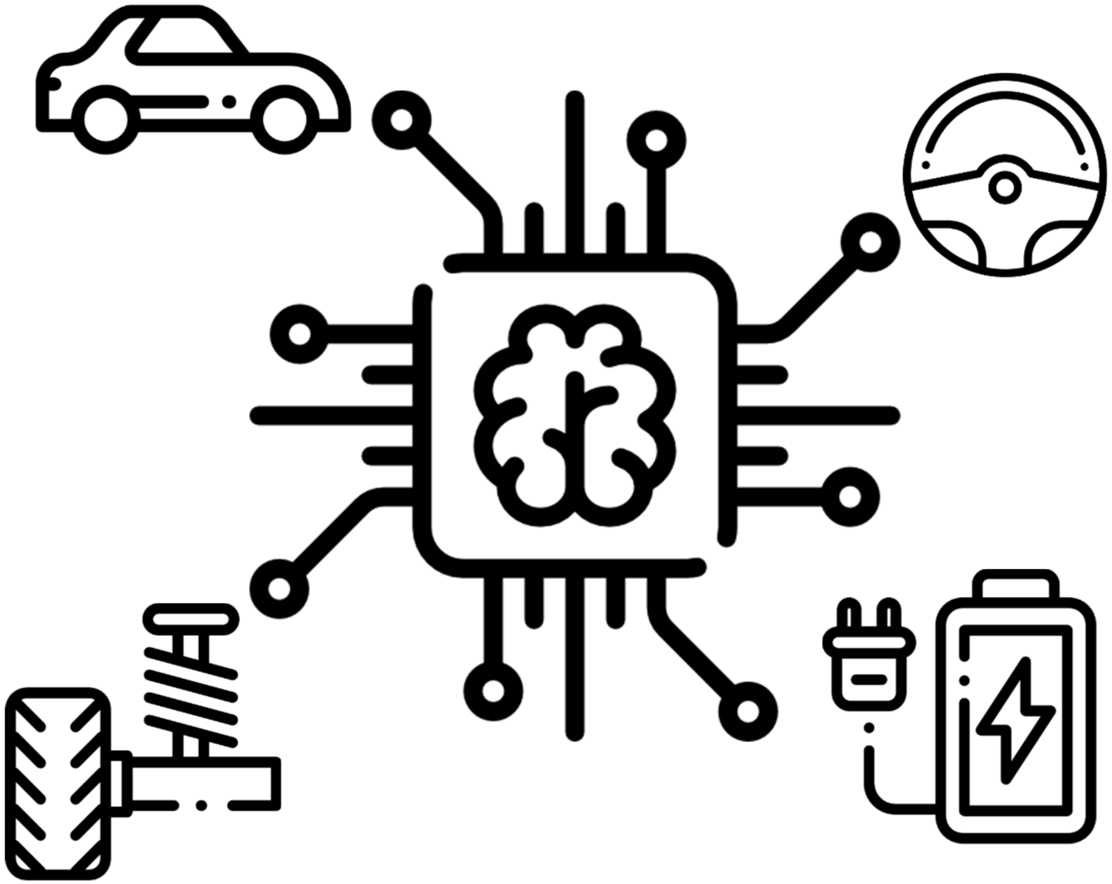
Chair of Automotive Technology
Department of Mechanical Engineering
Technical University of Munich

Artifical Intelligence in Automotive Technology

Johannes Betz / Prof. Dr.-Ing. Markus Lienkamp / Prof. Dr. Boris Lohmann



Feedback from last week

- Too much slides / Too much information in too little time

  → AI is a complex topic with different subjects

  → Overall lecture is not enough time (1.5h + Practice)

  → Use the summary and the learning results: Each slide is just for you to understand everything in total, but the focus of what you need to learn is clear based on learning results

  → **Trust us:** First lecture of his kind – We will combine learning results with exam questions

- Live Coding/ Practice Coding: the code could not be read on the beamer:

  → Uploading practice code and live code before the lecture

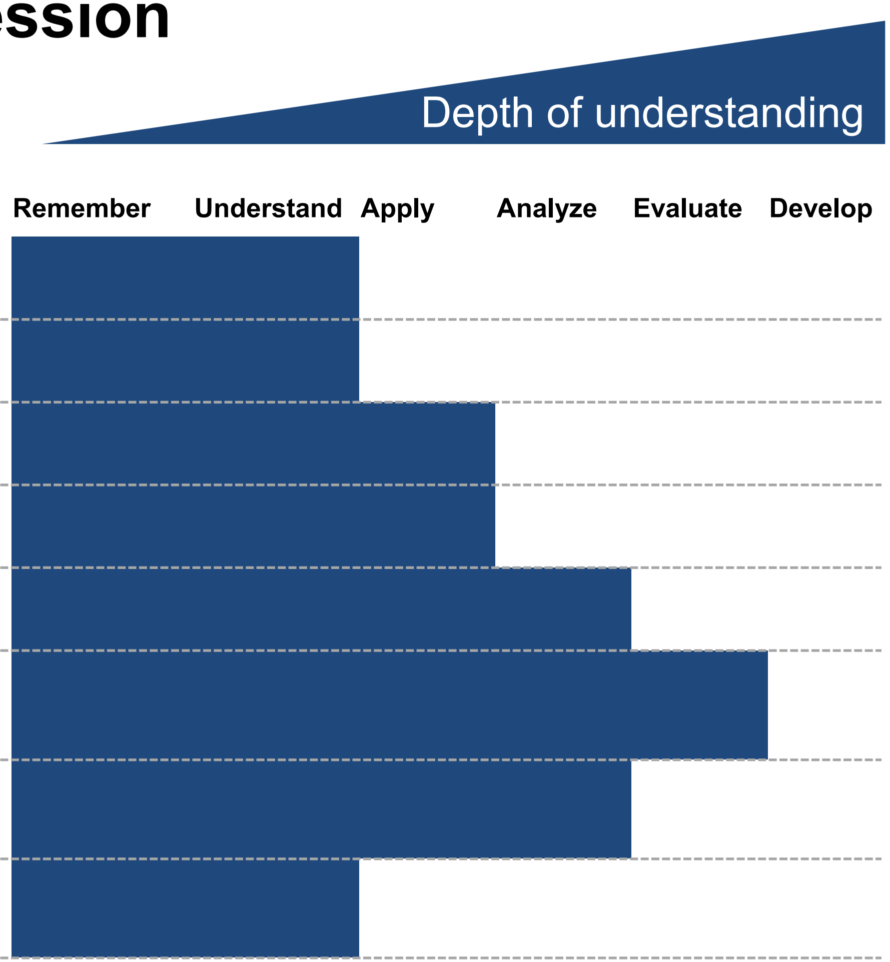
  → Use jupyter notebook for better explanation

Lecture Overview

Overall Introduction for the Lecture 18.10.2018 – Betz Johannes	**6 Pathfinding: From British Museum to A*** 29.11.2018 – Lennart Adenaw	**11 Reinforcement Learning** 17.01.2019 – Christian Dengler
1 Introduction: Artificial Intelligence 18.10.2018 – Betz Johannes	**P6:** 29.11.2018 – Lennart Adenaw	**P11** 17.01.2019 – Christian Dengler
P1: 18.10.2018 – Betz Johannes	**7 Introduction: Artificial Neural Networks** 06.12.2018 – Lennart Adenaw	**12 AI-Development** 24.01.2019 – Johannes Betz
2 Perception 25.10.2018 – Betz Johannes	**P7** 06.12.2018 – Lennart Adenaw	**P12** 24.01.2019 – Johannes Betz
P2: 25.10.2018 – Betz Johannes	**8 Deep Neural Networks** 13.12.2018 – Jean-Michael Georg	**13 Free Discussion** 31.01.2019 – Betz/Adenaw
3 Supervised Learning: Regression 08.11.2018 – Alexander Wischnewski	**P8** 13.12.2018 – Jean-Michael Georg	
P3: 08.11.2018 – Alexander Wischnewski	**9 Convolutional Neural Networks** 20.12.2018 – Jean-Michael Georg	
4 Supervised Learning: Classification 15.11.2018 – Jan-Cedric Mertens	**P9** 20.12.2018 – Jean-Michael Georg	
P4: 15.11.2018 – Jan-Cedric Mertens	**10 Recurrent Neural Networks** 10.01.2019 – Christian Dengler	
5 Unsupervised Learning: Clustering 22.11.2018 – Jan-Cedric Mertens	**P10** 10.01.2019 – Christian Dengler	

Objectives for Lecture 3: Regression

After the lecture you are able to…



… understand the definition of a Regression problem and know the relevant vocabulary.

…. understand in which areas of Automotive Technology Regression can be applied.

… know the strengths and weaknesses of different basis functions for applying a regression model

… choose the right loss function for training your model based on your dataset

…. analyze the result of the training process

…. apply and evaluate the usefulness of regularization techniques to control model complexity

…. analyze potential problems arising from real world datasets used in regression

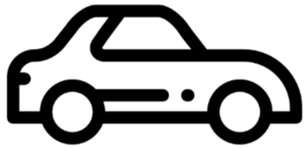
…. understand the basic mathematical principles behind regression

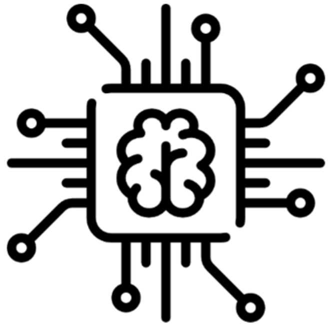
Supervised Learning: Regression

Johannes Betz / Prof. Dr. Markus Lienkamp / Prof. Dr. Boris Lohmann
(Alexander Wischnewski, M. Sc.)

Agenda

1. **Chapter: Motivation**
2. Chapter: Linear Models
3. Chapter: Loss functions
4. Chapter: Regularization & Validation
5. Chapter: Practical considerations
6. Chapter: Summary

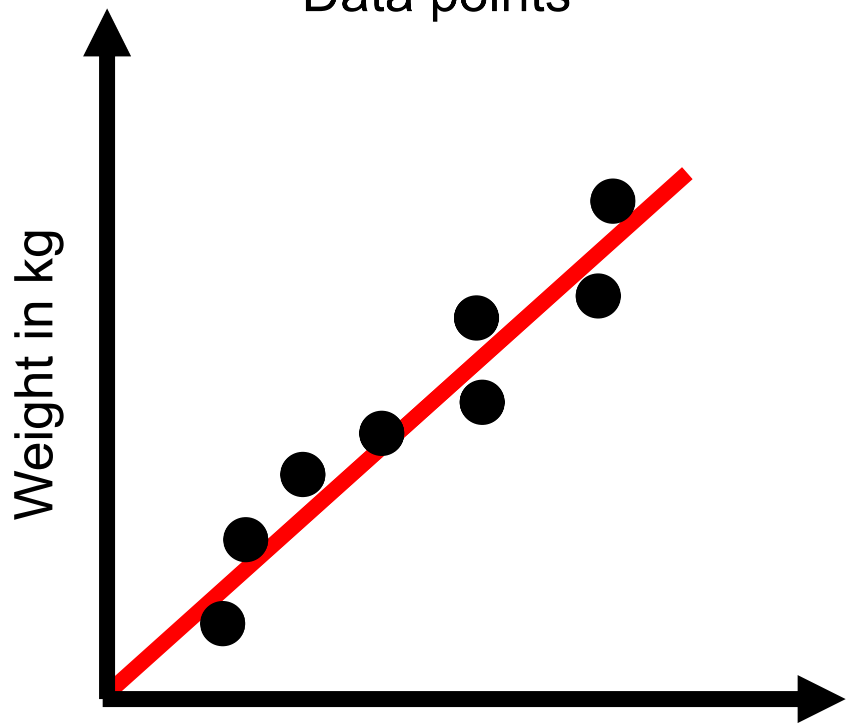




Motivation – Regression Example

Output variable
or Labels

Data points



Regression
result

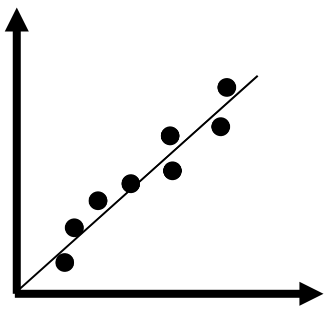
Input variables

Motivation – Algorithms in Machine Learning

Pattern Recognition

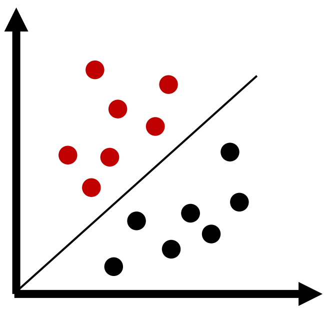
Regression

- Predict **continuous** valued output
- Supervised



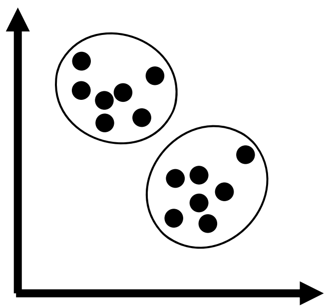
Classification

- Predict **discrete** valued output
- Supervised



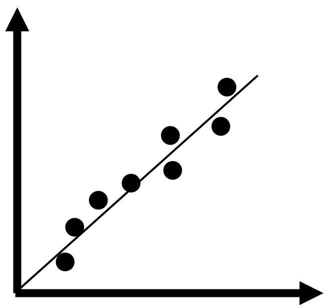
Clustering

- Predict discrete valued output
- **Unsupervised**



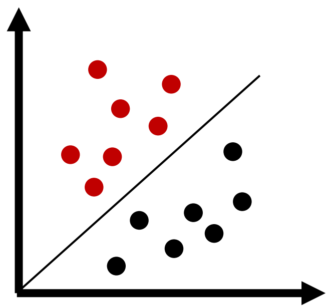
Motivation – Algorithms in Machine Learning

