

Linear Regression

Lecture *Machine Learning* vom 23.3.2022

Creating the data set &
defining the task

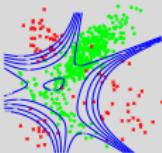
Choosing a model

Pre-processing the
data

Training and testing
the model

Gradient Descend

Lars Gabriel, Felix Becker, Mario Stanke
Institut für Mathematik und Informatik
Universität Greifswald



Creating the data set &
defining the task

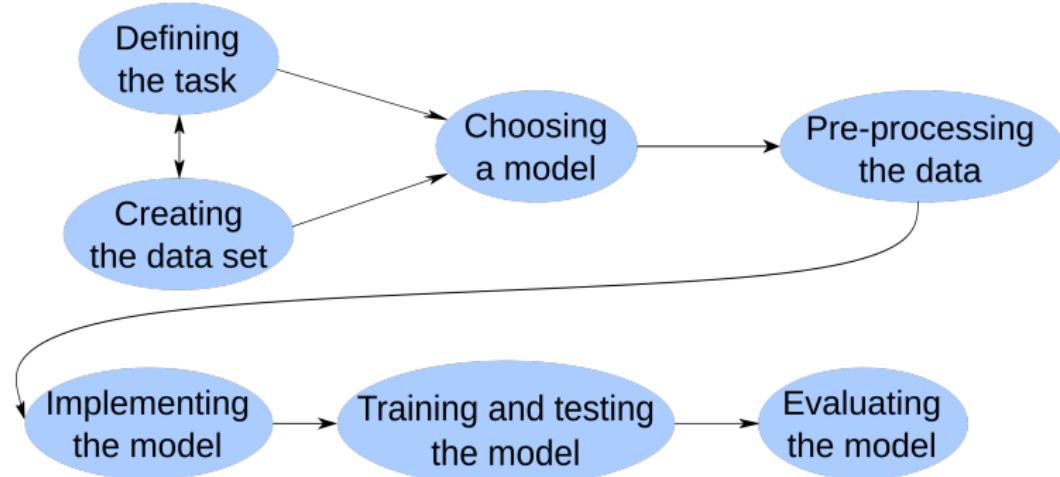
Choosing a model

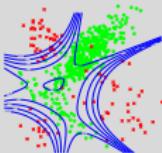
Pre-processing
the data

Training and testing
the model

Gradient Descend

Workflow



Creating the data set &
defining the task

Choosing a model

Pre-processing the
dataTraining and testing
the model

Gradient Descend

Bike Sharing Demand

Data set

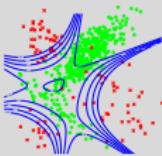
- hourly rental data for Washington bike sharing system
- data fields we will use:
 - *temp* - temperature in Celsius
 - *count* - number of total rents
- source:
www.kaggle.com/c/bike-sharing-demand

First steps

- read data from input file
- visualize data
- summarize data

Task

Predict the demand (count) of bikes based on the temperature

Creating the data set &
defining the task

Choosing a model

Pre-processing the
dataTraining and testing
the model

Gradient Descend

Choosing a model

Supervised learning

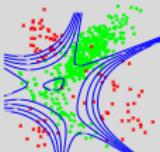
The parameter of a machine learning model are learned from a training set $x^{(1)}, x^{(2)}, \dots, x^{(n)}$ with known target variables (labels) $y^{(1)}, y^{(2)}, \dots, y^{(n)}$.

Classification: The target variable is categorical, i.e. an output is a discrete class label.

Regression: The target variable is continuous, i.e. an output is a numeric value.

Unsupervised learning

The parameter of a machine learning model are learned from a training set $x^{(1)}, x^{(2)}, \dots, x^{(n)}$ without known target variables.



