Model Selection and Estimation of the Generalization Cost

Volker Tresp
Summer 2015

Empirical Model Comparison

Model Comparison

- Let's consider the prediction of a model $\mathcal{M}$ for input $\mathbf{x}$ and with parameters $\mathbf{w}$

$$f(\mathbf{x}, \mathbf{w}, \mathcal{M})$$

- Example: $\mathcal{M}$ can be a neural network with a specific architecture. Another example: $\mathcal{M}$ is a linear regression model
- We define a cost function (loss function)

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

- Example (quadratic loss):

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

- We will use the terms cost, loss and error exchangeably

Further Examples of Cost Functions

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

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

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

$$\text{cost}_{\mathbf{x},y}[\mathbf{w},\mathcal{M}] = \sum_{i=1}^{N} \log\left(1 + \exp\left[-y_i f(\mathbf{x},\mathbf{w},\mathcal{M})\right]\right)$$

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

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

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

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

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

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

Generalization Cost

- In statistics one is often interest in the estimation of the value and the uncertainty of particular parameters. Example: is w_1 significantly less than zero?
- In machine learning one is often interested in the **generalization cost** which is the mean expected loss over all possible seen and unseen data, **for any $\mathbf{w}$**

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

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

Estimating the Generalization Cost Over Test Data

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

$$\text{cost}_{P(\mathbf{x},y)}[\mathbf{w}, \mathcal{M}] \approx \text{cost}_{\text{test}}[\mathbf{w}, \mathcal{M}] = \frac{1}{T} \sum_{i=1}^{T} \text{cost}_{\mathbf{x}_i, y_i}[\mathbf{w}, \mathcal{M}]$$

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

- This is an unbiased estimator, for any $\mathbf{w}$

Estimating the Generalization Cost Over Training Data

- An estimator of the generalization cost is also the **mean training set cost**

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

which is the mean cost on the N training data points with $(\mathbf{x}_i, y_i \in \text{train})$

- This is also an unbiased estimator, for any $\mathbf{w}$

Estimating the Generalization Cost of the Best Fit

- Now comes the difference: if, for each training set, I look at the parameters that minimize the training error, then, on average, the training set error is smaller than the generalization error for these parameters!

- Let $\widehat{\mathbf{w}}|\text{train}$ be the estimator (the minimizer of the training cost) on a training set $D = \text{train}$. In other words $\widehat{\mathbf{w}}|\text{train}$ minimizes the cost on the training data set.

- We define the **mean training set cost** of this best parameter vector, evaluated **on the training data** as

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

Here, $(\mathbf{x}_i, y_i) \in \text{train}$

Analysis of Different Quantities

- One quantity we are interested in is the generalization cost of a particular model $\mathcal{M}$ with particular best-fit parameters $\widehat{\mathbf{w}}|\text{train}$.

$$\text{cost}_{P(\mathbf{x},y)}[\widehat{\mathbf{w}}|\text{train}, \mathcal{M}]$$

- Another quantity of interest is the generalization cost averaged over all training sets of size N (where the training data are generated from $P(\mathbf{x}, y)$),

$$E_{\text{train}}\text{cost}_{P(\mathbf{x},y)}[\widehat{\mathbf{w}}|\text{train}, \mathcal{M}]$$

  Note that here we analyse the performance of a particular model $\mathcal{M}$!

Training Set Cost and Generalization Cost

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

$$E_{\text{train}} \left\{ \text{cost}_{\text{train}}[\hat{\mathbf{w}}|\text{train}, \mathcal{M}] - \text{cost}_{P(\mathbf{x},y)}[\hat{\mathbf{w}}|\text{train}, \mathcal{M}] \right\} \leq 0$$

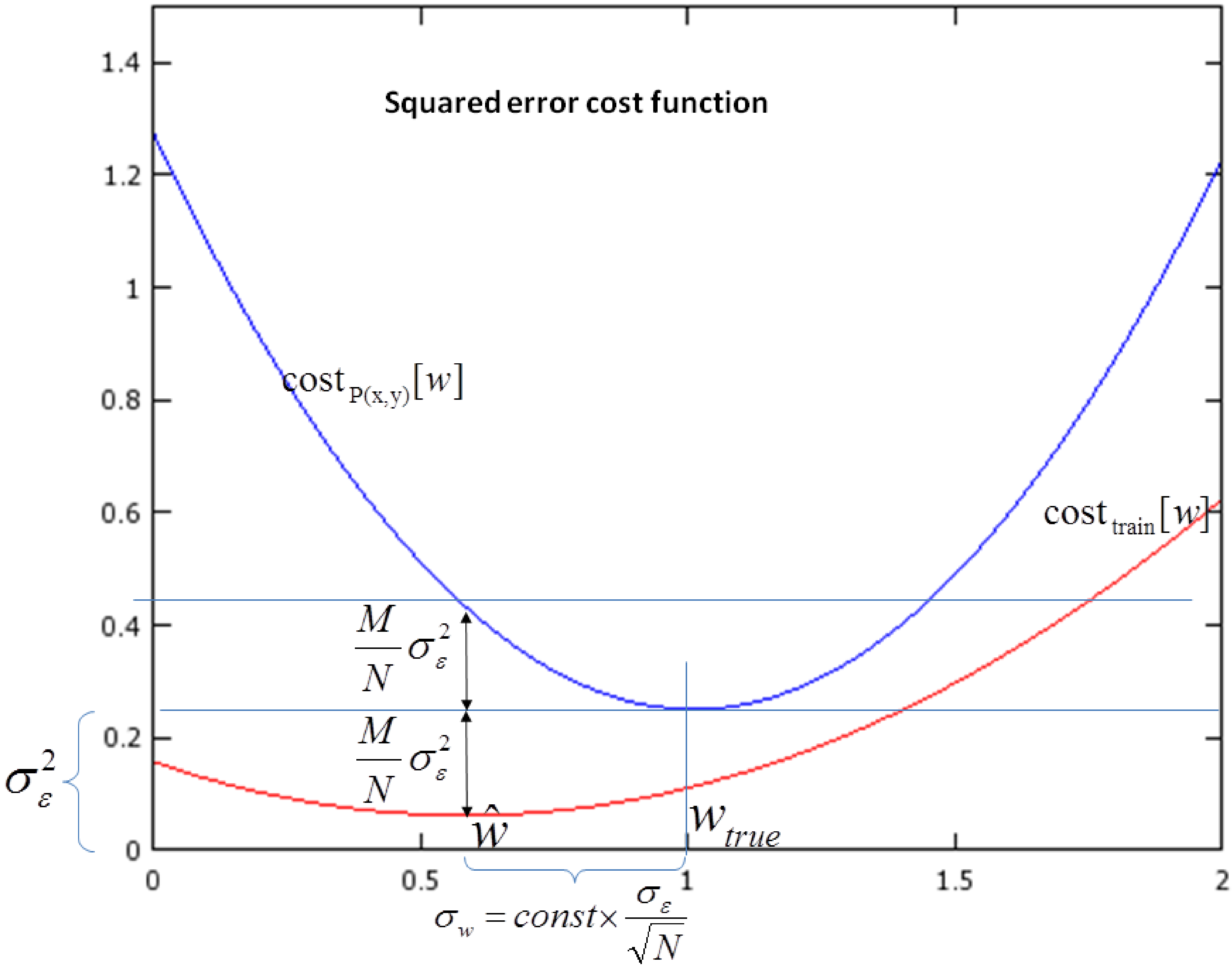
- Thus in expectation, the training cost underestimates the generalization cost for the estimated $\hat{w}$. Thus the performance of a trained model should not be evaluated on the training set but on the test set, which is an unbiased estimator of the generalization cost, also for $\hat{\mathbf{w}}|\text{train}$.