Regression



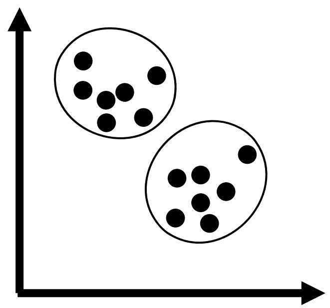
- House pricing
- Sales
- Persons weight

Classification



- Object detection
- Spam detection
- Cancer detection

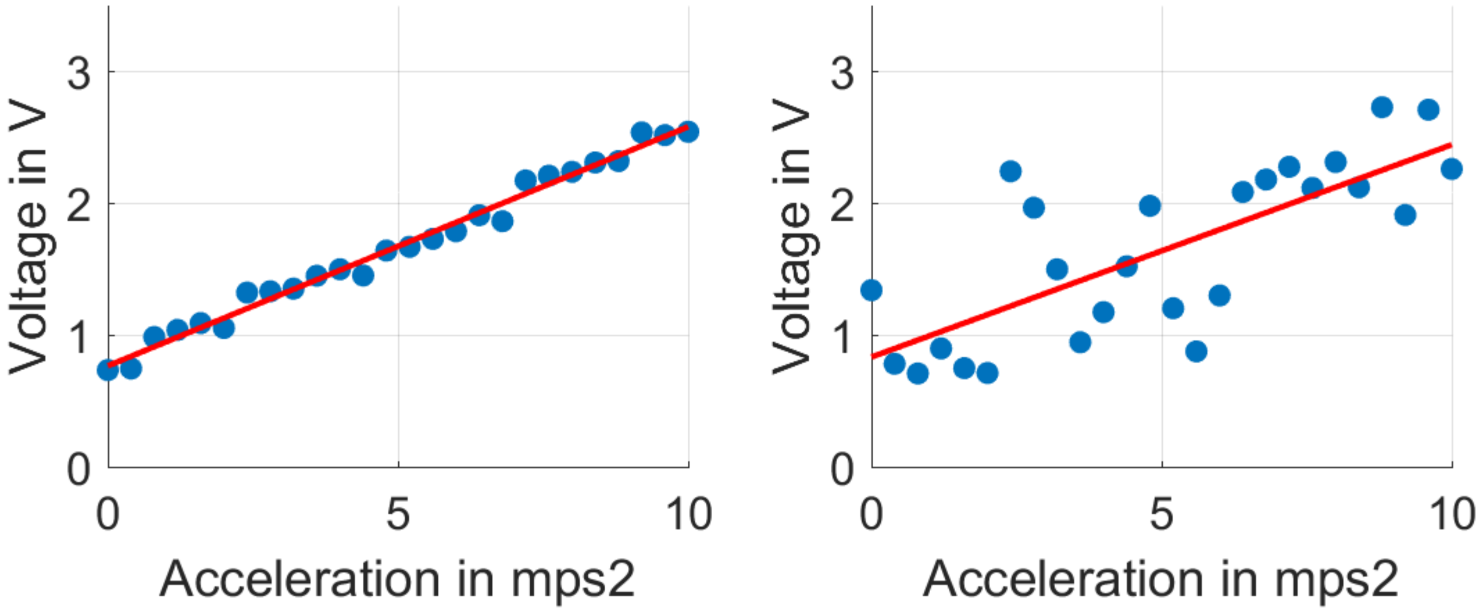
Clustering



- Genome patterns
- Google news
- Pointcloud (Lidar) processing

Motivation – Regression in Automotive Technology

Sensor calibration

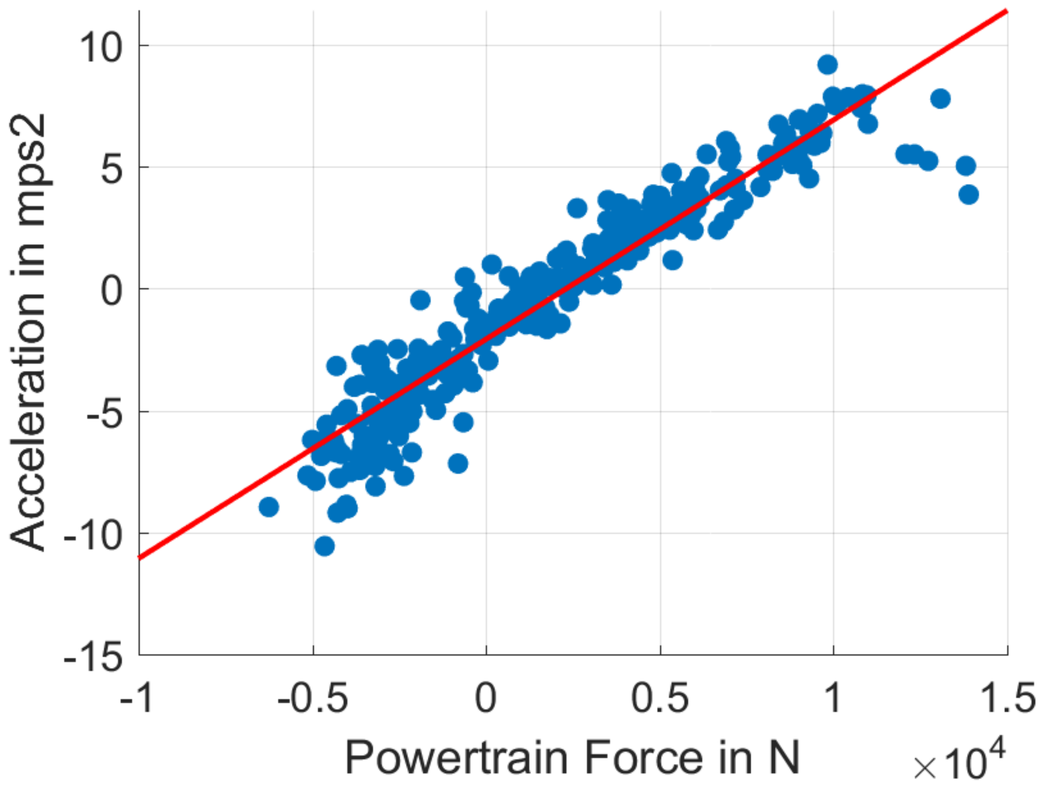


- Usually electric quantities are measured
- Necessary to convert them to physical quantities

- Examples:
  - Accelerometers
  - Gyroscopes
  - Displacement sensors

Motivation – Regression in Automotive Technology

Parameter estimation

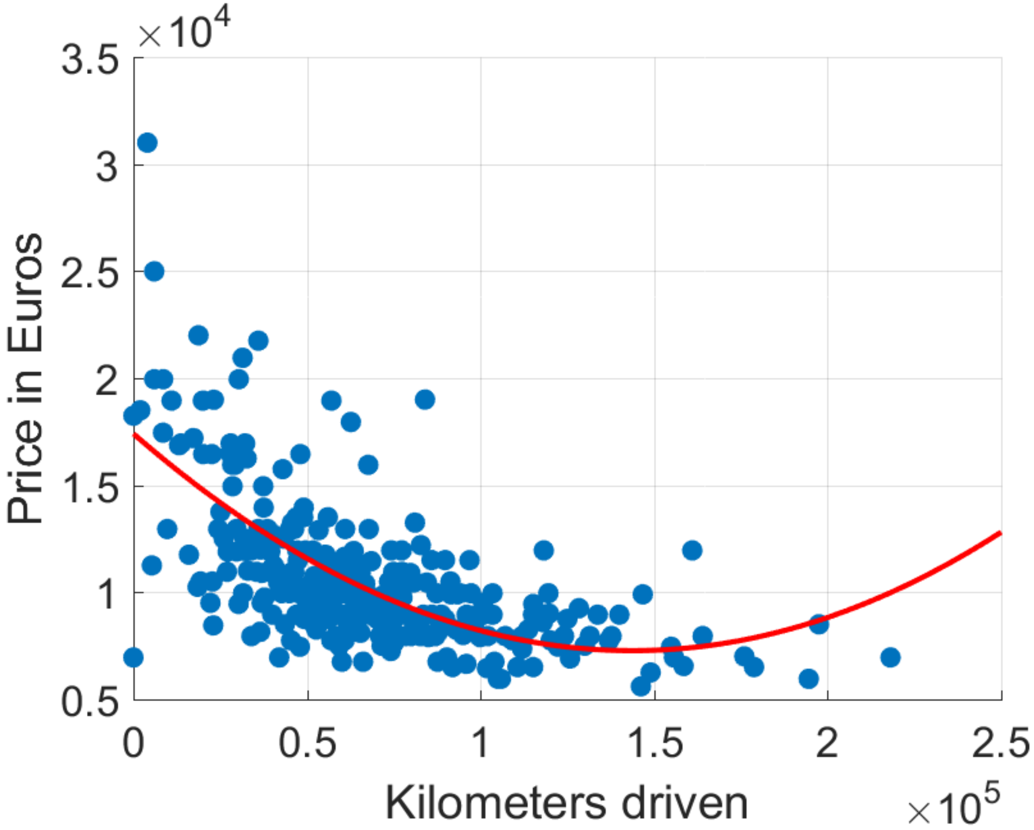


- Vehicle parameters like are often only roughly known
- Estimation via regression techniques

$$a_x = \frac{1}{m} F_{\mathrm{PT}}$$

Motivation – Regression in Automotive Technology

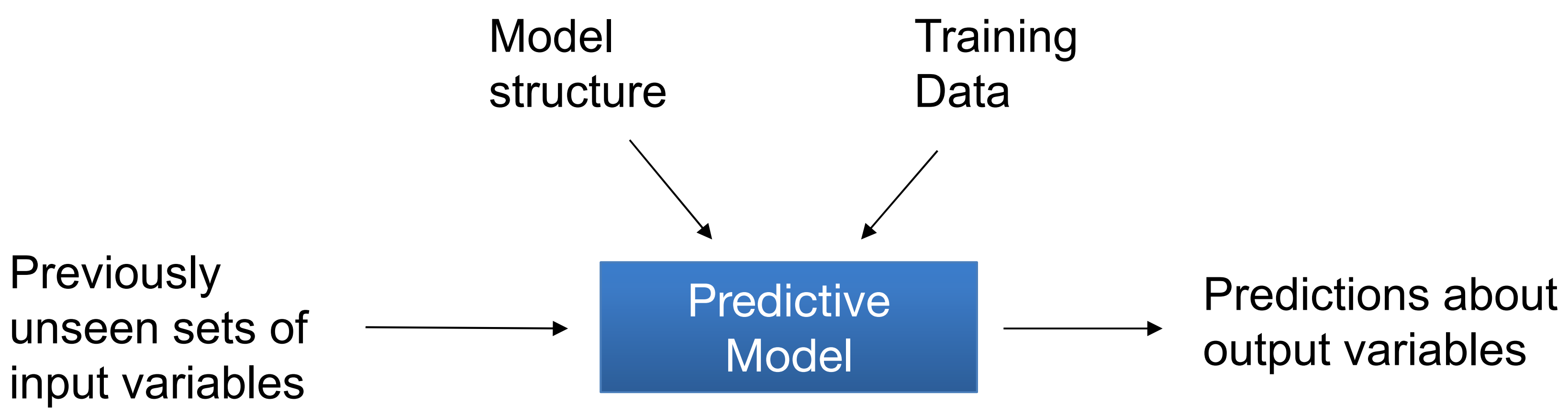
Vehicle pricing



- Regression is widely used for financial relations
- Allows to compress data into a simple model and evaluate derivatives

Motivation – Why should you use Regression?

- Based on the combination of data and model structure, it is possible to **predict the outcome** of a process or system
- Training data set is usually only a representation at sparse points and contains lots of noise
- Allows usage of information in simulation, optimization, etc.



Relation of statistics and machine learning

How can we extract information from data and use them to reason and predict in beforehand unseen cases? (learning)

- Nearly all classic machine learning methods can be reinterpreted in terms of statistics
- Focus in machine learning is mainly on prediction
- Statistics often focusses on relation analysis
- Lots of advanced regression techniques build upon a statistical interpretation of regression

Additional Information

Definition according to Wikipedia:

In statistical modeling, **regression analysis** is a set of statistical processes for estimating the relationships among variables. It includes many techniques for modeling and analyzing several variables, when the focus is on the relationship between a dependent variable and one or more independent variables (or 'predictors'). More specifically, regression analysis helps one understand how the typical value of the dependent variable (or 'criterion variable') changes when any one of the independent variables is varied, while the other independent variables are held fixed.

Source: https://en.wikipedia.org/wiki/Regression_analysis

Definition according to a Machine Learning Blog:

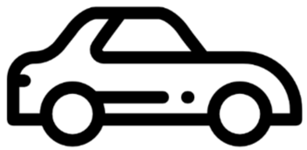
Regression is a method of modelling a target value based on independent predictors. This method is mostly used for forecasting and finding out cause and effect relationship between variables. Regression techniques mostly differ based on the number of independent variables and the type of relationship between the independent and dependent variables.

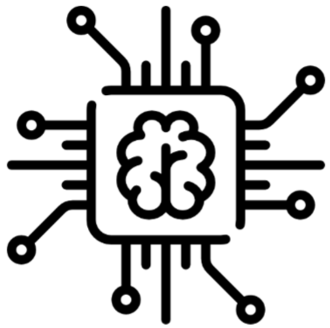
Source: https://towardsdatascience.com/introduction-to-machine-learning-algorithms-linear-regression-14c4e325882a

Supervised Learning: Regression

Johannes Betz / Prof. Dr. Markus Lienkamp / Prof. Dr. Boris Lohmann
(Alexander Wischnewski, M. Sc.)