Creating the data set &
defining the task

Choosing a model

Pre-processing the
data

Training and testing
the model

Gradient Descend

Linear regression

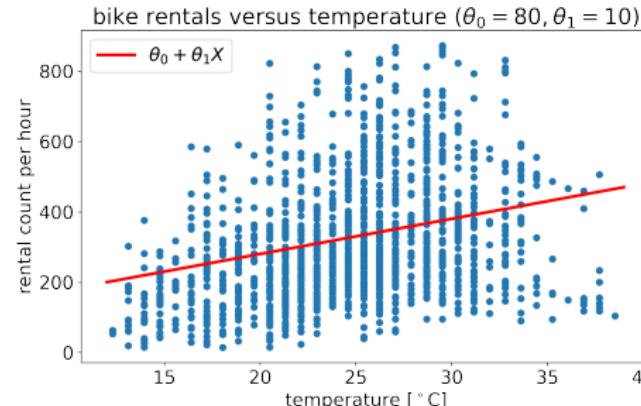
- input variables x_1, \dots, x_n
- output variable y
- linearity assumption:

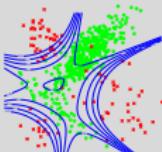
$$y = \theta_0 + \theta_1 x_1 + \dots + \theta_n x_n$$

- task: find $\theta_0, \dots, \theta_n$

Bike rental data set

- only one input variable
 x_1 : temperature
- y : number of rented bikes
- $y = \theta_0 + \theta_1 x_1$
- task: find θ_0, θ_1





Creating the data set &
defining the task

Choosing a model

Pre-processing the
data

Training and testing
the model

Gradient Descend

Hypothesis function

Data set

m training examples (input variables and output variable) with:

$$\mathbf{x}^{(i)} = (x_1^{(i)}, \dots, x_n^{(i)})^T, \quad y^{(i)}, \quad (i = 1, \dots, m)$$

Hypothesis function

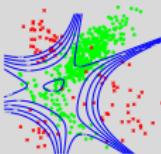
The task is to find $\theta = (\theta_0, \dots, \theta_n)^T$, such that:

$$y^{(i)} \approx \theta_0 + \theta_1 x_1^{(i)} + \dots + \theta_n x_n^{(i)} = \theta^T \begin{pmatrix} 1 \\ \mathbf{x}^{(i)} \end{pmatrix} =: h_{\theta}(\mathbf{x}^{(i)})$$

The function $h_{\theta}(\mathbf{x}^{(i)})$ is called the hypothesis.

Bike rental data set

$$h_{\theta}(\mathbf{x}^{(i)}) = (\theta_0, \theta_1) \begin{pmatrix} 1 \\ x_1^{(i)} \end{pmatrix}$$

Creating the data set &
defining the task

Choosing a model

Pre-processing the
dataTraining and testing
the model

Gradient Descend

Data matrix

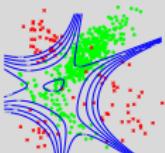
Data matrix

$$X := \begin{pmatrix} 1 & x_1^{(1)} & x_2^{(1)} & \cdots & x_n^{(1)} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_1^{(m)} & x_2^{(m)} & \cdots & x_n^{(m)} \end{pmatrix}$$

Hypothesis

The hypothesis for each training example can be computed with matrix multiplication:

$$\begin{pmatrix} h_{\theta}(x^{(1)}) \\ \vdots \\ h_{\theta}(x^{(m)}) \end{pmatrix} = \begin{pmatrix} 1 & x_1^{(1)} & x_2^{(1)} & \cdots & x_n^{(1)} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_1^{(m)} & x_2^{(m)} & \cdots & x_n^{(m)} \end{pmatrix} \begin{pmatrix} \theta_0 \\ \vdots \\ \theta_n \end{pmatrix} = X\theta$$