Visualization: Regression

- Input data is generated from a Gaussian with zero mean and unit variance. y is generated with $y = wx + \epsilon$ where in the true model $\mathbf{w}_{true} = 1$ and ϵ is Gaussian noise with $\sigma_\epsilon^2 = 0.25$. We consider the quadratic loss function.
- The blue line shows the generalization cost as a function of $\mathbf{w}$, i.e., $\text{cost}^q_{P(\mathbf{x},y)}[\mathbf{w}, \mathcal{M}]$
- The red line shows $\text{cost}^q_{\text{train}}[\mathbf{w}, \mathcal{M}]$ based on 5 specific training data points.
- We see that the location of the minimum $\hat{\mathbf{w}}|\text{train}$ is not identical to $\mathbf{w}_{true}$. Since the estimator is unbiased, $E_{\text{train}}(\hat{\mathbf{w}}|\text{train}) = \mathbf{w}_{true}$. We will see later that $\sigma_w = 0.5/\sqrt{5} = 0.22$. This is the focus in statistics: how well can I estimate parameters?

Visualization: Regression (cont'd)

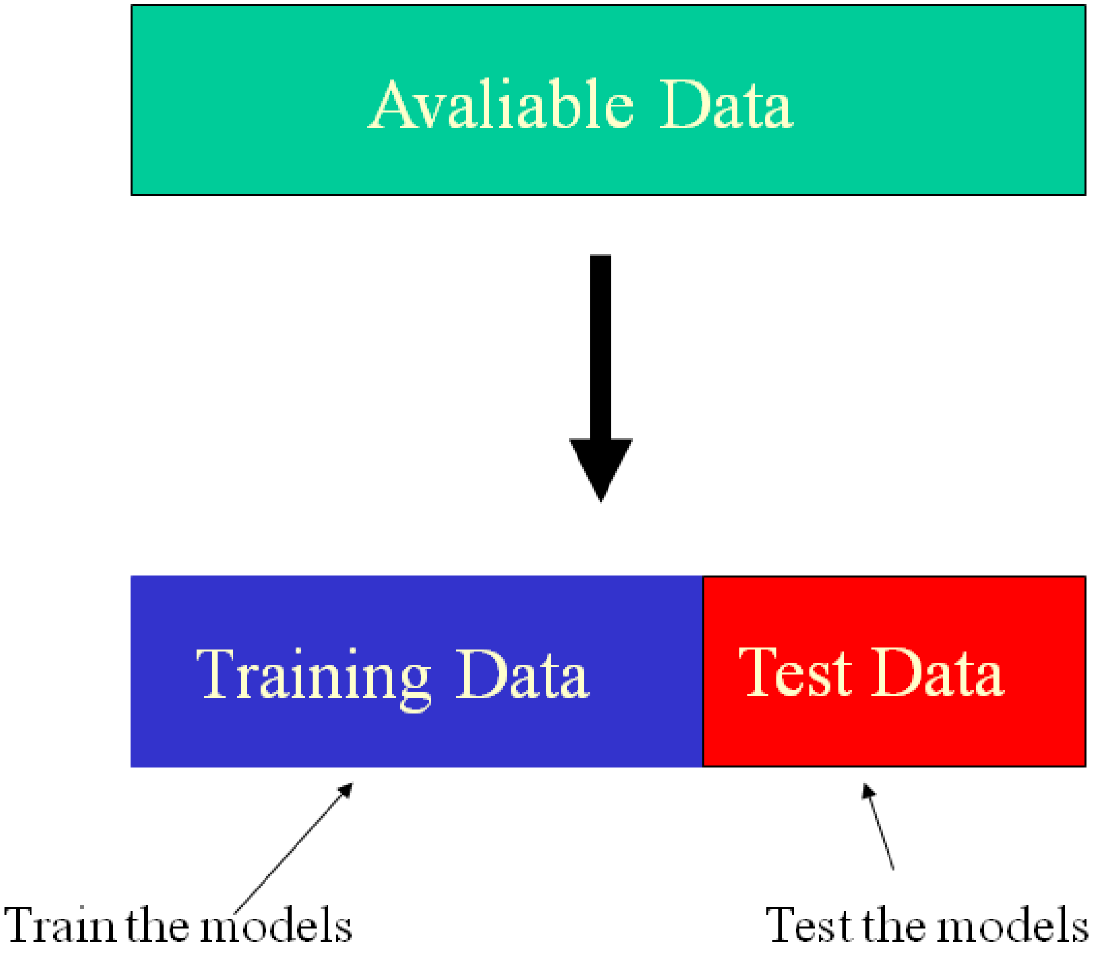
- Now we look at the costs
- For the model with the best possible generalization cost: $\text{cost}^{q}_{P(\mathbf{x},y)}[\mathbf{w}_{true}, \mathcal{M}] = \sigma^2_\epsilon$, of course
- We will see later that, for linear models, that, on average, the training error at $\widehat{w}$ is smaller than the generalization error of the best possible model (with $f_{true}(\mathbf{x})$) and the generalization error at $\widehat{w}$ is larger than the generalization error of the best possible model. The difference in both cases is $(M/N)\sigma^2_\epsilon$, where M is the number of parameters (in the example, $M = 1$)
- This is the focus in machine learning: for which model can we expect to obtain the best generalization performance?



