AutoML: Evaluation

Evaluation of ML Models (Review)

Bernd Bischl Frank Hutter Lars Kotthoff
Marius Lindauer Joaquin Vanschoren

Introduction

Performance estimation of a model Estimate generalization error of a model on new (unseen) data, drawn from the same data generating process.

Performance estimation of an algorithm Estimate generalization error of a learning algorithm, trained on a data set of a certain size, on new (unseen) data, all drawn from the same data generating process.

Model selection Select the best model from a set of potential candidate models (e.g., different model classes, different hyperparameter settings, different feature sets).

Performance Evaluation

ML performance evaluation provides clear and simple protocols for reliable model validation.

- often simpler than classical statistical model diagnosis
- relies only on few assumptions
- still hard enough and offers **lots** of options to cheat and make mistakes

Performance Measures

We measure performance using a statistical estimator for the **generalization error** (GE).
GE $=$ expected loss of a fixed model
$\hat{\text{GE}}$ $=$ estimated loss averaged across finite sample
Example: Mean squared error (L2 loss)

$$\hat{\text{GE}} = MSE = \frac{1}{n}\sum_{i=1}^{n}\left(y^{(i)} - \hat{y}^{(i)}\right)^2$$

Measures: Inner vs. Outer Loss

$$\begin{aligned} \text{Inner loss} &= \text{loss used in learning (training)} \\ \text{Outer loss} &= \text{loss used in evaluation (testing)} \\ &= \text{evaluation measure} \end{aligned}$$

Optimally: inner loss = outer loss

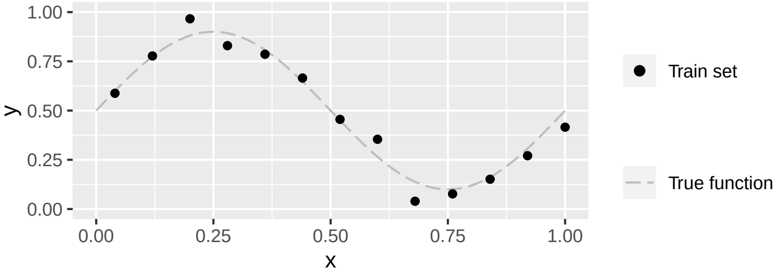
Not always possible:
some losses are hard to optimize or no loss is specified directly

Example:

Logistic Regression	$\rightarrow$ minimize binomial loss
kNN	$\rightarrow$ no explicit loss minimization

Inner Loss Example: Polynomial Regression I

Sample data from sinusoidal function $0.5 + 0.4 \cdot \sin(2\pi x) + \epsilon$ with measurement error ϵ.

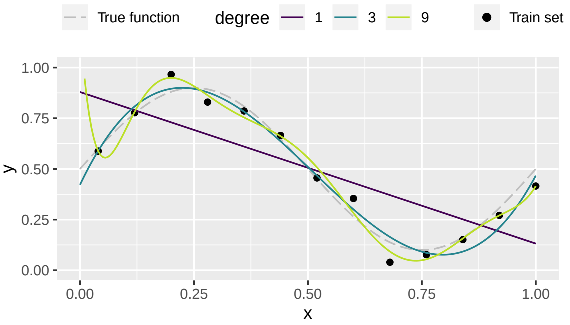


Assume data generating process unknown. Approximate with dth-degree polynomial:

$$f(\mathbf{x}|\theta) = \theta_0 + \theta_1 x + \cdots + \theta_d x^d = \sum_{j=0}^{d} \theta_j x^j$$

Inner Loss Example: Polynomial Regression II

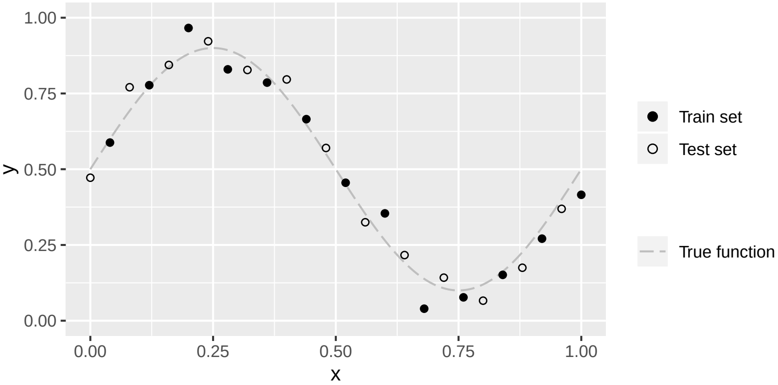
How should we choose d?



