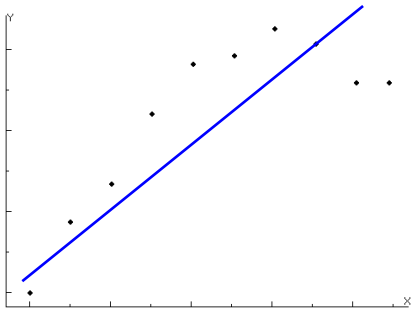
4

Model evaluation and selection

Resampling methods

Example: models of increasing complexity

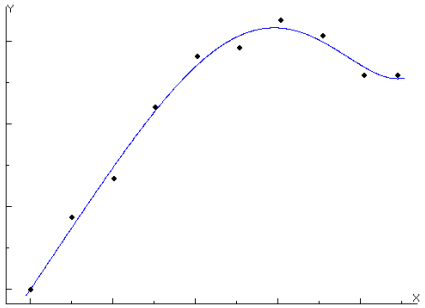
R^2 calculated in training dataset (like in the previous lecture)



$\hat{y} = a_1 + a_2 x$

($k = 2$)

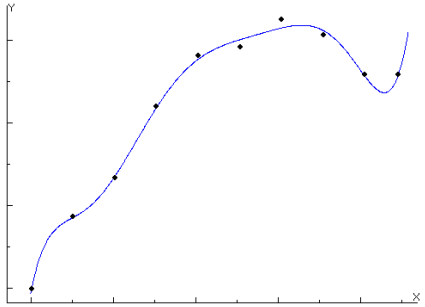
$R^2 \approx 0.6$



$\hat{y} = a_1 + a_2 x + a_3 x^4 + a_4 x^7$

($k = 4$)

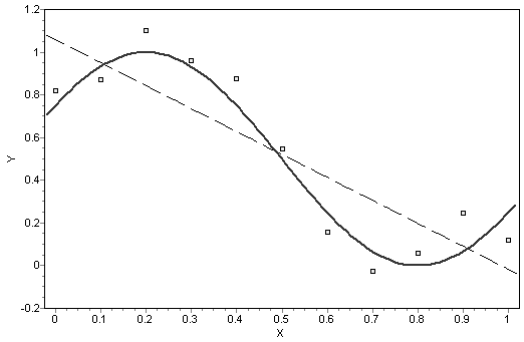
$R^2 \approx 0.96$



$\hat{y} = a_1 + a_2 x + a_3 x^2 + a_4 x^3 + a_5 x^4 + a_6 x^5 + a_7 x^6 + a_8 x^7$

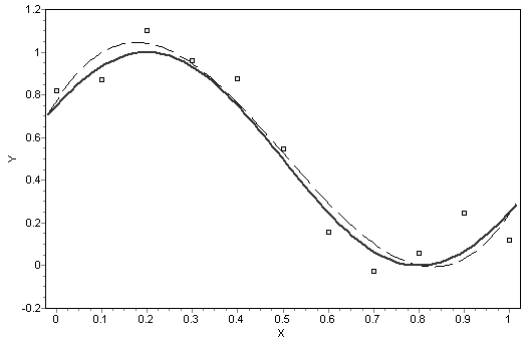
($k = 8$)

$R^2 \approx 0.99$



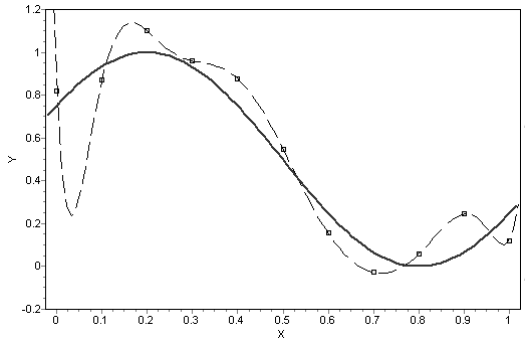
Polynomial of degree 1 ($k = 2$)

$R^2 = 0.72$



Polynomial of degree 3 ($k = 4$)

$R^2 = 0.92$



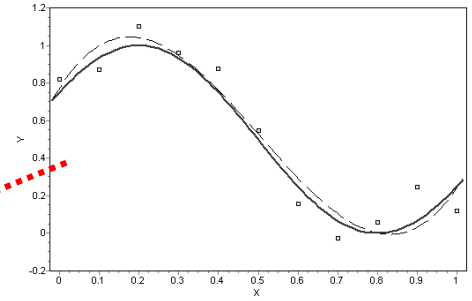
Polynomial of degree 10 ($k = 11$)