Squared error cost function
1.4
1.2
1
0.8
0.6
0.4
0.2
0
0
0.5
1
1.5
2
$\text{cost}_{\text{P(x,y)}}[w]$
$\text{cost}_{\text{train}}[w]$
$\frac{M}{N}\sigma_\varepsilon^2$
$\frac{M}{N}\sigma_\varepsilon^2$
σ_ε^2
$\hat{w}$
w_{true}
$\sigma_w = const \times \frac{\sigma_\varepsilon}{\sqrt{N}}$

Model Selection via Test Data

- This procedure can be applied with huge amounts of available data, $N >> M$
- 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 test set performance and get $\text{cost}_{\text{test}}[\hat{\mathbf{w}}|\text{train}, \mathcal{M}]$ for the different models, as an estimate of the generalization costs $\text{cost}_{P(\mathbf{x},y)}[\hat{\mathbf{w}}|\text{train}, \mathcal{M}]$ for a particular model $\mathcal{M}$ with a particular parameter vector $\hat{\mathbf{w}}|\text{train}$


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 is used for testing and the remaining data is used for training (finding the best parameters)

Evaluating Performance with Cross Validation

- For each model one gets K test costs

$$\text{cost}_{\text{test}_k}[\hat{\mathbf{w}}|\text{train}_k, \mathcal{M}], \quad k = 1, \ldots, K$$

- Now we now consider the generalization costs averaged over the parameter estimates obtained from different training data sets of size N

$$E_{\text{train}} \text{cost}_{P(\mathbf{x},y)}[\hat{\mathbf{w}}|\text{train}, \mathcal{M}]$$

- We can estimate this average as

$$E_{\text{train}} \text{cost}_{P(\mathbf{x},y)}[\hat{\mathbf{w}}|\text{train}, \mathcal{M}] \approx mean(\text{cost}_{\text{test}_k}[\hat{\mathbf{w}}|\text{train}_k, \mathcal{M}])$$

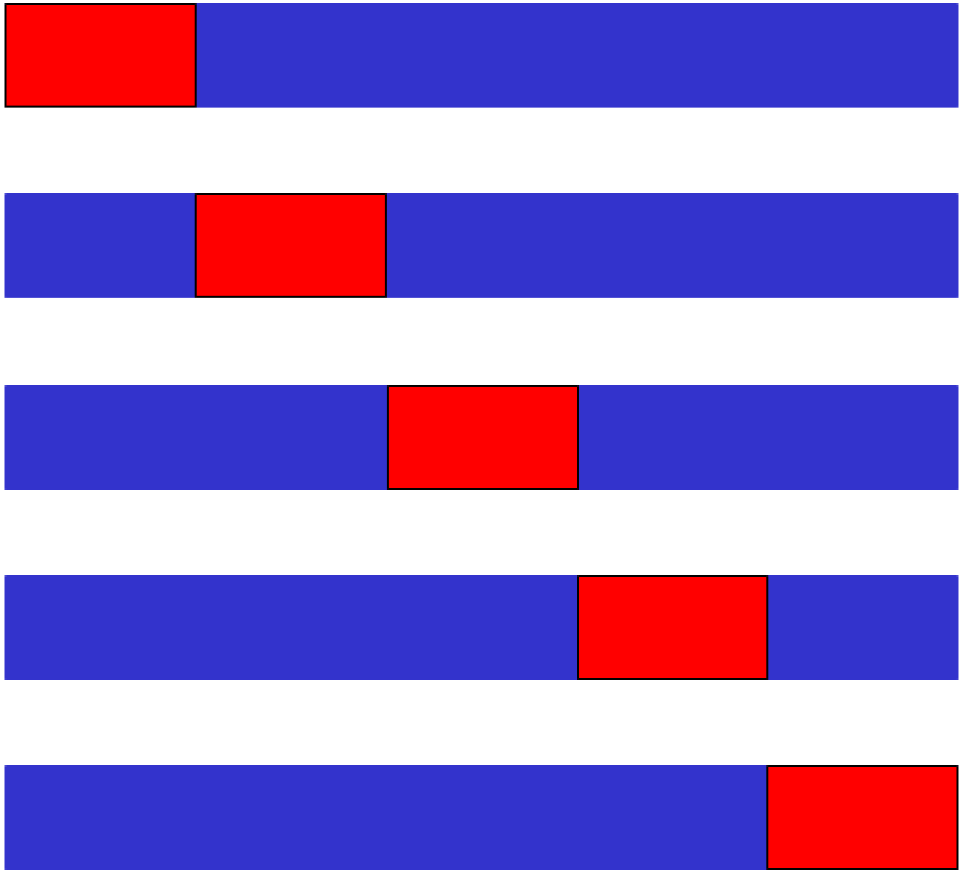
$$= \frac{1}{K} \sum_{k=1}^{K} \text{cost}_{\text{test}_k}[\hat{\mathbf{w}}|\text{train}_k, \mathcal{M}]$$

- If we simplify notation and set $m = mean(\text{cost}_{\text{test}_k}[\hat{\mathbf{w}}|\text{train}_k, \mathcal{M}])$ then we can estimate the variance of m as

