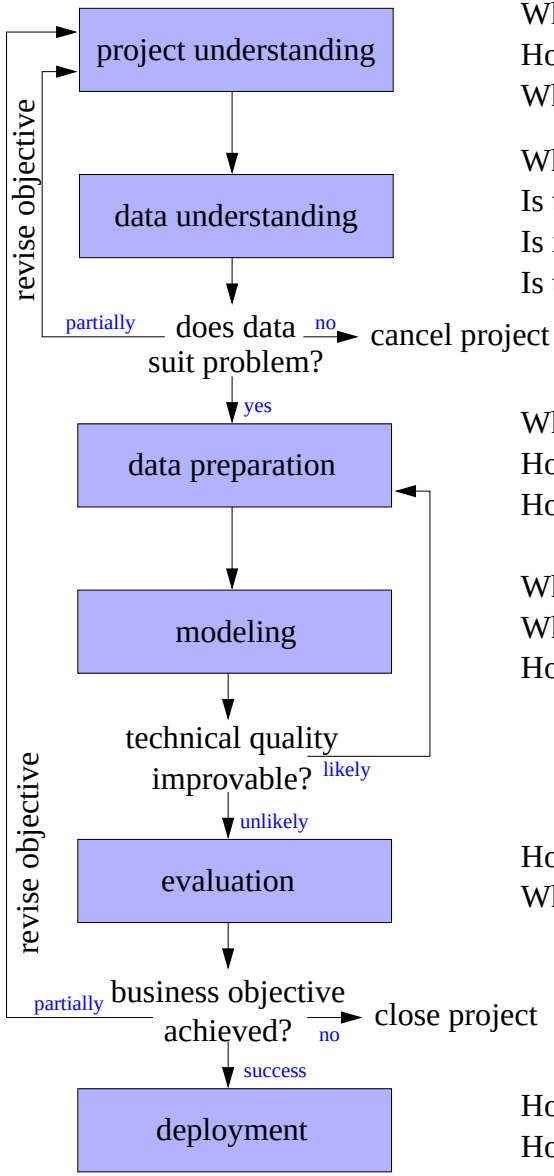
Modeling



What exactly is the problem, the expected benefit?
How would a solution look like?
What is known about the domain?

What data do we have available?
Is the data relevant to the problem?
Is it valid? Does it reflect our expectations?
Is the data quality, quantity, recency sufficient?

Which data should we concentrate on?
How is the data best transformed for modeling?
How may we increase the data quality?

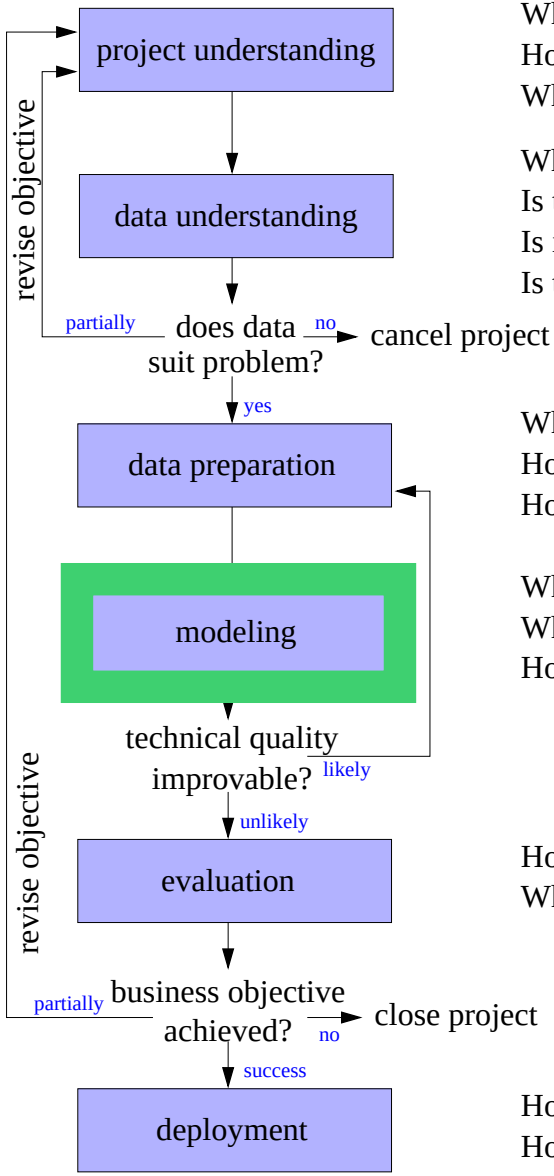
What kind of model architecture suits the problem best?
What is the best technique/method to get the model?
How good does the model perform technically?

How good is the model in terms of project requirements?
What have we learned from the project?

How is the model best deployed?
How do we know that the model is still valid?

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Modeling



What exactly is the problem, the expected benefit?
How would a solution look like?
What is known about the domain?

What data do we have available?
Is the data relevant to the problem?
Is it valid? Does it reflect our expectations?
Is the data quality, quantity, recency sufficient?

Which data should we concentrate on?
How is the data best transformed for modeling?
How may we increase the data quality?

What kind of model architecture suits the problem best?
What is the best technique/method to get the model?
How good does the model perform technically?

How good is the model in terms of project requirements?
What have we learned from the project?

How is the model best deployed?
How do we know that the model is still valid?

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

The Four Steps of Modeling

Select the **model class**

- General structure of the analysis result
- "Architecture" or "model class"
- Example: Linear or quadratic functions for regression problem

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

The Four Steps of Modeling

Select the **model class**

- General structure of the analysis result
- "Architecture" or "model class"
- Example: Linear or quadratic functions for regression problem

Select the **score function**

- Evaluate possible "models" using a score function

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

The Four Steps of Modeling

Select the **model class**

- General structure of the analysis result
- "Architecture" or "model class"
- Example: Linear or quadratic functions for regression problem

Select the **score function**

- Evaluate possible "models" using a score function

Apply the **algorithm**

- Compare models through the score function
- But: How do we find the models?

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

The Four Steps of Modeling

Select the **model class**

- General structure of the analysis result
- "Architecture" or "model class"
- Example: Linear or quadratic functions for regression problem

Select the **score function**

- Evaluate possible "models" using a score function

Apply the **algorithm**

- Compare models through the score function
- But: How do we find the models?

Validate the results

- We know: Best model among the chose ones
- But: Is this the best among very good or very bad choices?

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Model class?

- Model = The form or structure of the analysis result
- Here the parameters are **not** defined only the type is selected
- Examples:
  - Linear models ($y = ax + b$)
  - Constant values (e.g. mean)
  - Rule based models (if `A` buys `product one`, then `weather` is `sunny`)

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Model class - Requirements

Simplicity

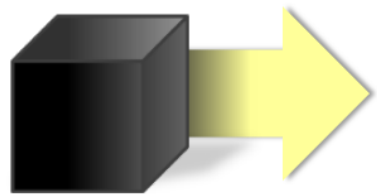
- **Occam's razor**:
  - Choose the simplest model that still "explains" the data.
  - Or : `Numquam ponenda est pluralitas sine necessitate`
    = [Plurality must never be posited without necessity]
- easier to understand
- lower complexity
- avoid overfitting(see Slide 21 ff.)

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Model class - Requirements

Simplicity

- **Occam's razor**:
  - Choose the simplest model that still "explains" the data.
  - Or : `Numquam ponenda est pluralitas sine necessitate`
    = [Plurality must never be posited without necessity]
- easier to understand
- lower complexity
- avoid overfitting(see Slide 21 ff.)



Interpretability

- Black-Boxes are mostly not a proper choice
- But: They can result in a very good accuracy(e.g. neural networks)

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Global vs. local models

- Global models provide a (not necessarily good) description for the whole data set.
  Example: Regression line
- Local models or patterns provide a description for only a part or subset of the data set.
  Example: Association rules

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Fitting Criteria and Score Function

- find an **objective function** $f : \mathcal{M} \rightarrow \mathbb{R}$
- Which, evaluates the quality of your model
- In order to detect the "best" model

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Fitting Criteria and Score Function

- find an **objective function** $f : \mathcal{M} \rightarrow \mathbb{R}$
- Which, evaluates the quality of your model
- In order to detect the "best" model

Example

Given: Dataset $D = \{d_1, d_2, ...d_n\} \in \mathbb{R}^m$ and "model" $\mathcal{M} : \mathbb{R}^m \rightarrow \mathbb{R}^m$
($\mathcal{M}$ predicts a value for a given data point).

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Fitting Criteria and Score Function

- find an **objective function** $f : \mathcal{M} \rightarrow \mathbb{R}$
- Which, evaluates the quality of your model
- In order to detect the "best" model

Example

Given: Dataset $D = \{d_1, d_2, ...d_n\} \in \mathbb{R}^m$ and "model" $\mathcal{M} : \mathbb{R}^m \rightarrow \mathbb{R}^m$
($\mathcal{M}$ predicts a value for a given data point).