$R^2 = 1$

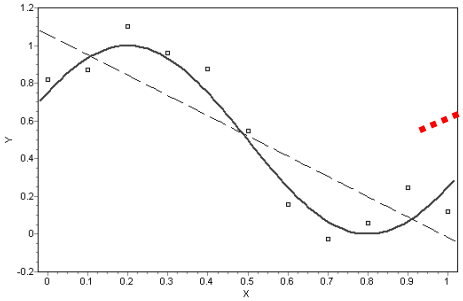
When the number of parameters is equal to the number of data points the curve goes exactly through all the points

Underfitting and overfitting

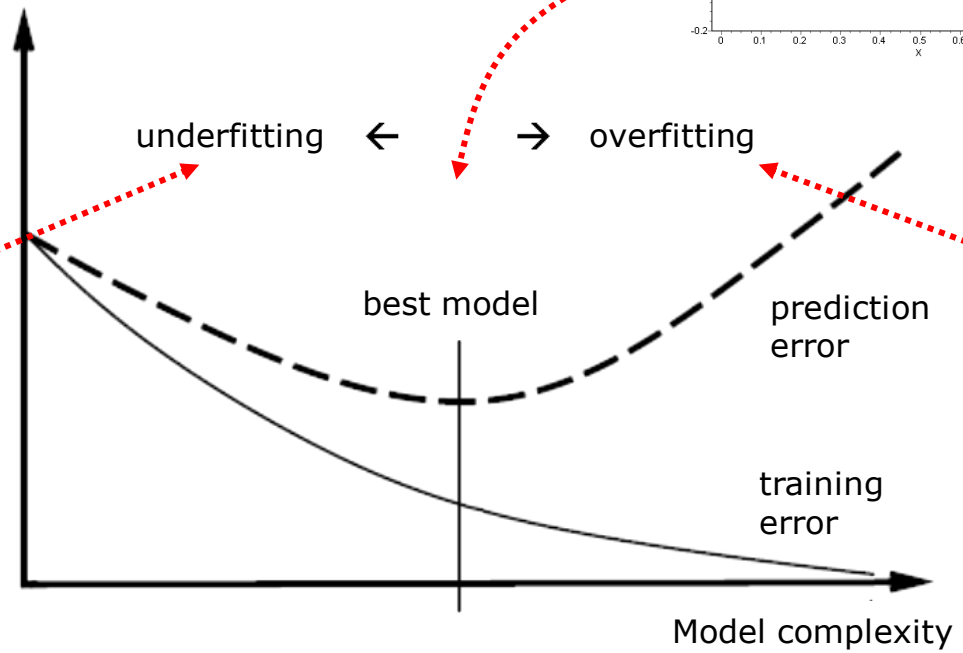
Training dataset alone does not give us sufficient information about model's predictive performance.

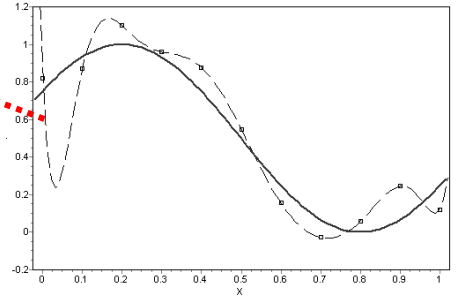


$\hat{y} = a_1 + a_2x + a_3x^2 + a_4x^3$
$(k = 4)$



$\hat{y} = a_1 + a_2x$
$(k = 2)$





$\hat{y} = a_1 + a_2x + a_3x^2 + a_4x^3 + a_5x^4 + a_6x^5 + a_7x^6 + a_8x^7 + a_8x^8 + a_8x^9 + a_8x^{10} + a_8x^{11}$
$(k = 11)$

Here "error" means, e.g., SAE, MAE, SSE, MSE, RMSE.
For R^2, this picture should be vertically flipped as R^2 is the opposite of error.

Underfitting and overfitting

- Underfitting and overfitting are universal learning problems that are important for all types of learning
- It's usually tempting to increase model's complexity. Incompetent use of evaluation criteria can easily hide the fact that a model is overfitted
- Advantages of simple models:
  - Better predictive performance (except if underfitted)
  - A simple model might use fewer features. This means that we will have to do fewer measurements or other kinds of data gathering to do predictions with such model
  - Work with simpler models is faster and needs less computer memory
  - Simpler models are easier to visualize and interpret

Estimation of true predictive performance

- How do we estimate true predictive performance of a model?
  - The training dataset alone can't help us

Idea

We know that when we use a model for prediction, we calculate y for given x which may not be (and usually is not) given in the training dataset.