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

  This can be used to decide if two models significantly differ in generalization performance

- Note, that we evaluate a model $\mathcal{M}$ and not some specific parameter vector



5-times cross validation:

Blue: Trainings Data

Red: Test Data

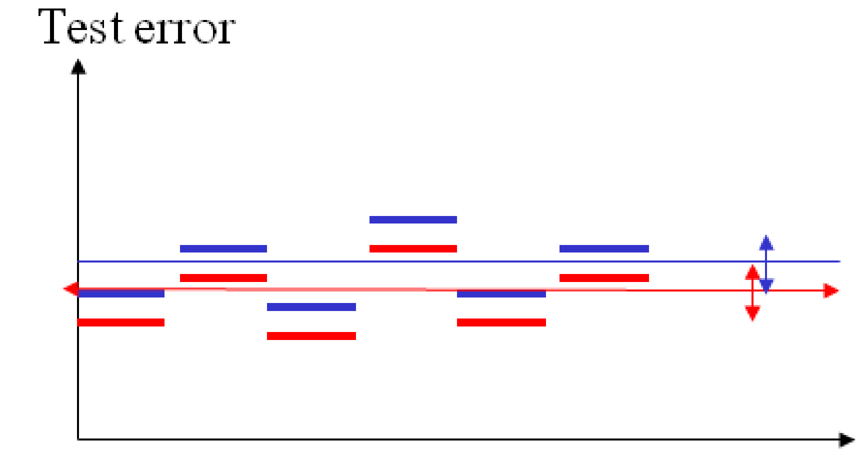
Paired Tests

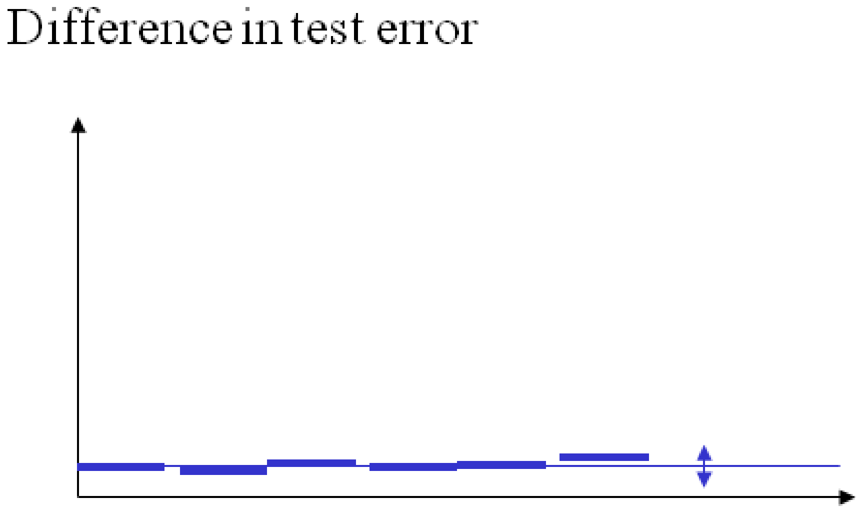
- With few data one can use a paired test
- Basic idea: let's assume that $K = 10$; if $\mathcal{M}_i$ in all test sets is better than $\mathcal{M}_j$, then this is s strong indication that $\mathcal{M}_i$ performs better, even if the variation in test set performance masks this behavior (error bars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

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

  and analyse if this difference is significantly different from zero; this can be shown by employing the test statistics for the paired t-test.

- Alternative: Wilcoxon signed-rank test





- 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)

