Model Selection and Estimation of Generalization Cost

Volker Tresp
Summer 2021

Generalization Cost and Training Cost

Cost Functions

- We define a cost function for a data point $\mathbf{x}, y$ of function $f_{\mathbf{w}}(\mathbf{x})$ as

$$\text{cost}_{\mathbf{x},y}[\mathbf{w}]$$

- We will use the terms **cost**, **loss** and **error** exchangeably
- Example (quadratic cost):

$$\text{cost}^{q}_{\mathbf{x},y}[\mathbf{w}] = (y - f_{\mathbf{w}}(\mathbf{x}))^2$$

Cost Functions (cont'd)

- Misclassification cost ($y \in \{-1, 1\}$):

$$\text{cost}^{m}_{\mathbf{x},y}[\mathbf{w}] = \frac{1}{2}|y - \text{sign}(f_{\mathbf{w}}(\mathbf{x}))|$$

- Absolute deviation (AD) ($y \in \{-1, 1\}$):

$$\text{cost}_{\mathbf{x},y}[\mathbf{w}] = \frac{1}{2}|y - f_{\mathbf{w}}(\mathbf{x})|$$

Cost Functions (cont'd)

- Perceptron ($y \in \{-1, 1\}$):

$$\text{cost}_{\mathbf{x},y}[\mathbf{w}] = |-yf_{\mathbf{w}}(\mathbf{x})|_+$$

- Vapnik's optimal hyperplanes ($y \in \{-1, 1\}$):

$$\text{cost}_{\mathbf{x},y}[\mathbf{w}] = |1-yf_{\mathbf{w}}(\mathbf{x})|_+$$

- Cross-entropy cost (negative log-likelihood cost)

$$\text{cost}^l_{\mathbf{x},y}[\mathbf{w}] = -\log P(y|f_{\mathbf{w}}(\mathbf{x}))$$

  For binary classification, identical to the logistic regression cost function

Model Selection Based on Generalization Cost

- In statistics one is often interested in the estimation of the value and the uncertainty of particular parameters. Example: is parameter w_1 significantly nonzero?
- In machine learning one is often interested in the **generalization cost** which is the expected cost over all possible data, **for any** fixed $\mathbf{w}$,

$$\text{cost}_{P(\mathbf{x},y)}[\mathbf{w}] = \int \text{cost}_{\mathbf{x},y}[\mathbf{w}] P(\mathbf{x}, y)\, d\mathbf{x}dy$$

- A typical assumption is that $P(\mathbf{x}, y) = P(\mathbf{x})P(y|\mathbf{x})$ is fixed but unknown, which implies that (the true) $f(\mathbf{x})$ is fixed but unknown

Average Test Set Cost Estimates the Generalisation Cost

- An estimator of the generalization cost is the **average test set cost**

$$\text{cost}_{\text{test}}[\mathbf{w}] = \frac{1}{T}\sum_{i=1}^{T} \text{cost}_{\mathbf{x}_i, y_i}[\mathbf{w}]$$

  which is the average cost on the T test data points with $(\mathbf{x}_i, y_i \in \text{test})$

- The test data are data points not used in training

- This is an unbiased estimator of the generalization cost, for any fixed $\mathbf{w}$; we can compare different models based on their average test set performance

- The variance of this estimator approaches zero for $T \to \infty$; typically one does not want to reserve a large set of available data as test data; a better alternative is a cross validation approach, described later

Average Training Set Cost

- To obtain a better understanding of model performance, one is interested in the relationship between average training set cost and generalization cost
- We define the **average training set cost** of the best parameter vector, trained and evaluated **on the training data** as

$$\text{cost}_{\text{train}}[\widehat{\mathbf{w}}(\text{train})] = \frac{1}{N}\sum_{i=1}^{N} \text{cost}_{\mathbf{x}_i, y_i}[\widehat{\mathbf{w}}(\text{train})]$$

Here, $(\mathbf{x}_i, y_i) \in$ train; we use train and D interchangeably

Analysis of Different Quantities

- So far we were interested in is the generalization cost of a particular model $\mathcal{M}$ with particular best-fit parameters $\widehat{\mathbf{w}}(\text{train})$.
- Another quantity of interest is the generalization cost **averaged over all possible training sets of size** N (where the training data set is generated from $P(\mathbf{x}, y)$)

Training Set Cost and Generalization Cost

- It turns out that if we calculate the expected average over all training sets of the same size, then

$$\mathbb{E}_{\text{train}} \left\{ \text{cost}_{P(\mathbf{x},y)}[\widehat{\mathbf{w}}(\text{train})] - \text{cost}_{\text{train}}[\widehat{\mathbf{w}}(\text{train})] \right\} \geq 0$$

- Thus in expectation, the average training cost underestimates the generalization cost for the estimated $\widehat{\mathbf{w}}$, optimized on the training data. Thus the performance of a trained model should not be evaluated on the training set but on the test set, which is an unbiased estimator of the generalization cost

- This expression is the focus in a frequentist analysis!

Preview: Theoretical Analysis

- Consider the special case of models with fixed basis functions and least squares cost and let's assume the model is **free of bias (no regularization, no structural bias)** (explained later)

- Then the generalization cost of the true model (the best possible model)

$$\text{cost}^{q}_{P(\mathbf{x},y)}[f(\cdot)] = \text{Residual} = \sigma^2$$

  If our training procedure would identify the true parameters, this would be the generalization cost

- The expected generalization cost of the fitted model is

$$\mathbb{E}_{\text{train}}\left\{\text{cost}^{q}_{P(\mathbf{x},y)}[\widehat{\mathbf{w}}(\text{train})]\right\} = \text{Residual} + \text{Var}$$

  This term is estimated by the average test set cost; in expectation, it is **larger** than the generalization costs of the best model by a term called the variance Var

Preview: Theoretical Analysis (cont'd)

- The expected average training set cost of the fitted model in some cases is

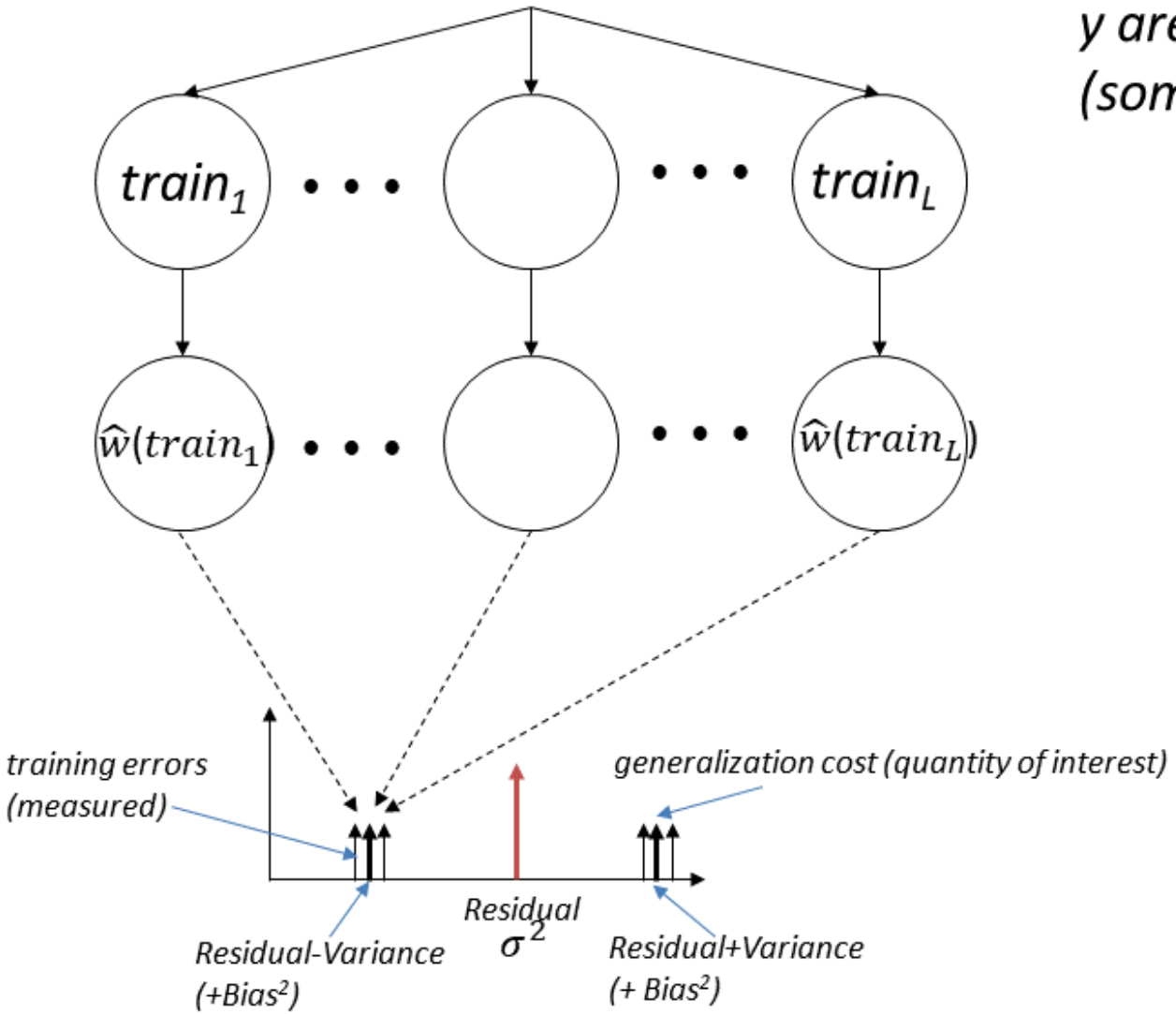
$$\mathbb{E}_{\text{train}}\left\{\text{cost}_{train}[\widehat{\mathbf{w}}(\text{train})]\right\} = \text{Residual} - \text{Var}$$

  This term is estimated by the average training costs; this term is **smaller** than the generalization costs of the best model by the variance Var

- Then we obtain

$$\mathbb{E}_{\text{train}}\left\{\text{cost}_{P(\mathbf{x},y)}[\widehat{\mathbf{w}}(\text{train})] - \text{cost}_{\text{train}}[\widehat{\mathbf{w}}(\text{train})]\right\} = 2\text{Var}$$