Agenda





Linear Basis Function Model

Bias Term

Input Variables

$$y = b + \sum w_i \phi_i(x)$$

Output Variables

Weight Parameters

Basis Functions

$$y = \begin{bmatrix} 1 & \phi_1(x) & \dots & \phi_k \end{bmatrix} \begin{bmatrix} b \\ w_1 \\ \vdots \\ w_k \end{bmatrix}$$

Weight Parameters

Basis Functions

Representing the dataset as a matrix

$$\begin{bmatrix} y_1 \\ \vdots \\ y_N \end{bmatrix} = \begin{bmatrix} 1 & \phi_1(x_1) & \dots & \phi_k(x_1) \\ \vdots & \ddots & \ddots & \vdots \\ 1 & \phi_1(x_N) & \dots & \phi_k(x_N) \end{bmatrix} \begin{bmatrix} b \\ w_1 \\ \vdots \\ w_k \end{bmatrix}$$

$$\vec{y} = X\vec{w}$$

Output vector

Design Matrix

Weight vector

Nonlinear regression

Nonlinear optimization

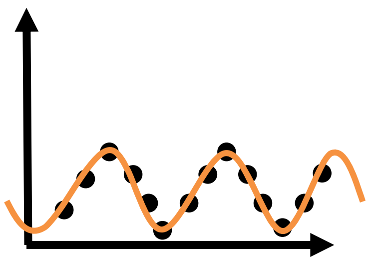
- Specify set of parameters
- Solve nonlinear optimization problem

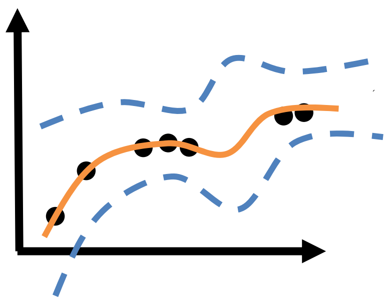
Gaussian Processes

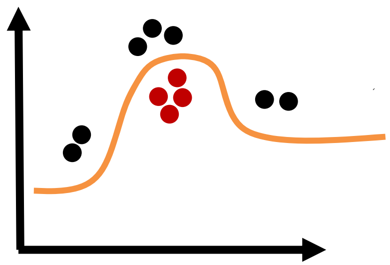
- Uses kernel methods
- No parameters
- Bayesian interpretation

Support Vector Machines

- Uses kernel methods
- No parameters
- Mostly used for classification







Workflow – How do we obtain model parameters?

1. Choose model

- Type of basis functions
- Number of basis functions

2. Find Parameters

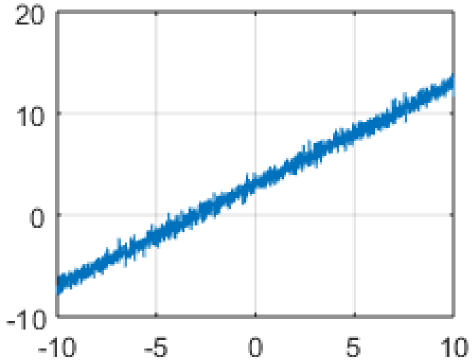
- Specify a **loss function** which measures „how good the model fits the data"
- Specify constraints on the parameters
- Use explicit solution or iterative algorithms to obtain parameters

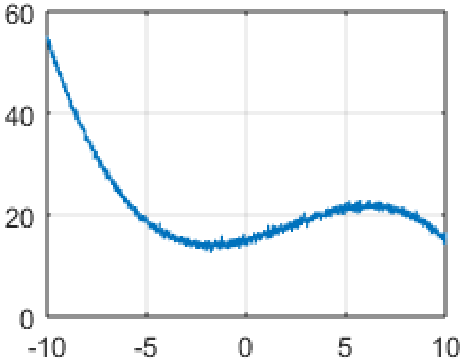
3. Validate Model

- Use a second dataset which was not used for training to evaluate the performance of your model
- This ensures the ability to generalize to unseen data

Basis functions – examples

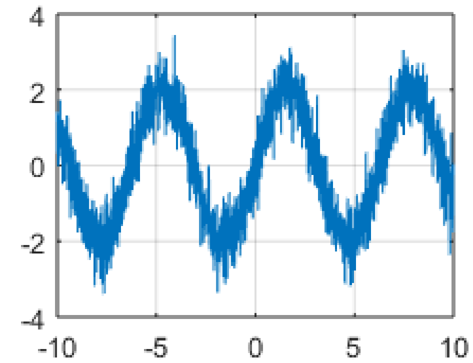
Linear
function

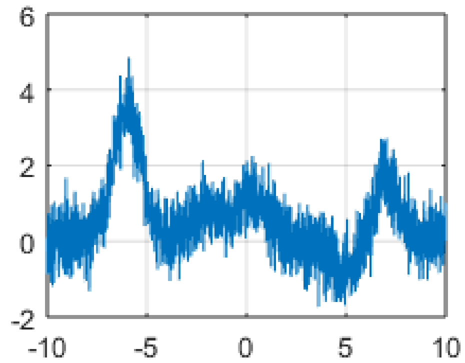




Polynomial
function

Sinusoidal
function

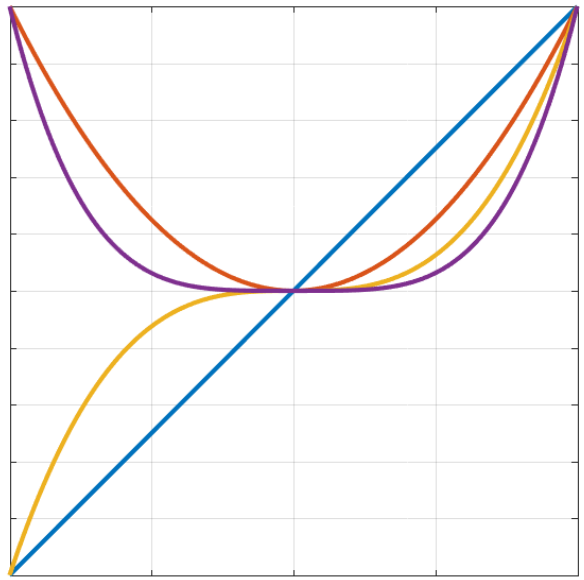




Gaussian
basis function

Basis functions – Polynomials

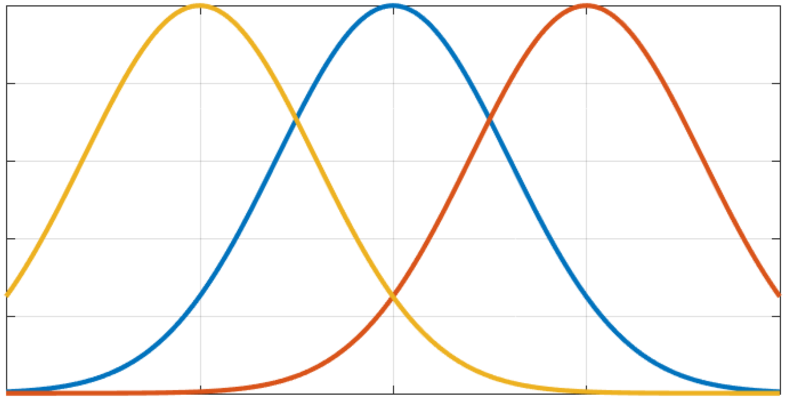
- Globally defined on the independent variable domain
- Design matrix becomes ill-conditioned for large input domain variables for standard polynomials
- Hyperparameter:
  - Polynomial degree



$$\phi_0(x) = 1 \qquad \phi_3(x) = x^3$$
$$\phi_1(x) = x \qquad \dots$$
$$\phi_2(x) = x^2 \qquad \phi_i(x) = x^i$$

Basis functions – Gaussians

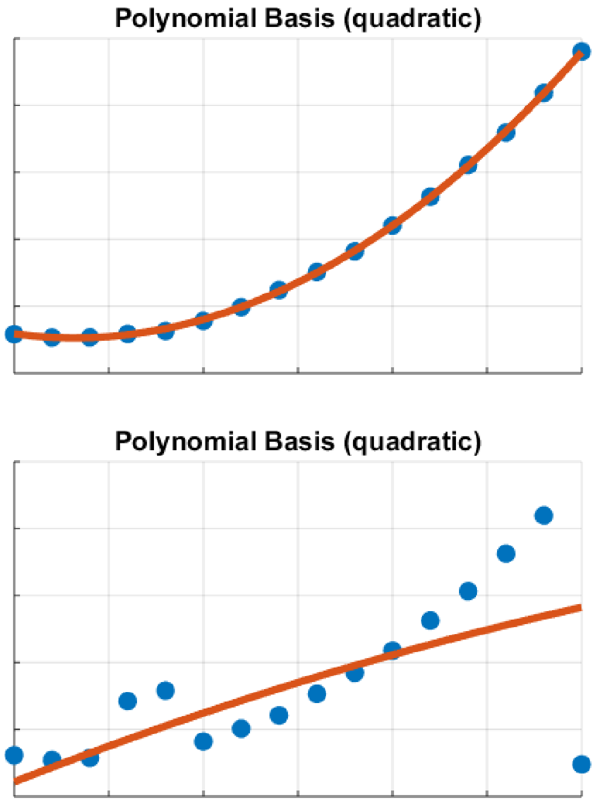
- Locally defined on the independent variable domain
- Sparse design matrix
- Infinitly differentiable
- Hyperparameter:
  - Number of Gaussian functions
  - Width s of each basis function
  - Mean μ of each basis function



$$\exp\left(\frac{-(x-\mu)^2}{2s^2}\right)$$

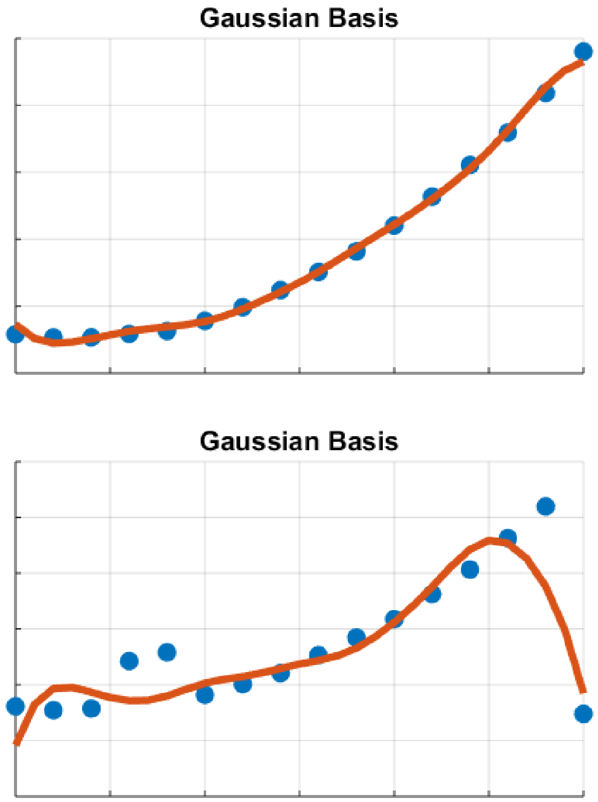
Basis functions – comparison of local and global

Global basis function



Local basis function

Spread parameter: 0.3



Additional Information

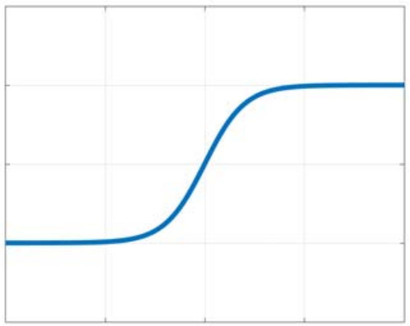
The previous slide compares the reaction of the regression to outliers if different basis functions are used. All examples are calculated using a quadratic loss function. Details on other possibilities follow in the upcoming slides.

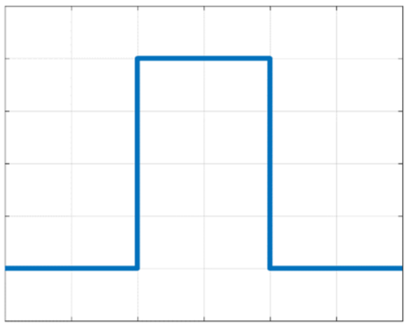
The upper plots are constructed based on slightly noisy observations of a quadratic polynomial function. The ground truth is $y = 3x^2 - 2x + 3$. The upper left plot uses a quadratic polynomial as regression model. Since this exactly matches the underlying generation model, the coefficients are estimated to 2.99, -1.96 and 2.96. The estimation quality is really good which is also reflected in the plot. The upper right plot uses seven gaussian basis function equally spaced along the relevant interval of independent variables. The prediction quality is only slightly worse than that of the polynomial model. However, note the little oscillations which are induced by the fact that the regression model does not perfectly fit the underlying model. This leads to local overfitting.

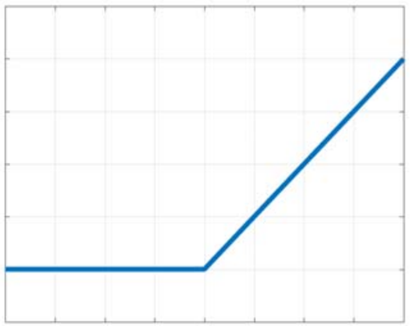
The lower plots contain four outliers. This heavily influences the polynomial model (right). Its characteristic does not fit the rest of the data at all. This is also reflected in the coefficients, which are estimated to -0.38, 5.55 and 0.97. Note that not even the coefficient signs are estimated correctly. As a consequence of the global nature of the basis functions, the model does not match the data over the complete interval. The gaussian model performs way better. It is only affected locally by the outliers and performs well on the rest of the intervall.

