(\mathbf{x}^{(i)}) - y^{(i)})^2, \text{ with } \hat{y} = f_\theta(\mathbf{x}) = \theta^T \cdot \mathbf{x} = \sum_{j=0}^{p}\theta_j x_j^{(i)}$$

$$L(\theta) = \frac{1}{2m}\sum_{i=1}^{m}\left(\sum_{j=0}^{p}\theta_j x_j^{(i)} - y^{(i)}\right)^2 = \frac{1}{2m}\left[\begin{pmatrix} 1 & x_1^{(1)} & \dots & x_p^{(1)} \\ 1 & x_1^{(2)} & \dots & x_p^{(2)} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_1^{(m)} & \dots & x_p^{(m)} \end{pmatrix}\begin{pmatrix}\theta_0\\ \theta_1 \\ \vdots \\ \theta_p\end{pmatrix} - \begin{pmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(m)}\end{pmatrix}\right]^T \cdot \left[\underbrace{\begin{pmatrix} 1 & x_1^{(1)} & \dots & x_p^{(1)} \\ 1 & x_1^{(2)} & \dots & x_p^{(2)} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_1^{(m)} & \dots & x_p^{(m)} \end{pmatrix}}_{=:X}\underbrace{\begin{pmatrix}\theta_0\\ \theta_1 \\ \vdots \\ \theta_p\end{pmatrix}}_{=:\theta} - \underbrace{\begin{pmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(m)}\end{pmatrix}}_{=:\mathbf{y}}\right]$$

$$= \frac{1}{2m}(X\theta - \mathbf{y})^T(X\theta - \mathbf{y})$$

Training Parametric Models

Analytical Solution – Normal equations

It can be shown that the gradient of J is as follows:

$$\nabla L(\theta) = \frac{1}{m}\left(X^T X\theta - X^T \mathbf{y}\right)$$

The gradient must disappear in a minimum of $L \rightarrow$ Necessary optimality condition:

Normal Equations Linear Least Squares Functional

$$X^T X\theta = X^T \mathbf{y}$$

Comments:

- The condition is a linear system of equations in the unknowns θ. If $\hat{\theta}$ is a solution of the system, then it is a critical point of the functional L.
- If the columns of the matrix X are linearly independent, the matrix $X^T X$ is invertible so that the normal equations are uniquely solvable:

$$\hat{\theta} = (X^T X)^{-1} X^T \mathbf{y}$$

Source: OTH-AW, Electrical Engineering, Media and Computer Science, Fabian Brunner – Vorlesung Deep Learning, Training parametrisierter ML-Modelle

Training Parametric Models

Analytical Solution – Normal equations

If $\hat{\theta}$ is a solution of the normal equations, there is initially only one critical point of L. To show that it is a (global) minimum of J, the Hessian matrix of L must be analyzed:

- The Hessian matrix of L is
$$H_L(\theta) = \frac{1}{m} X^T X \ .$$
- Obviously applies
$$\mathbf{x}^T X^T X \mathbf{x} = (X\mathbf{x})^T (X\mathbf{x}) = \|X\mathbf{x}\|_2^2 \geq 0 \text{ for all } \mathbf{x} \in \mathbb{R}^{p+1}$$
  i.e. H_L is positive semidefinite. Furthermore, if the columns of X are linearly independent, then even „>0“applies above, i.e. the matrix H_J is positive definite and therefore J is strictly convex
- At the point $\hat{\theta}$ there is therefore a clear global minimum of L

Source: OTH-AW, Electrical Engineering, Media and Computer Science, Fabian Brunner – Vorlesung Deep Learning, Training parametrisierter ML-Modelle

Training Parametric Models

Analytical Solution Linear Regression

Summary Linear Regression:

1. The parameters are determined by minimizing the least squares functional
2. The condition

$$\nabla L(\theta) = \mathbf{0}$$

   leads to a linear system of equations in the weights θ, the solution of which can be calculated explicitly:

$$\hat{\theta} = (X^T X)^{-1} X^T \mathbf{y}$$

3. The least squares functional is proven to be strictly convex, so that the explicit solution of the normal equations also represents the unique global minimum of L.