- Mean squared error : $f(x) = \frac{1}{n} \sum_{i=1}^{n} (x - \mathcal{M}(x))^2$

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Fitting Criteria and Score Function

- find an **objective function** $f : \mathcal{M} \rightarrow \mathbb{R}$
- Which, evaluates the quality of your model
- In order to detect the "best" model

Example

Given: Dataset $D = \{d_1, d_2, ...d_n\} \in \mathbb{R}^m$ and "model" $\mathcal{M} : \mathbb{R}^m \rightarrow \mathbb{R}^m$
($\mathcal{M}$ predicts a value for a given data point).

- Mean squared error : $f(x) = \frac{1}{n} \sum_{i=1}^{n} (x - \mathcal{M}(x))^2$
- Mean absolute error : $f(x) = \frac{1}{n} \sum_{i=1}^{n} |x - \mathcal{M}(x)|$

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Short comment : What is classification?

Example

Imagine a cup factory,
which wants to classify their cups as good or broken.

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Error functions for classification problems

How to set up an error function for those classification problems?

- Very common misclassification rate $= \frac{\#\text{ wrong classified}}{\#\text{ total classified}}$
- A low misclassification rate does not necessarily tell anything about the quality of a classifier.
- when classes are unbalanced (e.g. When 99% of the production are ok, a classifier always predicting ok will have a misclassification rate of 1%.)

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Cost matrix

- More general approach than the misclassification rate: cost function or cost matrix.
- The consequences (costs) for misclassification for one class might be different than for another class.

Example

Tea cup production.

Cost matrix

	predicted class	
true class	OK	broken
OK	0	c_1
broken	c_2	0

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Cost matrix

General form of a cost matrix for a multi-class classification problem:

	predicted class			
true class	c_1	c_2	$\dots$	c_m
c_1	0	$c_{1,2}$	$\dots$	$c_{1,m}$
c_1	$c_{2,1}$	0	$\dots$	$c_{2,m}$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
c_m	$c_{m,1}$	$c_{m,2}$	$\dots$	0

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Cost matrix

When such a cost matrix is provided, the expected loss given by

$$\text{loss}(c_i|E) = \sum_{j=1}^{m} P(c_j|E)c_{ji}$$

should be minimized.

- E is the evidence, i.e. the observed values of the predictor attributes used for the classification.
- $P(c_j|E)$ is the predicted probability that the true class is c_j given observation E.

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Cost matrix

Example

(Hypothetical) cost matrix for the tea cup production problem

true class	OK	broken
OK	0	1
broken	10	0

A classifier might classify a specific cup with 80% to the class ok and with 20% to the class broken.

- Expected loss for choosing ok: $0.8 \cdot 0 + 0.2 \cdot 10 = 2$.
- Expected loss for choosing broken: $0.8 \cdot 1 + 0.2 \cdot 0 = 0.8$.

Choose broken in this case to minimize the expected loss!

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Cost matrix

Using the cost matrix

	predicted class			
true class	c_1	c_2	$\ldots$	c_m
c_1	0	1	$\ldots$	1
c_1	1	0	$\ldots$	1
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
c_m	1	1	$\ldots$	0

corresponds to minimising the misclassification rate.

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Algorithms for model fitting

The objective function (scoring function) for models

- does not tell us how to find the best or a good model,
- it only provides a means for comparing models.

Optimisation algorithms to find the best or at least a good model are needed.

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Closed form solutions

In the best case, an explicit solution can be provided.

Example

Find a regression line $y = ax + b$ that minimizes the mean square error for the data set $(x_1, y_1), \ldots, (x_n, y_n)$. Computing partial derivatives of the objective (error) function

$$E(a,b) = \frac{1}{n}\sum_{i=1}^{n}(ax_i + b - y_i)^2$$

w.r.t. the paramaters a and b yields

$$\begin{aligned} \frac{\partial E}{\partial a} &= \frac{2}{n}\sum_{i=1}^{n}(ax_i + b - y_i)x_i &= 0, \\ \frac{\partial E}{\partial b} &= \frac{2}{n}\sum_{i=1}^{n}(ax_i + b - y_i) &= 0. \end{aligned}$$

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Closed form solutions

Example

The solution of this system of equations is

$$a = \frac{n\sum_{i=1}^{n} x_i y_i - \left(\sum_{i=1}^{n} x_i\right)\left(\sum_{i=1}^{n} y_i\right)}{n\sum_{i=1}^{n} x_i^2 - \left(\sum_{i=1}^{n} x_i\right)^2},$$

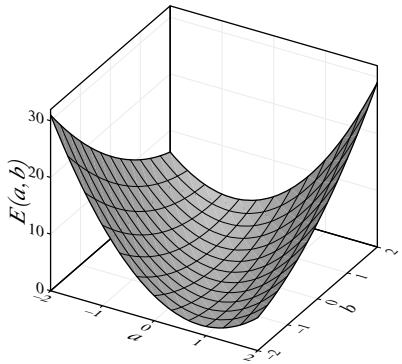
$$b = \bar{y} - a\bar{x}$$

where $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$ and $\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$.

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

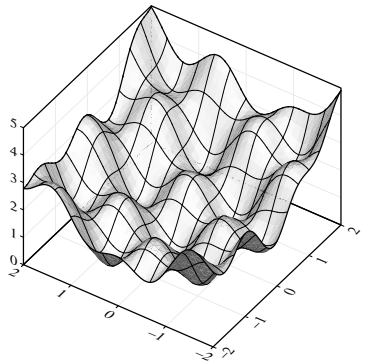
Algorithms for model fitting

For differentiable score functions, a gradient methods can be applied.



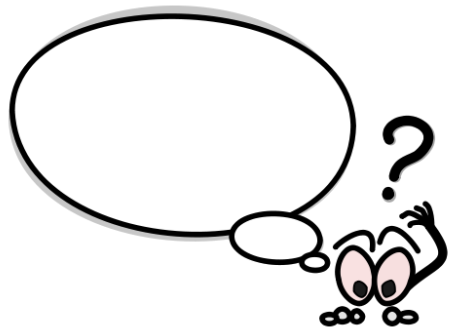
$k = 2$

A landscape for the mean squared error.



$k = 2$

A landscape with many local minima and maxima



$k > 2$

The objective function corresponds to a landscape in (k+1) dimensional space.

Problems

- Will only find local optima.
- Parameters (step width) must be adjusted or computed in each iteration step.

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Algorithms for model fitting

- For discrete problems with a finite search space (like finding association rules), combinatorial optimization strategies are needed.
- In principle, an exhaustive search of the finite domain $\mathcal{M}$ is possible, however, in most cases it is not feasible, since $\mathcal{M}$ is much too large.

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Algorithms for model fitting

- For discrete problems with a finite search space (like finding association rules), combinatorial optimization strategies are needed.
- In principle, an exhaustive search of the finite domain $\mathcal{M}$ is possible, however, in most cases it is not feasible, since $\mathcal{M}$ is much too large.

Example

- Finding the best possible association rules with an underlying set of 1000 items (products).
- Every combination of items, i.e. every nonempty subset is a possible candidate set from which several rules may be constructed.
- The number of nonempty subsets alone contains $2^{1000} - 1 > 10^{300}$ elements.

Heuristic strategies are therefore needed.

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Algorithms for model fitting: Heuristic strategies

- **Random search.** Create random solutions and choose the best one among them.

  Very inefficient

- **Greedy strategies.** Formulate an algorithm that tries to improve the solution in each step.
  - Example. Gradient method.
  - Example. Hillclimbing.
    Start with a random solution,
    generate new solutions in the "neighbourhood" of the solution.
    If a new solution is better than the old one, generate new solutions in its "neighbourhood".

  Can find a solution quickly, but get stuck in local optima.

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Algorithms for model fitting: Heuristic strategies

- **Simulated annealing is a mixture between random search and a greedy strategy. Simulated annealing is a modified version of hillclimbing, sometimes replacing better solutions by worse ones with a (low) probability.**
  This probability is decreased in each iteration step.
- **Evolutionary algorithms** like evolution strategies or genetic algorithms combine random with greedy components, using a population of solutions in order to explore the search space in parallel and efficiently.
- **Alternating optimisation** can be applied when the set of parameters can be split into disjoint subsets in such a way that for each subset an analytical solution for the optimum can be provided, given the parameters in the other subsets are fixed. Alternating optimization computes the analytical solution for the parameter subsets alternatingly and iterates this scheme until convergence.

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Algorithms for model fitting: Heuristic strategies

- **Simulated annealing** is a mixture between random search and a greedy strategy. Simulated annealing is a modified version of hillclimbing, sometimes replacing better solutions by worse ones with a (low) probability.
  This probability is decreased in each iteration step.
- **Evolutionary algorithms like evolution strategies or genetic algorithms combine random with greedy components, using a population of solutions in order to explore the search space in parallel and efficiently.**
- **Alternating optimisation** can be applied when the set of parameters can be split into disjoint subsets in such a way that for each subset an analytical solution for the optimum can be provided, given the parameters in the other subsets are fixed. Alternating optimization computes the analytical solution for the parameter subsets alternatingly and iterates this scheme until convergence.