- For certain models, we can estimate Var



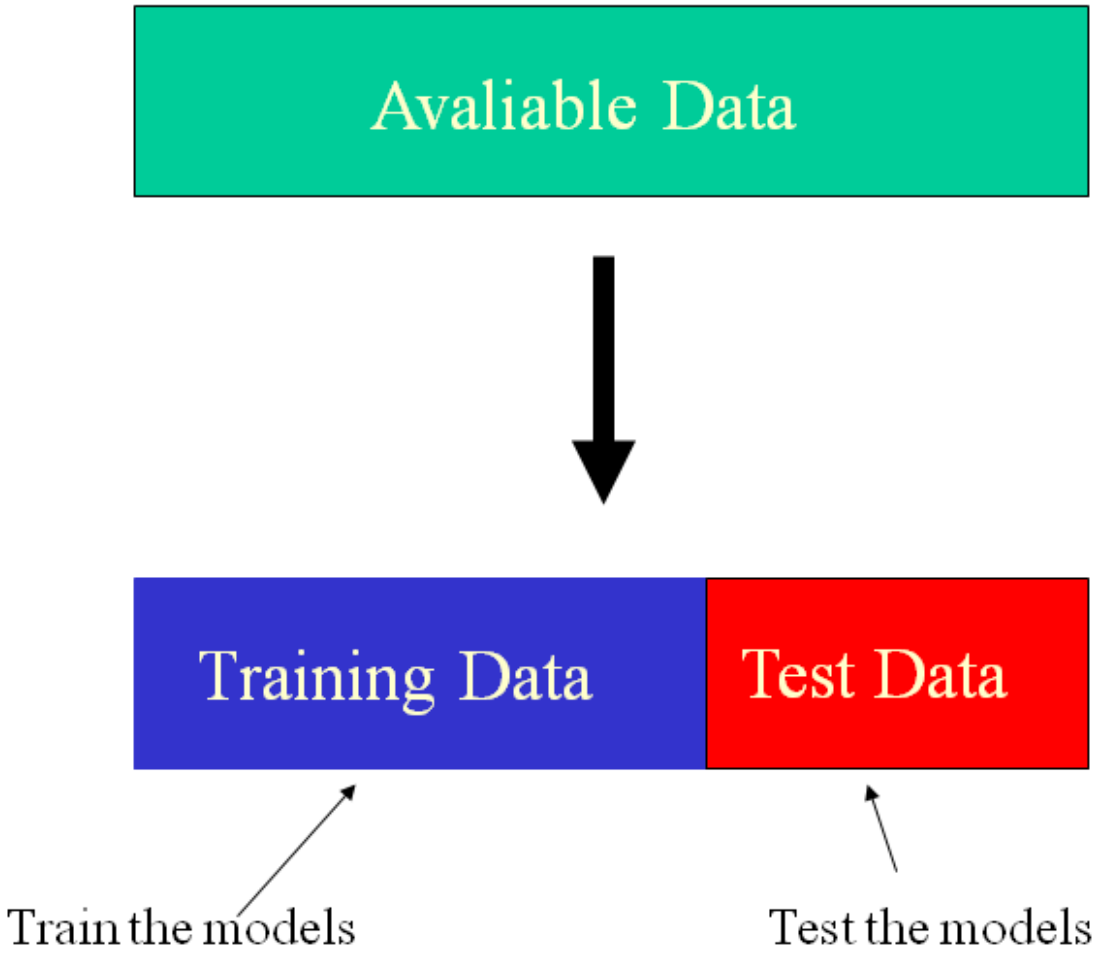
- *f() is fixed*
- *In each experiment, new y are generated (sometimes also new x)*

L -> infinity

Empirical Model Comparison

Model Selection via Training and Test Data Performance

- This procedure can be applied with huge amounts of available data, $N >> M_p$, where M_p is the number of model parameters
- This procedure is typically used in deep learning with large data sets, where a cross validation approach would be too costly
- Divide the data set randomly into a training data set and a test data set
- Train all models only on the training data: find the best parameters for each model under consideration
- Evaluate the generalization performance based on the average test set performance and get $\text{cost}_{\text{test}}[\widehat{\mathbf{w}}(\text{train})]$ for the different models, as an estimate of the generalization costs $\text{cost}_{P(\mathbf{x},y)}[\widehat{\mathbf{w}}(\text{train})]$


Avaliable Data
Training Data
Test Data
Train the models
Test the models

Cross Validation

- Cross validation uses all data in turn for testing
- Consider K- fold cross validation; typical: $K = 5$ oder $K = 10$
- The data is partitioned into K sets of approximately the same size
- For $k = 1, \ldots, K$: The $k-$th fold (test_k) is used for testing and the remaining data (train_k) is used for training (finding the best parameters)

Evaluating Performance with Cross Validation

- For each model one gets K test costs

$$\text{cost}_{\text{test}_k}[\widehat{\mathbf{w}}(\text{train}_k)], \quad k = 1, \ldots, K$$

- Now we now consider the generalization costs averaged over the parameter estimates obtained from different training data sets of size N
- We can estimate this expectation as

$$\mathbb{E}_{\text{train}}\text{cost}_{P(\mathbf{x},y)}[\widehat{\mathbf{w}}(\text{train})] \approx \frac{1}{K}\sum_{k=1}^{K}\text{cost}_{\text{test}_k}[\widehat{\mathbf{w}}(\text{train}_k)] = m_{\mathcal{M}}$$

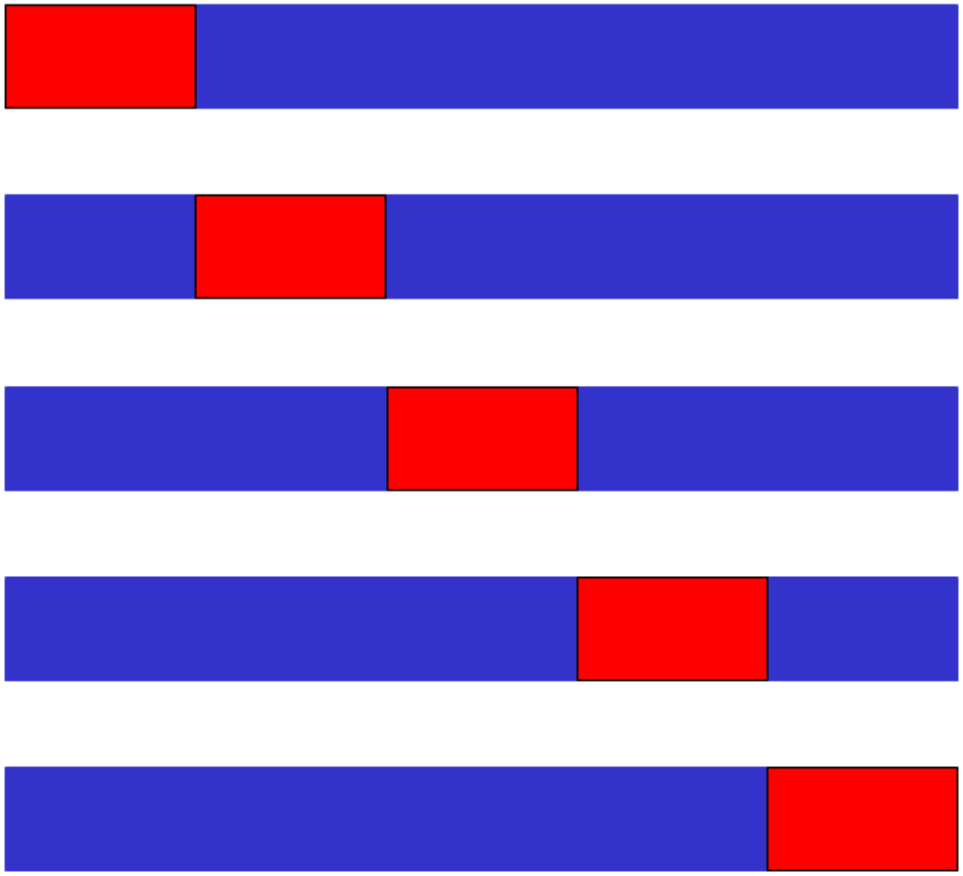
Variance Estimate

- We can estimate the uncertainty of $m_{\mathcal{M}}$ as the variance

$$\widehat{Var}_{\mathcal{M}} = \frac{1}{K(K-1)} \sum_{k=1}^{K} (\text{cost}_{\text{test}}[\hat{\mathbf{w}}(\text{train}_k), \mathcal{M}] - m_{\mathcal{M}})^2$$

- Mean and mean-variance estimates can be used to decide if two models significantly differ in generalization performance: typically, one accepts that model $\mathcal{M}_i$ has smaller generalization cost than $\mathcal{M}_j$, if

$$m_{\mathcal{M}_i} + \sqrt{\widehat{Var}_{\mathcal{M}_i}} < m_{\mathcal{M}_j} - \sqrt{\widehat{Var}_{\mathcal{M}_j}}$$



5-times cross validation:

Blue: Trainings Data

Red: Test Data

Paired Tests

- With few data one can use a paired test
- Basic idea: let's assume that $K = 10$; if $\mathcal{M}_i$ in all test sets is better than $\mathcal{M}_j$, then this is a strong indication that $\mathcal{M}_i$ performs better, even if the variation in test set performance masks this behavior (error bars of the estimate are too large)
- Calculate the average difference between both model costs

$$\mathsf{MeanDiff}_{i,j} = \frac{1}{K} \sum_{k=1}^{K} \mathsf{cost}_{\mathsf{test}_k}[\widehat{\mathbf{w}}(\mathsf{train}_k), \mathcal{M}_j] - \mathsf{cost}_{\mathsf{test}_k}[\widehat{\mathbf{w}}(\mathsf{train}_k), \mathcal{M}_i]$$

and analyse if this difference is significantly larger than zero

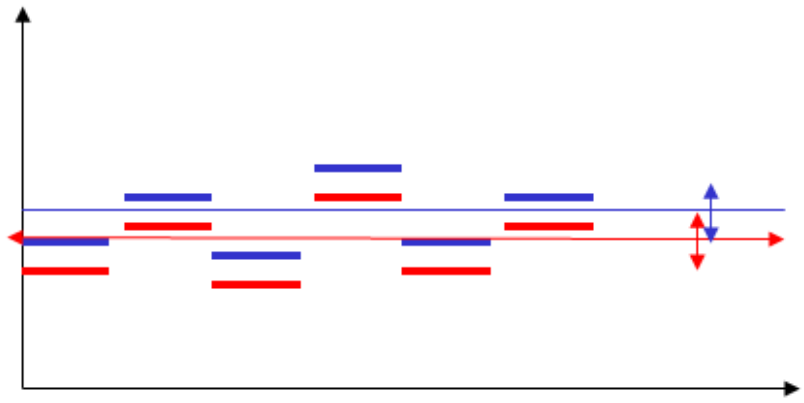
Paired Tests (cont'd)