Source: OTH-AW, Electrical Engineering, Media and Computer Science, Fabian Brunner – Vorlesung Deep Learning, Training parametrisierter ML-Modelle

Training Parametric Models

Logistic regression

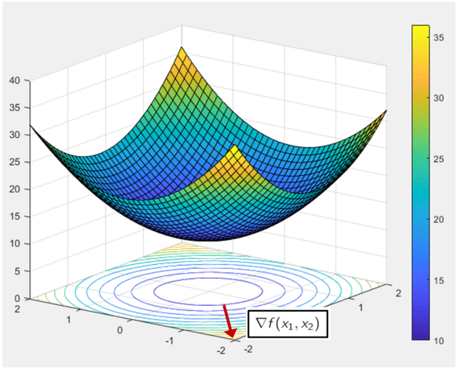
- The parameters are also determined by minimizing a functional, which may depend on the type of application
- The condition $\nabla L(\theta) = \mathbf{0}$ typically leads to a system of non-linear equations that can no longer be solved analytically. The solution would require an iterative approximation method (e.g. Newton method)
- Instead of solving a non-linear system of equations iteratively, a minimum of the functional is determined iteratively, e.g. using the gradient method
- In logistic regression, the error functional is still convex $\rightarrow$ Programming Excerise!
- With more complex neural networks, convexity is typically no longer present. This can result in local minima.

Source: OTH-AW, Electrical Engineering, Media and Computer Science, Fabian Brunner – Vorlesung Deep Learning, Training parametrisierter ML-Modelle

Training Parametric Models

Interpretation Gradient

- The gradient at a point $\mathbf{x} = (x_1, x_2)$ always points in the direction of the steepest incline of the function f:



Source: OTH-AW, Electrical Engineering, Media and Computer Science, Fabian Brunner – Vorlesung Deep Learning, Mathematische Grundlagen

Training Parametric Models

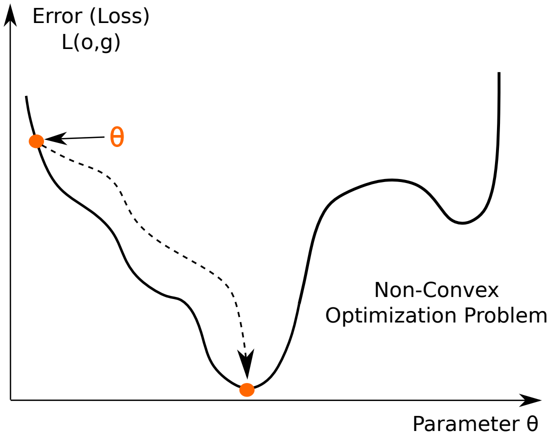
The Gradient Method

- In the case of linear regression, we were able to calculate the minimum of the least squares error functional analytically
- This is often not possible. Then approximation methods are used
- One of the most important approximation methods for approximating minima of real functions is the so-called **gradient method** or **gradient descent method**
- It is based on the fact that the gradient of a real function at a point always points in the direction of the strongest incline of the function at this point
- Starting from an initial value, a minimum is determined iteratively by adding a part (specified by the so-called **learning rate**) of the negative gradient as an update in each iteration step

Source: OTH-AW, Electrical Engineering, Media and Computer Science, Fabian Brunner – Vorlesung Deep Learning, Training parametrisierter ML-Modelle

Training Parametric Models

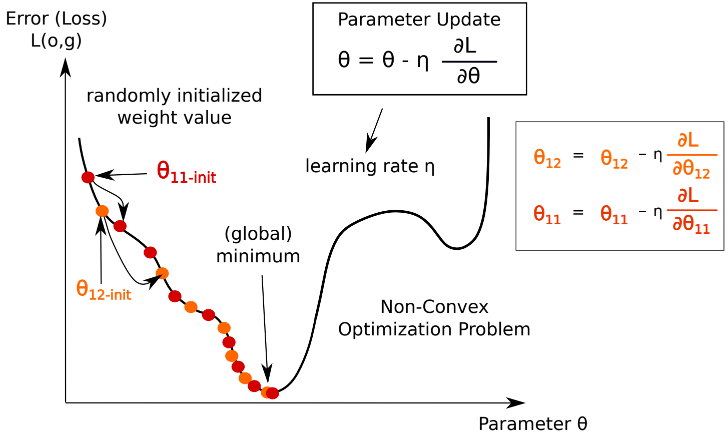
The Gradient Method



- How to find the minimum of function $\mathtt{f}(\theta)$ w.r.t. θ?

Training Parametric Models

The Gradient Method



$$\theta_{12} = \theta_{12} - \eta \frac{\partial L}{\partial \theta_{12}}$$

$$\theta_{11} = \theta_{11} - \eta \frac{\partial L}{\partial \theta_{11}}$$

- Goal: Gradient descent method to approach a minimum of L
- Attention: Non-convex functions can have many local minima (one global!!!)

