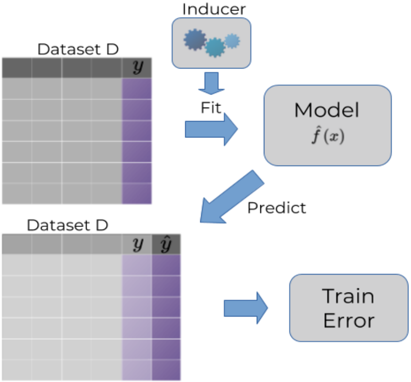
Introduction to Machine Learning

Evaluation: Train and Test Error

Department of Statistics – LMU Munich

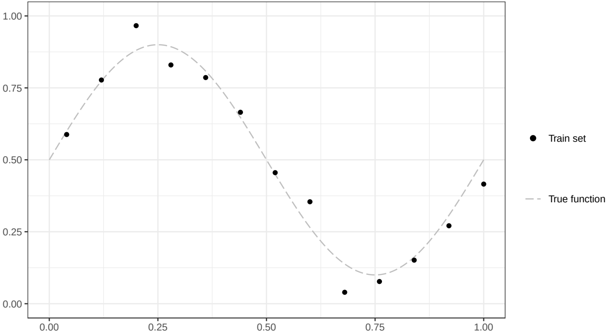
TRAINING ERROR

The *training error* (also called apparent error or resubstitution error) is estimated by the averaging error over the same data set we fitted on:



EXAMPLE: POLYNOMIAL REGRESSION

Assume an (unknown) sinusoidal function that $0.5 + 0.4 \cdot \sin(2\pi x) + \epsilon$ that we sample from with some measurement error ϵ.

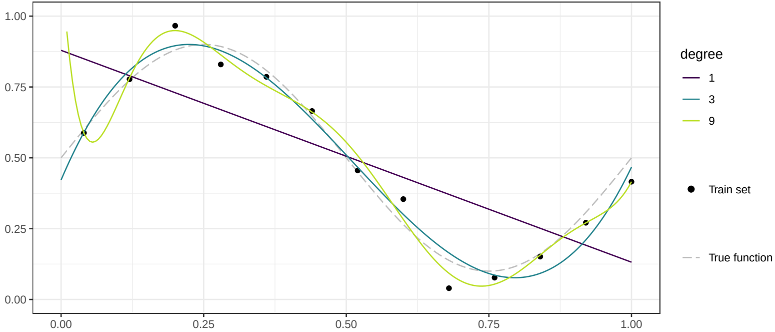


We try to approximate it with a *d*th-degree polynomial

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

EXAMPLE: POLYNOMIAL REGRESSION

Models of different *complexity*, i.e., of different orders of the polynomial are fitted. How should we choose *d*?



- d=1: 0.036: Clear underfitting
- d=3: 0.003: Pretty OK?
- d=9: 0.001: Clear overfitting

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

TRAINING ERROR PROBLEMS

- The training error is usually a very unreliable and overly optimistic estimator of future performance. Modelling the training data is not of interest, but modelling the general structure in it. We do not want to overfit to noise or peculiarities.
- The training error of 1-NN is always zero, as each observation is its own NN during test time (assuming we do not have repeated measurements with conflicting labels).
- Extend any ML training in the following way: After normal fitting, we also store the training data. During prediction, we first check whether x is already stored in this set. If so, we replicate its label. The train error of such an (unreasonable) procedure will be 0.
- There are so called interpolators - interpolating splines, interpolating Gaussian processes - whose predictions can always perfectly match the regression targets, they are not necessarily good as they will interpolate the noise, too