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

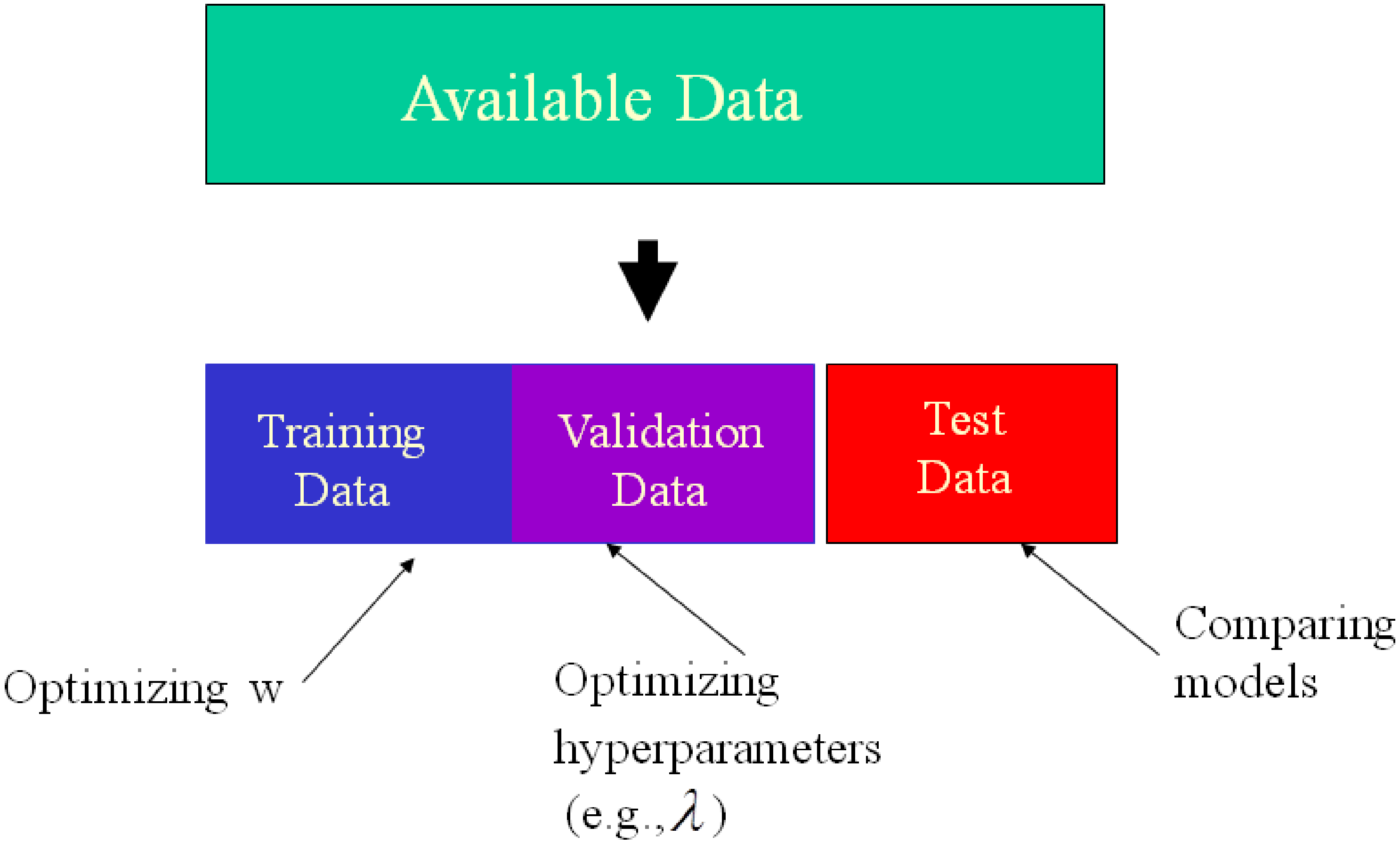
Empirical Tuning of Hyperparameters

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

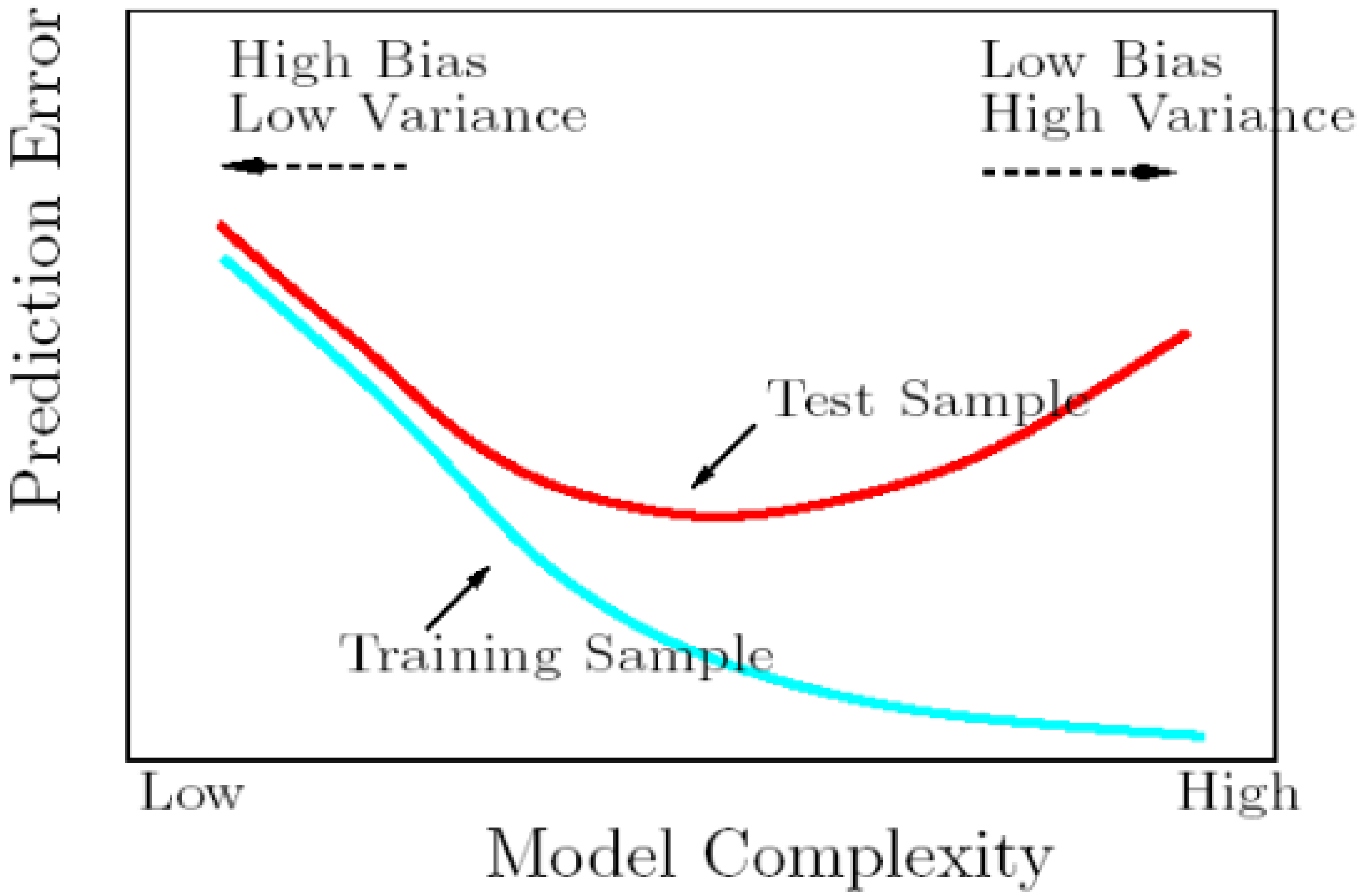
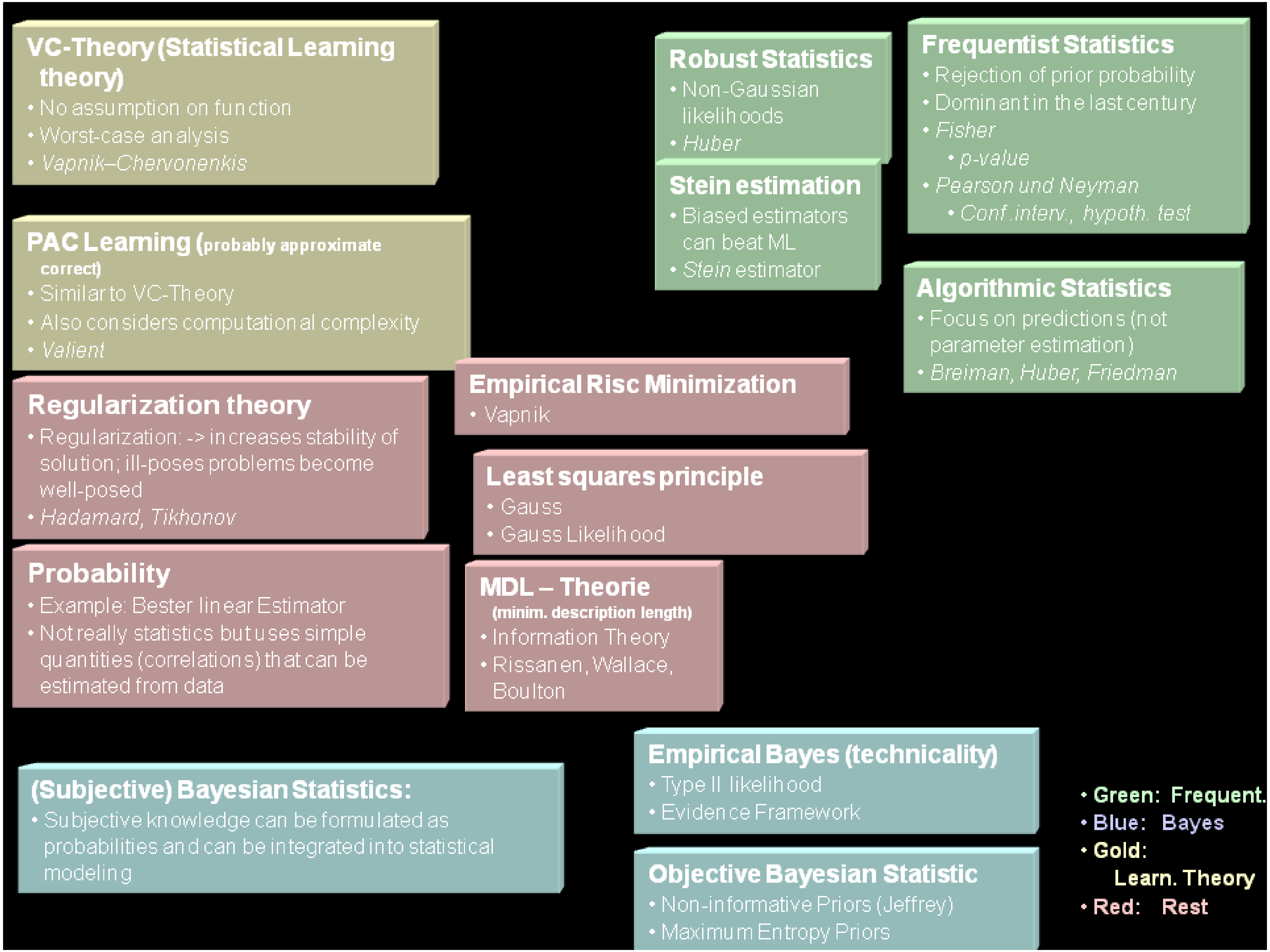