- In the case of high correlation, the variance of $\mathsf{MeanDiff}_{i,j}$ can be much smaller than the variances of $m_{\mathcal{M}_i}$ and $m_{\mathcal{M}_i}$, due to the rule

$$var(X - Y) = var(X) + var(Y) - 2cov(X, Y)$$

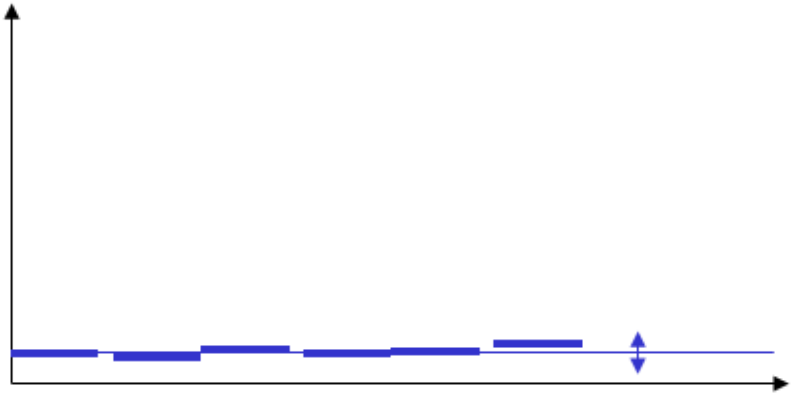
- For a statistical analysis, one employs the test statistics for the paired t-test
- Alternative approach: Wilcoxon signed-rank test

Test error



- Based on the test error on the different folds alone it is not possible to decide that model 1 (red) is significantly bettwer than model 2 (blue)

Difference in test error



- If we look at the difference in performance, it is clear that model 2 (red) performs better

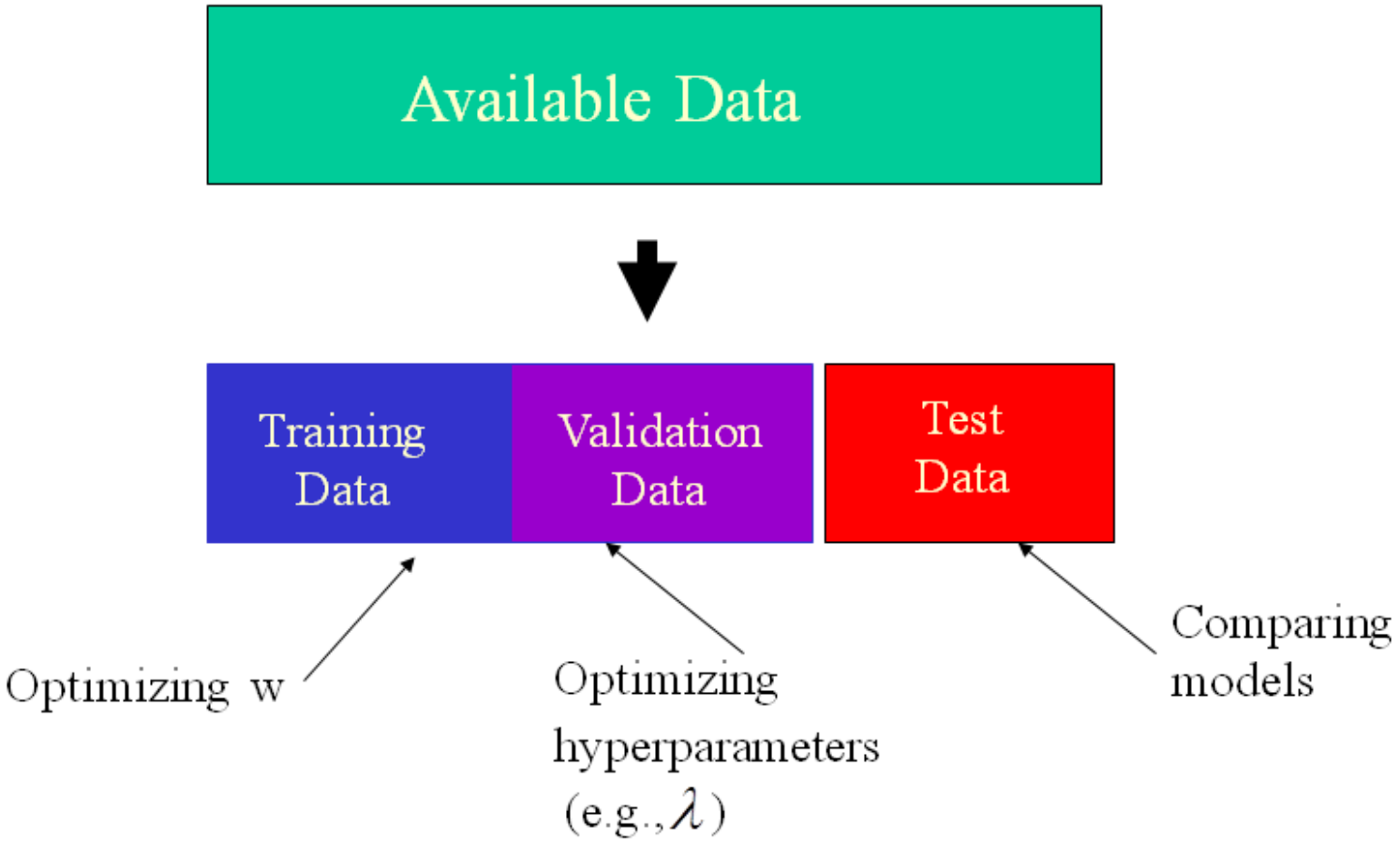
Empirical Tuning of Hyperparameters

Hyperparameter

- In addition to the normal parameters, often one or several hyperparameters need to be tuned as well. Example: regularization weight λ
- The tuning should be done on the training fold. Part of the training fold becomes another fold on which the hyperparameters are tuned

Hyperparameters(cont'd)

- Let's call the folds parameter training fold, hyperparameter fold, and test fold
- In the outer loop we generate training data and test data (as part of K-fold cross validation)
- In the inner loop we divide the training data into parameter training fold and hyperparameter fold. We train the parameters using the parameter training fold with different values of the hyperparameters. We then select the hyperparameter values which give best performance on the hyper-parameter fold
- We use these hyperparameter values to optimize the model on all training data, and evaluate this model on the test set


Available Data
Training Data
Validation Data
Test Data
Optimizing w
Optimizing hyperparameters (e.g., λ)
Comparing models