TRAINING ERROR PROBLEMS

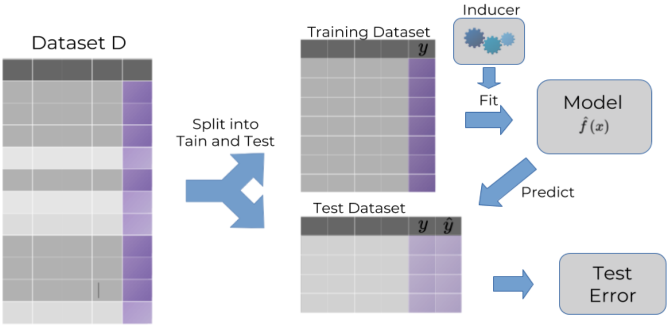
- Goodness-of-fit measures like (classical) R^2, likelihood, AIC, BIC, deviance are all based on the training error.
- For models of severely restricted capacity, and given enough data, the training error might provide reliable information. E.g. consider a linear model with $p = 5$ features, with 10^6 training points. But: What happens if we have less data or as p increases? Not possible to determine when training error becomes unreliable.

TEST ERROR AND HOLD-OUT SPLITTING

- The fundamental idea behind test error estimation (and everything that will follow) is quite simple
- To measure performance, let’s simulate how our model will be applied on new, unseen data
- So, to evaluate a given model do exactly that, predict only on data not used during training and measure performance there
- That implies that for a given dataset $\mathcal{D}$, we have to preserve some data for testing that we cannot use for training

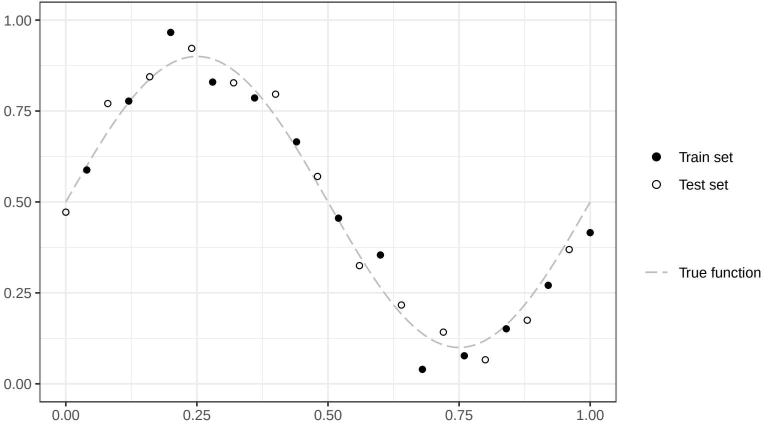
TEST ERROR AND HOLD-OUT SPLITTING

- Split data into 2 parts, e.g. a common setup is 2/3 for training, 1/3 for testing
- Evaluate on data not used for model building, no way to “cheat”

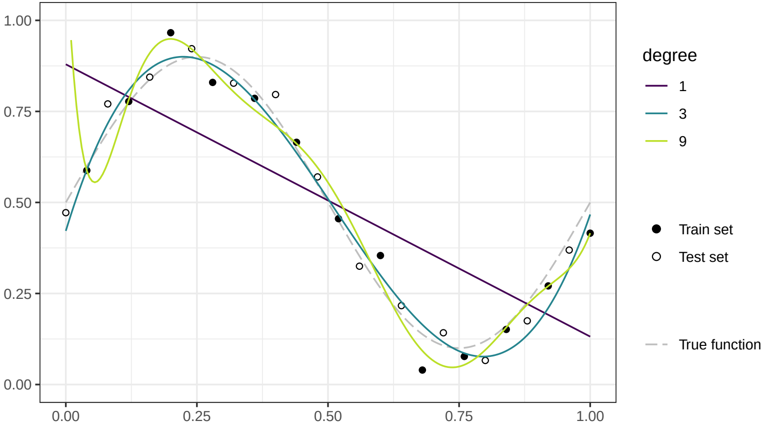


TEST ERROR AND HOLD-OUT SPLITTING

Let's consider some clean test data for our sinusoidal example:

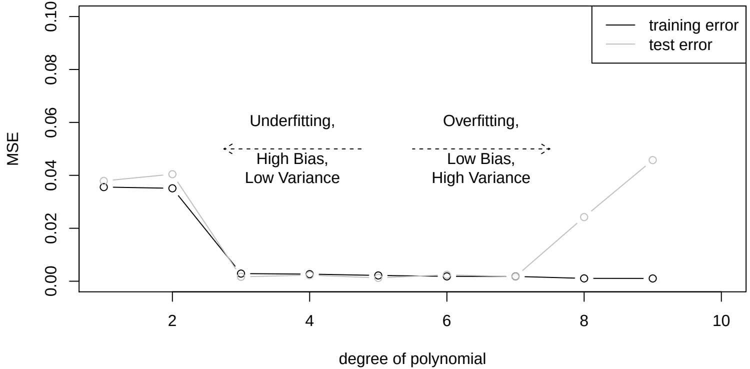


TEST ERROR AND HOLD-OUT SPLITTING



- d=1: 0.038: Clear underfitting
- d=3: 0.002: Pretty OK?
- d=9: 0.046: Clear overfitting

TEST ERROR AND HOLD-OUT SPLITTING



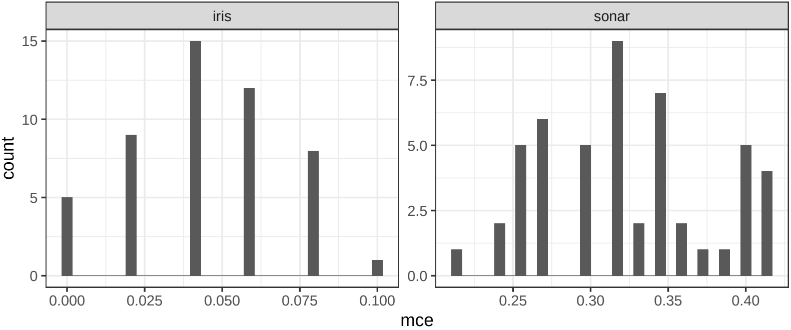
We can also plot error measures for all polynomial degrees. We see the common monotonous decrease in training error, if we increase model complexity (we can adapt better to data with more flexibility) and we see the common U-shape of the test error. First we underfit, then we over-fit, sweet-spot is in the middle. Numerically best for $d = 3$.

PROBLEMS OF TEST ERROR

- In general, the test is a good way to estimate future performance, **given** that the test data is i.i.d. compared to the data we will see when we apply the model.
- The estimator on the will suffer from high variance and be less reliable if the test set is too small. If the test set contains less than a few hundred observations, you should worry.
- Small sample size problems can come in different shapes and forms in ML. Maybe your test set (for binary classification) is large, but one of the two classes is small.
- Try out different train-test splits in your own experiments and study error measurement fluctuation.

PROBLEMS OF TEST ERROR

Let's produce repeated 2/3 training, 1/3 testing splits on two ML tasks: a) iris (n = 150) b) sonar (n = 208). So we have about 50 (iris) and ca 70 (sonar) observations in our respective test sets. The plots below show the strong stochastic fluctuation of test errors (50 repeats).



PROBLEMS OF TEST ERROR

A major point of confusion:

- In ML we are in a weird situation. We are usually given one data set. At the end of our model selection and evaluation process we will likely fit one model on exactly that complete data set. As training error evaluation does not work, we have now nothing left to evaluate exactly that model.
- Holdout splitting (and the soon following resampling) are tools just to estimate that future performance, to put that next to our final model. All of the models produced during that phase of evaluation are basically intermediate results.
- Keep that already in mind now, it will help to avoid confusion when we move on to cross-validation and nested cross-validation.