Basis functions – other

Sigmoid Functions	Bin functions	Piecewise linear
• Bounded • Globally defined	• Bounded • Locally defined • No continuity	• Not bounded • Globally defined







Additional Information

Background details on different basis functions:

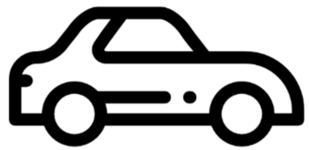
- http://madrury.github.io/jekyll/update/statistics/2017/08/04/basis-expansions.html

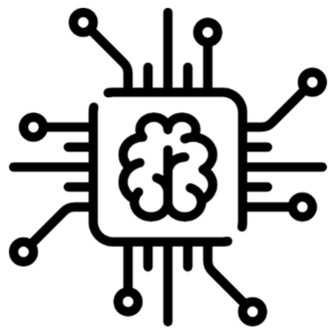
Supervised Learning: Regression

Johannes Betz / Prof. Dr. Markus Lienkamp / Prof. Dr. Boris Lohmann
(Alexander Wischnewski, M. Sc.)

Agenda

1. Chapter: Motivation
2. Chapter: Linear Models
3. **Chapter: Loss functions**
4. Chapter: Regularization & Validation
5. Chapter: Practical considerations
6. Chapter: Summary





Loss functions

- The loss functions measures the accuracy of the model based on the training dataset
- The best model we can obtain, is the minimum loss model

  → Choice of a loss function is fundamental in the regression problem

$$\underset{\vec{w}}{\text{minimize}}\ L(\vec{x}, \vec{y}, \vec{w})$$

→ Minimize the loss function L for the training dataset consisting of independent variables $\vec{x}$ and target variables $\vec{y}$ by variation of the basis function weights $\vec{w}$.

Loss functions – Mean Squared Error (MSE or L2)

Pro‘s:

- Very important in practical applications
- Solution can be easily obtained analytically

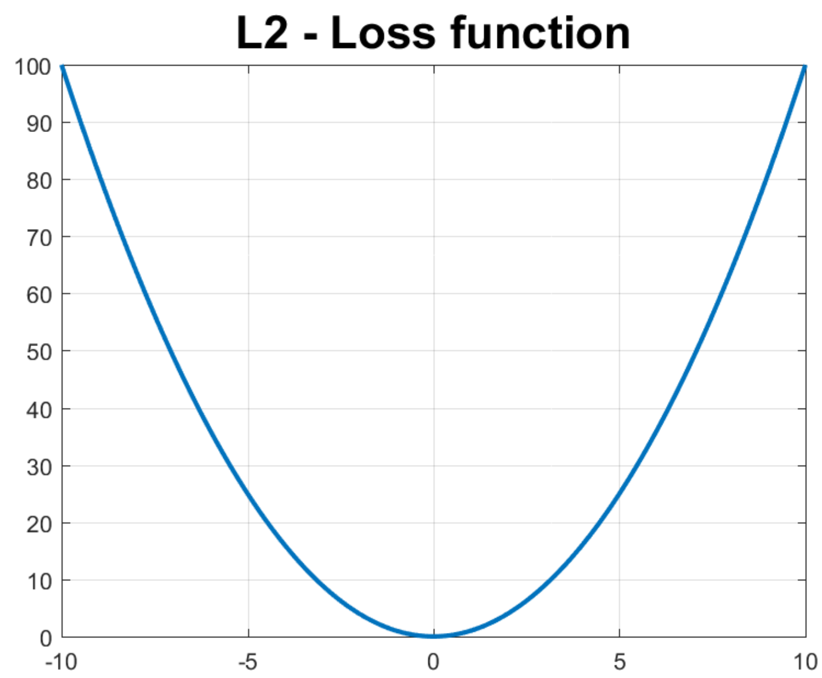
Con‘s:

- Not robust to outliers

Examples:

- Basic regression
- Energy optimization
- Control applications

$$L = \frac{1}{N} \sum (y - \hat{y})^2$$



Loss functions – Mean Absolute Error (MAE or L1)

$$L = \frac{1}{N} \sum | y - \hat{y} |$$

Pro‘s:

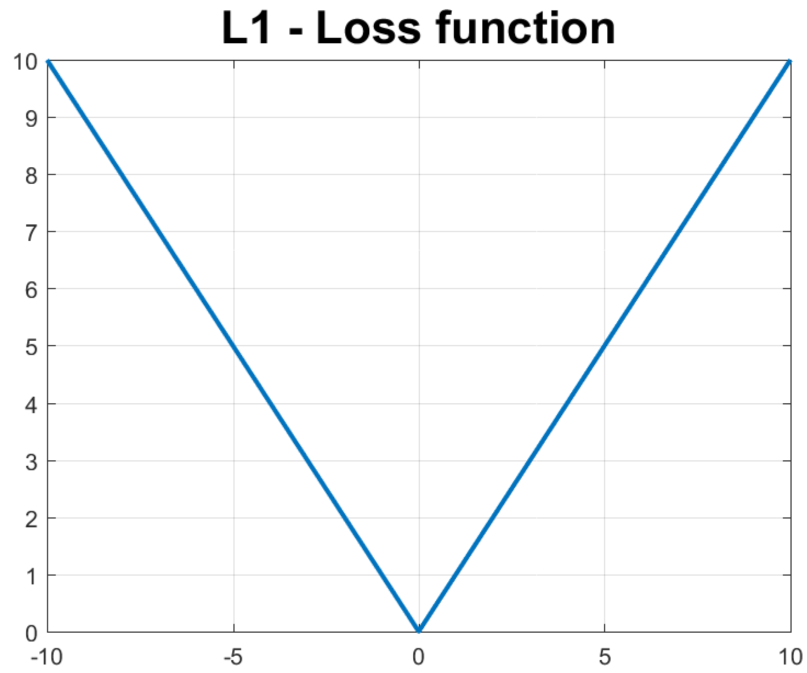
- Robust to outliers

Con‘s:

- No analytical solution
- Non-differentiable in the origin

Examples:

- Financial applications



Loss functions – Huber Loss

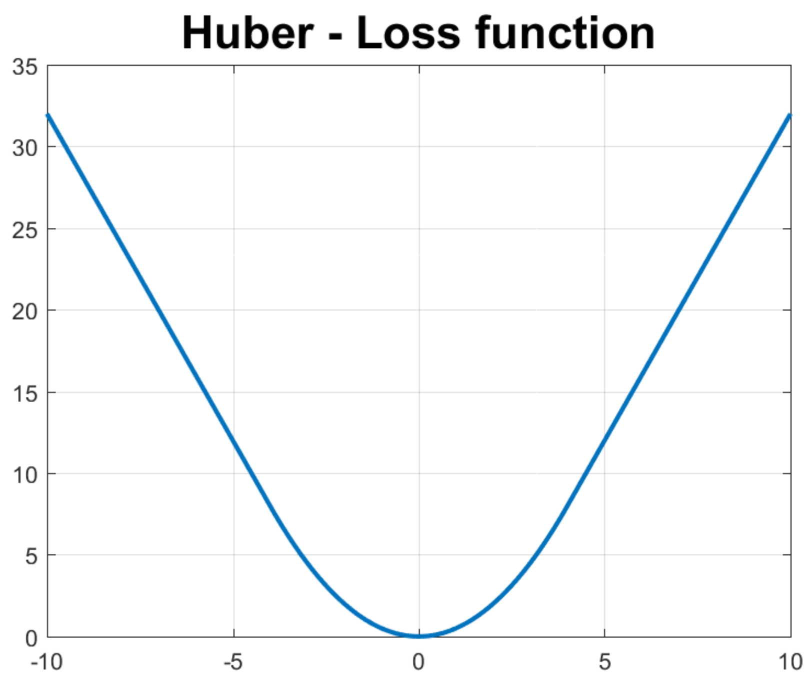
$$L = \sum \begin{cases} 0.5\,(y-\hat{y})^2\,, & \text{for } \mid y-\hat{y} \mid \leq \delta \\ \delta \mid y-\hat{y} \mid -0.5\delta^2, & \text{otherwise} \end{cases}$$

Pro‘s:

- Combines strengths and weaknesses of L1 and L2 loss functions
- Robust + differentiable

Con‘s:

- More hyperparameters
- No analytical solution

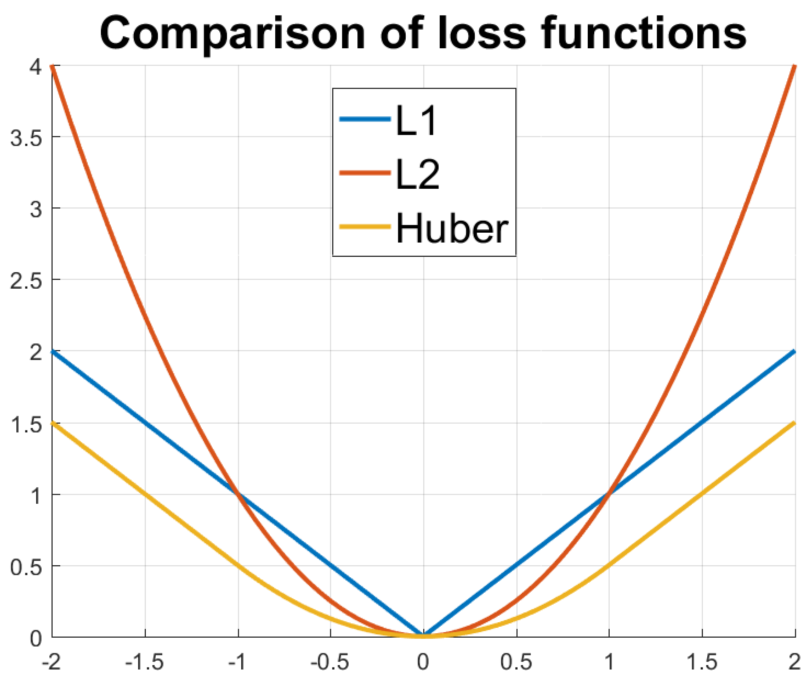


Loss functions – Comparison

- L2 loss is differentiable
- L1 loss is more intuitive
- Huber Loss combines theoretical strengths of both

Practical hints:

- Start with L2 loss whenever possible
- Think about physical insights and your intent!



Additional Information

Additional material on loss functions:

- Good overview and basic discussion of pro‘s and con‘s of different loss functions

https://heartbeat.fritz.ai/5-regression-loss-functions-all-machine-learners-should-know-4fb140e9d4b0

In detail comparison of MSE and MAE loss functions and their robustness

http://rishy.github.io/ml/2015/07/28/l1-vs-l2-loss/

Analytic Solution – Low dimensional example

Solve the optimization problem

$$\min_{\vec{w}} \frac{1}{2} \sum (y - \hat{y})^2$$

with the model $\hat{y} = w_1 x + b$

Insert the model and data points $\{x_1, y_1\}, \{x_2, y_2\}, \{x_3, y_3\}$

$$\min_{w_1, b} \frac{1}{2} \left[(y_1 - w_1 x_1 - b)^2 + (y_2 - w_1 x_2 - b)^2 + (y_3 - w_1 x_3 - b)^2 \right]$$

In general, optimal solutions are obtained at the points where the gradient vanishes to zero.

Analytic Solution – Low dimensional example

Calculate the gradient

$$\Delta L(x,y,w) = \begin{bmatrix} \frac{\partial L(x,y,w)}{\partial w_1} \\ \frac{\partial L(x,y,w)}{\partial b} \end{bmatrix}$$

$$= \begin{bmatrix} -\left(y_1 - w_1 x_1 - b\right) x_1 - \left(y_2 - w_1 x_2 - b\right) x_2 - \left(y_3 - w_1 x_3 - b\right) x_3 \\ -\left(y_1 - w_1 x_1 - b\right) - \left(y_2 - w_1 x_2 - b\right) - \left(y_3 - w_1 x_3 - b\right) \end{bmatrix}$$

$$= \begin{bmatrix} x_1^2 + x_2^2 + x_3^2 & x_1 + x_2 + x_3 \\ x_1 + x_2 + x_3 & 3 \end{bmatrix} \begin{bmatrix} w_1 \\ b \end{bmatrix} - \begin{bmatrix} y_1 x_1 + y_2 x_2 + y_3 x_3 \\ y_1 + y_2 + y_3 \end{bmatrix}$$

and set it equal to zero

$$\begin{bmatrix} x_1^2 + x_2^2 + x_3^2 & x_1 + x_2 + x_3 \\ x_1 + x_2 + x_3 & 3 \end{bmatrix} \begin{bmatrix} w_1 \\ b \end{bmatrix} - \begin{bmatrix} y_1 x_1 + y_2 x_2 + y_3 x_3 \\ y_1 + y_2 + y_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

Analytic Solution – Low dimensional example

Solve the resulting equation (also called normal equation):

$$\begin{bmatrix} x_1^2 + x_2^2 + x_3^2 & x_1 + x_2 + x_3 \\ x_1 + x_2 + x_3 & 3 \end{bmatrix} \begin{bmatrix} w_1 \\ b \end{bmatrix} - \begin{bmatrix} y_1 x_1 + y_2 x_2 + y_3 x_3 \\ y_1 + y_2 + y_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} w_1 \\ b \end{bmatrix} = \begin{bmatrix} x_1^2 + x_2^2 + x_3^2 & x_1 + x_2 + x_3 \\ x_1 + x_2 + x_3 & 3 \end{bmatrix}^{-1} \begin{bmatrix} y_1 x_1 + y_2 x_2 + y_3 x_3 \\ y_1 + y_2 + y_3 \end{bmatrix}$$

Analytic Solution – General form

- Minimizing MSE loss function can be rewritten in matrix form

$$\min_{\vec{w}} \frac{1}{2} \sum (y - \hat{y})^2$$

$$\min_{\vec{w}} \frac{1}{2} (\vec{y} - X\vec{w})^T (\vec{y} - X\vec{w})$$

- Optimum value for *w* is equal to setting the gradient to zero and solve for

$$\vec{w} = (X^T X)^{-1} X^T \vec{y}$$

- The importance of this loss function is tightly related to the fact that the analytical solution is available and can be calculated explicitly for low- to medium sized datasets!