What if we could "simulate" this process?

We could use additional evaluation dataset which is not included in the training dataset.

Validation set and test set

- If the mentioned additional dataset is used for model selection the dataset is called validation set. If it is used for final estimation of prediction error it is called test set. (*in literature this terminology is not always consistent*)
- The aims of model selection and final evaluation are different:
  - Model selection aims to select the best model by evaluating predictive performance of models-candidates

    (here we mostly pay attention to the relative differences between evaluations of different models)
  - Final estimation of the true prediction error aims to estimate the true prediction error as closely as possible in this way giving information about the expected error of the model in its future applications
- *(The more models you consider, the lower is the probability that the estimated prediction error is near the real one)*

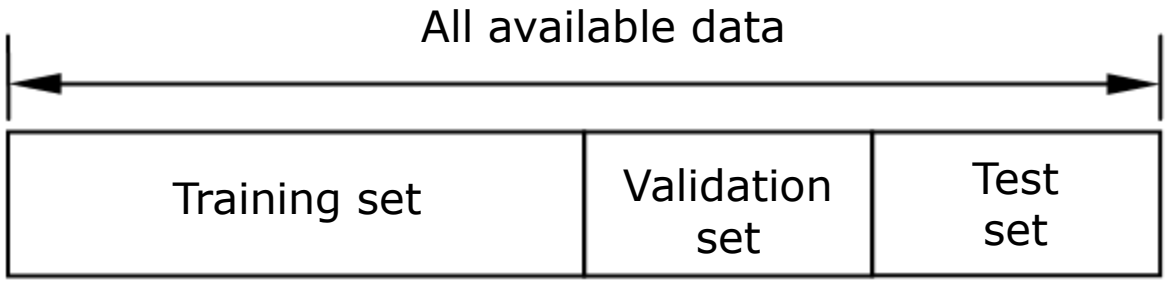
Resampling methods

- In both cases (model selection and true prediction error estimation) we can use the so-called resampling methods
- The idea: evaluate the model on (additional) data that was not included in the training set
- The data may be
  - Additionally generated (but this can be very expensive or even impossible to do)
  - Simply subtracted from the already existing full dataset and set aside
- It's important that these (additional) data points would not be included in the training set, i.e., the data points would not be used for building the models (estimation of model parameters etc.)

The three datasets

- Basic idea – divide the whole dataset into three subsets (or two, if either validation or testing is not required):
  - Training set

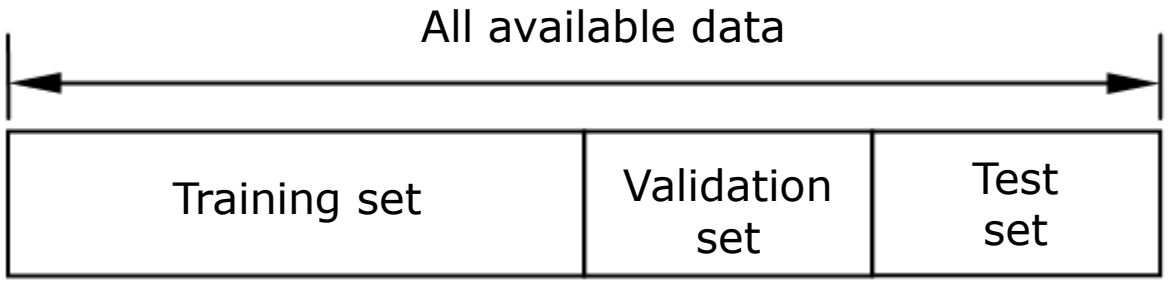
    (usually the largest set, especially if data is scarce)
  - Validation set
  - Test set



For moderate data sizes the division is usually 60:20:20 or 70:30

Hold-Out

- Each model-candidate is evaluated using the validation set
  - Calculate the error between model's predicted $\hat{y}$ value and the y value given in the validation dataset (at the given x value)
- In the end, the one "best" model is evaluated in the same manner using the test set