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

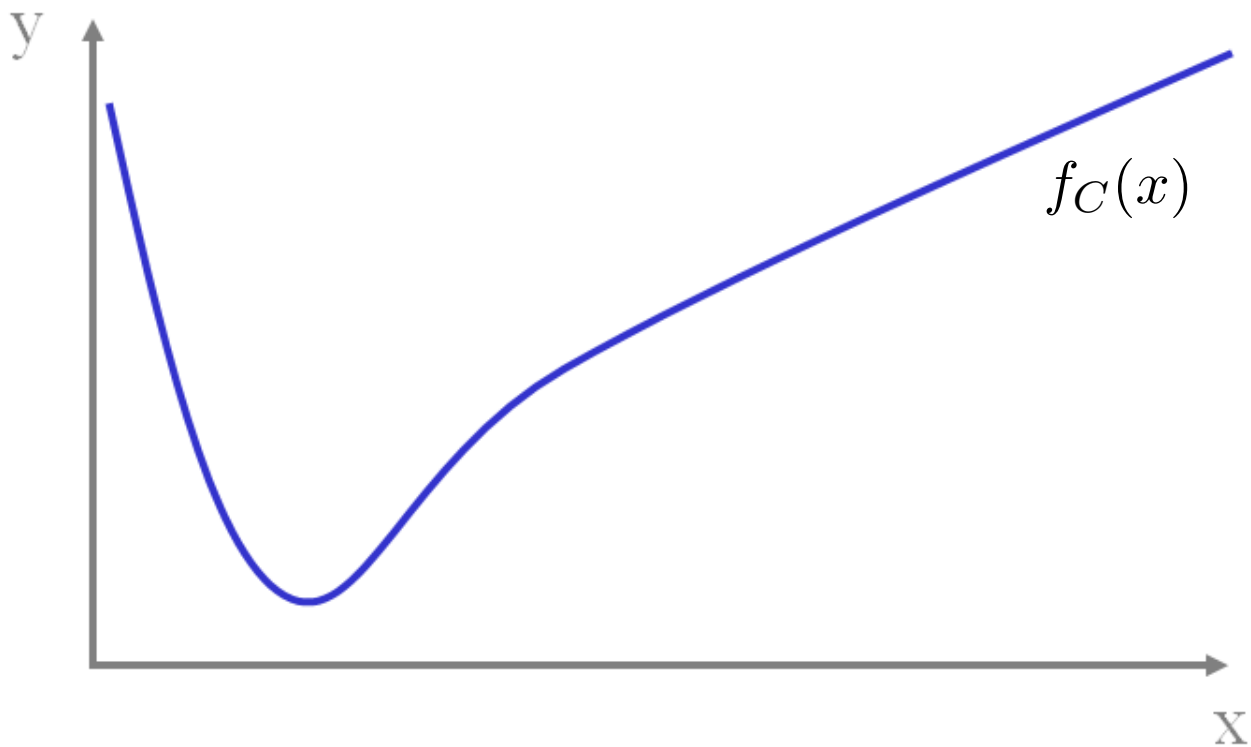
Algorithms for model fitting: Heuristic strategies

- **Simulated annealing** is a mixture between random search and a greedy strategy. Simulated annealing is a modified version of hillclimbing, sometimes replacing better solutions by worse ones with a (low) probability.
  This probability is decreased in each iteration step.
- **Evolutionary algorithms** like evolution strategies or genetic algorithms combine random with greedy components, using a population of solutions in order to explore the search space in parallel and efficiently.
- **Alternating optimisation can be applied when the set of parameters can be split into disjoint subsets in such a way that for each subset an analytical solution for the optimum can be provided, given the parameters in the other subsets are fixed. Alternating optimization computes the analytical solution for the parameter subsets alternatingly and iterates this scheme until convergence.**

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Overfitting

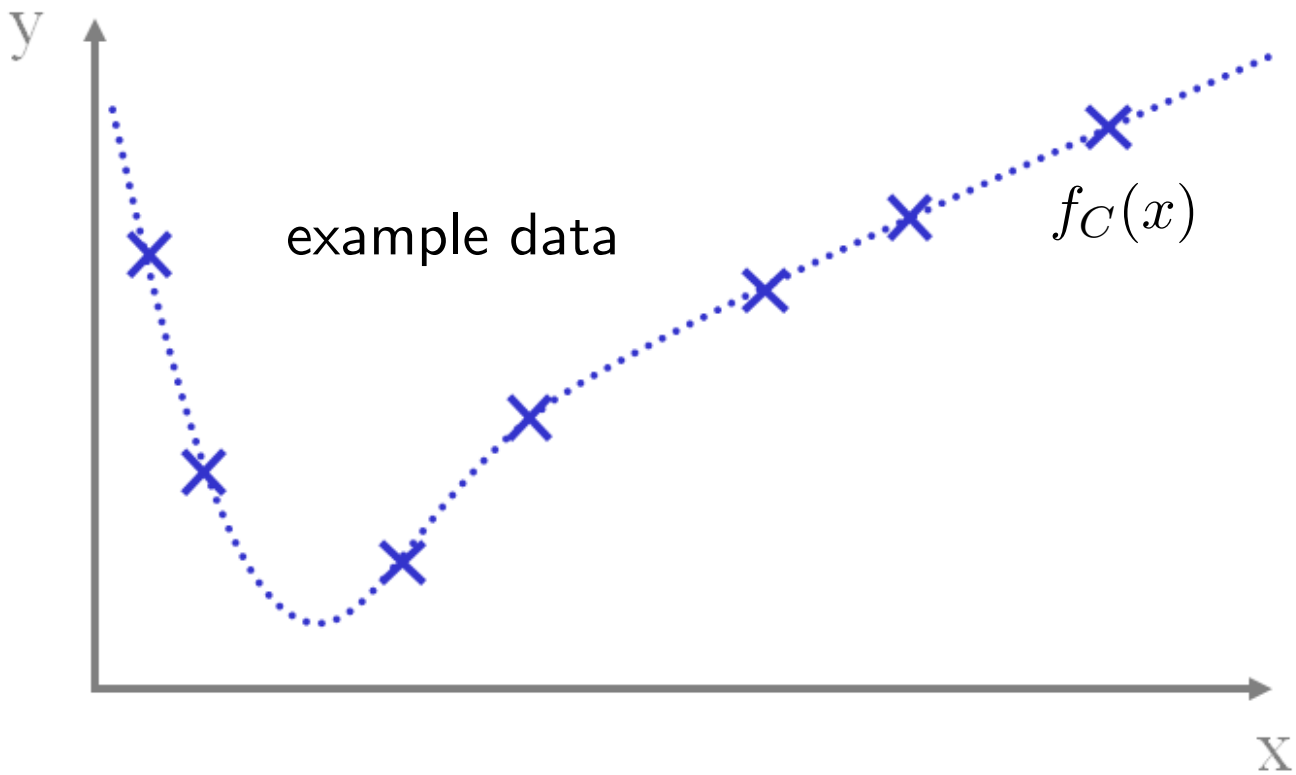
Good fit to data is not necessarily the same as a good fit to concept!