d=1: MSE = 0.036 – clear underfitting, d=3: MSE = 0.003 – ok?, d=9: MSE = 0.001 – clear overfitting

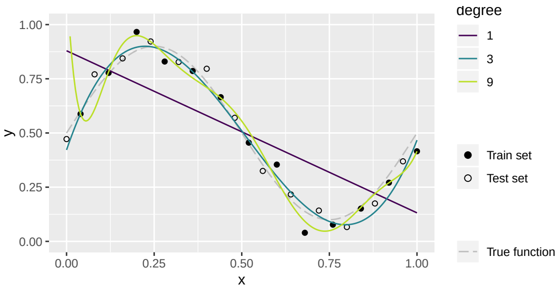
Simply using the training error seems to be a bad idea.

Outer Loss Example: Polynomial Regression I



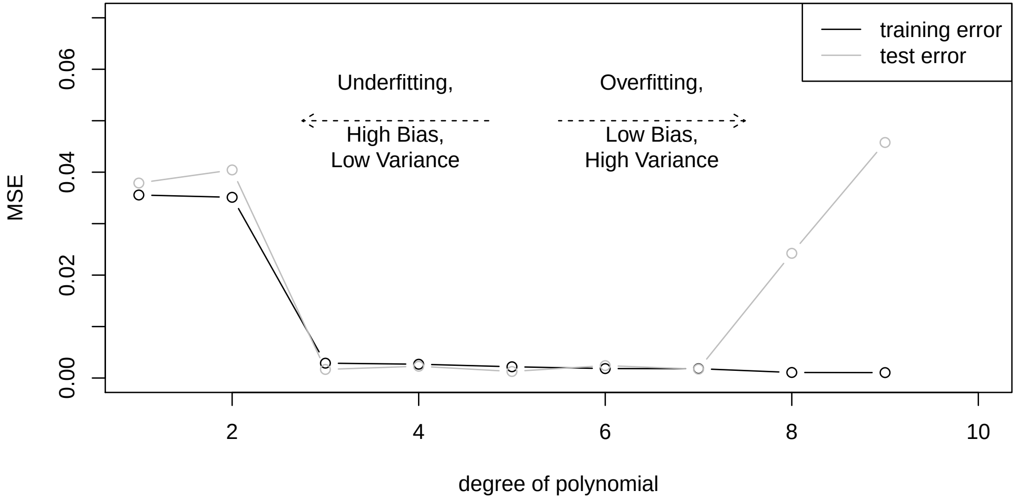
Outer Loss Example: Polynomial Regression II

How should we choose d?

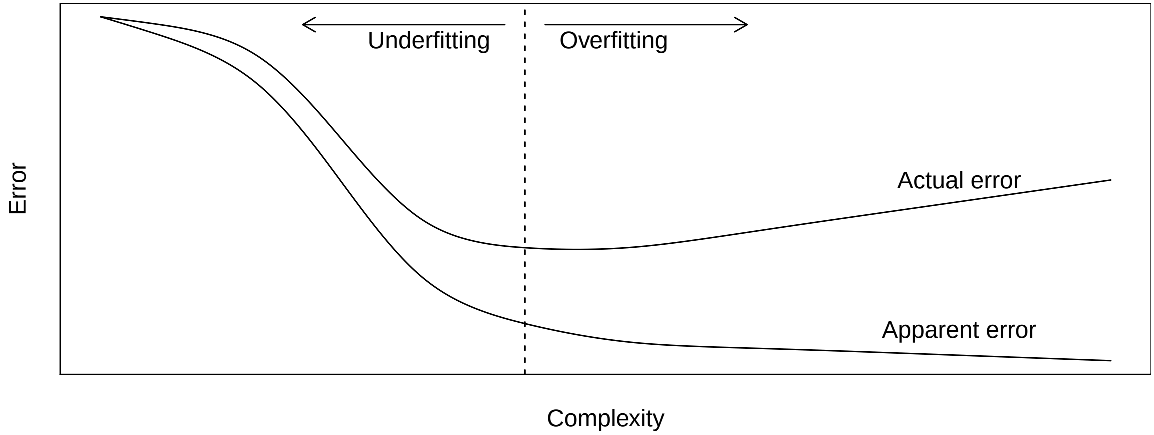


d=1: MSE = 0.038 – clear underfitting, d=3: MSE = 0.002 – ok?, d=9: MSE = 0.046 – clear overfitting