Additional Information

Detailed derivation of least squares solution

$$\min_{\vec{w}} L(\vec{x}, \vec{y}, \vec{w})$$

$$\min_{\vec{w}} \frac{1}{2} (\vec{y} - X\vec{w})^T (\vec{y} - X\vec{w})$$

$$\min_{w} \frac{1}{2} \left(\vec{y}^T \vec{y} - 2\vec{w}^T X^T \vec{y} + \vec{w}^T X^T X \vec{w}\right)$$

Taking the gradient with respect to w

$$\Delta_{\vec{w}} L(\vec{x}, \vec{y}, \vec{w}) = -2X^T \vec{y} + 2X^T X \vec{w}$$

From the necessary condition for a minimum, it follows that

$$-2X^T \vec{y} + 2X^T X \vec{w} = 0$$

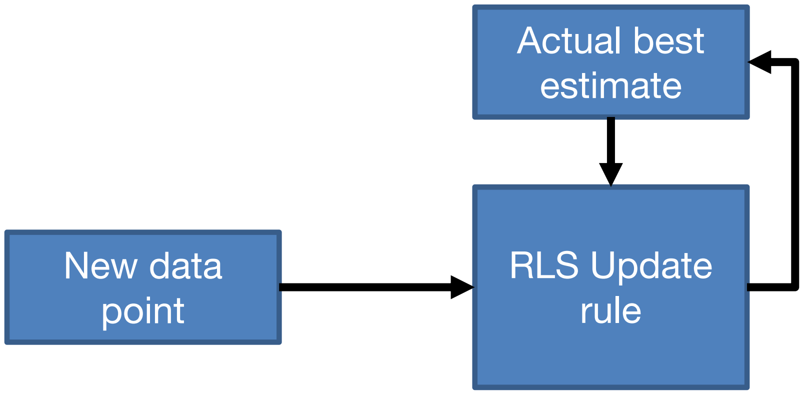
$$\vec{w} = \left(X^T X\right)^{-1} X^T \vec{y}$$

The term $\left(X^T X\right)^{-1} X^T$ is also called the Moore-Penrose pseudo-inverse. It is also the MSE solution of an overdetermined system of equations of the form $\vec{b} = A\vec{x}$ with A being a tall, full-rank matrix. Note the equivalence of the problem formulation using the Data Matrix before.

Sequential Analytic Solution - Motivation

- Consider the following cases:
  - Apply regression during operation of the product
  - There is not enough memory to store all data points

- A possible solution is given by Recursive Least Squares (RLS)



Sequential Analytic Solution – The algorithm

- New data point: $\{\bar{x}, \bar{y}\}$
- Update the parameters

Prediction based on old parameters

$$\vec{w}(k+1) = \underbrace{\vec{w}(k)}_{\text{Old parameter estimate}} + \underbrace{P(k)\bar{x}^T \left(I + \bar{x}^T P(k)\bar{x}\right)^{-1}}_{\text{Correction gain}} \underbrace{\left(\bar{y} - \bar{x}^T \vec{w}(k)\right)}_{\text{Residual}}$$

- And the memory matrix P

$$P(k+1) = \left(I - P(k)\bar{x}^T \left(I + \bar{x}^T P(k)\bar{x}\right)^{-1} \bar{x}\right) P(k)$$

- with I being the identity matrix of appropriate dimension

Additional Information

Please note the following facts:

The initialization of this algorithm is a crucial part in its application. There are two major possibilities:

- Initialize by using the first datapoints available
- Initialize using values which you assume to be right, e.g. obtained from first principles

The first technique leads to the solution of the ordinate least squares problem. The second technique however, is tighly related to the regularization techniques already discussed. If one initializes the parameters as follows

$$P(0) = \lambda^{-1} I$$
$$w(0) = 0$$

the resulting solution $w(k)$ is equivalent to the solution of the L2 regularized least squares problem. Moreover, you have the possibility to set a *starting value*. This can be helpful if a first principles model is available which should be refined online.

Moreover, the presented algorithm is tightly connected to the Kalman Filter algorithm. In this case, P takes the role of the estimation covariance matrix and w is the corresponding state estimate. More details on this correspondence can be understood from the carefuly comparison of the update equations of both algorithms. This constitutes the fact, that the Kalman Filter delivers the least squares state estimate.

Sequential Analytic Solution – Forgetting factor

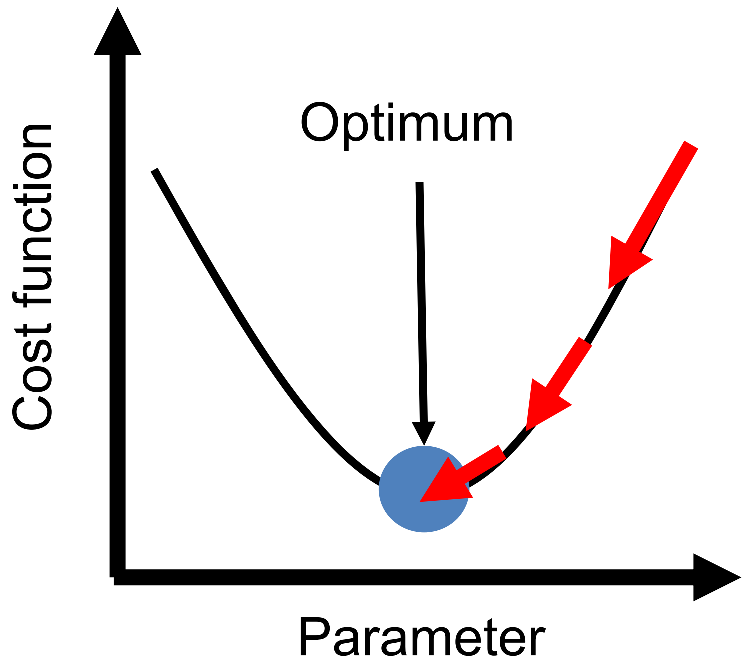
- Some applications show slowly varying conditions in the long term, but can be considered stationary on short to medium time periods
  - Aging of products leads to slight parameter changes
  - Vehicle mass is usually constant over a significant period of time
- The RLS algorithm can deal with this by introduction of a forgetting factor γ. This leads to a reduction of weight for old samples.

$$\vec{w}(k+1) = \vec{w}(k) + P(k)\bar{x}^T \left(\gamma I + \bar{x}^T P(k)\bar{x}\right)^{-1} \left(\bar{y} - \bar{x}^T \vec{w}(k)\right)$$

$$P(k+1) = \gamma^{-1} \left(I - P(k)\bar{x}^T \left(\gamma I + \bar{x}^T P(k)\bar{x}\right)^{-1} \bar{x}\right) P(k)$$

Numerical Iterative Solutions

- Regression can be solved numerically
- Important for large-scale problems and for non-quadratic loss functions
- Popular methods:
  - Gradient descent
  - Gauss-Newton
  - Levenberg-Marquardt



Pro‘s:

- Very generic

Con‘s:

- Knowledge about numeric optimization necessary

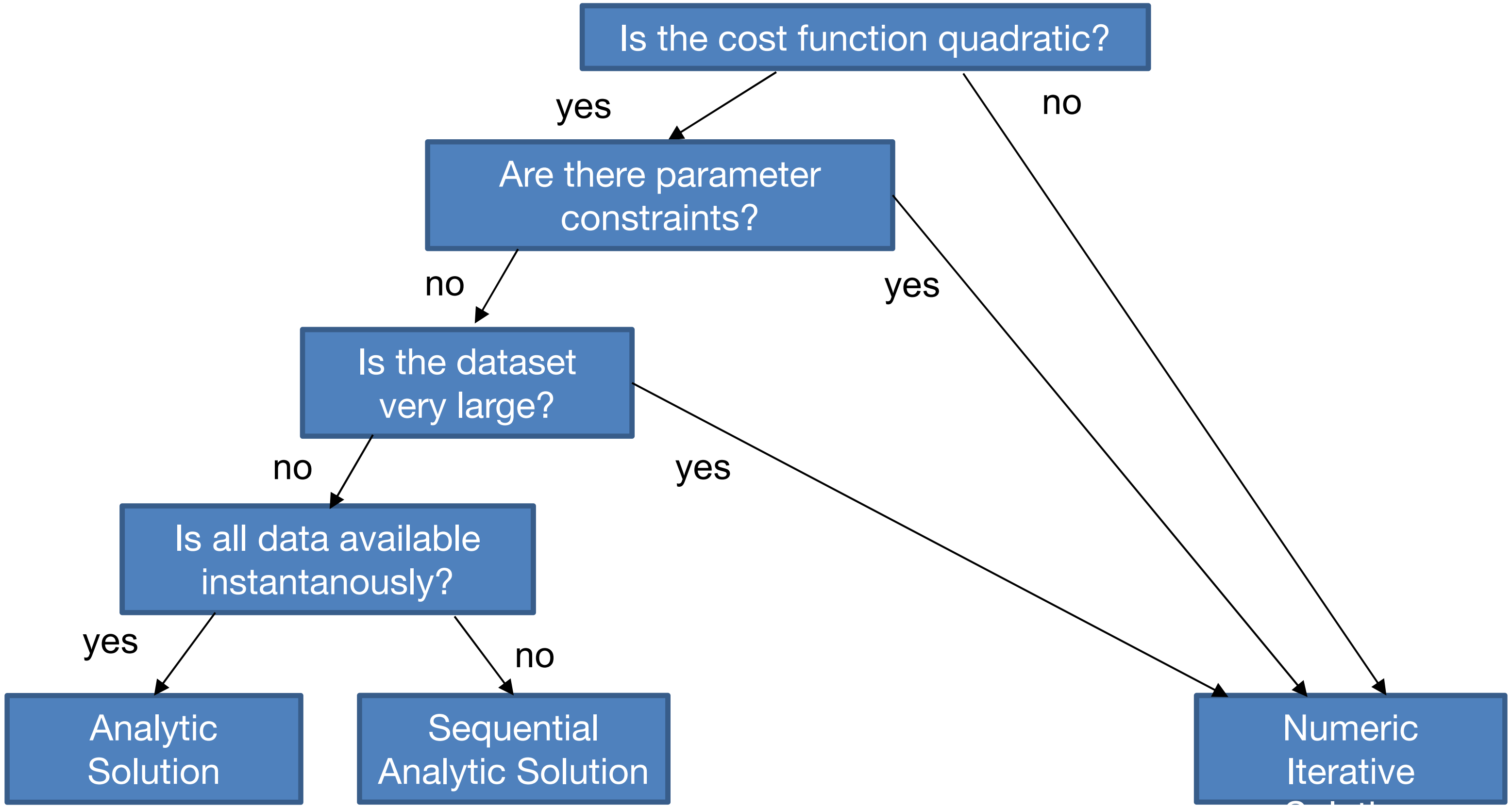
Constraints on the weights

- Weights can be interpreted as physical quantities
  - Temperature (non-negative)
  - Spring constants (non-negative)
  - Mass (non-negative)
- A valid range is known for the weights
  - Tire and other friction models
  - Efficiency (0 – 100 %)

$$\underset{\vec{w}}{\text{minimize}}\ L(\vec{x}, \vec{y}, \vec{w})$$
$$\text{subject to } \vec{c}_1 \leq \vec{w} \leq \vec{c}_2$$

- Improves robustness
- More difficult to solve

How to solve the regression problem?

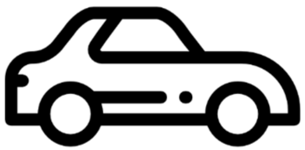


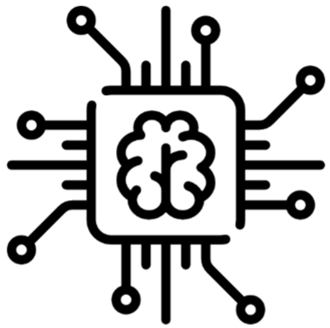
Supervised Learning: Regression

Johannes Betz / Prof. Dr. Markus Lienkamp / Prof. Dr. Boris Lohmann
(Alexander Wischnewski, M. Sc.)

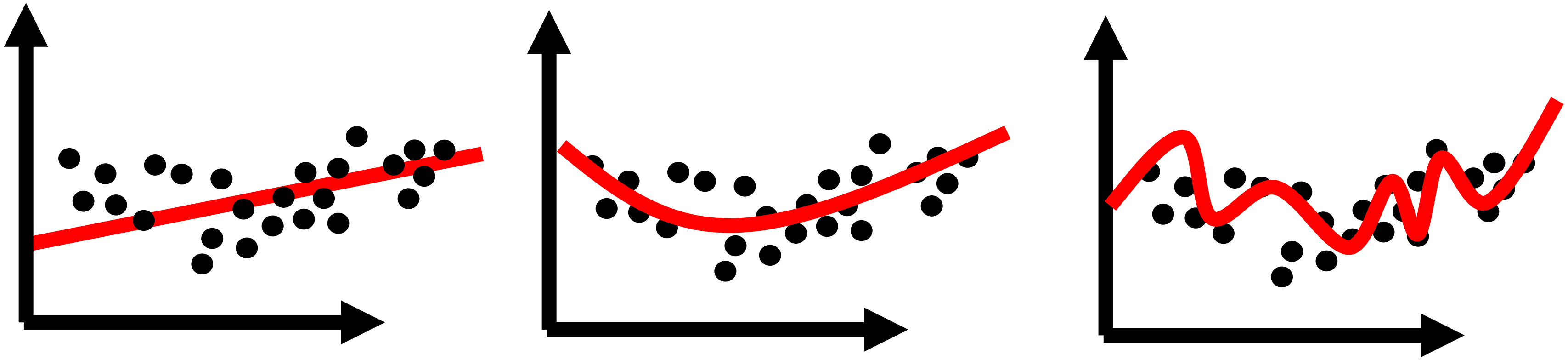
Agenda

1. Chapter: Motivation
2. Chapter: Linear Models
3. Chapter: Loss functions
4. **Chapter: Regularization & Validation**
5. Chapter: Practical considerations
6. Chapter: Summary





How to choose the model?