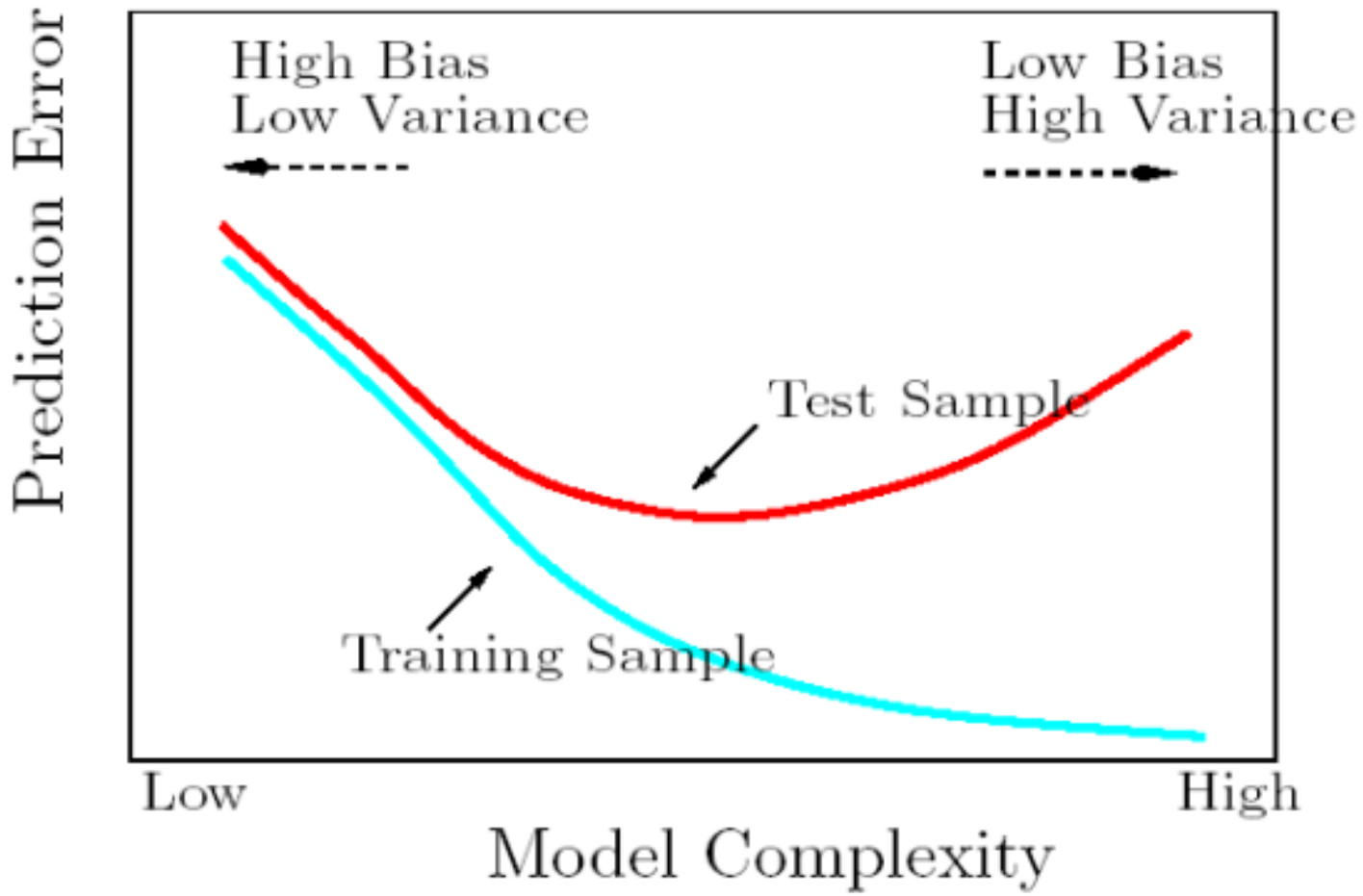


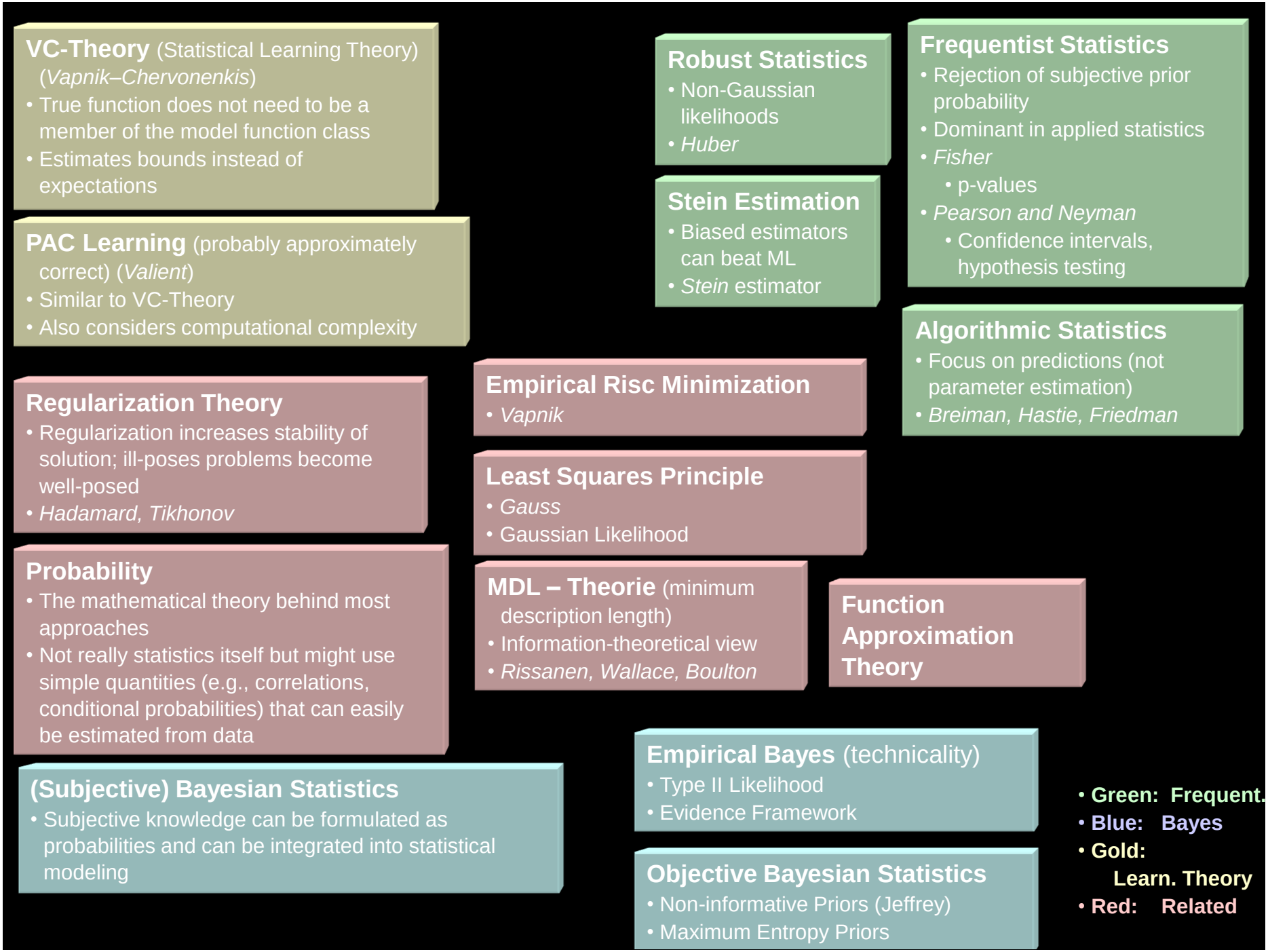
Figure 7.1: *Behavior of test sample and training sample error as the model complexity is varied.*

How to Search for Best Hyperparameters

- Optimizing one (or two hyperparameters) one can perform some form of grid search or consider random choices of parameters
- With many hyperparameters, a random selection of hyperparameters works surprisingly well: one explanation is that some of the hyperparameters might be rather irrelevant and a random search strongly explores the space of potentially relevant subspace

Learning Theories

Overview: Statistical Theories and Learning Theories


VC-Theory (Statistical Learning Theory) (*Vapnik–Chervonenkis*)
• True function does not need to be a member of the model function class
• Estimates bounds instead of expectations
PAC Learning (probably approximately correct) (*Valient*)
• Similar to VC-Theory
• Also considers computational complexity
Regularization Theory
• Regularization increases stability of solution; ill-poses problems become well-posed
• *Hadamard, Tikhonov*
Probability
• The mathematical theory behind most approaches
• Not really statistics itself but might use simple quantities (e.g., correlations, conditional probabilities) that can easily be estimated from data
(Subjective) Bayesian Statistics
• Subjective knowledge can be formulated as probabilities and can be integrated into statistical modeling
Robust Statistics
• Non-Gaussian likelihoods
• *Huber*
Stein Estimation
• Biased estimators can beat ML
• *Stein* estimator
Frequentist Statistics
• Rejection of subjective prior probability
• Dominant in applied statistics
• *Fisher*
• p-values
• *Pearson and Neyman*
• Confidence intervals, hypothesis testing
Algorithmic Statistics
• Focus on predictions (not parameter estimation)
• *Breiman, Hastie, Friedman*
Empirical Risc Minimization
• *Vapnik*
Least Squares Principle
• *Gauss*
• Gaussian Likelihood
MDL – Theorie (minimum description length)
• Information-theoretical view
• *Rissanen, Wallace, Boulton*
Function Approximation Theory
Empirical Bayes (technicality)
• Type II Likelihood
• Evidence Framework
Objective Bayesian Statistics
• Non-informative Priors (Jeffrey)
• Maximum Entropy Priors
• **Green: Frequent.**
• **Blue: Bayes**
• **Gold: Learn. Theory**
• **Red: Related**

Learning Theories

- A: Classical Frequentist Approaches
  - C_p Statistics
  - Akaikes Information Criterion (AIC)
- B: Bayesian approaches
  - Strict Bayes: model averaging instead of model selection
  - Bayesian model selection and Bayesian Information Criterion (BIC)
- C: Modern Frequentist Approaches
  - Minimum Description Length (MDL) Principle
  - Statistical Learning Theory (Vapnik-Chervonenkis (VC) Theory)

A:Classical Frequentist Approaches

Frequentist Approaches

- We are again interested in the generalization cost averaged over the parameter estimates from different training data sets of size N

$$\mathbb{E}_{\text{train}}\text{cost}_{P(\mathbf{x},y)}[\widehat{\mathbf{w}}(\text{train})]$$

- Thus we evaluate the quality of a particular model $\mathcal{M}$ and not a *particular* parameter vector

Bias-Variance Decomposition

- We assume a fixed $P(\mathbf{x})$ and than $y = f(\mathbf{x}) + \epsilon$, where ϵ is uncorrelated noise
- We use a quadratic cost function. Then one can decompose for the *squared cost*

$$\mathbb{E}_{\mathsf{train}}\mathsf{cost}^{q}_{P(\mathbf{x},y)}[\widehat{\mathbf{w}}(\mathsf{train})] = \mathsf{Bias}^2 + \mathsf{Var} + \mathsf{Residual}$$

Residual

- The residual cost is simply the cost of the true model

$$\text{Residual} = \int (f(\mathbf{x}) - y)^2 P(\mathbf{x}, y) d\mathbf{x} dy = \text{cost}^q_{P(\mathbf{x},y)}[f(\cdot)]$$

- In regression, this is simply the noise variance σ^2

Bias

- The bias is the mean square of the difference between the true model and the average prediction of all models trained with different training sets of size N. A regularized model with $\lambda > 0$ would typically be biased. A linear model is biased if the true dependency is quadratic. With $m(\mathbf{x}) = \mathbb{E}_{\text{train}}(f(\mathbf{x}, \widehat{\mathbf{w}}(\text{train})))$

$$\text{Bias}^2 = \int [m(\mathbf{x}) - f(\mathbf{x})]^2 \, P(\mathbf{x}) d\mathbf{x}$$

Variance

- The variance is the mean square of the difference between trained models and the average prediction of all models trained with different training sets of size N

$$\mathsf{Var} = \int \mathbb{E}_{\mathsf{train}}[f(\mathbf{x}, \widehat{\mathbf{w}}(\mathsf{train})) - m(\mathbf{x})]^2 P(\mathbf{x}) d\mathbf{x}$$

Background: Some Rules for Variances and Traces

- Let $\mathbf{y}$ be a random vector with covariance $\mathsf{Cov}(\mathbf{y})$ and let A be a fixed matrix

  If $\mathbf{z} = A\mathbf{y}$, then: $\mathsf{Cov}(\mathbf{z}) = A\mathsf{Cov}(\mathbf{y})A^T$

- The trace is the sum over the diagonal elements of a matrix. One can show that

$$\mathbf{trace}[\Phi(\Phi^T\Phi)^{-1}\Phi^T] = M$$

  where M is the number of columns of the matrix Φ. Special case: when Φ is a square matrix and has an inverse, then $\Phi(\Phi^T\Phi)^{-1}\Phi^T = I$ and the trace of I is obviously M

Example: One Parameter, Unbiased

- Model is $y = w_0 + \epsilon = f + \epsilon$
- The (unbiased) estimate is $\widehat{w}(\text{train}) = f(\widehat{w}(\text{train})) = \frac{1}{N}\sum_{i=1}^{N} y_i$
- *Variance*: $\text{Var} = \text{Var}(\widehat{w}(\text{train})) = \frac{1}{N^2}(N\sigma^2) = \frac{1}{N}\sigma^2$
- $\text{Bias} = 0$
- We get

$$\mathbb{E}_{\text{train}}\text{cost}^q_{P(x,y)}[\widehat{w}(\text{train})] = \left(\frac{1}{N} + 1\right)\sigma^2 = \frac{N+1}{N}\sigma^2$$

- An unbiased estimate is

$$\text{Residual} \approx \hat{\sigma}^2 = \frac{1}{N-1}\sum_{i=1}^{N}(y_i - \widehat{w}_{0,ls})^2 = \frac{N}{N-1}\text{cost}^q_{\text{train}}[\widehat{w}(\text{train})]$$

Example: One Parameter, Unbiased (cont'd)

- Now, we can actually estimate the variance as well as

$$\mathsf{Var} \approx \frac{1}{N}\hat{\sigma}^2 = \frac{1}{N-1}\mathsf{cost}^q_{\mathsf{train}}[\hat{w}(\mathsf{train})]$$

- So we get as an estimate,

$$\mathbb{E}_{\mathsf{train}}\mathsf{cost}^q_{P(x,y)}[\hat{w}(\mathsf{train})] \approx \frac{N+1}{N-1}\mathsf{cost}^q_{\mathsf{train}}[\hat{w}(\mathsf{train})]$$

- Often one is interested in the difference between generalization error and training error:

$$\mathbb{E}_{\mathsf{train}}\mathsf{cost}^q_{P(x,y)}[\hat{w}(\mathsf{train})] - \mathsf{cost}^q_{\mathsf{train}}[\hat{w}](\mathsf{train})$$

$$\approx \mathsf{cost}^q_{\mathsf{train}}[\hat{w}(\mathsf{train})]\frac{2}{N-1} = 2\mathsf{Var}$$

  As mentioned before, the difference between both is twice the variance

- The great thing about using this last equation is that we never have to estimate the bias term explicitly!