Training Parametric Models

The Gradient Method

Ostbayerische Technische Hochschule Amberg-Weiden

Gradient Descent

- Choose a starting value $\mathbf{x}^0$ and a learning rate $\alpha > 0$.
- As long as the cancellation criterion is not met, add an update in the k-th step as follows

$$\mathbf{x}^k := \mathbf{x}^{k-1} - \alpha \nabla f(\mathbf{x}^{k-1}) \ .$$

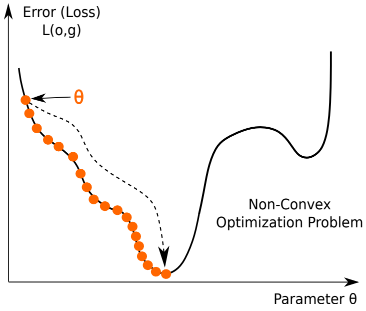
Comments:

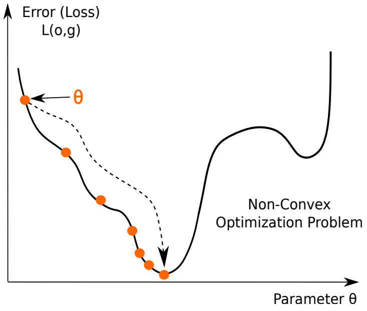
- **Stopping criterion:** the iteration is cancelled when a specified number of iterations is reached or when the update „is sufficiently small“ (e.g.: $\|\Delta \mathbf{x}^k\| < 10^{-6}$)
- **learning rate**: a learning rate that is too small means that potentially many iterations are required to reach the minimum. If the learning rate is too high, you may miss the target „“. Often the only thing that helps is to try out which learning rate works best

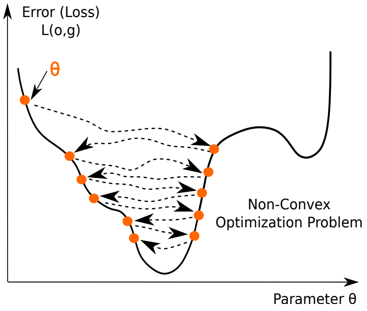
Source: OTH-AW, Electrical Engineering, Media and Computer Science, Fabian Brunner – Vorlesung Deep Learning, Training parametrisierter ML-Modelle

Training Parametric Models

The Gradient Method







- **Goal:** Gradient descent method together with a proper learning rate α
- **Question:** What type of learning rate scenarios are visualized?

Training Parametric Models

The Gradient Method for Linear Regression

In the case of linear regression, we had already calculated the gradient of the least squares functional:

$$\nabla L(\theta) = \frac{1}{m}(X^T X\theta - X^T \mathbf{y})$$

The update rule of the gradient method for this case is therefore

Gradient Method Least-Squares-Functional for Linear Regression

$$\theta^k = \theta^{k-1} - \frac{\alpha}{m}(X^T X\theta^{k-1} - X^T \mathbf{y})$$

Comment: The above representation allows an implementation of the gradient method in vectorised form

Source: OTH-AW, Electrical Engineering, Media and Computer Science, Fabian Brunner – Vorlesung Deep Learning, Training parametrisierter ML-Modelle

Training Parametric Models

Logistic Regression Model Approach

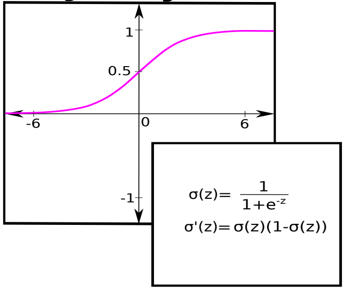
Model Function Logistic Regression

In logistic regression, the model function has the following form

$$f_\theta(\mathbf{x}) = \sigma(\theta^T \mathbf{x}) = \sigma(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \ldots + \theta_p x_p), \; \textit{with } \sigma(x) = \frac{1}{1 + e^{-x}}$$

- $z = \theta^T x + b \rightarrow \sigma(z) = \frac{1}{1+e^{-z}} = \frac{1}{1+e^{(\theta_0+\theta_1 x_1+\ldots+\theta_p x_p)}}$
- If the weighted sum (linear combination) is high, the probability should be close to 1, otherwise close to 0
- Sigmoid Function σ returns a probability $(0-1)$!
- Functional property: $1 - \sigma(z) = \sigma(-z)$