Underfitted | Well done | Overfitted

- Not enough features
- Wrong structure

- Too many features
- Unrelevant features

Overfitting – Choice of hyperparameters

- Overfitting is the failure to generalize properly between the data points
- Cost function decreases with increased model complexity
- Noise and unrelevant effects become too important

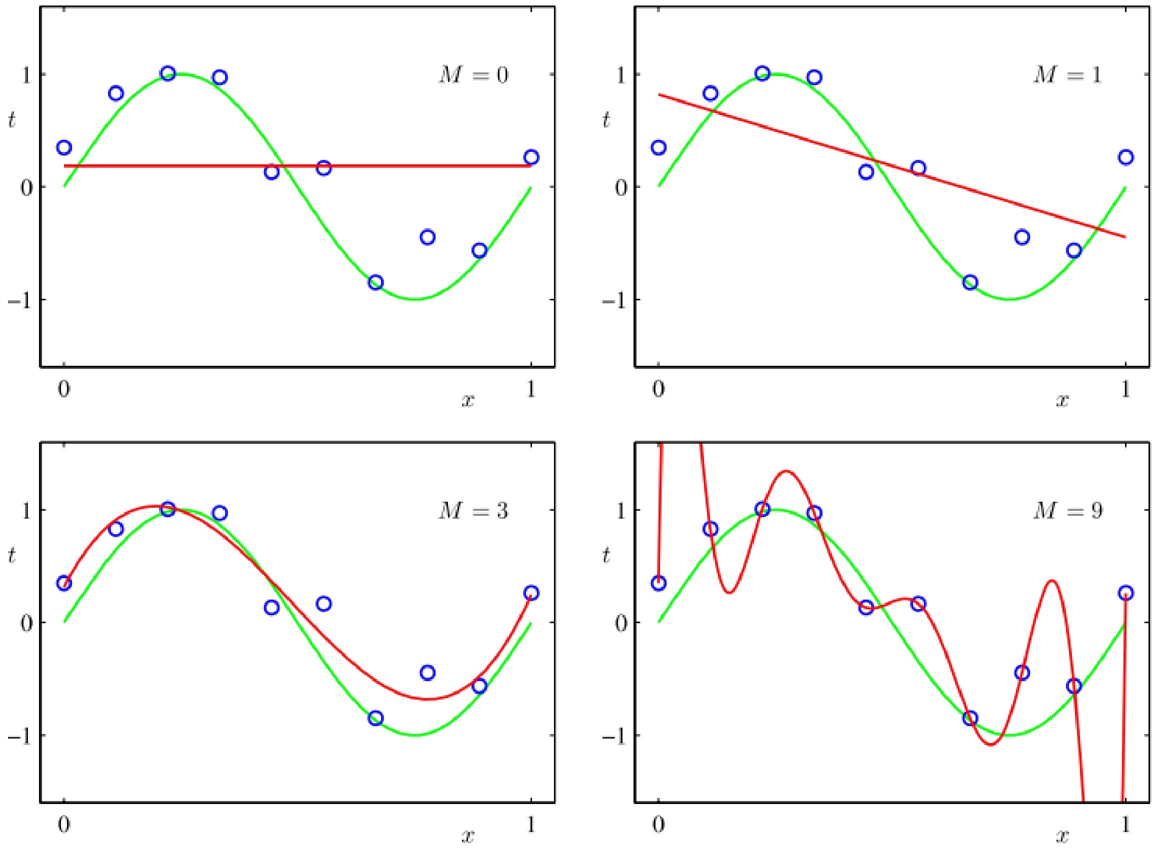
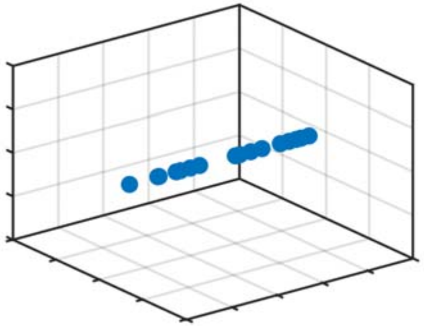
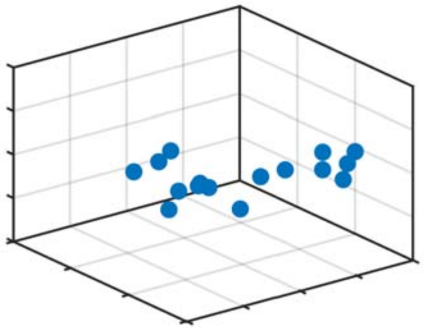


Figure source: Bishop – Pattern Recognition and Machine Learning

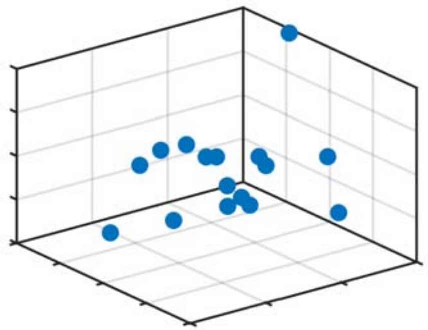
Overfitting – Curse of dimensionality

- Overfitting occurs if
  - data points are sparse
  - Model complexity is high
- Sparsity of data points is difficult to grasp
- Sparsity increases fast with increased input dimension





16 samples
in one, two and
three dimensional
space



Additional Information

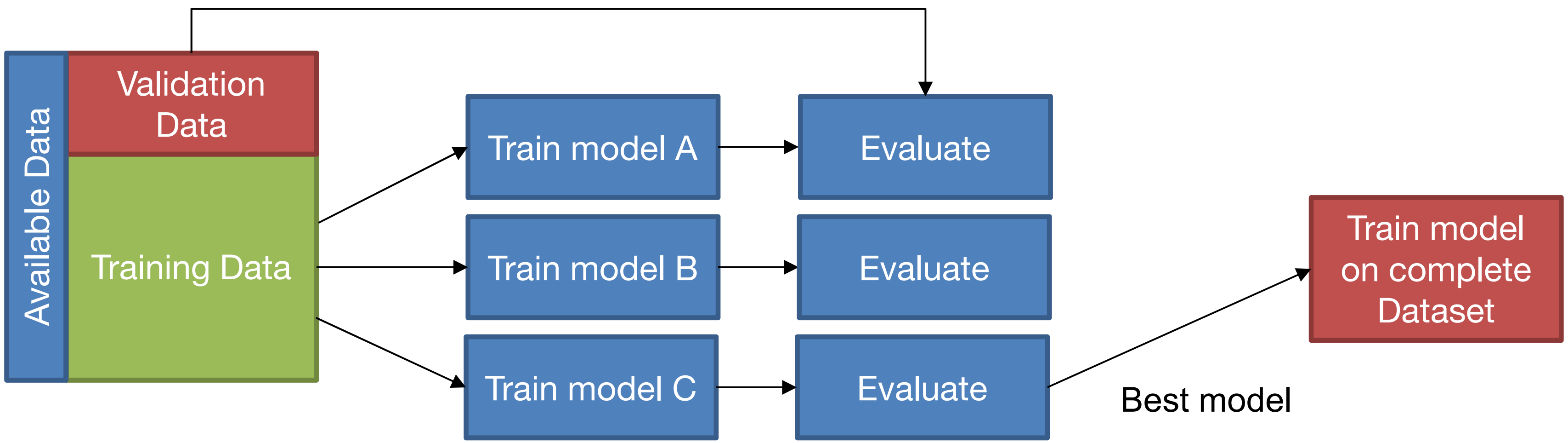
The curse of dimensionality is a widespread problem along all data based algorithms. The most intuitive is the combinatorical aspect mentioned on the slide before. It affects the required sample size as well as the computational effort needed. Beside the mathematical implications, it is therefore of big impact regarding feasible implementations in a descent time and can be considered critical for the application of systems with real time constraints.

Despite the raw number of samples, it is helpful in regression if the samples are „nicely“ located in the sample space. This usually transforms to well-spread along the domain of interest. It is often possible to design the sample points in advance to the experiment carried out. In this case, techniques like Design of Experiment can help to optimize the location of sample points. This helps increasing the model quality with a limited sample size. (Further details can be found here: https://en.wikipedia.org/wiki/Design_of_experiments)

There are other effects related to the dimensionality of the spaces as well. A fundamental one which is more difficult to grasp is the fact, that distances become less meaningful in high dimensional space. (Details can be found here: https://en.wikipedia.org/wiki/Curse_of_dimensionality)

Validation datasets

- Difficult to judge overfitting in high-dimensional domains and autonomous systems
- A standard technique is to separate the data into training and validation data



Validation datasets

- Different hyperparameters can be used to tune the model
- Validation technique works for all of them

Figure source: Bishop – Pattern Recognition and Machine Learning

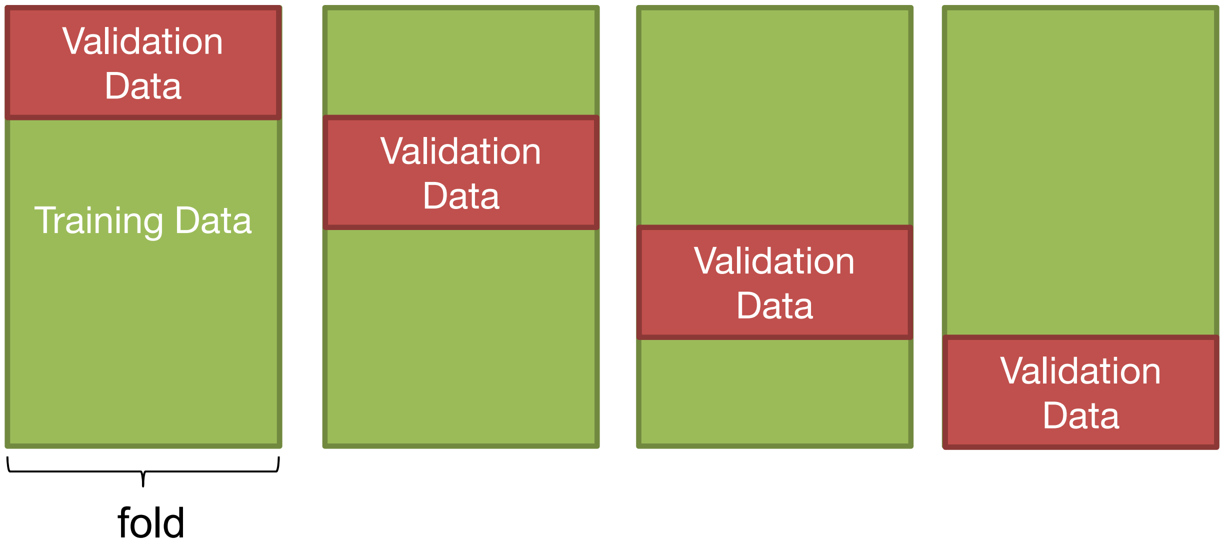
Common pitfalls with validation datasets

- Be aware that your validation dataset must reflect the future properties of the underlying physical relationship.
- Do not reuse validation datasets. If the same validation set is used again and again for testing the model performance it is somehow incorporated into the modelling process and does not give the expected results anymore!
- Split the data before fitting the model is therefore essential. Taking 2/3 of the data as training data is a good starting value.