©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Overfitting

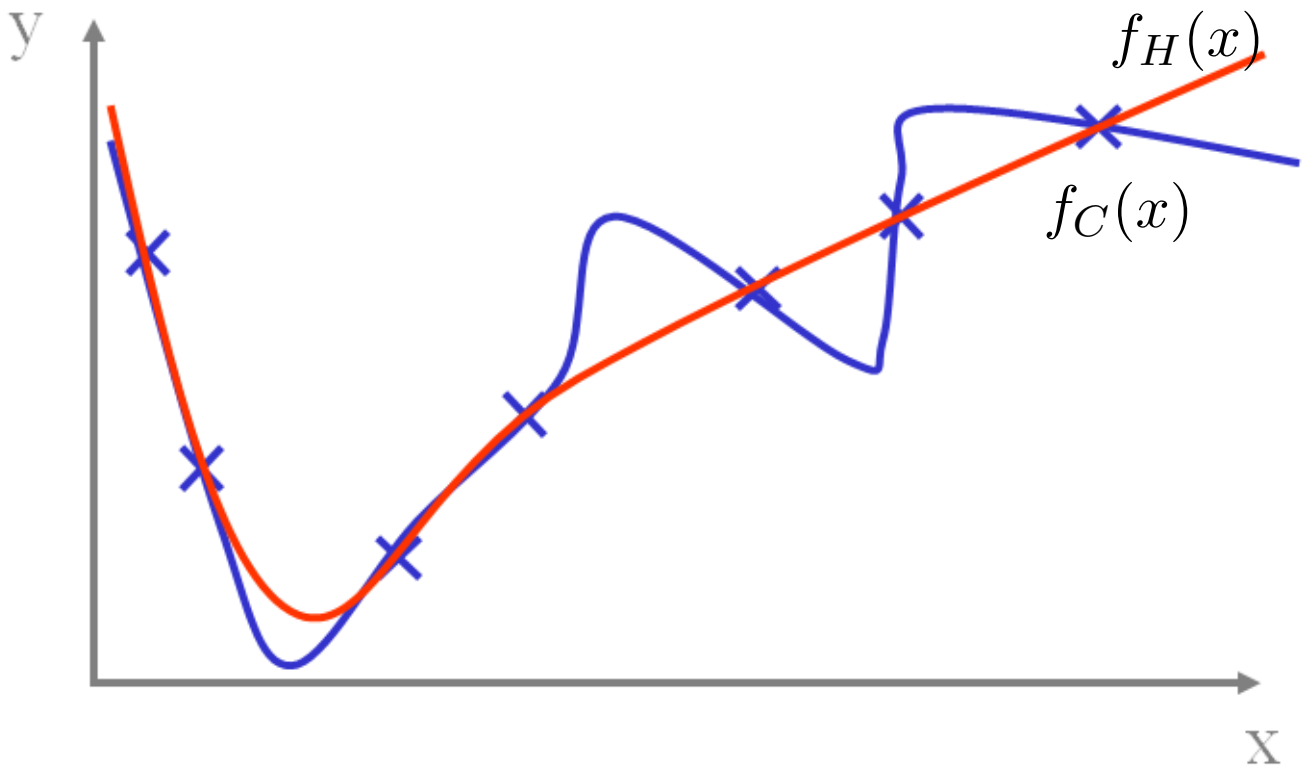
Good fit to data is not necessarily the same as a good fit to concept!



©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Overfitting

Good fit to data is not necessarily the same as a good fit to concept!



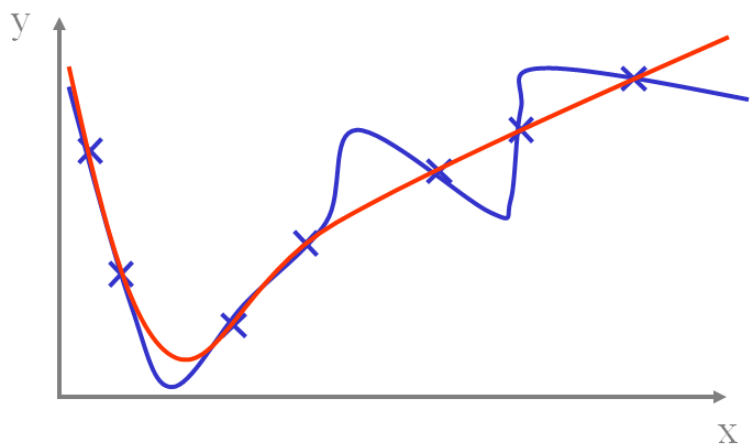
©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Overfitting

Overfitting

Fitting the noise rather than fitting the underlying relationship.

Typical indicator for overfitting: "Perfect fit" e.g. the error gets near to zero.
ONE solution: Choose a less flexible model.



©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Model error

- There are 4 parts of potential error origins, which sum up to the overall error meassure
- Error = Experimental error
  + Sample error
  + Model error
  + Algorithmic error

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Experimental Error

- The **pure error** or **experimental error** is inherent in the data and due to noise, random variations, imprecise measurements or the influence of hidden variables that cannot be observed.
- It is impossible to overcome this error by the choice of a suitable model.
- Also called **intrinsic error**.
- In the context of classification problems it is also called **Bayes error**.

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

ROC curves

How can a classifier be judged when no explicit cost matrix is known and the misclassification rate might not be a good choice?

Consider a two class problem. (positive and negative)

Very often, classifiers provide a score for each class.
e.g. a likelihood

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

ROC curves

Example

Assume, only the attribute Sepal Length should be used to distinguish Iris versicolor from Iris virginica by a simple rule of the form

$$\text{If Sepal Length} < c\text{, then versicolor, otherwise virginica}$$

where c can be chosen in the range of the values of the attribute Sepal Length.

The attribute Sepal Length provides the "score" in this case.

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

ROC curves

S. Length	Species	S. Length	Species	S. Length	Species
4.9	versicolor	5.6	versicolor	5.9	versicolor
4.9	virginica	5.6	versicolor	5.9	versicolor
5.0	versicolor	5.6	virginica	5.9	virginica
5.0	versicolor	5.7	versicolor	6.0	versicolor
5.1	versicolor	5.7	versicolor	6.0	versicolor
5.2	versicolor	5.7	versicolor	6.0	versicolor
5.4	versicolor	5.7	versicolor	6.0	versicolor
5.5	versicolor	5.7	versicolor	6.0	virginica
5.5	versicolor	5.7	virginica	6.0	virginica
5.5	versicolor	5.8	versicolor	6.1	versicolor
5.5	versicolor	5.8	versicolor	6.1	versicolor
5.5	versicolor	5.8	versicolor	6.1	versicolor
5.6	versicolor	5.8	virginica	6.1	versicolor
5.6	versicolor	5.8	virginica	6.1	virginica
5.6	versicolor	5.8	virginica	⋮	⋮

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

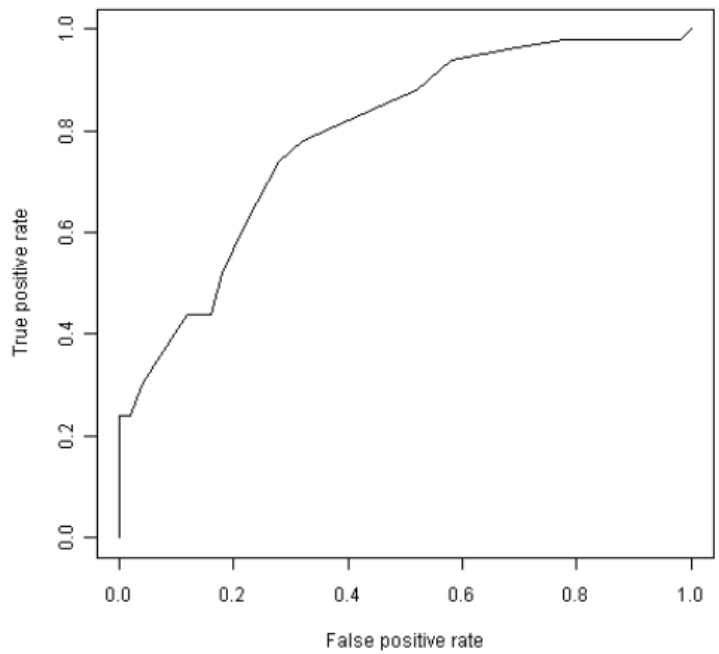
ROC curves

- Consider versicolor as the "positive" and virginica as the "negative" class.
- The higher the threshold c is chosen, the more instances are classified as "positive" (versicolor).
- there are four possibilities :
  - true positive TP classified as Pos and is Pos
  - false positive FP classified as Pos and is Neg
  - true negative TN classified as Neg and is Neg
  - false negative FN classified as Neg and is Pos
- Increasing the threshold c, will increase the true positives as well as the true negatives.
- Ideal case: only true positives and no false negatives

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

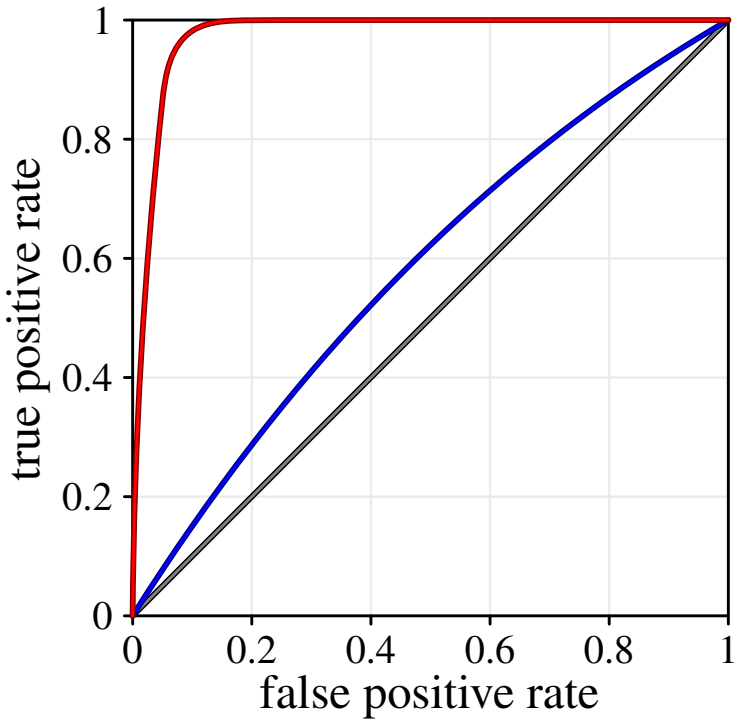
ROC curves

The ROC curve (receiver operating characteristic curve) draws the false positive rate against the false negative rate (depending on the choice of the threshold c).