Example: Linear Models

- We assume that the data has been generated with

$$y_i = \phi(\mathbf{x}_i)\mathbf{w} + \epsilon_i$$

where: ϵ_i is independent noise with variance σ^2

- We take the ML estimator which is known to be unbiased and is

$$\hat{\mathbf{w}} = (\Phi^T\Phi)^{-1}\Phi^T\mathbf{y}$$

Thus we now know that $\mathsf{Bias} = 0$.

- With the rule we just learned we can calculate the parameter covariance

$$\mathsf{Cov}(\hat{\mathbf{w}}) = (\Phi^T\Phi)^{-1}\Phi^T\mathsf{Cov}(\mathbf{y})\Phi(\Phi^T\Phi)^{-1}$$

$$= \sigma^2(\Phi^T\Phi)^{-1}\Phi^T\Phi(\Phi^T\Phi)^{-1} = \sigma^2(\Phi^T\Phi)^{-1}$$

- Great, now we know how certain the parameters are. We can now evaluate Var by taking a large sample of $P(\mathbf{x})$. Unfortunately such a large sample is not available and *we simply approximate it with the training data inputs*.

Variance Estimate

- The mean predictions of our model at the training data inputs is then $\mathbf{f} = \Phi\hat{\mathbf{w}}$
- Applying the covariance formula again as before, we get

$$\mathsf{Cov}(\mathbf{f}) = \Phi\mathsf{Cov}(\hat{\mathbf{w}})\Phi^T$$

- For the variance we really only need the mean over the diagonal terms

$$\mathsf{Var}(\mathbf{f}) = \frac{1}{N}trace(\Phi\mathsf{Cov}(\hat{\mathbf{w}})\Phi^T)$$

Example: Linear Models (cont'D)

- Substituting, we get

$$\widehat{\mathsf{Var}} = \frac{1}{N} trace(\Phi \mathsf{Cov}(\hat{\mathbf{w}}) \Phi^T) = \frac{\sigma^2}{N} trace(\Phi(\Phi^T \Phi)^{-1} \Phi^T)$$

- Now we apply our trace-rule and get

$$\widehat{\mathsf{Var}} = \frac{M_p}{N} \sigma^2$$

  where M_p is the number of parameters

- The solution is surprisingly simple, but makes sense: The predictive variance increases with more noise on the data and with more free parameters and decreases with more data!

Generalization Cost of the Best Fit

- Thus the generalization cost for the parameters that minimize the training set costs is on average

$$\mathbb{E}_{\text{train}}\text{cost}^{q}_{P(\mathbf{x},y)}[\widehat{\mathbf{w}}(\text{train})] \approx \sigma^2 + \frac{M_p}{N}\sigma^2 = \sigma^2\frac{M_p + N}{N}$$

- Thus on average the generalization cost for the parameters optimized on the training set is larger by $\frac{M_p}{N}\sigma^2$, if compared to the generalization cost of the best possible model

Estimating the Residual and the Variance

- We estimate the Residual as

$$\text{Residual} \approx \hat{\sigma}^2 = \frac{N}{N - M_p} \text{cost}^q_{\text{train}}[\widehat{\mathbf{w}}(\text{train})]$$

- Then,

$$\text{Var} \approx \frac{M_p}{N - M_p} \text{cost}^q_{\text{train}}[\widehat{\mathbf{w}}(\text{train})]$$

C_P-statistics

- By substitution we now get

$$\mathbb{E}_{\text{train}}\text{cost}^q_{P(\mathbf{x},y)}[\widehat{\mathbf{w}}(\text{train})] \approx \frac{N+M_p}{N-M_p}\text{cost}^q_{\text{train}}[\widehat{\mathbf{w}}(\text{train})]$$

- This is called Mallot's C_P-statistics
- Thus in model selection on would chose the model where Mallot's C_P is smallest
- Often one is interested in the difference between generalization error and training error:

$$\mathbb{E}_{\text{train}}\text{cost}^q_{P(\mathbf{x},y)}[\widehat{\mathbf{w}}(\text{train})] - \text{cost}^q_{\text{train}}[\widehat{\mathbf{w}}(\text{train})]$$

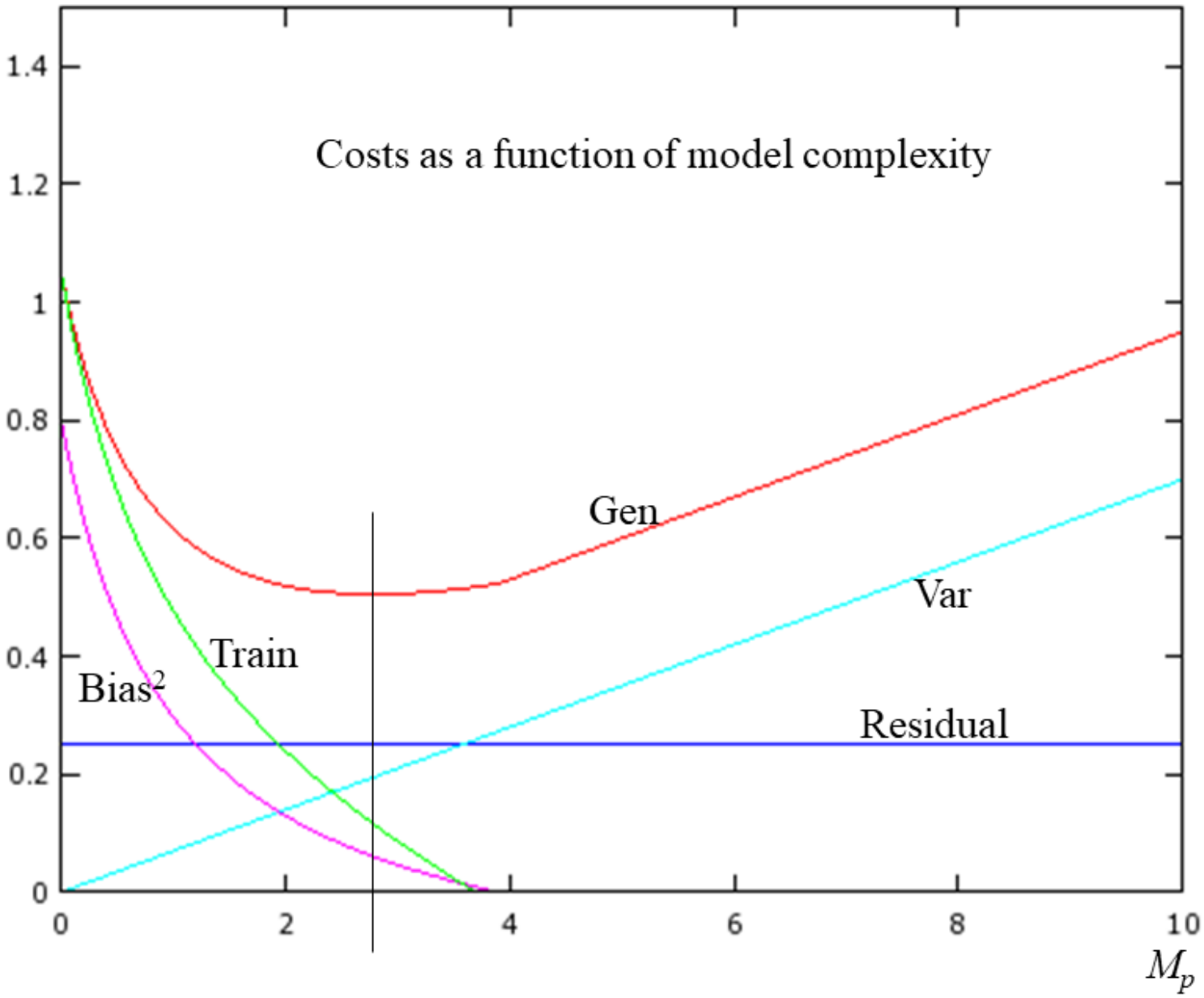
$$\approx \text{cost}^q_{\text{train}}[\widehat{\mathbf{w}}(\text{train})]\frac{2}{N-M_p} = 2\text{Var}$$

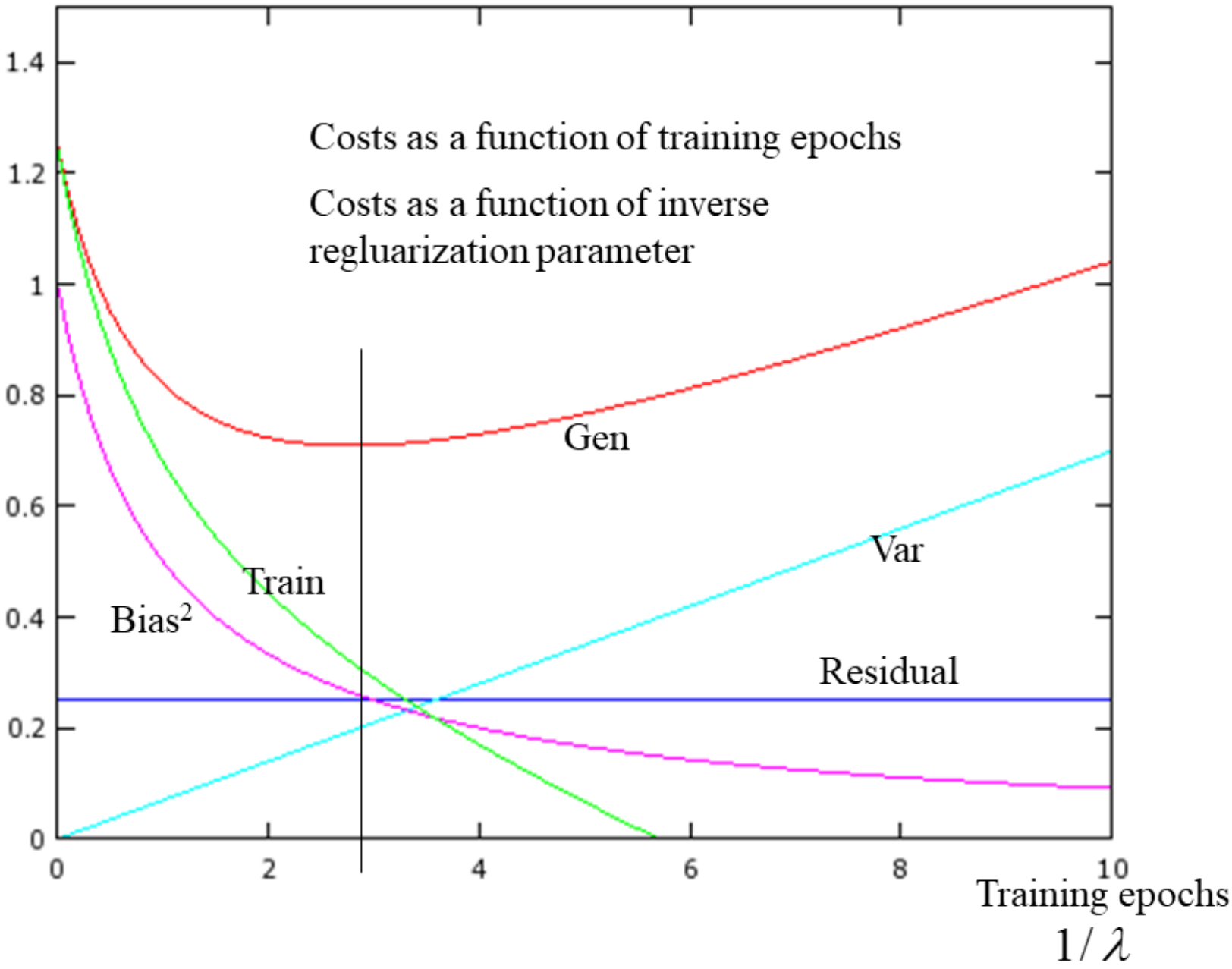
  As mentioned before, the difference between both is twice the variance

- The great thing about using this last equation is that we never have to estimate the bias term explicitly!