Visualize your data as much as possible!

k-Fold Cross-validation

- In case of limited data size sets, one may not want to remove a substantial part of the data for the fitting process
- One can use smaller validations sets to estimate the true prediction error by splitting the data into multiple ‚folds‘
- Variance of the estimation error is an indicator for model stability

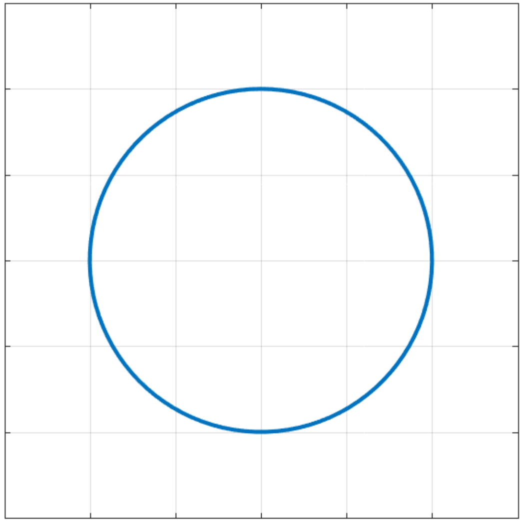


Regularization

- From a design point of view, we want to choose model structure based on underlying physical principles and not on the characteristics of the dataset
  - Polynomial basis functions tend to have large coefficients for sparse datasets
  - Gaussian basis functions tend to overfit locally, which leads to single, large coefficients
- A technique to circumvent this is regularization
  - Penalize high coefficients in the optimization prevents these effects
  - Weighting of penalty term gives an intuitive hyperparameter to control model complexity

Typical Regularization – Ridge Regression

- Other names: L2 regularization, Thikonov regularization
- Prevents overfitting well
- Analytic solution is available as an extension to the MSE problem
- Difficult to apply and tune in high-dimensional feature spaces



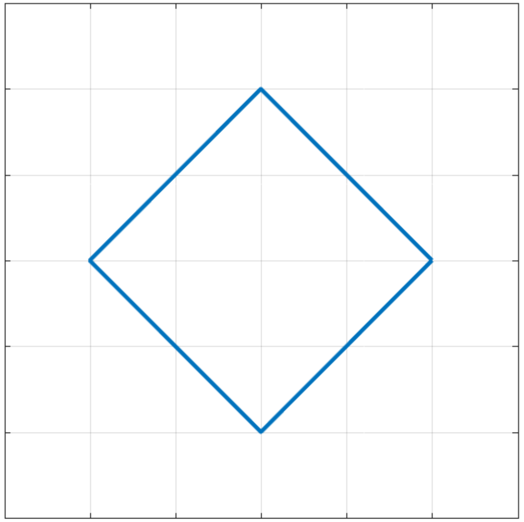
$$\min_{\vec{w}} \; L(\vec{x}, \vec{y}, \vec{w}) + \lambda \vec{w}^T \vec{w}$$

Regularization Term

Typical Regularization – Lasso Regression

- Other names: L1 regularization
- Tends to produce sparse solutions and can therefore be applied for feature selection
- Sparse solution means, that several coefficients go to zero:

$$\vec{w} = \begin{bmatrix} 0 & w_1 & 0 & 0 & 0 & w_2 \end{bmatrix}$$



$$\min_{\vec{w}} L(\vec{x}, \vec{y}, \vec{w}) + \underbrace{\lambda \sum_i \mid w_i \mid}_{\text{Regularization Term}}$$

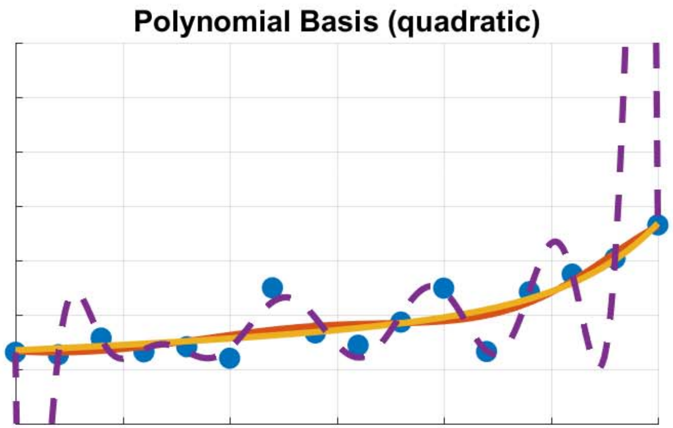
Additional Information

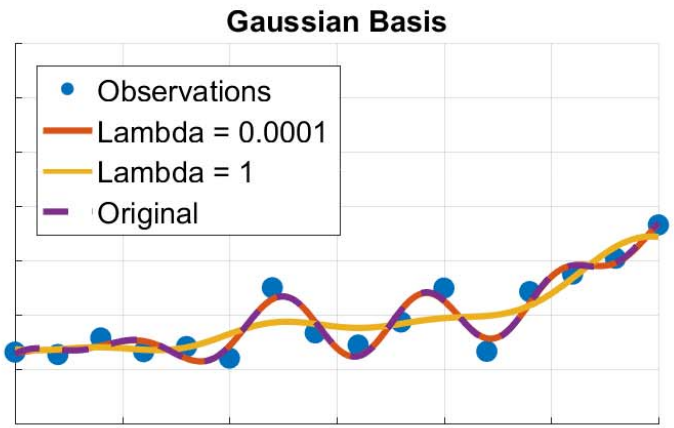
Comparison between ridge and lasso regularization:

- https://www.analyticsvidhya.com/blog/2016/01/complete-tutorial-ridge-lasso-regression-python/
- https://codingstartups.com/practical-machine-learning-ridge-regression-vs-lasso/

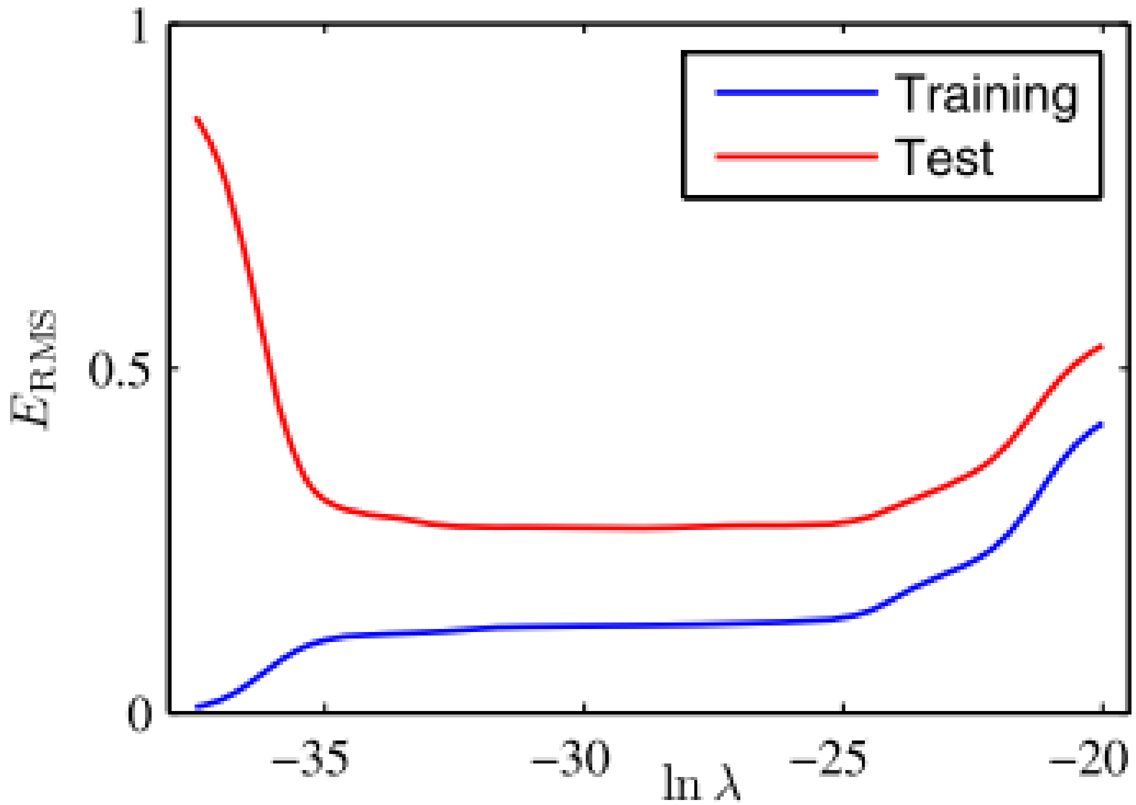
Application of regularization

- Use high dimensional input space for test model
- Ridge regression is applied
- Regularized solutions perform far better at interpolation
- Note: You must evaluate at points BETWEEN your sample points





Tuning regularization with validation data



Increased Model Complexity

Figure source: Bishop – Pattern Recognition and Machine Learning

Bias Variance Trade Off

- The issue of over- and underfitting can be formalized
- Study the predictor performance on a previously unseen data set
- Assume that the observations $y = f(x) + \epsilon$ are corrupted by noise
- The variance of the test data is then written as

$$\mathbb{E}\left(\left(y - \hat{f}(x)\right)^2\right) = \mathbb{E}\left(f(x) - \hat{f}(x)\right)^2 + \mathbb{E}\left(\hat{f}(x)^2\right) - \left(\mathbb{E}\left(\hat{f}(x)\right)\right)^2 + \sigma^2$$

Bias Variance Noise

Additional Information

Derivation of the Bias-Variance decomposition following
https://towardsdatascience.com/mse-and-bias-variance-decomposition-77449dd2ff55

$$\mathbb{E}\left(\left(y-\hat{f}(x)\right)^2\right)$$

$$\mathbb{E}\left(y^2\right)+\mathbb{E}\left(\hat{f}(x)^2\right)-\mathbb{E}\left(2y\hat{f}\right)$$

Using the fact that $\mathbb{E}\left(a^2\right)+\mathrm{Var}(x)+\mathbb{E}(a)^2$ holds for the first and the second term we obtain

$$\mathrm{Var}(y)+\mathbb{E}(y)^2+\mathrm{Var}\left(\hat{f}(x)\right)+\mathbb{E}\left(\hat{f}(x)\right)^2-\mathbb{E}(2y\hat{f}(x))$$

$$\mathrm{Var}(y)+\mathbb{E}(y)^2+\mathrm{Var}\left(\hat{f}(x)\right)+\mathbb{E}\left(\hat{f}(x)\right)^2-2\mathbb{E}\left(f(x)\hat{f}(x)\right)-2\mathbb{E}\left(\epsilon\hat{f}(x)\right)$$

Since we evaluate on previously unseen data, the last term vanishes (The noise of new measurements is independent of the fitted model and zero mean).

$$\mathrm{Var}\left(f(x)\right)+\mathrm{Var}(\epsilon)+\mathbb{E}(y)^2+\mathrm{Var}\left(\hat{f}(x)\right)+\mathbb{E}\left(\hat{f}(x)\right)^2-2\mathbb{E}\left(f(x)\hat{f}(x)\right)$$

$$\mathrm{Var}\left(f(x)\right)+\mathrm{Var}(\epsilon)+\mathbb{E}(y)^2+\mathrm{Var}\left(\hat{f}(x)\right)+\mathbb{E}\left(\hat{f}(x)\right)^2-2\mathbb{E}\left(f(x)\right)\mathbb{E}\left(\hat{(}x)\right)-2\mathrm{cov}\left(f(x),\hat{f}(x)\right)$$

$$\mathrm{Var}\left(f(x)-\hat{f}(x)\right)+\mathrm{Var}(\epsilon)+\mathbb{E}\left(y\right)^2+\mathbb{E}\left(\hat{f}(x)\right)^2-2\mathbb{E}\left(f(x)\right)\mathbb{E}\left(\hat{(}x)\right)$$

$$\mathrm{Var}\left(f(x)-\hat{f}(x)\right)+\mathrm{Var}(\epsilon)+\mathbb{E}\left(f(x)\right)^2+\mathbb{E}\left(\epsilon\right)^2+2\mathbb{E}\left(f(x)\right)\mathbb{E}\left(\epsilon\right)+\mathbb{E}\left(\hat{f}(x)\right)^2-2\mathbb{E}\left(f(x)\right)\mathbb{E}\left(\hat{(}x)\right)$$

$$\mathrm{Var}\left(f(x)-\hat{f}(x)\right)+\mathrm{Var}(\epsilon)+\left(\mathbb{E}\left(f(x)\right)-\mathbb{E}\left(\hat{f}(x)\right)\right)^2+\mathbb{E}\left(\epsilon\right)^2+2\mathbb{E}\left(f(x)\right)\mathbb{E}\left(\epsilon\right)$$

$$\mathrm{Var}\left(f(x)-\hat{f}(x)\right)+\left(\mathbb{E}\left(f(x)\right)-\mathbb{E}\left(\hat{f}(x)\right)\right)^2+\mathrm{Var}(\epsilon)$$

Regularization in context of Bias-Variance Trade-Off

- Imagine you can repeat the fitting process lots of times
- This renders the fitting process a random experiment
- Model Bias: The mean error over all possible fits of the model
- Model Variance: The variance of all possible fits of the model