©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

ROC curves



A ROC curve of a better performing and a classifier with a performance. The area under curve (AUC), i.e. the area under the ROC curve, is an indicator how well the classifier solves the problem. The larger the area, the better the solution for the classification problem.

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Confusion matrix

A confusion matrix is a table where the rows represent the true classes and the columns the predicted classes. Each entry specifies how many objects from a given class are classified into the class of the corresponding column.

true class	predicted class		
	Iris setosa	Iris versicolor	Iris virginica
Iris setosa	50	0	0
Iris versicolor	0	47	3
Iris virginica	0	2	48

A possible confusion matrix for the Iris data set

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Sample Error

Sample Error

- The data is not a perfect representation of the underlying data
- The smaller the sample the smaller the probability for a perfect model

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Sample Error

Sample Error

- The data is not a perfect representation of the underlying data
- The smaller the sample the smaller the probability for a perfect model

Example

- Throw a dice
- Sample Bias: what if the dice was not fair?

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

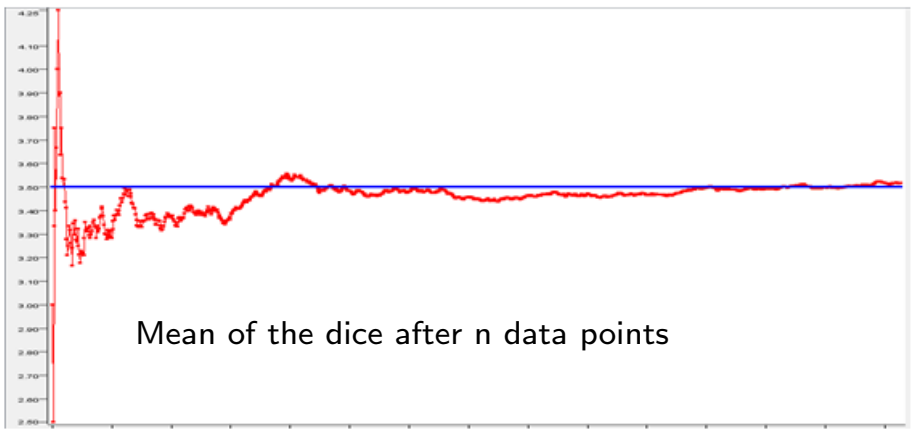
Sample Error

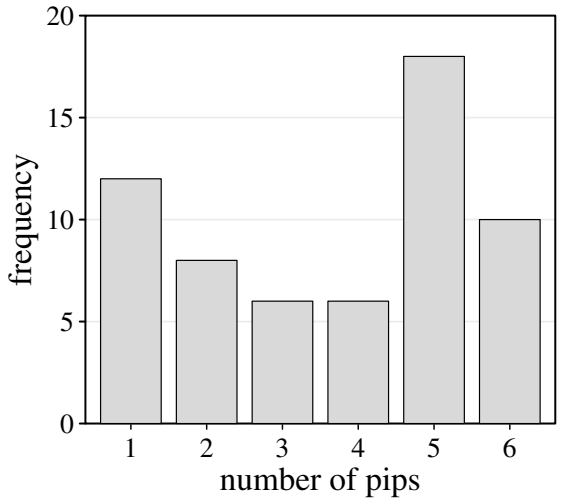
Sample Error

- The data is not a perfect representation of the underlying data
- The smaller the sample the smaller the probability for a perfect model

Example

- Throw a dice
- Sample Bias: what if the dice was not fair?





©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Model Error

There are different models for the data

- simpler model $\Longrightarrow$ bigger error
- more complex model $\Longrightarrow$ overfitting and larger error on new data
- type of model $\Longrightarrow$ different "fit" to data

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Algorithmic Error

- Based on the selected algorithm

Algorithmic Error

- Based on the selected algorithm
- For example

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Algorithmic Error

- Based on the selected algorithm
- For example
  - Gradient descend $\Longrightarrow$ local minima

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Algorithmic Error

- Based on the selected algorithm
- For example
  - Gradient descend $\Longrightarrow$ local minima
  - Randomized method $\Longrightarrow$ too much randomness

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Algorithmic Error

- Based on the selected algorithm
- For example
  - Gradient descend $\Longrightarrow$ local minima
  - Randomized method $\Longrightarrow$ too much randomness
- The algorithmic error can often not be measured (several runs of similarly biased)

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Algorithmic Error

- Based on the selected algorithm
- For example
  - Gradient descend $\Longrightarrow$ local minima
  - Randomized method $\Longrightarrow$ too much randomness
- The algorithmic error can often not be measured (several runs of similarly biased)
- Normalilty: we assume that our algorithm is good enough

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Algorithmic Error

- Based on the selected algorithm
- For example
  - Gradient descend $\Longrightarrow$ local minima
  - Randomized method $\Longrightarrow$ too much randomness
- The algorithmic error can often not be measured (several runs of similarly biased)
- Normalilty: we assume that our algorithm is good enough
  - (otherwise : choose another)

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Machine Learning Bias and Variance

Machine Learners have a slightly different view:

- Machine Learning Bias: Model and Algorithmic Error
- Variance: Intrinsic and Sample Error

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Machine Learning Bias and Variance

Machine Learners have a slightly different view:

- Machine Learning Bias: Model and Algorithmic Error
- Variance: Intrinsic and Sample Error

Alternative View: Error Decomposition:

$$MSE = Var(\theta^*) + (Bias\,(\theta^*))^2$$

(θ^* is an estimator for unknown parameter(s) θ.
MSE: Mean Squared Error)

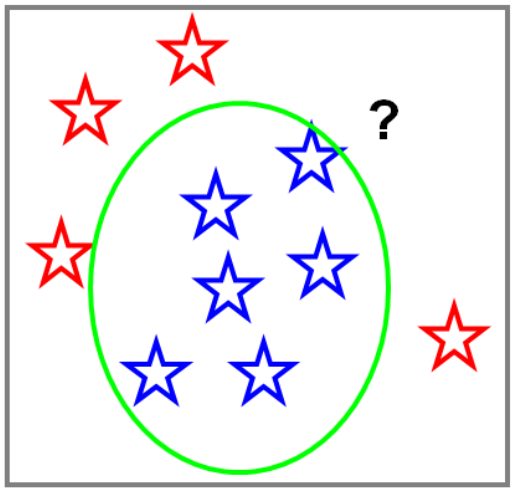
©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

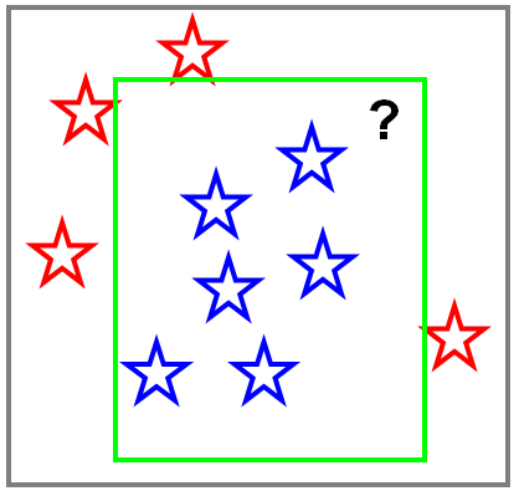
Learning without Bias?

- Can we find a good model without model or algorithmic bias?

Learning without Bias?

- Can we find a good model without model or algorithmic bias?
- Remember? Version Space Learning

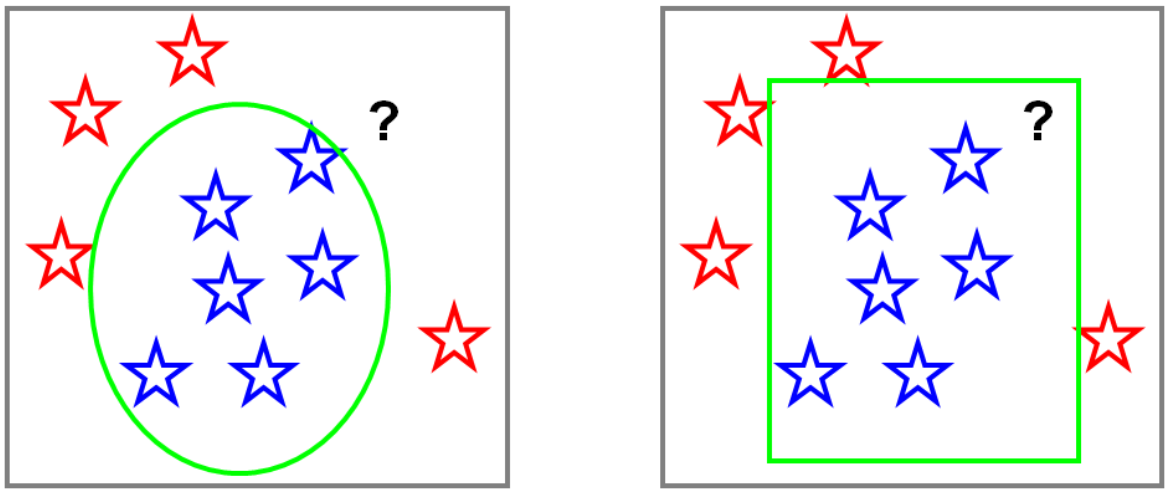




©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Learning without Bias?