Creating the data set & defining the task

Choosing a model

Pre-processing the data

Training and testing the model

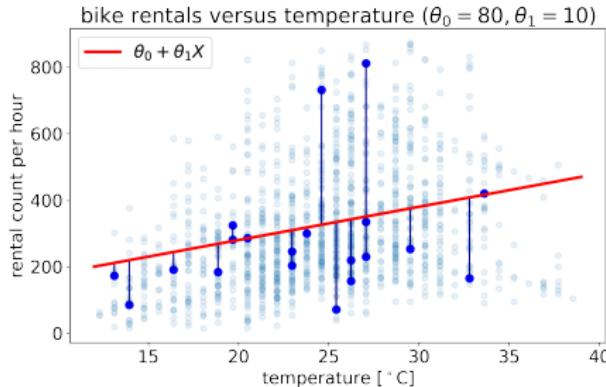
Gradient Descend

Loss Function

Mean squared error

Define the mean squared error function

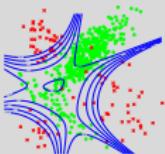
$$E(\theta) := \frac{1}{m} \sum_{i=1}^m (h_\theta(x^{(i)}) - y^{(i)})^2 = \frac{1}{m} (X\theta - y)^2.$$



Machine learning task

Find θ with minimal loss:

$$E(\theta) \rightarrow \min$$



Creating the data set &
defining the task

Choosing a model

Pre-processing the
data

Training and testing
the model

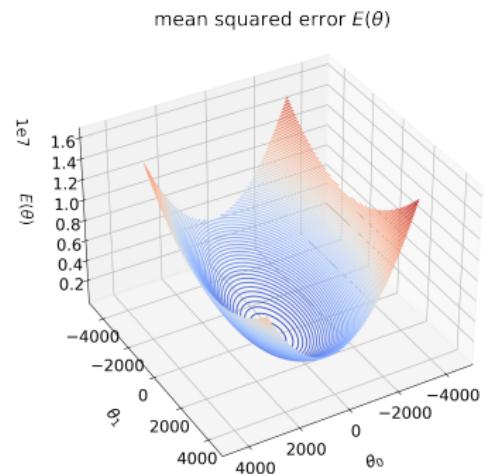
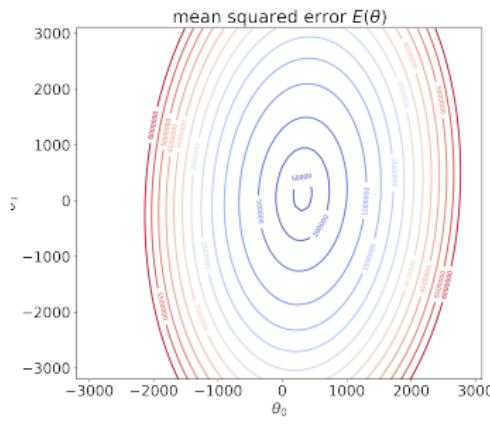
Gradient Descend

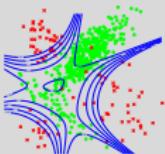
Gradient Descend

Machine learning task

Find θ with minimal loss:

$$E(\theta) \rightarrow \min$$





Creating the data set &
defining the task

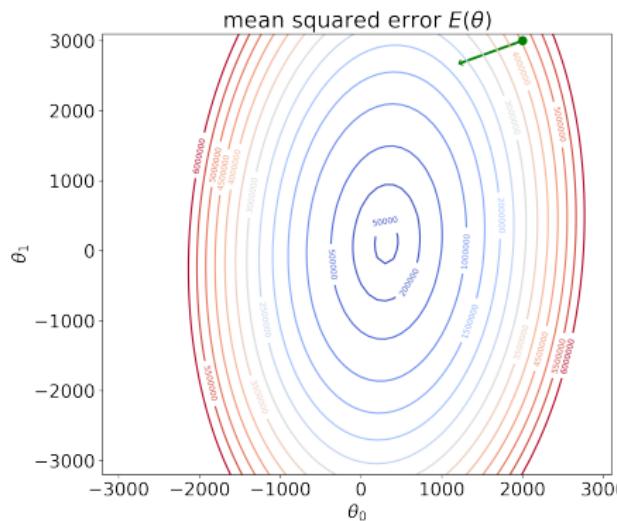
Choosing a model

Pre-processing the
data

Training and testing
the model

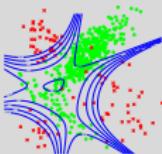
Gradient Descend

Gradient descend



Input

- $\hat{\theta}$: starting parameter
- α : learning rate



Creating the data set &
defining the task

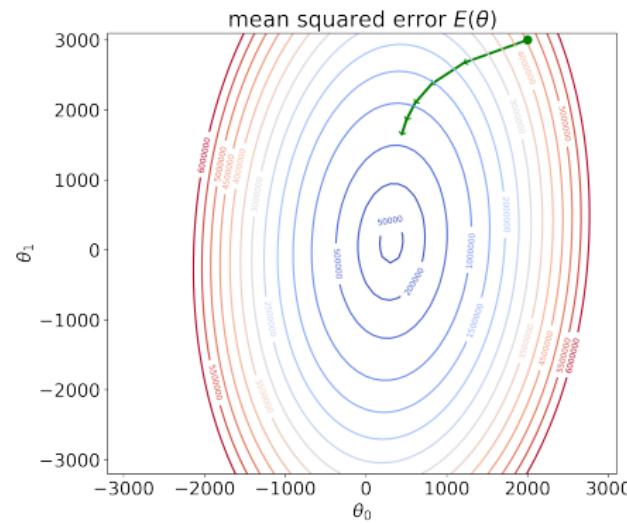
Choosing a model

Pre-processing the
data

Training and testing
the model

Gradient Descend

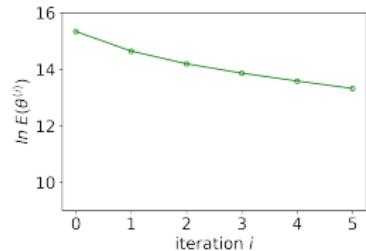
Gradient Descend

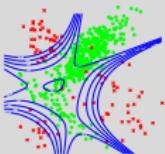


Input

- $\hat{\theta}$: starting parameter
- α : learning rate

Log error by iteration of gradient descent





Creating the data set &
defining the task

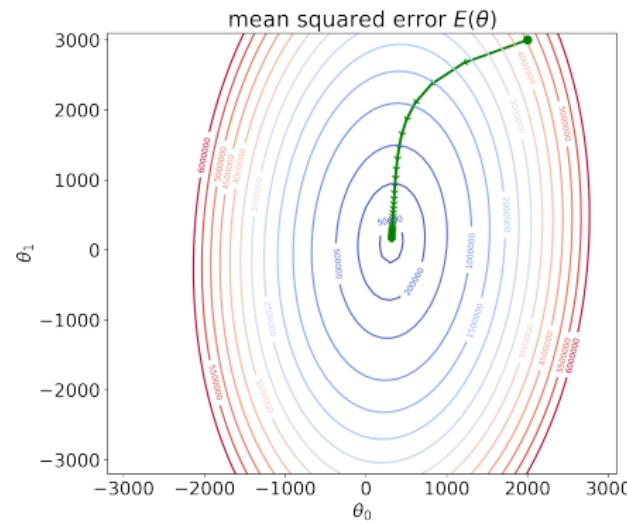
Choosing a model

Pre-processing the
data

Training and testing
the model

Gradient Descend

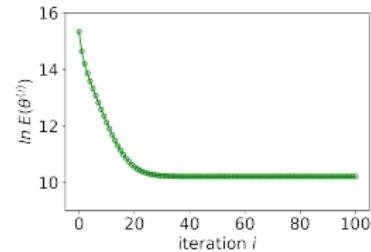
Gradient Descend

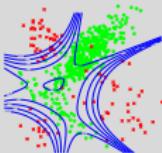


Input

- $\hat{\theta}$: starting parameter
- α : learning rate

Log error by iteration of gradient descent





Creating the data set &
defining the task

Choosing a model

Pre-processing the
data

Training and testing
the model

Gradient Descend

Gradient Descend

Parameter update

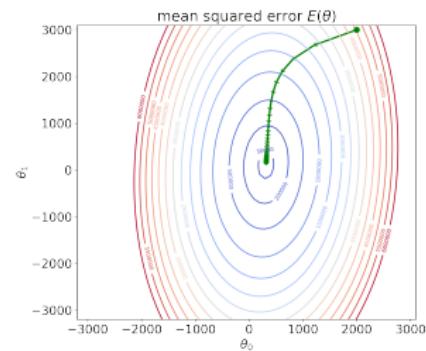
Parameters are iteratively updated, by moving each time a (small) step into the direction of the steepest descend of the target function to be minimized:

$$\theta \leftarrow \theta - \alpha \cdot \nabla E(\theta)$$

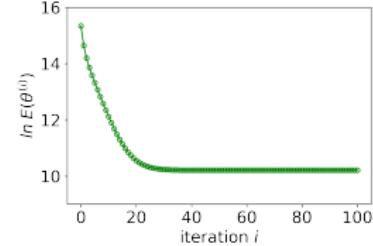
Here,

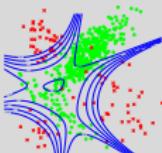
$$\nabla E(\theta) = \left(\frac{\partial E(\theta)}{\partial \theta_0}, \dots, \frac{\partial E(\theta)}{\partial \theta_n} \right)^T$$

is the gradient of the mean squared error function in the point θ and α is called the learning rate.



Log error by iteration of gradient descent





Creating the data set &
defining the task

Choosing a model

Pre-processing the
data

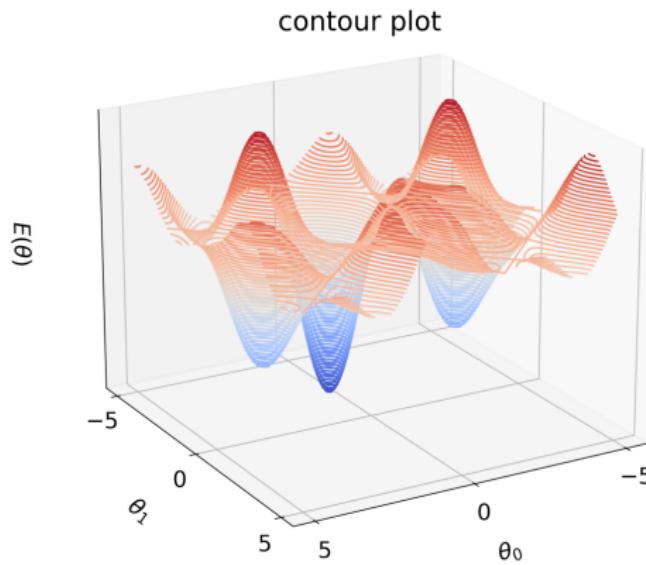
Training and testing
the model

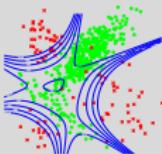
Gradient Descend

Gradient Descend

Disadvantages

- may converge to a local optimum
- can diverge, if the learning rate is too high
- X is used to compute the gradient in each step





Creating the data set & defining the task

Choosing a model

Pre-processing the data

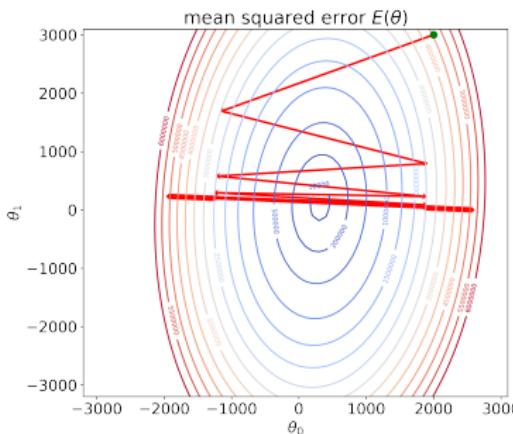
Training and testing the model

Gradient Descend

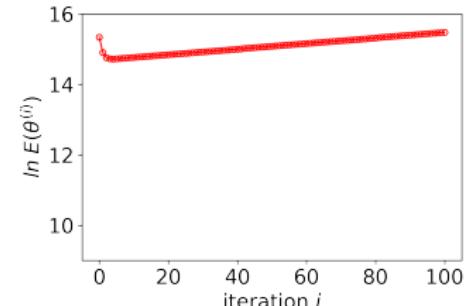
Gradient Descend

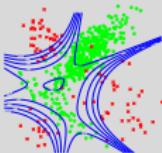
Disadvantages

- may converge to a local optimum
- can diverge, if the learning rate is too high
- X is used to compute the gradient in each step