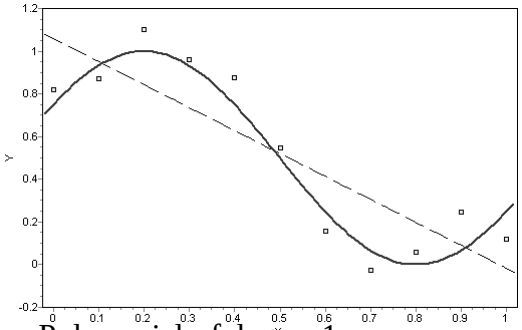


It is important that the order of the data points is ***randomized****!*

Example: using Hold-Out

- MSE in validation and test sets is calculated in the same manner as in training set

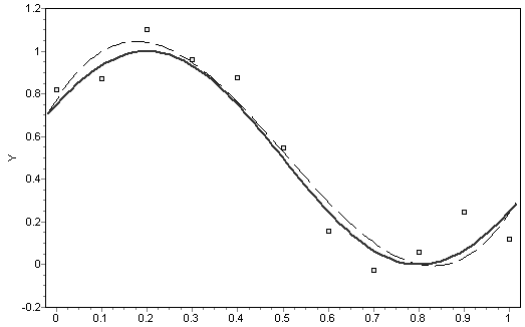
$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$