Conceptual Plots

- The next figures show the behavior of bias, variance and residual and average training and average test costs
- The complexity is controlled by the number of parameters M_p, or the number of epochs (stopped training), of the inverse of the regularization parameter
- Note that the best models have a Bias > 0


Costs as a function of model complexity
1.4
1.2
1
0.8
0.6
0.4
0.2
0
0
2
4
6
8
10
M_p
Gen
Var
Train
Bias^2
Residual


Costs as a function of training epochs
Costs as a function of inverse regluarization parameter
Gen
Var
Train
Bias²
Residual
1.4
1.2
1
0.8
0.6
0.4
0.2
0
0
2
4
6
8
10
Training epochs
1 / λ

Akaikes Information Criterion (AIC)

- The analysis so far was only valid for models that minimized the squared cost. Consider the cross entropy cost function which minimizes the negative log-likelihood

$$\text{cost}^{l}_{\mathbf{x},y}[\mathbf{w}] = -\log P(y|\mathbf{x},\mathbf{w})$$

- The negative log-likelihood of the ML-solution is

$$-\log L = -\sum_{i=1}^{N} \log P(y_i|\mathbf{x}_i,\mathbf{w}_{ML})$$

- Here, one can apply Akaike's *Information Criterion* (*AIC*) (as defined in Wikipedia)

$$AIC = -2\log L + 2M_p$$

- A model with a smaller *AIC* is preferred

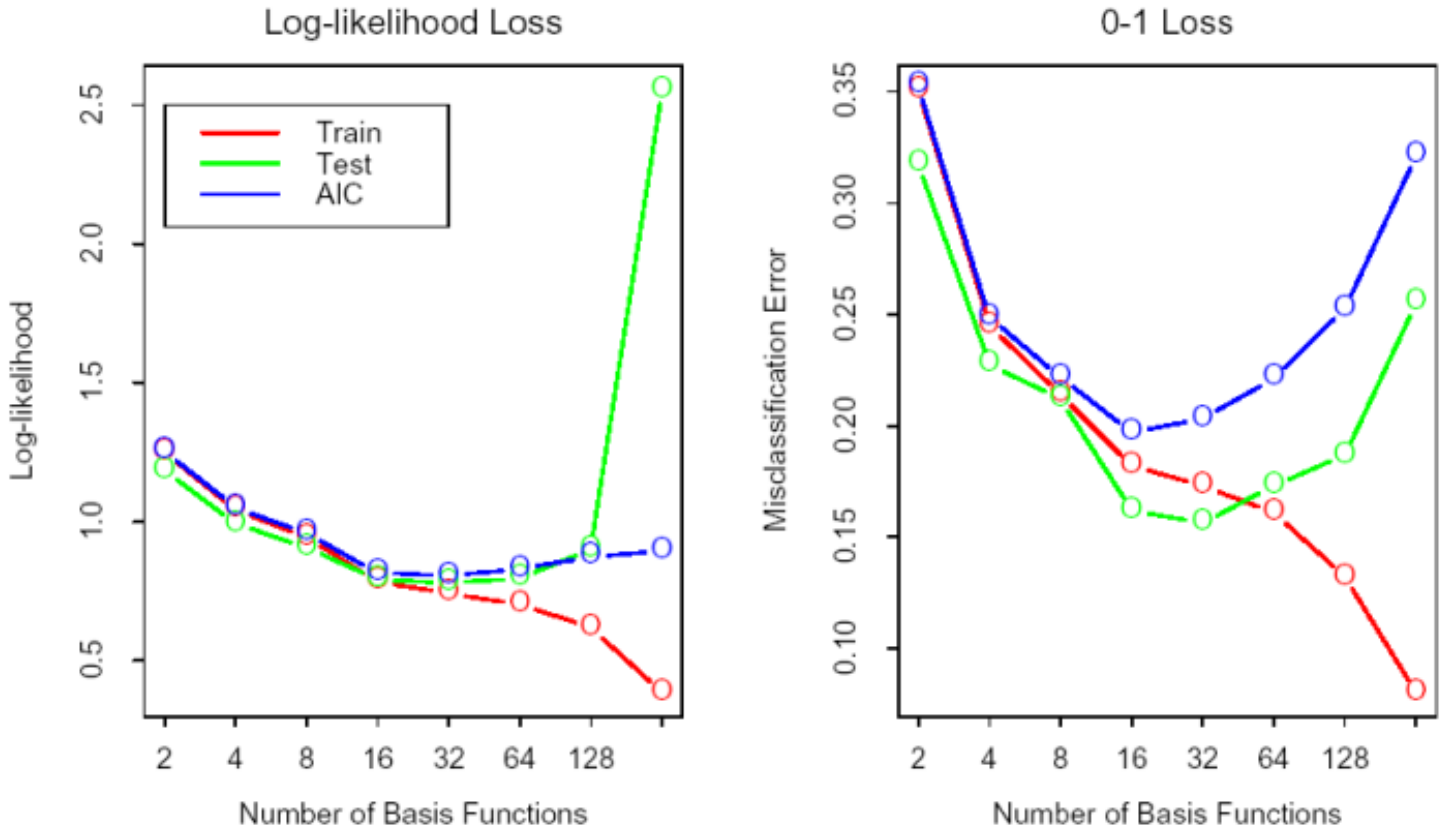
Comments on AIC

- The expression

$$\frac{AIC}{2N} = \left(\text{cost}^{l}_{\text{train}}[\widehat{\mathbf{w}}(\text{train})] + \frac{M_p}{N} \right)$$

  estimates the generalization log-likelihood cost

AIC for Likelihood Cost Function and for 1/0 Cost Function



Proof: Bias-Variance Decomposition

- One can reduce the problem to estimating the decomposition for one parameter. μ is the parameter and x are the data. We add and subtract $\mathbb{E}_{\text{train}}(\widehat{\mu})$ and we add and subtract μ (true parameter). Then,

$$\mathbb{E}_{\text{train}}\mathbb{E}_x(\widehat{\mu}-x)^2 = \mathbb{E}_{\text{train}}\mathbb{E}_x\left[(\widehat{\mu} - \mathbb{E}_{\text{train}}(\widehat{\mu})) + (\mathbb{E}_{\text{train}}(\widehat{\mu}) - \mu) + (\mu - x)\right]^2$$

- One gets

$$\mathbb{E}_{\text{train}}\mathbb{E}_x(\widehat{\mu} - x)^2 = \text{Bias}^2 + \text{Var} + \text{Residual}$$

$$\text{Residual} = \mathbb{E}_x(x - \mu)^2$$

$$\text{Bias} = \mathbb{E}_{\text{train}}(\widehat{\mu}) - \mu$$

$$\text{Var} = \mathbb{E}_{\text{train}}[\widehat{\mu} - \mathbb{E}_{\text{train}}(\widehat{\mu})]^2$$

Proof: Bias-Variance Decomposition (cont'd)

$$\mathbb{E}_{\text{train}}\mathbb{E}_x(\widehat{\mu}-x)^2 = \mathbb{E}_{\text{train}}\mathbb{E}_x\left[(\widehat{\mu}-\mathbb{E}_{\text{train}}(\widehat{\mu}))+(\mathbb{E}_{\text{train}}(\widehat{\mu})-\mu)+(\mu-x)\right]^2$$

- We get 6 terms. Three are: Bias^2, Var, Residual. We need to show that the three cross terms become zero.

$$\mathbb{E}_{\text{train}}\mathbb{E}_x[(\widehat{\mu}-\mathbb{E}_{\text{train}}(\widehat{\mu}))(\mathbb{E}_{\text{train}}(\widehat{\mu})-\mu)] = \mathbb{E}_{\text{train}}[(\widehat{\mu}-\mathbb{E}_{\text{train}}(\widehat{\mu}))(\mathbb{E}_{\text{train}}(\widehat{\mu})-\mu)]$$

$$= (\mathbb{E}_{\text{train}}(\widehat{\mu})-\mu)\mathbb{E}_{\text{train}}[\widehat{\mu}-\mathbb{E}_{\text{train}}(\widehat{\mu})] = \text{Bias}\times 0 = 0$$

$$\mathbb{E}_{\text{train}}\mathbb{E}_x[(\widehat{\mu}-\mathbb{E}_{\text{train}}(\widehat{\mu}))(\mu-x)] = \mathbb{E}_{\text{train}}[\widehat{\mu}-\mathbb{E}_{\text{train}}(\widehat{\mu})]\mathbb{E}_x[\mu-x] = 0\times 0 = 0$$

$$\mathbb{E}_{\text{train}}\mathbb{E}_x[(\mathbb{E}_{\text{train}}(\widehat{\mu})-\mu)(\mu-x)] = \mathbb{E}_{\text{train}}[\mathbb{E}_{\text{train}}(\widehat{\mu})-\mu]\mathbb{E}_x[\mu-x] = \text{Bias}\times 0 = 0$$

Bayesian Approaches

The Bayesian Perspective

- The Bayesian approach does not require model selection!
- One formulates all plausible models under consideration and specifies a prior probability for those models

$$P(\mathcal{M}_i)$$

- The posterior prediction becomes

$$P(y|\mathbf{x}) = \sum_i P(\mathcal{M}_i|D) \int P(y|\mathbf{x}, \mathbf{w}, \mathcal{M}_i) P(\mathbf{w}|D, \mathcal{M}_i) d\mathbf{w}$$