Figure 7.1: *Behavior of test sample and training sample error as the model complexity is varied.*

Learning Theories

Overview: Statistical Theories and Learning Theories



VC-Theory (Statistical Learning theory)

- No assumption on function
- Worst-case analysis
- *Vapnik–Chervonenkis*

PAC Learning (probably approximate correct)

- Similar to VC-Theory
- Also considers computational complexity
- *Valient*

Regularization theory

- Regularization: -> increases stability of solution; ill-poses problems become well-posed
- *Hadamard, Tikhonov*

Probability

- Example: Bester linear Estimator
- Not really statistics but uses simple quantities (correlations) that can be estimated from data

Empirical Risc Minimization

- Vapnik

Least squares principle

- Gauss
- Gauss Likelihood

MDL – Theorie (minim. description length)

- Information Theory
- Rissanen, Wallace, Boulton

Robust Statistics

- Non-Gaussian likelihoods
- *Huber*

Stein estimation

- Biased estimators can beat ML
- *Stein* estimator

Frequentist Statistics

- Rejection of prior probability
- Dominant in the last century
- *Fisher*
  - *p-value*
- *Pearson und Neyman*
  - *Conf.interv., hypoth. test*

Algorithmic Statistics

- Focus on predictions (not parameter estimation)
- *Breiman, Huber, Friedman*

(Subjective) Bayesian Statistics:

- Subjective knowledge can be formulated as probabilities and can be integrated into statistical modeling

Empirical Bayes (technicality)

- Type II likelihood
- Evidence Framework

Objective Bayesian Statistic

- Non-informative Priors (Jeffrey)
- Maximum Entropy Priors

- Green: Frequent.
- Blue: Bayes
- Gold: Learn. Theory
- Red: Rest

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 (Appendix)
  - 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

$$E_{\text{train}}\text{cost}_{P(\mathbf{x},y)}[\hat{\mathbf{w}}|\text{train}, \mathcal{M}]$$

- Thus we evaluate the quality of a particular model $\mathcal{M}$

Bias-Variance Decomposition

- We assume that the (fixed) true function can be realized by a function out off the function class and we assume a quadratic cost function. Then one can decompose for the *squared loss*

$$E_{\text{train}} \text{cost}^{q}_{P(\mathbf{x},y)}[\hat{\mathbf{w}}|\text{train}, \mathcal{M}] = \text{Bias}^2 + \text{Var} + \text{Residual}$$

Residual

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

$$\text{Residual} = \int (f_{true}(x) - y)^2 P(\mathbf{x}, y) dx dy = \text{cost}^q_{P(\mathbf{x},y)}[f_{true}]$$