Polynomial of degree 1

MSE = 0.0439

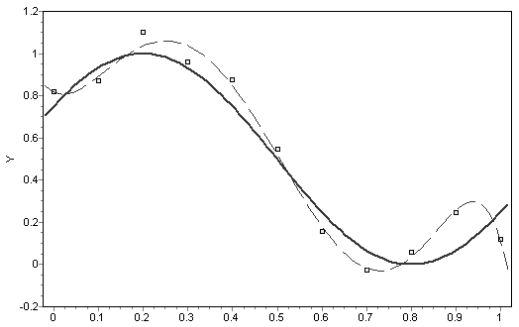
R^2 = 0.72



Polynomial of degree 3

MSE = 0.0123

R^2 = 0.92

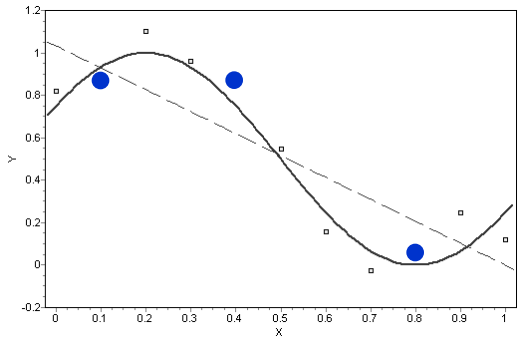


Polynomial of degree 7

MSE = 0.0012

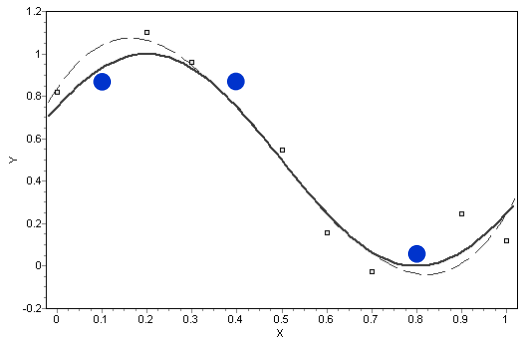
R^2 = 0.99

Evaluation in training dataset



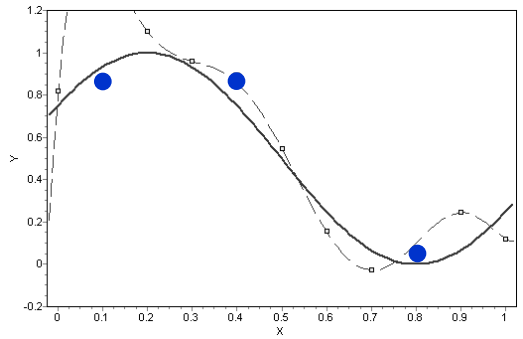
MSE = 0.0307

R^2 = 0.81



MSE = 0.0188

R^2 = 0.88



MSE = 0.1199

R^2 = 0.25

Evaluation using Hold-out

Advantages and disadvantages of Hold-Out

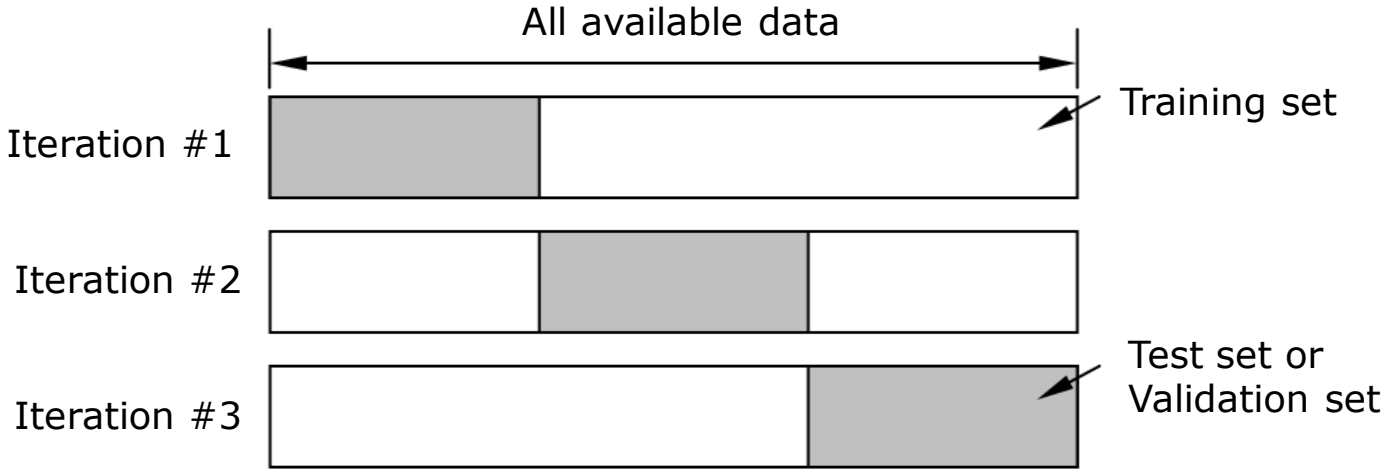
- Advantages
  - Easy to implement and use
  - Efficiency of computations – nothing is to be computed more than once (in contrast to most other resampling methods)
- Disadvantages
  - Considerably reduces the size of the training dataset

    (big problem if data is scarce)
  - We can get "unlucky" data point combinations in any of the sets