Bayesian Model Selection

- The principled Bayesian approach is sometimes impractical and a model selection is performed
- A posteriori model probability

$$P(\mathcal{M}|D) \propto P(\mathcal{M})P(D|\mathcal{M})$$

- If one assumes that all models have the same prior probability, and the important term is the so-called marginal likelihood, or model evidence

$$P(D|\mathcal{M}) = \int P(D|\mathbf{w}, \mathcal{M})P(\mathbf{w}|\mathcal{M})d\mathbf{w}$$

Our Favorite Linear Model

- Fortunately we can sometimes calculate the evidence without solving complex integrals. From Bayes formula we get

$$P(\mathbf{w}|D,\mathcal{M}) = \frac{P(D|\mathbf{w},\mathcal{M})P(\mathbf{w}|\mathcal{M})}{P(D|\mathcal{M})}$$

  and thus

$$P(D|\mathcal{M}) = \frac{P(D|\mathbf{w},\mathcal{M})P(\mathbf{w}|\mathcal{M})}{P(\mathbf{w}|D,\mathcal{M})}$$

- This equation must be true for any $\mathbf{w}$. Let's substitute $\mathbf{w}_{MAP}$ and take the log

$$\log P(D|\mathcal{M}) =$$

$$\log P(D|\mathbf{w}_{MAP},\mathcal{M}) + \log P(\mathbf{w}_{MAP}|\mathcal{M}) - \log P(\mathbf{w}_{MAP}|D,\mathcal{M})$$

- The first term is the log-likelihood and the second the prior. Both are readily available. So we only need to take care of the last term

Our Favorite Linear Model (cont'd)

- Recall from a previous lecture that for a Bayesian approach to linear regression, we get

$$P(\mathbf{w}|D, \mathcal{M}) = \mathcal{N}\left(\mathbf{w}; \mathbf{w}_{MAP}, cov(\mathbf{w}|D)\right)$$

Here,

$$\text{cov}(\mathbf{w}|D, \mathcal{M}) = \sigma^2 \left(\Phi^T \Phi + \frac{\sigma^2}{\alpha^2} I\right)^{-1}$$

Our Favorite Linear Model (cont'd)

- Thus at $\mathbf{w}_{MAP}$ the exponent is zero ($\exp(0) = 1$) and we are left with the *normalization term*

$$\log P(\mathbf{w}_{MAP}|D, \mathcal{M}) = \log \frac{1}{\sqrt{(2\pi)^{M_p} \det \mathrm{cov}(\mathbf{w}|D)}}$$

$$= -\frac{M_p}{2} \log(2\pi) - \frac{1}{2} \log \det \mathrm{cov}(\mathbf{w}|D)$$

Our Favorite Linear Model (cont'd)

- Thus,

$$\log P(D|\mathcal{M}) = \log P(D|\mathbf{w}_{MAP}, \mathcal{M}) + \log P(\mathbf{w}_{MAP}|\mathcal{M})$$

$$+\frac{M_p}{2}\log(2\pi) + \frac{1}{2}\log\det \mathrm{cov}(\mathbf{w}|D)$$

- For large N, one can approximate

$$\log\det \mathrm{cov}(\mathbf{w}|D) \approx -M_p \log N + constants$$

(for large N, $\mathrm{cov}(\mathbf{w}|D)$ becomes diagonal and the diagonal entries become proportional to $1/N$; thus $\det(\mathrm{cov}(\mathbf{w}|D)) \propto (1/N)^{M_p} = N^{-M_p}$ is then the product over the diagonals)

Our Favorite Linear Model (cont'd)

- Thus

$$\log P(D|\mathcal{M}) \approx \log P(D|\mathbf{w}_{MAP}, \mathcal{M}) + \log P(\mathbf{w}_{MAP}|\mathcal{M})$$

$$+\frac{M_p}{2}\log(2\pi) - \frac{1}{2}M_p \log N + constants$$

- If we consider models with different number of parameters M_p, then

$$\log P(D|w_{MAP}, \mathcal{M}) + \log P(w_{MAP}|\mathcal{M})$$

might produce a larger value (better fit) for the model with the larger M_p. But for the larger model, we subtract a larger $M_p \log N$, so we obtain a compromise between both terms at the optimum

Laplace Approximation of the Marginal Likelihood

- By simplifying the previous equation (we only keep terms that depend on N) and using the ML (Maximum Likelihood) estimate instead of the MAP estimate one obtains

$$\log P(D|\mathcal{M}) \approx \log P(D|\hat{\mathbf{w}}_{ML}, \mathcal{M}) - \frac{M_p}{2} \log N$$

- The *Bayesian information criterion* (BIC) is -2 times this expression (definition in Wikipedia)

$$\text{BIC} = -2 \log P(D|\hat{\mathbf{w}}_{ML}, \mathcal{M}) + M_p \log N$$

  (a better model has a smaller BIC)

- This approximation is generally applicable (not just for regression)

Bayesian Information Criterion (BIC)

- We get

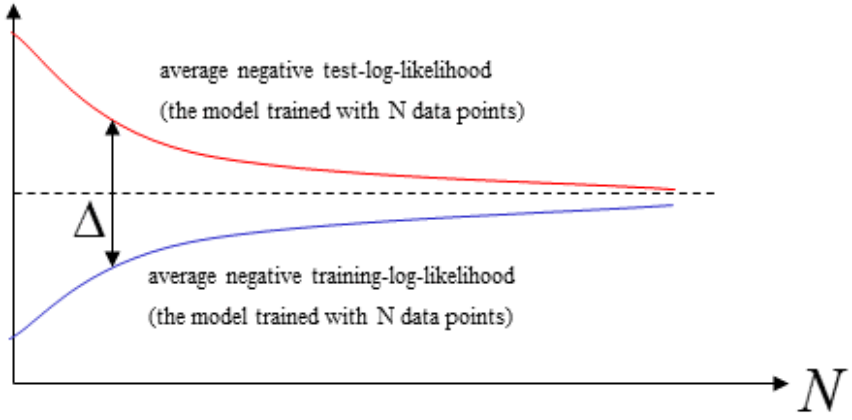
$$\frac{BIC}{2N} = \text{cost}^{l}_{\text{train}}[\widehat{\mathbf{w}}(\text{train})] + \frac{1}{2} M_p \frac{\log N}{N}$$

  Compare

$$\frac{AIC}{2N} = \text{cost}^{l}_{\text{train}}[\widehat{\mathbf{w}}(\text{train})] + \frac{M_p}{N}$$

- $\frac{1}{2}\frac{M_p}{N} \log N$ is an estimate of the difference between the average test likelihood and the average training log-likelihood
- BIC corection is by a factor $\frac{1}{2} \log N$ larger than the AIC correction and decreases more slowly $(\log N)/N$ with the number of training examples

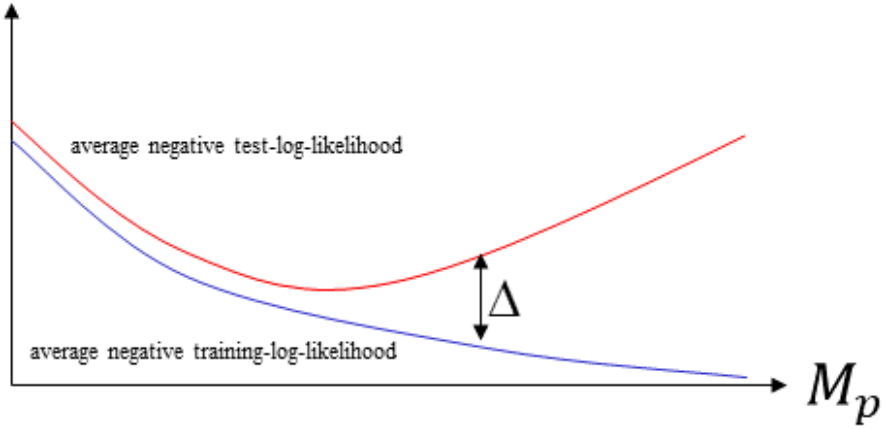
Comparison: AIC and BIC



Schätzung von D:

AIC: $\Delta = \frac{M_p}{N}$

BIC: $\Delta = \frac{M_p}{N}\frac{1}{2}\log N$



With the number of data points N, Δ decreases, with increasing complexity M_p

Δ increases

C: Modern Frequentist Approaches

Minimum Description Length (MDL)

- Based on the concept of algorithmic complexity (Kolmogorov, Solomonoff, Chaitin)
- Based on these ideas: Rissanen (and Wallace, Boulton) introduced the principal of the minimum description length (MDL)
- Under simplifying assumptions the MDL criterion becomes the BIC criterion

Statistical Learning Theory

- The Statistical Learning Theory (SLT) is in the tradition of the Russian mathematicians Andrey Kolmogorov and Valery Ivanovich Glivenko and the Italian mathematician Francesco Paolo Cantelli
- SLT was founded by Vladimir Vapnik and Alexey Chervonenkis (VC-Theory)
- Part of *Computational Learning Theory* (COLT); similar to PAC (*probably approximately correct*) Learning (Leslie Valiant)

Function Classes

- As before: data is generated according to some distribution $P(\mathbf{x}, y)$. This distribution is fixed but otherwise arbitrary
- As before: SLT considers functions out of a model function class $f_{\mathbf{w}}(\mathbf{x}) \in \mathcal{M}$. Example: $\mathcal{M}$ is the class of all linear classifiers with $M_p = M + 1$ parameters (for simplicity, we consider that an element of the function class can be described by a parameter vector $\mathbf{w}$)
- SLT does not assume that the best possible (true or target) function $f(\mathbf{x})$ is contained in $\mathcal{M}$
- We now consider binary classification without noise (i.e., classes are in principal separable)

SLT Bounds

- As before: train is a random sample of $P(\mathbf{x}, y)$
- SLT considers the difference between $\text{cost}^m_{P(\mathbf{x},y)}[\mathbf{w}]$ and $\text{cost}^m_{\text{train}}[\mathbf{w}]$ (recall that cost^m denotes the misclassification cost); we are interested in the difference between generalization cost and average training cost for any $\mathbf{w}$ (not just for $\widehat{\mathbf{w}}(\text{train})$)
- As before: probabilities are calculated w.r.t. all training sets of size N

SLT Bounds (cont'd)

- SLT shows that with (large) probability $1 - \eta$ that for any $\mathbf{w}$,

$$\text{cost}^m_{P(\mathbf{x},y)}[\mathbf{w}] \leq \text{cost}^m_{\text{train}}[\mathbf{w}] + \epsilon$$

- Model selection is performed on

$$\text{cost}^m_{\text{train}}[\widehat{\mathbf{w}}(\text{train})] + \epsilon$$

(note; for model selection, we consider the particular parameter choice $\widehat{\mathbf{w}}(\text{train})$)