Log error by iteration of gradient descent



Creating the data set &
defining the task

Choosing a model

Pre-processing the
dataTraining and testing
the model

Gradient Descend

Stochastic Gradient Descend

Stochastic Gradient Descend

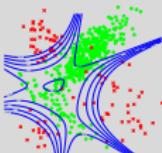
- minimizes the loss function $E(\theta; (X, y))$ by iteratively updating θ based on the gradient ∇E
- uses only a randomly sampled subset (X_{batch}, y_{batch}) (called **batch**) of the test data (X, y) for each update
- the **learning rate** α plays a similar role as in regular gradient descent
- $0 \leq \mu \leq 1$ is called the **momentum**

Algorithm

$$v \leftarrow 0$$

While the *termination condition* is not satisfied

- || Sample a random batch (X_{batch}, y_{batch}) from the training data (X, y)
- || $v \leftarrow \mu \cdot v - \alpha \cdot \nabla E(\theta; (X_{batch}, y_{batch}))$
- || $\theta \leftarrow \theta + v$



Creating the data set &
defining the task

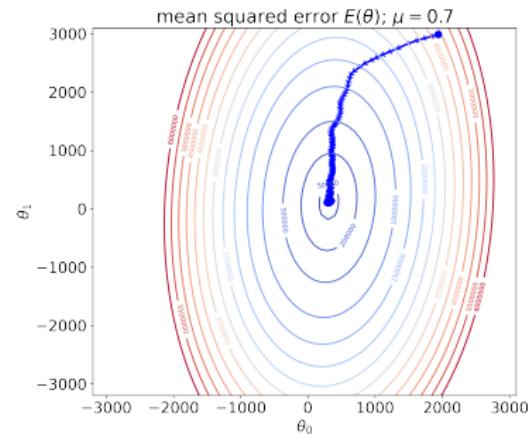
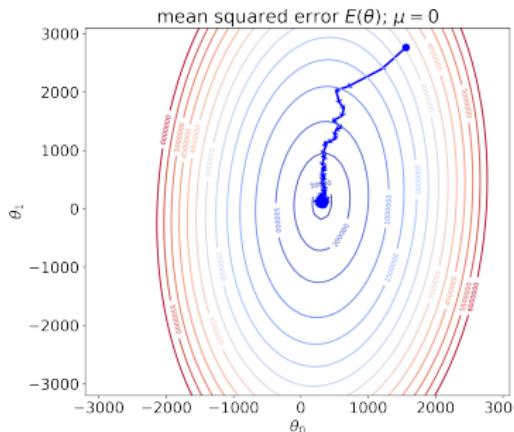
Choosing a model

Pre-processing the
data

Training and testing
the model

Gradient Descend

Stochastic Gradient Descend

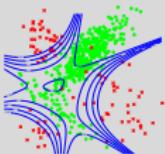


Algorithm

$$v \leftarrow 0$$

While the *termination condition* is not satisfied

- || Sample a random batch (X_{batch}, y_{batch}) from the training data (X, y)
- || $v \leftarrow \mu \cdot v - \alpha \cdot \nabla E(\theta; (X_{batch}, y_{batch}))$
- || $\theta \leftarrow \theta + v$



Creating the data set & defining the task

Choosing a model

Pre-processing the data

Training and testing the model

Gradient Descent

Stochastic Gradient Descent

Termination condition

- a fixed number of iterations has been done
 - the accuracy/error on a separate validation set satisfies some condition (e.g. it is plateauing)
 - user interruption

Algorithm

$V \leftarrow 0$

While the *termination condition* is not satisfied

Sample a random batch (X_{batch}, y_{batch}) from the training data (X, y)

$$v \leftarrow \mu \cdot v - \alpha \cdot \nabla E(\theta; (X_{batch}, y_{batch}))$$

$$\theta \leftarrow \theta + v$$