- In regression simply the noise variance σ_ϵ^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 is biased. With $m(x) = E_{\text{train}}(f(\mathbf{x}, \hat{\mathbf{w}}|\text{train}))$

$$\text{Bias}^2 = \int [m(x) - f_{true}(x)]^2 \, P(x) dx$$

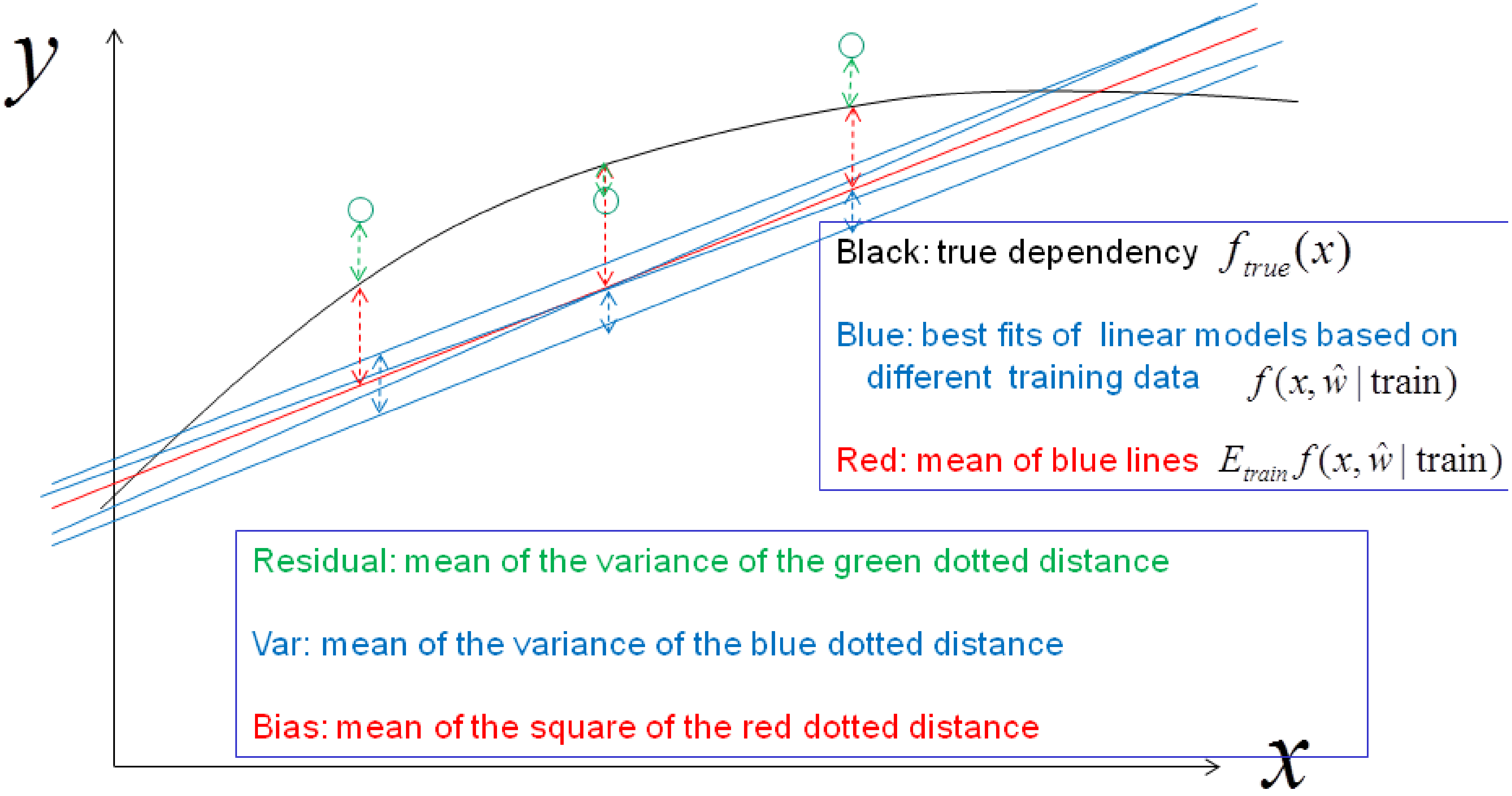
$$= \text{cost}^q_{y=f_{true}(x), P(x)}[m(x)]$$

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

$$\text{Var} = \int E_{\text{train}}[f(\mathbf{x}, \widehat{\mathbf{w}}|\text{train}) - m(x)]^2 P(x)dx$$

$$= E_{\text{train}} \text{cost}^q_{\{y=m(x),P(x)\}}[(f(\mathbf{x}, \widehat{\mathbf{w}}|\text{train}))]$$



Black: true dependency $f_{true}(x)$

Blue: best fits of linear models based on different training data $f(x, \hat{w} \mid \text{train})$

Red: mean of blue lines $E_{train} f(x, \hat{w} \mid \text{train})$

Residual: mean of the variance of the green dotted distance

Var: mean of the variance of the blue dotted distance

Bias: mean of the square of the red dotted distance

Background: Some Rules for Variances and Traces

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

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

- 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 has an inverse, then $\Phi(\Phi^T\Phi)^{-1}\Phi^T = I$ and the trace of I is obviously M

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 $Bias = 0$.

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

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

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

- Great, now we know how certain the parameters are. We can now evaluate Var by taking a large sample of $P(x)$. 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 $\mathbf{f} = \Phi\mathbf{w}$
- Applying the covariance formula again as before, we get

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

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

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

Example: Linear Models (cont'D)

- Substituting, we get

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

- Now we apply our trace-rule and get

$$\widehat{\mathrm{Var}} = \frac{M}{N} \sigma^2$$

- 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 Error of the Best Fit

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

$$E_{\mathsf{train}}\mathsf{cost}^{q}_{P(\mathbf{x},y)}[\widehat{\mathbf{w}}|\mathsf{train},\mathcal{M}] \approx \sigma^2 + \frac{M}{N}\sigma^2 = \sigma^2\frac{M+N}{N}$$

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

Training Error of the Best Fit

- We estimate σ^2 as

  $$\hat{\sigma}^2 = \frac{N}{N-M} \text{cost}^q_{\text{train}}[\hat{\mathbf{w}}|\text{train}]$$