TRAINING VS. TEST ERROR

The training error

- is an over-optimistic (biased) estimator as the performance is measured on the same data the learned model was trained for
- decreases with smaller training set size as it is easier for the model to learn the underlying structure in the training set perfectly
- decreases with increasing model complexity as the model is able to learn more complex structures

The test error

- will typically decrease when the training set increases as the model generalizes better with more data (more data to learn)
- will have higher variance with decreasing test set size
- will have higher variance with increasing model complexity

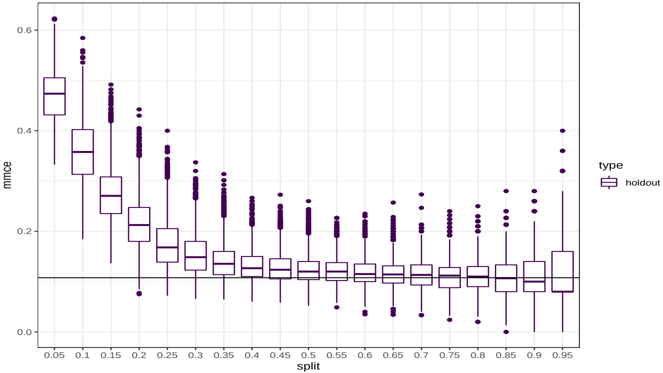
BIAS-VARIANCE OF HOLD-OUT

- If the size of our initial, complete data set $\mathcal{D}$ is limited, single train-test splits can be problematic.
- The smaller our single test set is, the higher the variance of our estimated performance error (e.g., if we test on one observation, in the extreme case). But note that by just making the test set smaller, we do not introduce any bias, as we simply average losses on i.i.d. observations from $\mathbb{P}_{xy}$.
- The smaller our training set becomes, the more pessimistic bias we introduce into the model. Note that if $|\mathcal{D}| = n$, our aim is to estimate the performance of a model fitted on n observations (as this is what we will do in the end). If we fit on less data during evaluation, our model will learn less, and perform worse. Very small training sets will also increase variance a bit.

BIAS-VARIANCE OF HOLD-OUT - EXPERIMENT

- Data: simulate spiral data (sd = 0.1) from `mlbench`
- Learner: CART (`classif.rpart` from `mlr`)
- Goal: estimate real performance of a model with $|\mathcal{D}_{\text{train}}| = 500$
- Get the "true" estimator by repeatedly sampling 500 observations from the simulator, fit the learner, then evaluate on 10^5 observations - obviously cannot be done in practice
- Sample $\mathcal{D}$ with $|\mathcal{D}| = 500$ and analyze different split-rate $s \in \{0.05, 0.1, ..., 0.95\}$ for training with $|\mathcal{D}_{\text{train}}| = s \cdot 500$
- Estimate performance on $\mathcal{D}_{\text{test}}$ with $|\mathcal{D}_{\text{test}}| = 500 \cdot (1 - s)$
- Repeat the experiment for each split rate 50 times

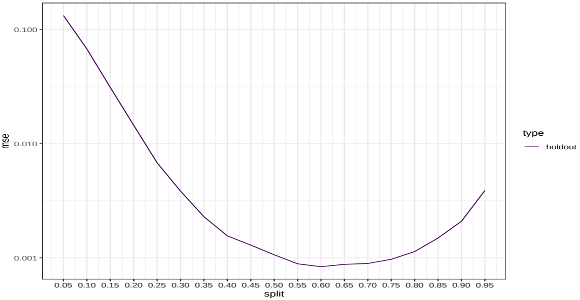
BIAS-VARIANCE OF HOLD-OUT - EXPERIMENT



- We clearly see the pessimistic bias for small training sets – we cannot learn well here with much less data compared to n=500
- We see the increased variance, when test sets become smaller

BIAS-VARIANCE OF HOLD-OUT - EXPERIMENT

- We now plot the mean quadratic error between the true performance (line in 1st plot) and the hold-out values in each boxplot
- The split rate with the lowest MSE value produces the best estimator, which is pretty much 2/3 data for training
- NB: This is a single experiment and not a scientific study, but this rule-of-thump is also validated in larger studies


mse
split
type
holdout