Logistic (Sigmoid σ)

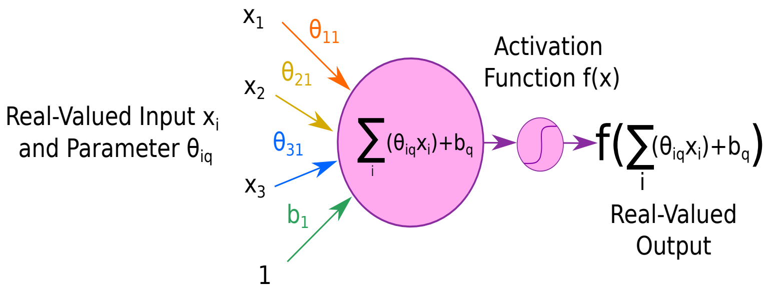


Training Parametric Models

Logistic Regression Model Approach

- Logistic regression can be understood as a neural network consisting of only a single artificial neuron with the activation function σ

Mathematical Representation of a Neuron



x_1
θ_{11}
x_2
θ_{21}
Real-Valued Input x_i and Parameter θ_{iq}
θ_{31}
x_3
b_1
1
$\sum_i (\theta_{iq} x_i) + b_q$
Activation Function f(x)
$f(\sum_i (\theta_{iq} x_i) + b_q)$
Real-Valued Output

- Hidden Neuron $h_1^{(1)} = \mathtt{f}(\theta_{11}x_1 + \theta_{21}x_2 + \theta_{31}x_3 + b_1)$

Training Parametric Models

Logistic Regression Model Training

Given: Data set with examples $(\mathbf{x}^{(1)}, y^{(1)}), \ldots, (\mathbf{x}^{(m)}, y^{(m)})$ where $\mathbf{x}^{(i)} \in \mathbb{R}^p$ contains the feature values and $y^{(i)} \in \{0, 1\}$ contains the labels.

Model Training

Determine the weights so that the cross-entropy error function is minimized

$$L(\theta) = -\frac{1}{m} \sum_{i=1}^{m} y^{(i)} \log\left(f_\theta(\mathbf{x}^{(i)})\right) + (1 - y^{(i)}) \log\left(1 - f_\theta(\mathbf{x}^{(i)})\right)$$

Question of Understanding: Why is not the least squares functional minimized?

$$L(\theta) = \frac{1}{2m} \sum_{i=1}^{m} (f_\theta(\mathbf{x}^{(i)}) - y^{(i)})^2$$

Source: OTH-AW, Electrical Engineering, Media and Computer Science, Fabian Brunner – Vorlesung Deep Learning, Training parametrisierter ML-Modelle

Training Parametric Models

Properties Logistic Regression

- For the logistic function: σ, $\sigma'(x) = \sigma(x)(1 - \sigma(x))$
- The loss function L in logistic regression is convex
- The gradient of L is

$$\nabla L(\theta) = \frac{1}{m} \sum_{i=1}^{m} (\sigma(\theta^T \cdot \mathbf{x}^{(i)}) - y^{(i)}) \mathbf{x}^{(i)} \ ,$$

or in vectorized form

$$\nabla L(\theta) = \frac{1}{m} X^T (\sigma(X\theta) - \mathbf{y}) \ .$$

Observation: The gradient ∇L depends non-linearly on θ!

Source: OTH-AW, Electrical Engineering, Media and Computer Science, Fabian Brunner – Vorlesung Deep Learning, Training parametrisierter ML-Modelle

Training Parametric Models

No Analytical Solution

Summary Logistic Regression

1. The parameters are determined by minimizing the cross-entropy loss function
2. The condition
$$\nabla L(\theta) = \mathbf{0}$$
leads to a non-linear system of equations with respect to the weights θ, while the solution of it cannot be explicitly calculated
3. As with linear regression, the cost function is convex, so there can be no local minima

Source: OTH-AW, Electrical Engineering, Media and Computer Science, Fabian Brunner – Vorlesung Deep Learning, Training parametrisierter ML-Modelle

Training Parametric Models
Summary and Outlook

Summary

- Principle of model training for parameterised ML models
- Gradient method
- Vectorization
- Examples: Linear regression and logistic regression

Outlook

- Linear neural networks
- Multi-class classification with softmax regression

Source: OTH-AW, Electrical Engineering, Media and Computer Science, Fabian Brunner – Vorlesung Deep Learning, Training parametrisierter ML-Modelle