- Can we find a good model without model or algorithmic bias?
- Remember? Version Space Learning



We cannot learn without a bias

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Model Validation

- The error for unseen data will most probably always be bigger than for the data used for training.
- How do we find out which model is actually suited best to our problem?

Training and Test Data

- Split data into two subsets: training and test data
- Train your model on the training data and measure the model quality on the test data
- Typically 2/3 training 1/3 test (usually more training)
- Splitting strategies
  - Random (distribution in both sets should be roughly same)
  - Stratification (i.e. the distribution of one class should remain)
- Split into training, test and validation
  - Choose for each model kind the best based on the test data
  - Test the best models on the validation data set.

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Validation of Models

Estimating the Generalization ability of a Model using a separate data set T_{eval}

$$T = T_{train} \cup T_{eval}$$

$$T_{train} \cap T_{eval} = \emptyset$$

$$P(E) = \approx \frac{\sum_{x \in T_{eval}} g(H(x), C(x))}{|T_{eval}|}$$

$g(x, y) = 1$ if $x = y$ and $g(x, y) = 0$ if $x \neq y$

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Cross-Validation

- **Split the data multiple times to validate the results, to dimish the effect of the single estimation**
- Use a combination of the received models, or the best.
- **K-fold Cross-Validation** (e.g. k=10):
  - Divide into k subsets
  - Test data is always another subset, training data the rest
  - Average of the k model error is supposed as the model error
- **Leave-One-Out Method**:

Cross-Validation

- Split the data multiple times to validate the results, to dimish the effect of the single estimation
- **Use a combination of the received models, or the best.**
- **K-fold Cross-Validation** (e.g. k=10):
  - Divide into k subsets
  - Test data is always another subset, training data the rest
  - Average of the k model error is supposed as the model error
- **Leave-One-Out Method**:

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Cross-Validation

- Split the data multiple times to validate the results, to dimish the effect of the single estimation
- Use a combination of the received models, or the best.
- **K-fold Cross-Validation (e.g. k=10):**
  - Divide into k subsets
  - Test data is always another subset, training data the rest
  - Average of the k model error is supposed as the model error
- Leave-One-Out Method:

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Cross-Validation

- Split the data multiple times to validate the results, to dimish the effect of the single estimation
- Use a combination of the received models, or the best.
- **K-fold Cross-Validation** (e.g. k=10):
  - **Divide into k subsets**
  - Test data is always another subset, training data the rest
  - Average of the k model error is supposed as the model error
- **Leave-One-Out Method**:

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Cross-Validation

- Split the data multiple times to validate the results, to dimish the effect of the single estimation
- Use a combination of the received models, or the best.
- **K-fold Cross-Validation** (e.g. k=10):
  - Divide into k subsets
  - **Test data is always another subset, training data the rest**
  - Average of the k model error is supposed as the model error
- **Leave-One-Out Method**:
  - For very small data sets
  - Use everything except of one data point for training
  - This single data point is used for testing

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Cross-Validation

- Split the data multiple times to validate the results, to dimish the effect of the single estimation
- Use a combination of the received models, or the best.
- **K-fold Cross-Validation** (e.g. k=10):
  - Divide into k subsets
  - Test data is always another subset, training data the rest
  - **Average of the k model error is supposed as the model error**
- **Leave-One-Out Method**:
  - For very small data sets
  - Use everything except of one data point for training
  - This single data point is used for testing

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Cross-Validation

- Split the data multiple times to validate the results, to dimish the effect of the single estimation
- Use a combination of the received models, or the best.
- **K-fold Cross-Validation** (e.g. k=10):
  - Divide into k subsets
  - Test data is always another subset, training data the rest
  - Average of the k model error is supposed as the model error
- **Leave-One-Out Method:**
  - **For very small data sets**
  - **Use everything except of one data point for training**
  - **This single data point is used for testing.**

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Cross-Validation

- Split the data multiple times to validate the results, to dimish the effect of the single estimation
- Use a combination of the received models, or the best.
- **K-fold Cross-Validation** (e.g. k=10):
  - Divide into k subsets
  - Test data is always another subset, training data the rest
  - Average of the k model error is supposed as the model error
- **Leave-One-Out Method**:
  - **For very small data sets**
  - Use everything except of one data point for training
  - This single data point is used for testing.

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Cross-Validation

- Split the data multiple times to validate the results, to dimish the effect of the single estimation
- Use a combination of the received models, or the best.
- **K-fold Cross-Validation** (e.g. k=10):
  - Divide into k subsets
  - Test data is always another subset, training data the rest
  - Average of the k model error is supposed as the model error
- **Leave-One-Out Method**:
  - For very small data sets
  - **Use everything except of one data point for training**
  - This single data point is used for testing.

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Cross-Validation

- Split the data multiple times to validate the results, to dimish the effect of the single estimation
- Use a combination of the received models, or the best.
- **K-fold Cross-Validation** (e.g. k=10):
  - Divide into k subsets
  - Test data is always another subset, training data the rest
  - Average of the k model error is supposed as the model error
- **Leave-One-Out Method**:
  - For very small data sets
  - Use everything except of one data point for training
  - **This single data point is used for testing.**

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Bootstrapping

- **Overall goal: Draw samples multiple times to underline your model (e.g. with the variance of the parameters)**
- Pseudo algorithm:
  - Draw k bootstrap samples from the data
  - Learn the model on each sample
  - Calculate the mean and standard deviation
  - Small standard deviation supports the model.
- Bagging (Use Bootstrapping to improve the results)
- The final parameters can be achieved by averaging the k sets of parameters
- We will discuss bagging and more ensemble methods in a later lecture.

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Bootstrapping

- Overall goal: Draw samples multiple times to underline your model (e.g. with the variance of the parameters)
- **Pseudo algorithm:**
  - **Draw k bootstrap samples from the data**
  - **Learn the model on each sample**
  - **Calculate the mean and standard deviation**
  - **Small standard deviation supports the model.**
- Bagging (Use Bootstrapping to improve the results)
- The final parameters can be achieved by averaging the k sets of parameters
- We will discuss bagging and more ensemble methods in a later lecture.

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Bootstrapping

- Overall goal: Draw samples multiple times to underline your model (e.g. with the variance of the parameters)
- Pseudo algorithm:
  - **Draw k bootstrap samples from the data**
  - Learn the model on each sample
  - Calculate the mean and standard deviation
  - Small standard deviation supports the model.
- Bagging (Use Bootstrapping to improve the results)
- The final parameters can be achieved by averaging the k sets of parameters
- We will discuss bagging and more ensemble methods in a later lecture.

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Bootstrapping

- Overall goal: Draw samples multiple times to underline your model (e.g. with the variance of the parameters)
- Pseudo algorithm:
  - Draw k bootstrap samples from the data
  - **Learn the model on each sample**
  - Calculate the mean and standard deviation
  - Small standard deviation supports the model.
- Bagging (Use Bootstrapping to improve the results)
- The final parameters can be achieved by averaging the k sets of parameters
- We will discuss bagging and more ensemble methods in a later lecture.

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Bootstrapping

- Overall goal: Draw samples multiple times to underline your model (e.g. with the variance of the parameters)
- Pseudo algorithm:
  - Draw k bootstrap samples from the data
  - Learn the model on each sample
  - **Calculate the mean and standard deviation**
  - Small standard deviation supports the model.
- Bagging (Use Bootstrapping to improve the results)
- The final parameters can be achieved by averaging the k sets of parameters
- We will discuss bagging and more ensemble methods in a later lecture.

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Bootstrapping

- Overall goal: Draw samples multiple times to underline your model (e.g. with the variance of the parameters)
- Pseudo algorithm:
  - Draw k bootstrap samples from the data
  - Learn the model on each sample
  - Calculate the mean and standard deviation
  - **Small standard deviation supports the model.**
- Bagging (Use Bootstrapping to improve the results)
- The final parameters can be achieved by averaging the k sets of parameters
- We will discuss bagging and more ensemble methods in a later lecture.