Outer Loss Example: Polynomial Regression III



General Trade-Off Between Error and Complexity



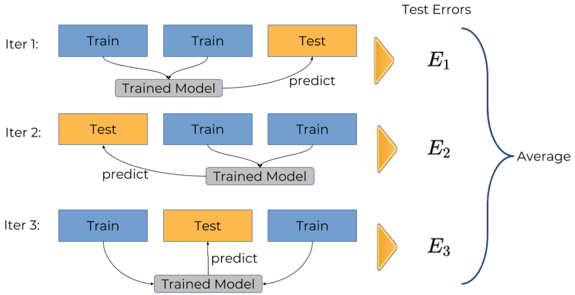
Resampling

- uses data efficiently
- repeatedly split in train and test, average results
- make training sets large (to keep the pessimistic bias small), reduce variance introduced by smaller test sets through many repetitions and averaging of results

Cross-Validation

- split data into k roughly equally-sized partitions
- use each part as test set and join the $k-1$ others for training, repeat for all k combinations
- obtain k test errors and average

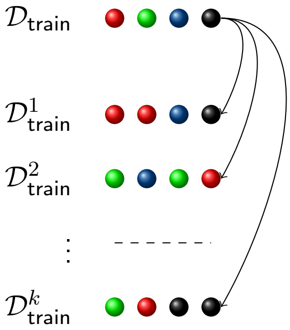
Example 3-fold cross-validation:



10-fold cross-validation is common.

Bootstrap

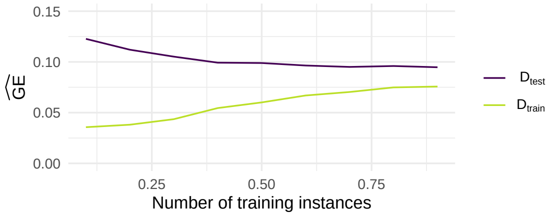
- randomly draw k training sets of size n with replacement from the data
- evaluate on observations from the original data that are not in the training set
- obtain k test errors and average



Training sets will contain about 63.2% of observations on average.

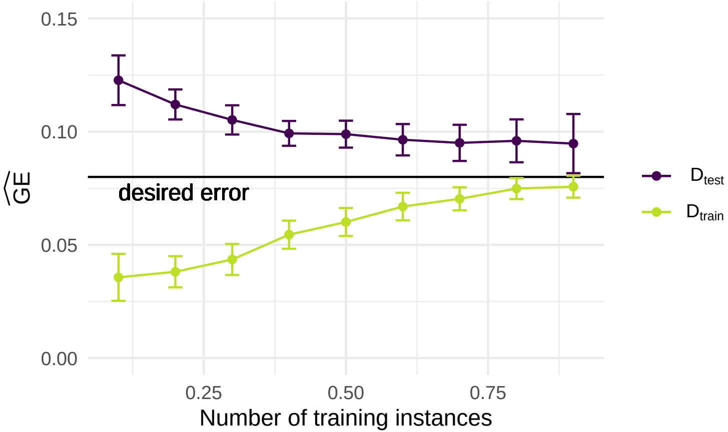
Learning Curves I

- compares performance of a model on training and test data over a varying number of training instances $\rightarrow$ how fast can a learner learn the given relationship in the data?
- can also be over number of iterations of a learner (e.g. epochs in deep learning), or AutoML system over time
- learning usually fast in the beginning
- visualizes when a learner has learned as much as it can:
  - when performance on training and test set reach a plateau
  - when gap between training and test error remains the same



Learning Curves II

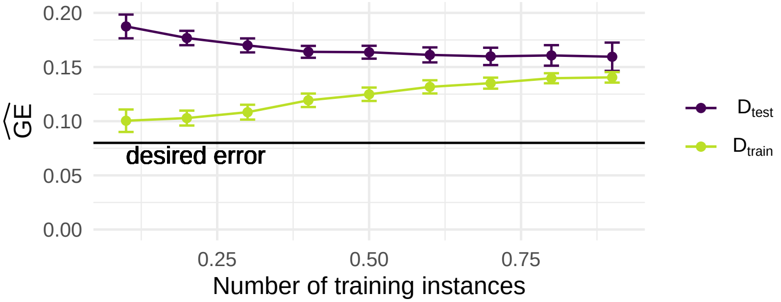
Ideal learning curve:



Learning Curves III

In general, there are two reasons for a bad learning curve:

1. high bias in model/underfitting
   - training and test errors converge at a high value
   - model can't learn underlying relationship and has high systematic errors, no matter how big the training set
   - poor fit, which also translates to high test error



Learning Curves IV

2. high variance in model/overfitting
   - large gap between training and test errors
   - model requires more training data to improve
   - model has a poor fit and does not generalize well