    (big problem if data is scarce)

Cross-Validation

- *k-fold Cross-Validation*
  - All examples of the full dataset are randomly reordered and divided in k subsets (folds) of equal size. k iterations are done: in each iteration jth subset is used as a validation (test) set and the other $k-1$ subsets together are used as training set.
  - In total, the model is created (structure, parameters) k times and validation (test) error is calculated k times. In the end we get final evaluation of the model by calculating the average of the errors.



Calculations

- MSE (or other error criteria) in validation set and in test set is calculated in the same way as in training set:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

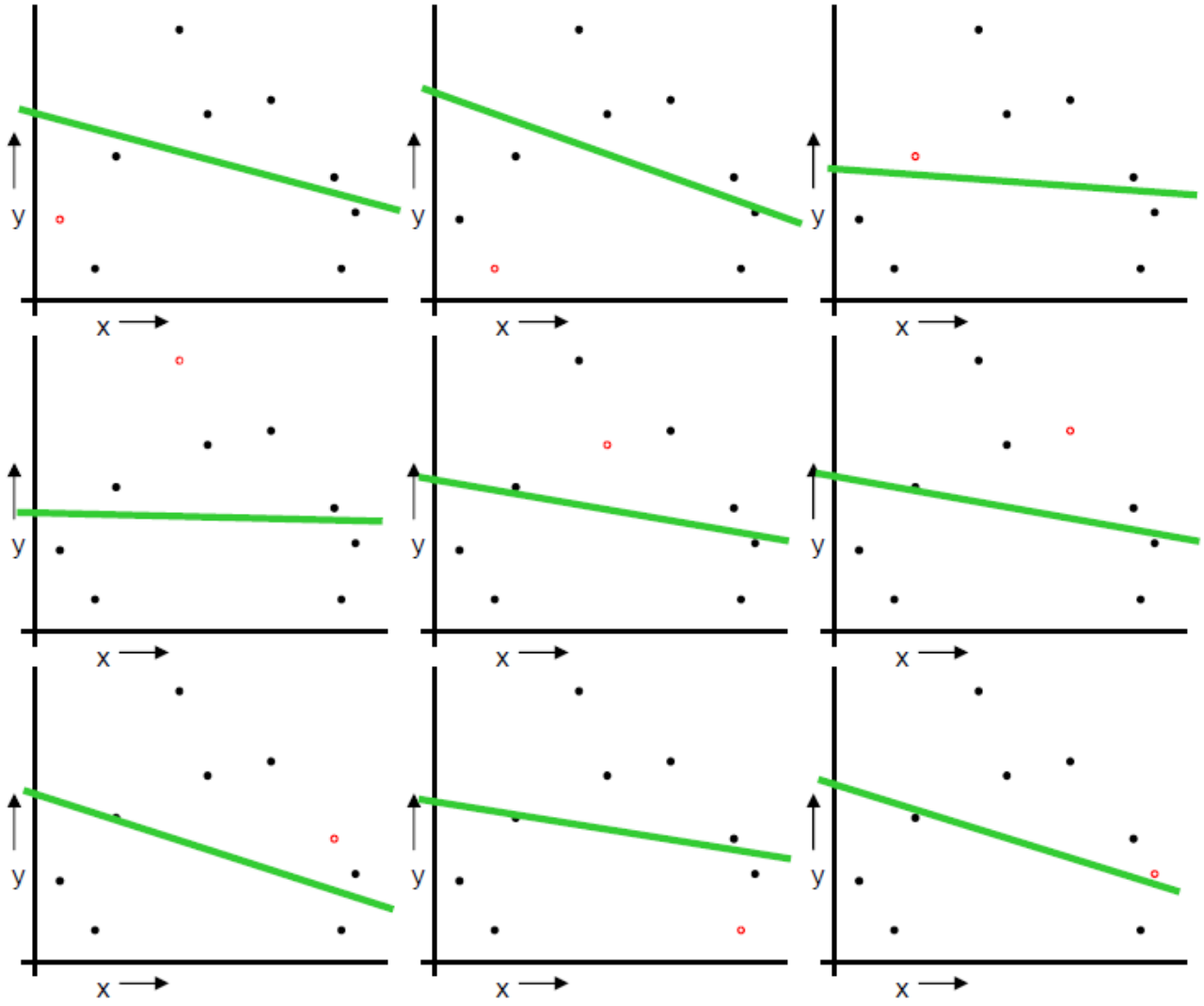
- The average Cross-Validation error (MSE_j is MSE of jth iteration):