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Bootstrapping

- Overall goal: Draw samples multiple times to underline your model (e.g. with the variance of the parameters)
- Pseudo algorithm:
  - Draw k bootstrap samples from the data
  - Learn the model on each sample
  - Calculate the mean and standard deviation
  - Small standard deviation supports the model.
- **Bagging (Use Bootstrapping to improve the results)**
- The final parameters can be achieved by averaging the k sets of parameters
- We will discuss bagging and more ensemble methods in a later lecture.

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Bootstrapping

- Overall goal: Draw samples multiple times to underline your model (e.g. with the variance of the parameters)
- Pseudo algorithm:
  - Draw k bootstrap samples from the data
  - Learn the model on each sample
  - Calculate the mean and standard deviation
  - Small standard deviation supports the model.
- Bagging (Use Bootstrapping to improve the results)
- **The final parameters can be achieved by averaging the k sets of parameters**
- We will discuss bagging and more ensemble methods in a later lecture.

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Bootstrapping

- Overall goal: Draw samples multiple times to underline your model (e.g. with the variance of the parameters)
- Pseudo algorithm:
  - Draw k bootstrap samples from the data
  - Learn the model on each sample
  - Calculate the mean and standard deviation
  - Small standard deviation supports the model.
- Bagging (Use Bootstrapping to improve the results)
- The final parameters can be achieved by averaging the k sets of parameters
- **We will discuss bagging and more ensemble methods in a later lecture.**

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Measures for Model Complexity

- The goal is to fulfill Occam's razor :
  Choose the simplest model that still explains the data.
- How do we measure the complexity of the model?
- Two ideas:
  - The Minimum Description Length Principle
  - Akaike's and the Bayesian Information Criterion

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

The Minimum Description Length Principle

- **Basic Idea: Regard the routine as a way of data compression**
- To recreate the data, two things are needed:
  - The decompression rule
  - The compressed data
- The quality is than measured by the number of bits needed to code these two
- The simplest cases
- Goal: find a solution inbetween

The Minimum Description Length Principle

- Basic Idea: Regard the routine as a way of data compression
- **To recreate the data, two things are needed:**
  - **The decompression rule**
  - **The compressed data**
- The quality is than measured by the number of bits needed to code these two
- The simplest cases
- Goal: find a solution inbetween

The Minimum Description Length Principle

- Basic Idea: Regard the routine as a way of data compression
- To recreate the data, two things are needed:
  - **The decompression rule**
  - The compressed data
- The quality is than measured by the number of bits needed to code these two
- The simplest cases
- Goal: find a solution inbetween

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

The Minimum Description Length Principle

- Basic Idea: Regard the routine as a way of data compression
- To recreate the data, two things are needed:
  - The decompression rule
  - **The compressed data**
- The quality is than measured by the number of bits needed to code these two
- The simplest cases
  - Compressed data of size 1, by adding the data to the rule (the compressed data is 1 than data as original)
  - Decompression rule of size 1, by saving the real data as the compressed
- Goal: find a solution inbetween

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

The Minimum Description Length Principle

- Basic Idea: Regard the routine as a way of data compression
- To recreate the data, two things are needed:
  - The decompression rule
  - The compressed data
- **The quality is than measured by the number of bits needed to code these two**
- The simplest cases
  - Compressed data of size 1, by adding the data to the rule (If compressed data == 1 than data == original)
  - Decompression rule of size 1, by saving the real data as the compressed
- Goal: find a solution inbetween

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

The Minimum Description Length Principle

- Basic Idea: Regard the routine as a way of data compression
- To recreate the data, two things are needed:
  - The decompression rule
  - The compressed data
- The quality is than measured by the number of bits needed to code these two
- **The simplest cases**
  - **Compressed data of size 1, by adding the data to the rule (If compressed data = 1 than data = original)**
  - **Decompression rule of size 1, by saving the real data as the compressed**
- Goal: find a solution inbetween

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

The Minimum Description Length Principle

- Basic Idea: Regard the routine as a way of data compression
- To recreate the data, two things are needed:
  - The decompression rule
  - The compressed data
- The quality is than measured by the number of bits needed to code these two
- The simplest cases
  - **Compressed data of size 1, by adding the data to the rule (If compressed data = 1 than data = original)**
  - Decompression rule of size 1, by saving the real data as the compressed
- Goal: find a solution inbetween

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

The Minimum Description Length Principle

- Basic Idea: Regard the routine as a way of data compression
- To recreate the data, two things are needed:
  - The decompression rule
  - The compressed data
- The quality is than measured by the number of bits needed to code these two
- The simplest cases
  - Compressed data of size 1, by adding the data to the rule (If compressed data = 1 than data = original)
  - **Decompression rule of size 1, by saving the real data as the compressed**
- Goal: find a solution inbetween

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

The Minimum Description Length Principle

- Basic Idea: Regard the routine as a way of data compression
- To recreate the data, two things are needed:
  - The decompression rule
  - The compressed data
- The quality is than measured by the number of bits needed to code these two
- The simplest cases
  - Compressed data of size 1, by adding the data to the rule (If compressed data = 1 than data = original)
  - Decompression rule of size 1, by saving the real data as the compressed
- **Goal: find a solution inbetween**

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Example for the MDL

- We restrict the coding of the decimals to two digits reversed and ignore the sign
- Example:
  - code 0.73 as 37
  - 1.23 as 321
  - -0.06 as 6

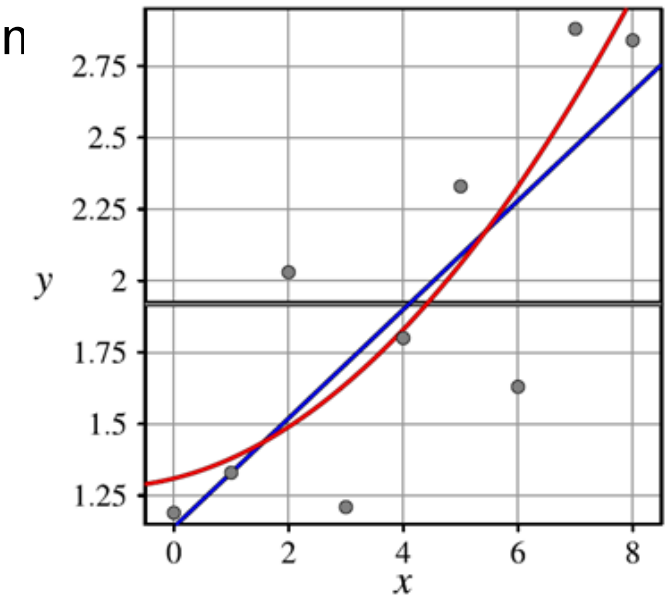


Table 5.7 The errors for the three regression functions

ID	1	2	3	4	5	6	7	8	9
$y = 1.92$	0.73	0.59	−0.11	0.71	0.12	−0.41	0.29	−0.96	−0.92
$y = 1.14 + 0.19x$	−0.05	0.00	−0.51	0.50	0.10	−0.24	0.65	−0.41	−0.18
$y = 1.31 + 0.05x + 0.02x^2$	0.12	0.05	−0.54	0.43	0.03	−0.27	0.70	−0.24	0.15

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Example for the MDL

- Coding the constant function:
- 3 digits for the value 1.92
- And for each "error" 2 digits resolves in:
- $21 = 3 + 9 * 2$

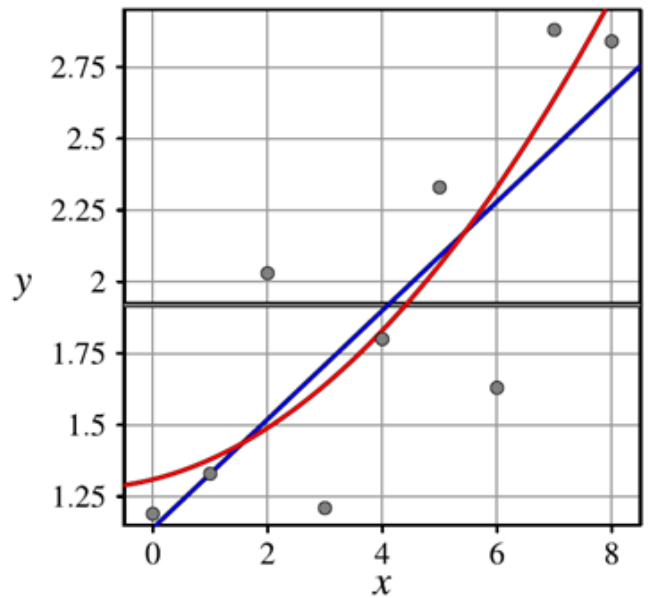


Table 5.7 The errors for the three regression functions

ID	1	2	3	4	5	6	7	8	9
$y = 1.92$	0.73	0.59	−0.11	0.71	0.12	−0.41	0.29	−0.96	−0.92
$y = 1.14 + 0.19x$	−0.05	0.00	−0.51	0.50	0.10	−0.24	0.65	−0.41	−0.18
$y = 1.31 + 0.05x + 0.02x^2$	0.12	0.05	−0.54	0.43	0.03	−0.27	0.70	−0.24	0.15