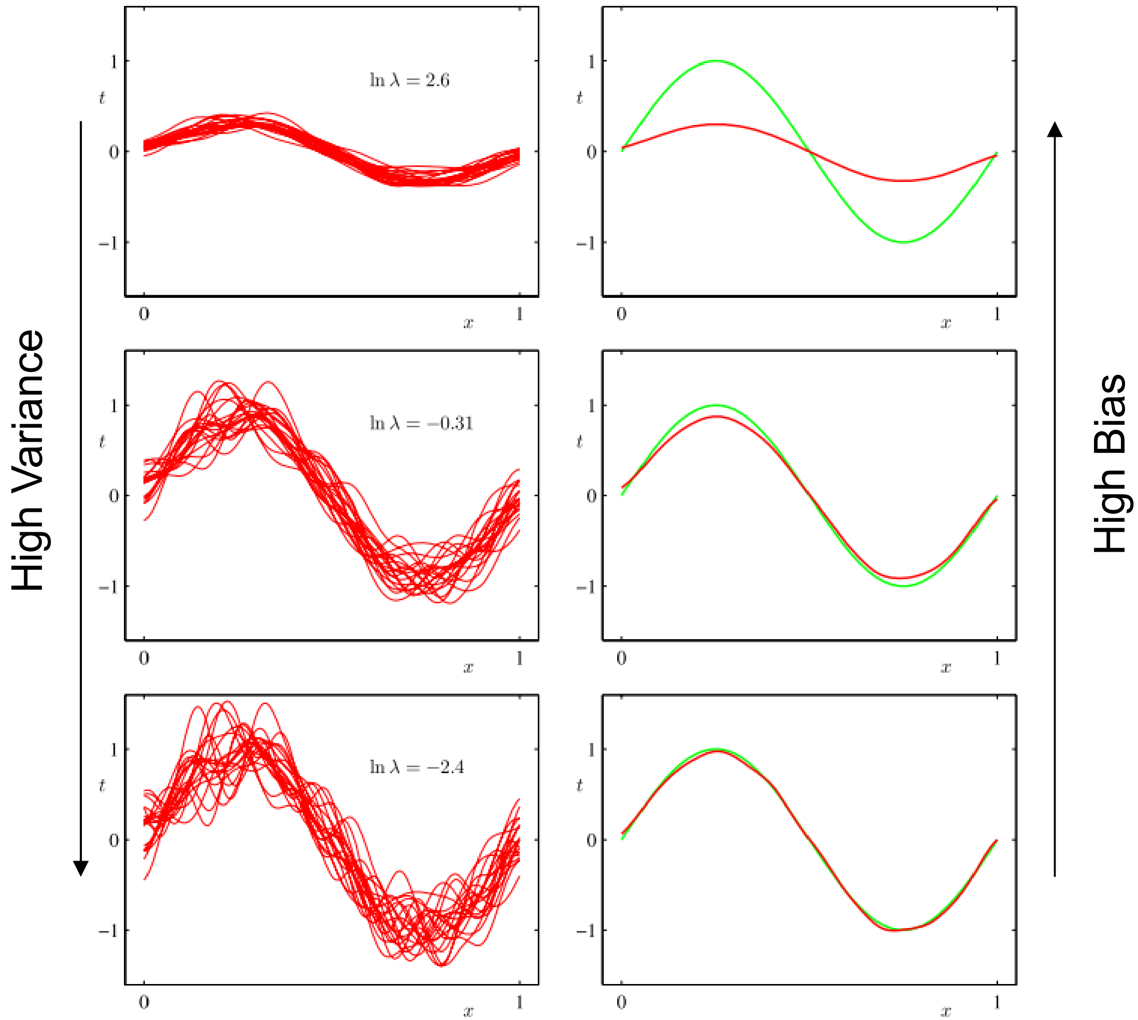
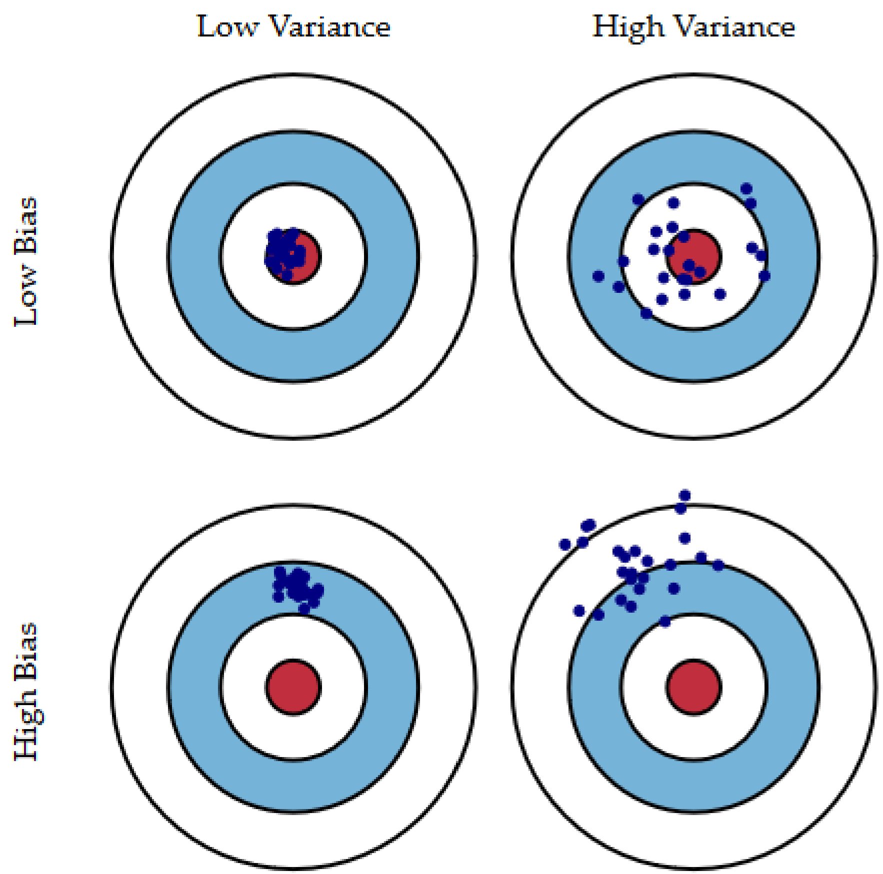


Figure source: Bishop – Pattern Recognition and Machine Learning

Regularization in context of Bias-Variance Trade-Off

- Bulls-Eye: True model
- Each hit is a realization of prediction model
- In general, we cannot ensure to reach low bias and low variance at the same time
- We have to balance bias and variance according to our objective
- Low bias and low variance require large, high quality datasets



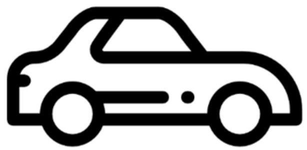
Source: Scott Fortmann-Roe

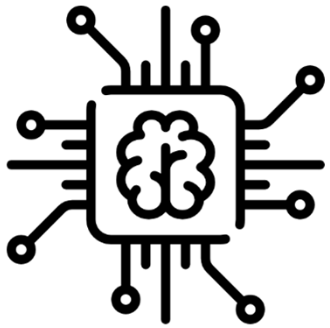
Supervised Learning: Regression

Johannes Betz / Prof. Dr. Markus Lienkamp / Prof. Dr. Boris Lohmann
(Alexander Wischnewski, M. Sc.)

Agenda

1. Chapter: Motivation
2. Chapter: Linear Models
3. Chapter: Loss functions
4. Chapter: Regularization & Validation
5. **Chapter: Practical considerations**
6. Chapter: Summary

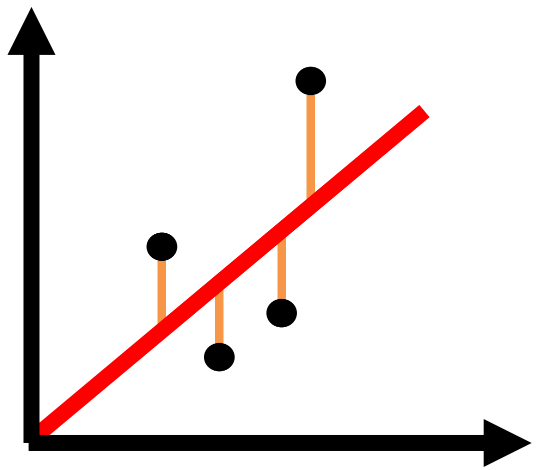




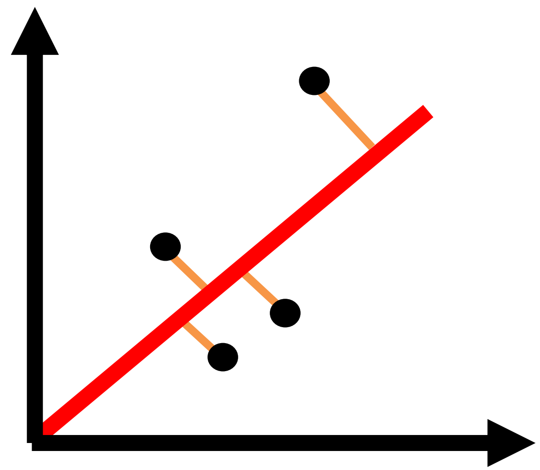
Comparison regression and intuitive solution

- Regression: Minimize the error in output domain
- Independent Variables are assumed noise free
- Often not true in engineering applications

Standard linear regression



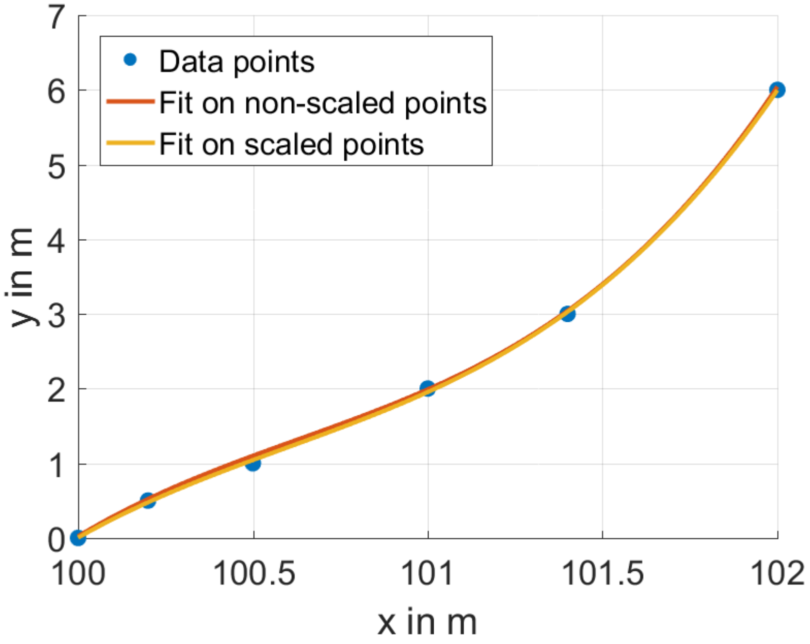
Total least squares
Principal component analysis

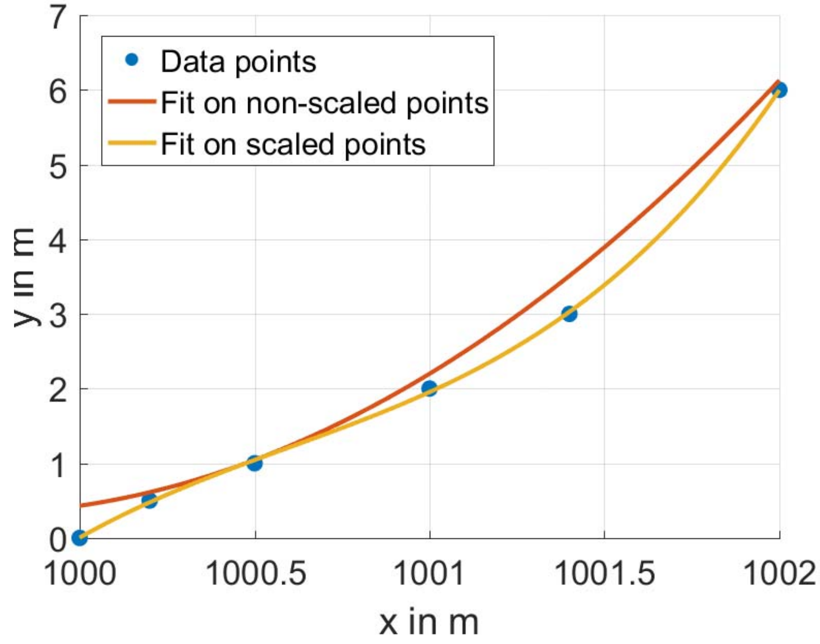


Normalization – Input data

- Regression implies inversion of design matrix
- Numerically expensive and can get unstable
- In general, all matrix entries variables should lie within the same order of magnitude!
- Examples for scaling and mean value normalization
  - Multiple length measures should be transformed to the same unit
  - Variables which do not have any physical relation should be transformed based on their min/max values
  - Temperatures are likely to be far away from zero, they should be normalized by a mean temperature

Normalization – Input data





Additional Information

The plots in the slide before were obtained by fitting a regression based on the simple polynomial basis to the data points. The basis functions in this example are

$$1 \quad x \quad x \quad x^2$$

This leads to the fact, that the design matrix becomes ill-conditioned very fast for large input data. It results from the polynomials which lead to a design matrix (depicted for x = 1000 and 1001) of

$$\begin{bmatrix} 1 & 1000 & 1000000 & 1000000000 \\ 1 & 1001 & 1002001 & 1003003001 \\ \dots & \dots & \dots & \dots \end{bmatrix}$$

Instead of (depicted for x = 0 and x = 1)

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

The latter matrix is far easier to solve by numeric computations. The intuition behind this is that the ‚spread‘ between the matrix entries is much smaller. This helps the computer to obtain a numerically stable result since the precision is limited even in floating point operations. As a result from this, the maximum ‚spread‘ leading to reliable results is depending on the available computation precision. This is much more critical on embedded hardware.

Normalization – Output data

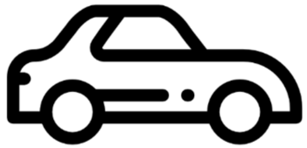
- In engineering applications we often face the task to fit parameters such that multiple objectives are met
- These may not lie within the same units or scale
- Example:
  - Estimate the driven curvature κ based on speed v, yaw rate $\dot{\psi}$ and lateral acceleration a_y
  - Model equations: $\dot{\psi} = v\kappa$ and $a_y = v^2\kappa$
  - Scale varies heavily with speed and is inconsistent between the two equations
  - Better use: $\frac{\dot{\psi}}{v} = \kappa$ and $\frac{a_y}{v^2} = \kappa$
- Scaling the output equations can also be used to weight equations which are more important than others

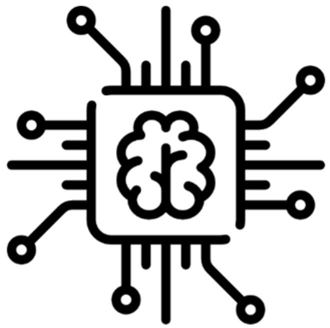
Supervised Learning: Regression

Johannes Betz / Prof. Dr. Markus Lienkamp / Prof. Dr. Boris Lohmann
(Alexander Wischnewski, M. Sc.)

Agenda

1. Chapter: Motivation
2. Chapter: Linear Models
3. Chapter: Loss functions
4. Chapter: Regularization & Validation
5. Chapter: Practical considerations
6. **Chapter: Summary**





Summary

- We learned about:
  - the characteristics of a **regression** problem and learned about the **differences** to **clustering** and **classification**
  - the difference between **linear and nonlinear regression**
  - **use cases** and **applications** for regression in Automotive Technology
  - different applications for **local and global basis functions**
  - different **loss functions**
  - **iterative** and **analytic** solution methods for model training
  - **regularization** techniques and **validation** techniques
  - details which are important for succesful application of regression to real world data

Related Literature

- Christopher M. Bishop, *Machine Learning and Pattern Recognition*
- Trevor Hastie, Robert Tibshirani, Jerome Friedman, *The Elements of Statistical Learning*
- Kevin P. Murphy*, Machine Learning – A Probabilistic Perspective*