VC-Dimension

- The statement is trivially true for a large enough ϵ; the science (and the art) is now to find the smallest possible ϵ
- Of great importance is the so-called VC-dimension dim_{VC} of the model class
- Wikipedia: In Vapnik-Chervonenkis theory, the VC dimension (for Vapnik-Chervonenkis dimension) is a measure of the capacity (complexity, expressive power, richness, or flexibility) of a space of functions that can be learned by a statistical classification algorithm. It was originally defined by Vladimir Vapnik and Alexey Chervonenkis
- The VC-dimension can be finite or infinite. For linear classifiers, $dim_{VC} = M_p = M + 1$, which means that the VC-dimension is simply the number of parameters
- For systems with a finite VC dimension, the bound decreases with N when $N > dim_{VC}$.
- In practice, $\text{cost}^m_{\text{train}}[\widehat{\mathbf{w}}(\text{train})] + \epsilon$ is much larger than the average test set error, which limits the application of the theory in practice

Typical Bound

- A typical estimate is

$$\epsilon = \sqrt{\frac{1}{N}\left[dim_{VC}\left(\log\left(\frac{2N}{dim_{VC}}\right)+1\right)-\log\left(\frac{\eta}{4}\right)\right]}$$

Comparison

- Comparison of the ϵ

C_P	$\propto \frac{M_P}{N}$
AIC	$\propto \frac{M_P}{N}$
BIC	$\propto \frac{M_P \log(N)}{N}$
VC	$\propto \frac{\sqrt{M_P}}{\sqrt{N}} = \frac{\sqrt{M_P N}}{N}$

Comparing SLT and a Frequentist Approach

- SLT does not make any assumption about the true function; in a frequentist view, this can mean that the bias can be large
- SLT makes a statement about the worst case (supremum); often the supremum corresponds to the weight vector that minimizes the training cost, i.e. $\widehat{\mathbf{w}}(\mathrm{train})$
- Note: training data is are not worse case (they are not selected by a demon to fool you): they are assumed to be generated i.i.d. from a fixed $P(y|\mathbf{x})P(\mathbf{x})$

Conclusion

- Machine learning focusses on generalization costs and traditionally not as much on parameter estimation, although explainable AI is gaining interest
- Empirical model selection is most often used. If possible, cross validation results should be reported
- Frequentist approaches typically estimate $\mathbb{E}_{\text{train}}\text{cost}_{P(\mathbf{x},y)}[\widehat{\mathbf{w}}(\text{train}), \mathcal{M}]$. We have studied C_P and the AIC
- Bayesian approaches do model averaging instead of model selection. The BIC criterion is useful, if model selection needs to be performed
- An advantage of the SLT is that the true function does not need to be included in the function class; the derived bounds are typically often rather conservative
- SLT has been developed in the Machine Learning community, whereas frequentist and Bayesian approaches originated in statistics

Appendices

P-norm Cost Functions

- A p-norm is

$$\|x\|_p := \left(\sum_{i=1}^{N} |x_i|^p \right)^{1/p}$$

- Let $\mathbf{y}$ be targets on the training data and $\hat{\mathbf{y}}$ the vector of predictions of the classifier; based on the p-norm, we can define different distances between $\mathbf{y}$ and $\hat{\mathbf{y}}$, which then imply different costs
- Then the quadratic cost can be written as $\|\mathbf{y} - \hat{\mathbf{y}}\|_2^2$ (squared Euclidean distance)
- The absolute value cost is $\|\mathbf{y} - \hat{\mathbf{y}}\|_1$ (Manhatten distance)
- $\|\mathbf{y} - \hat{\mathbf{y}}\|_\infty$ is the infinity norm which is equal to the maximum absolute value of its components (Maximumsnorm)
- ℓ^p-norms generalize the p-norm to infinite sequences
- L^p-norms generalize the p-norm to function spaces

Evaluation

- The cost functions defined before can also be used to evaluate the performance on the test data; but measures like the cross-entropy on test data are difficult to interpret and the goal is to have performance measures independent of the particular cost function used in training
- For classification one considers the 2×2 table of $y \in \{0, 1\}$ and $\widehat{y} \in \{0, 1\}$ on the test data
- For balanced classes one typically reports mutual information and accuracy
- For unbalanced classes one focusses on measures derived from the co-occurrence distribution

Measures for Performance Evaluation

- True Positive (a.k.a. hit)$(y, \widehat{y} \in \{0, 1\})$:

$$\mathsf{TP}[\mathbf{w}] = \sum_{i \in test} y_i \widehat{y}_i$$

- True Negative:

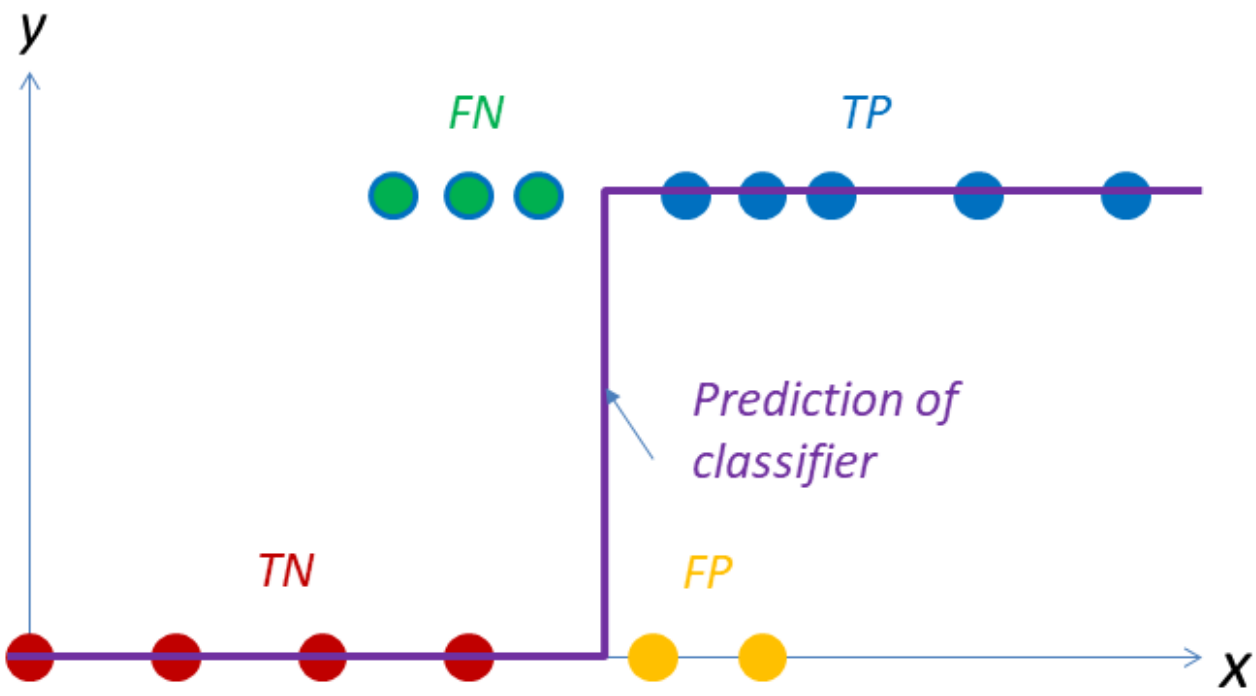
$$\mathsf{TN}[\mathbf{w}] = \sum_{i \in test} (1 - y_i)(1 - \widehat{y}_i)$$

- False Positive (a.k.a. false alarm, type I error):

$$\mathsf{FP}[\mathbf{w}] = \sum_{i \in test} (1 - y_i)\widehat{y}_i$$

- False Negative (a.k.a. miss, type II error):

$$\mathsf{FN}[\mathbf{w}] = \sum_{i \in test} y_i(1 - \widehat{y}_i)$$



FN=3	TP=5
TN=4	FP=2

y (label)

0

$\hat{y}$ (predicted)

Precision, Recall, and Specificity

Based on these, one defines

- Precision (positive predicted value):

$$\frac{\mathsf{TP}}{\mathsf{TP}+\mathsf{FP}} \approx P(Y=1|\widehat{Y}(X)=1)$$

- Recall (sensitivity, true positive rate, hit rate, or detection rate):

$$\frac{\mathsf{TP}}{\mathsf{TP}+\mathsf{FN}} \approx P(\widehat{Y}(X)=1|Y=1)$$

- Specificity (True Negative Rate):

$$\frac{\mathsf{TN}}{\mathsf{FP}+\mathsf{TN}} \approx P(\widehat{Y}(X)=0|Y=0)$$

F1 and Accuracy

- F1-score

$$\mathsf{F1} = 2\frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

- Accuracy

$$\frac{\mathsf{TP} + \mathsf{TN}}{\mathsf{TP} + \mathsf{TN} + \mathsf{FP} + \mathsf{FN}}$$

Interpretation: Inverse Model

- Consider a fire detector where X is the degree of smoke associated with a fire $Y = 1$; the fire is the cause, the smoke is the effect, $Y \rightarrow X$
- The fire detector $\widehat{Y}(X)$ tries to model the inverse: predict the probability of a fire Y given the degree of smoke X
- Recall (Sensitivity) is the probability that there is an alarm when there is a fire: $P(\widehat{Y}(X) = 1|Y = 1)$
- Specificity is the probability that there is no alarm when there is no fire: $P(\widehat{Y}(X) = 0|Y = 0)$
- **Recall and Specificity** condition on Y and are thus independent on the prevalence of fire in a neighborhood. They measure the **performance of a fire detector** and are the basis for the receiver operating characteristic (ROC) curve (and the area under ROC (AU-ROC))

Interpretation: Inverse Model (cont'd)

-

- Precision is the probability that there is a fire when there is an alarm: $P(Y = 1|\widehat{Y}(X) = 1)$
- **Usefulness** in an application: **Recall and Precision** is what you typically care about in an application: you don't want to have false alarms (you want to have a high precision) and you don't want to miss a fire (you want to have a high recall)
- But Precision is not independent of the prevalence of fire in an environment!
- Precision and Recall are used in the Precision Recall (PR) curve (and the area under PR (AU-PR))

Interpretation: Forward Model

- Y is the interest of a user (the effect), X is the web page (the cause) and $\widehat{Y}(X)$ is the estimated interest, $X \rightarrow Y$
- The model tries to model the forward model: predict the probability of a user interest given X
- Recall (Sensitivity) is the probability that we label each relevant web page as being interesting (each relevant page should be shown to you):
  $P(\widehat{Y} = 1(X)|Y = 1)$
- Precision is the probability that a web page is of interest when we label it to be interesting (if a page is shown to you, it should be relevant):
  $P(Y = 1|\widehat{Y}(X) = 1)$
- Again, in the application one is interested in a high Recall and a high Precision