$$MSE_{CV} = \frac{1}{k}\sum_{j=1}^{k} MSE_j$$

Leave-One-Out Cross-Validation

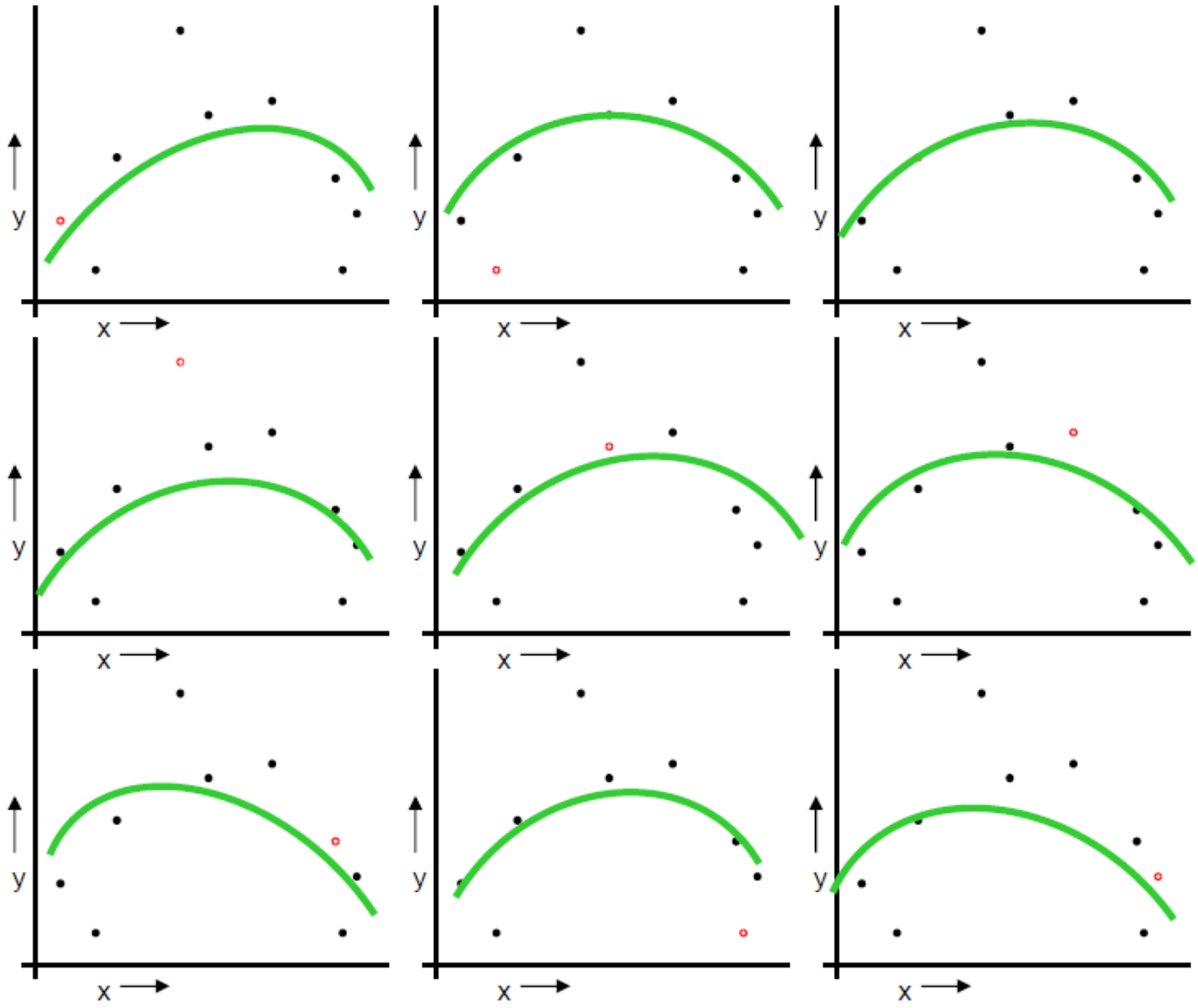
- If $k = n$, the method is called *Leave-One-Out Cross-Validation (LOOCV)*
  - The number of iterations is equal to the number of examples in the data
  - The validation (test) set always includes only one example but overall through all iterations the evaluation is done on all examples
  - This is a more reliable alternative when the data is very scarce, because you remove only one example from the training set and you evaluate the model as many times as the possible