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Example for the MDL

- Coding the linear function:
- 3 digits for the value 1.14 and 2 digits for the offset 0.19
- Error : 7 times 2 digits and for 1: 1 digit and for 2 : 0 digits
- $20 = 5 + 7 * 2 + 1 + 0$

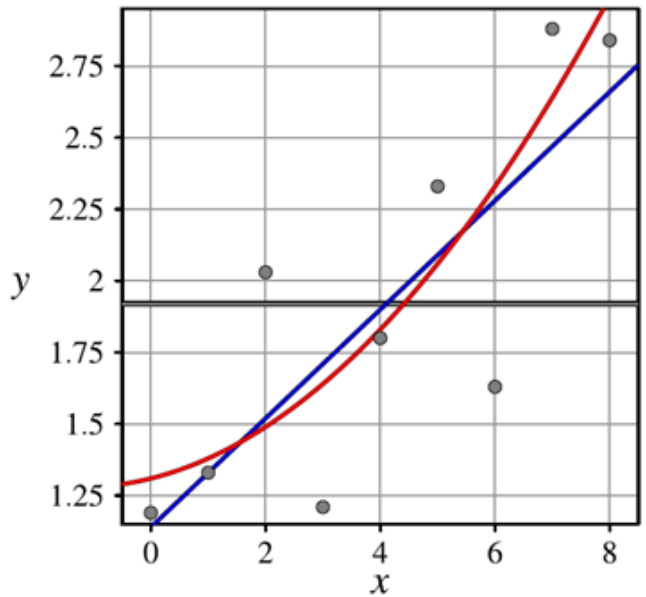


Table 5.7 The errors for the three regression functions

ID	1	2	3	4	5	6	7	8	9
$y = 1.92$	0.73	0.59	−0.11	0.71	0.12	−0.41	0.29	−0.96	−0.92
$y = 1.14 + 0.19x$	−0.05	0.00	−0.51	0.50	0.10	−0.24	0.65	−0.41	−0.18
$y = 1.31 + 0.05x + 0.02x^2$	0.12	0.05	−0.54	0.43	0.03	−0.27	0.70	−0.24	0.15

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Example for the MDL

- Coding the quadratic function:
- 3 digits for the value 1.31 and 1 digit for the raising 0.05 and 1 for 0.02
- Error : 7 times 2 digits and for ID 2 and 5 only 1 digit
- $21 = 5 + 7 * 2 + 2 * 1$

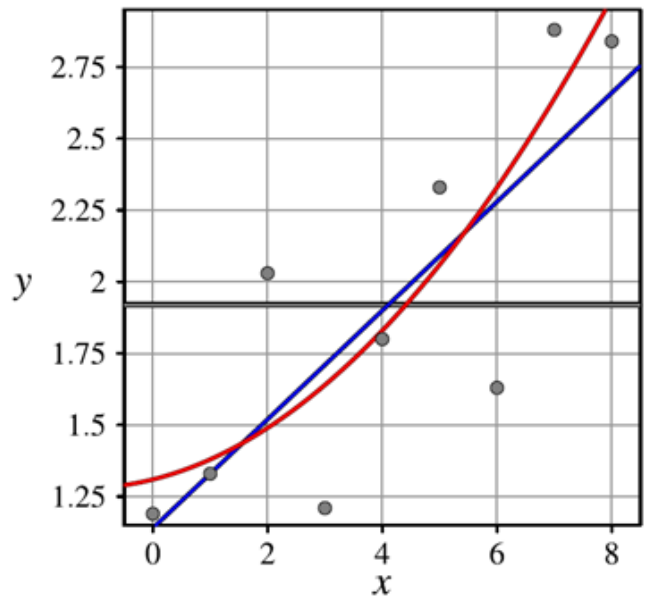


Table 5.7 The errors for the three regression functions

ID	1	2	3	4	5	6	7	8	9
$y = 1.92$	0.73	0.59	−0.11	0.71	0.12	−0.41	0.29	−0.96	−0.92
$y = 1.14 + 0.19x$	−0.05	0.00	−0.51	0.50	0.10	−0.24	0.65	−0.41	−0.18
$y = 1.31 + 0.05x + 0.02x^2$	0.12	0.05	−0.54	0.43	0.03	−0.27	0.70	−0.24	0.15

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Example for the MDL

Summary:

- Constant = 21
- Linear = 20
- Quadratic = 21
- Recommendation would be to use the model "linear function" for this data

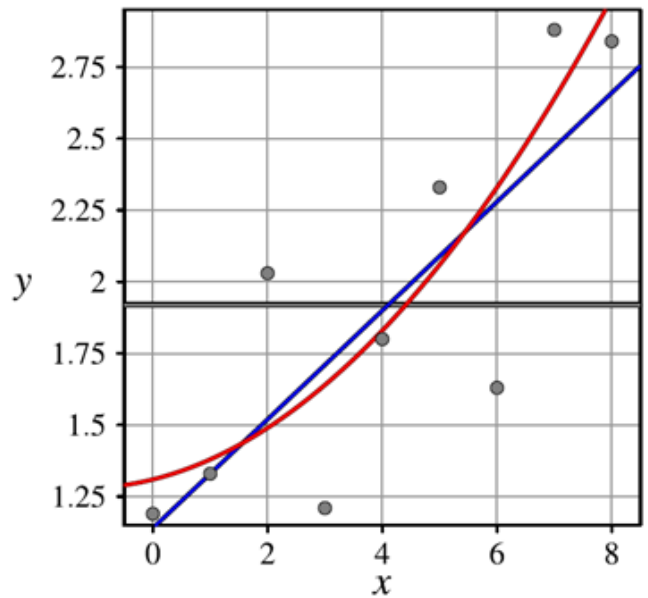


Table 5.7 The errors for the three regression functions

ID	1	2	3	4	5	6	7	8	9
$y = 1.92$	0.73	0.59	−0.11	0.71	0.12	−0.41	0.29	−0.96	−0.92
$y = 1.14 + 0.19x$	−0.05	0.00	−0.51	0.50	0.10	−0.24	0.65	−0.41	−0.18
$y = 1.31 + 0.05x + 0.02x^2$	0.12	0.05	−0.54	0.43	0.03	−0.27	0.70	−0.24	0.15

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Example for the MDL

- We restrict the coding of the decimals to two digits reversed and ignore the sign
- Example:
  - code 0.73 as 37
  - 1.23 as 321
  - -0.06 as 6

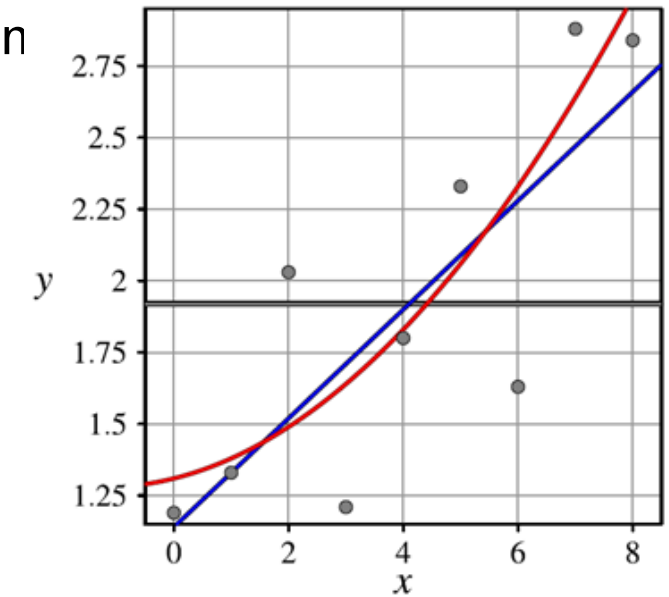


Table 5.7 The errors for the three regression functions

ID	1	2	3	4	5	6	7	8	9
$y = 1.92$	0.73	0.59	−0.11	0.71	0.12	−0.41	0.29	−0.96	−0.92
$y = 1.14 + 0.19x$	−0.05	0.00	−0.51	0.50	0.10	−0.24	0.65	−0.41	−0.18
$y = 1.31 + 0.05x + 0.02x^2$	0.12	0.05	−0.54	0.43	0.03	−0.27	0.70	−0.24	0.15

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä

Other Model Selection Criteria

- Akaike's Information Criterion measures
  - model complexity by number of parameters (k)
  - Fit to data by probability of data generated by model (L)
  - $AIC = 2k - 2\ln(L)$
- Notes:
  For error assumed to be normal distributed, MSE models the likelihood directly.
- Other Measures: Bayesian information criterion.

©Michael R. Berthold, Christian Borgelt, Frank Höppner, Frank Klawonn and Iris Adä