LOOCV example – a line



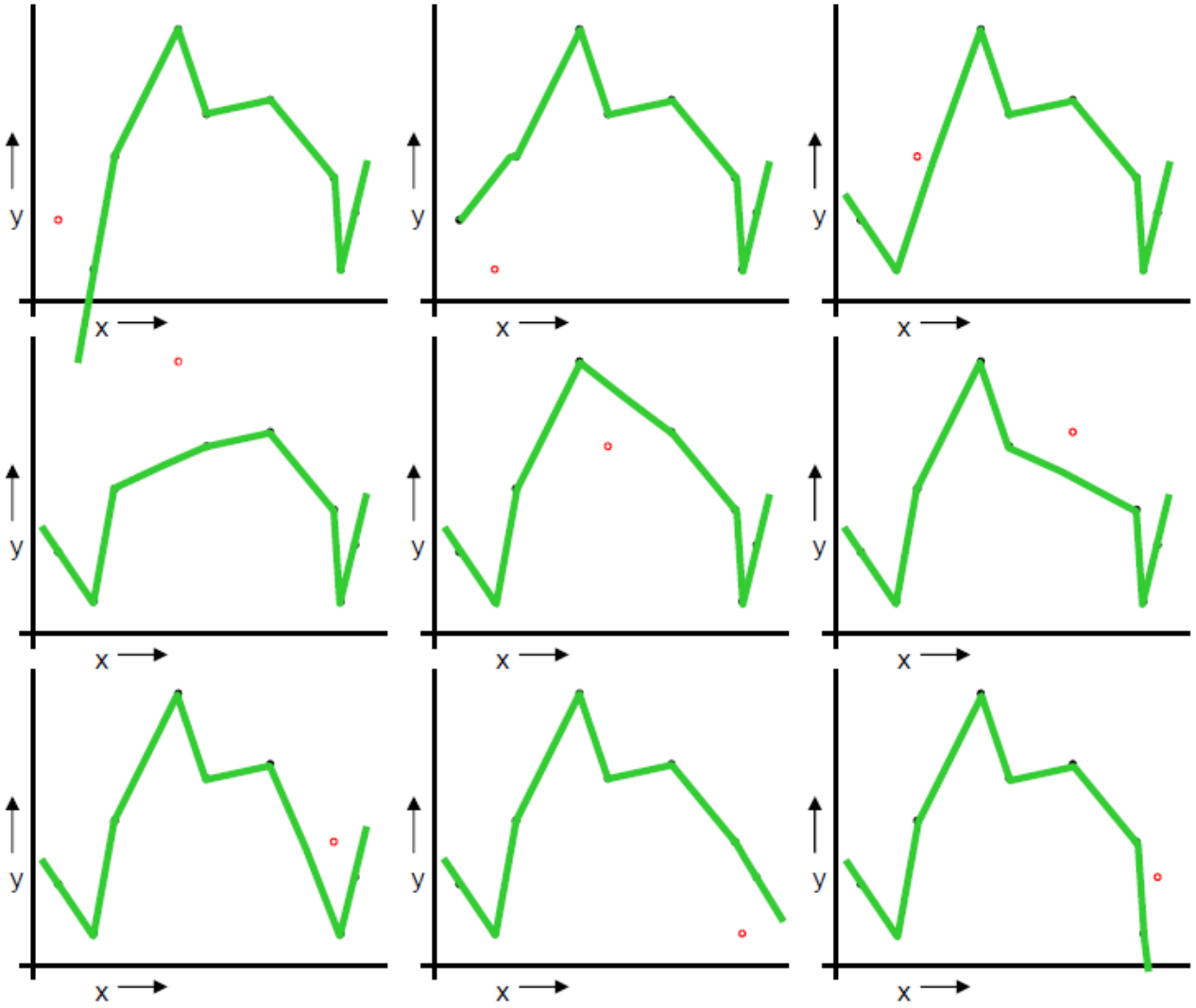
$MSE_{LOOCV} = 2.12$

LOOCV example – parabola



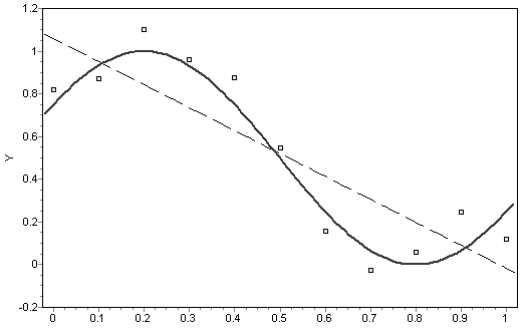
MSE_{LOOCV} =0.962

LOOCV example – complex model

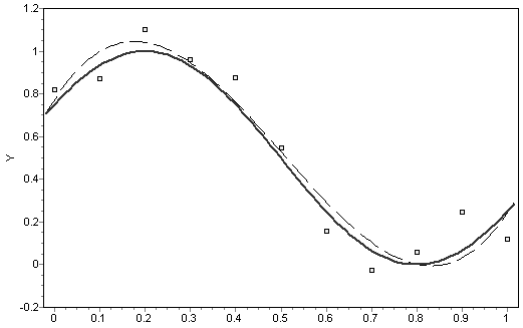


$MSE_{LOOCV} = 3.33$

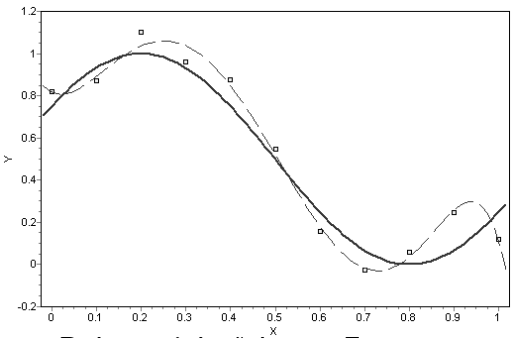
Example: using Cross-Validation



Polynomial of degree 1



Polynomial of degree 3



Polynomial of degree 7

MSE = 0.0439 R^2 = 0.72	MSE = 0.0123 R^2 = 0.92	MSE = 0.0012 R^2 = 0.99	Evaluation in training dataset
MSE = 0.0307 R^2 = 0.81	MSE = 0.0188 R^2 = 0.88	MSE = 0.1199 R^2 = 0.25	Evaluation using Hold-out
MSE = 0.0647 R^2 = 0.59	MSE = 0.0540 R^2 = 0.66	MSE = 1.5395 R^2 = -8.66	Evaluation using Cross-Validation

Advantages and disadvantages of Cross-Validation

- Advantages
  - All the data is used for calculations of model parameters, model selection, and model testing
  - Usually, if you have small amount of data, Cross-Validation is more reliable than Hold-Out
- Disadvantages
  - Requires much more computations than Hold-Out because the whole process (estimation of model parameters and model evaluation) is repeated k times. If creating models is a slow process, Cross-Validations might turn out to be impractical.

- Hold-Out and Cross-Validation are the most popular resampling methods
- Other resampling methods exist as well