  We get

  $$E_{\text{train}} \text{cost}^q_{\text{train}}[\hat{\mathbf{w}}|\text{train}] = \frac{N-M}{N}\sigma^2 = \sigma^2 - \frac{M}{N}\sigma^2$$

- Thus the training error is on average smaller than the generalization of the best possible model and the difference is again $\frac{M}{N}\sigma^2$

C_P-statistics

- By substitution we now get

$$E_{\text{train}}\text{cost}^q_{P(\mathbf{x},y)}[\hat{\mathbf{w}}|\text{train},\mathcal{M}] \approx \frac{M+N}{N-M}\text{cost}^q_{\text{train}}[\hat{\mathbf{w}}|\text{train}]$$

$$= \text{cost}^q_{\text{train}}[\hat{\mathbf{w}}|\text{train}] + 2\frac{M}{N}\hat{\sigma}^2$$

- This is called Mallot's C_P-statistics
- Thus in model selection on would chose the model where Mallot's C_P is smallest

Considering Bias

- Discussion: if two models are unbiased, the estimate of the noise level should be more or less the same and the C_P-statistics will chose the smaller one. If the model becomes too simple (we get bias), the training error and thus also the estimate of the noise will also start to contain the bias term

$$\hat{\sigma}^2 = \frac{N}{N-M} \text{cost}^q_{\text{train}}[\hat{\mathbf{w}}|\text{train}]$$

- Thus, as a heuristics, we can also apply the C_P-statistics to biased models if each model estimates its individual noise variance

- Thus one can argue that the smallest unbiased model will give the optimal C_P-statistics

Bias Estimate

- One might estimate

$$\widehat{\mathsf{Bias}}^2 = \hat{\sigma}^2 - \hat{\sigma}^2_\infty$$

whereas $\hat{\sigma}^2_\infty$ is the estimate for a sufficiently large unbiased model

Conclusion

- The estimates are

$$\widehat{\text{Residual}} = \hat{\sigma}^2_\infty$$

$$\widehat{\text{Bias}} = \hat{\sigma}^2 - \hat{\sigma}^2_\infty$$

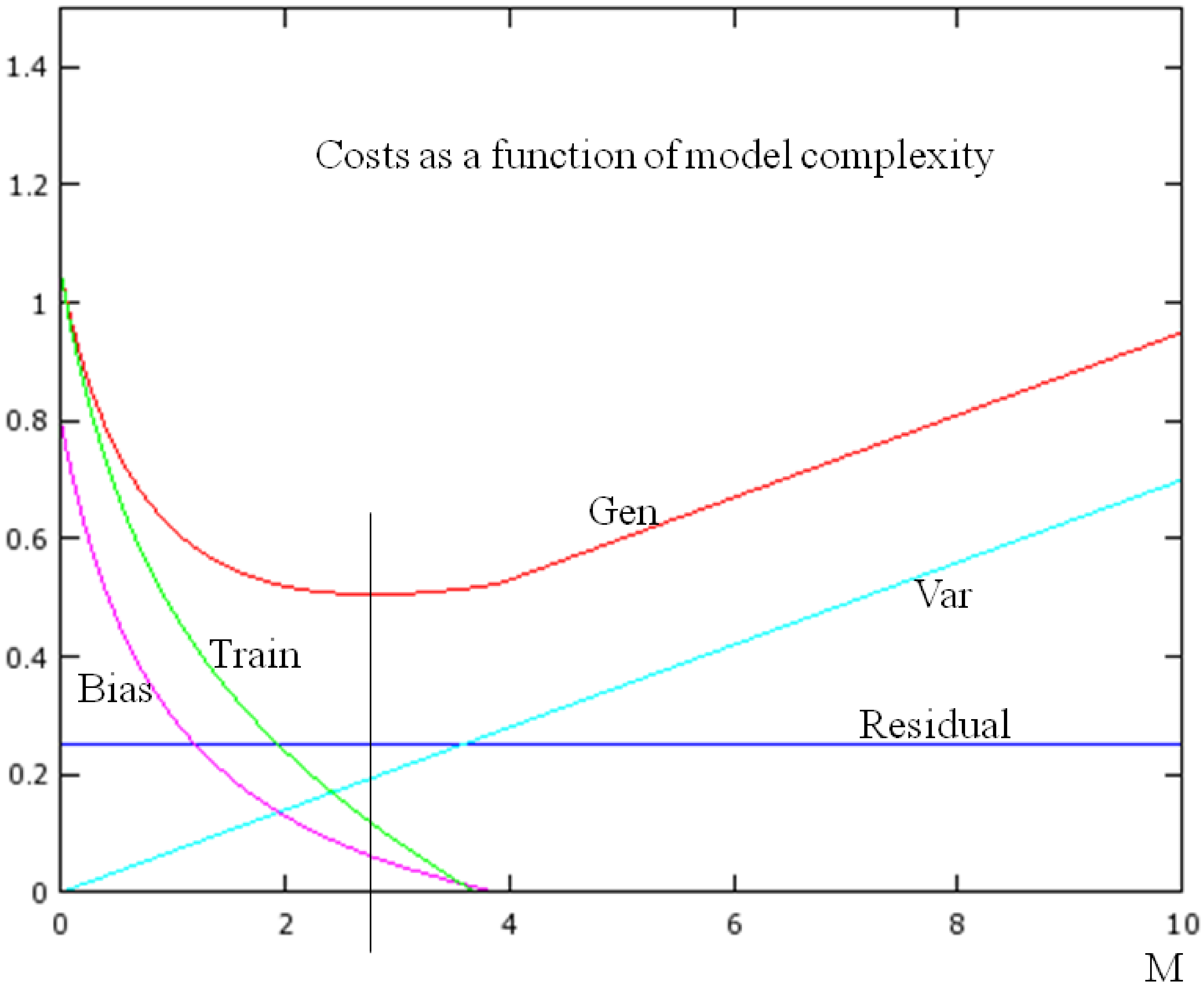
$$\widehat{\text{Var}} = \frac{M}{N}\hat{\sigma}^2_\infty$$

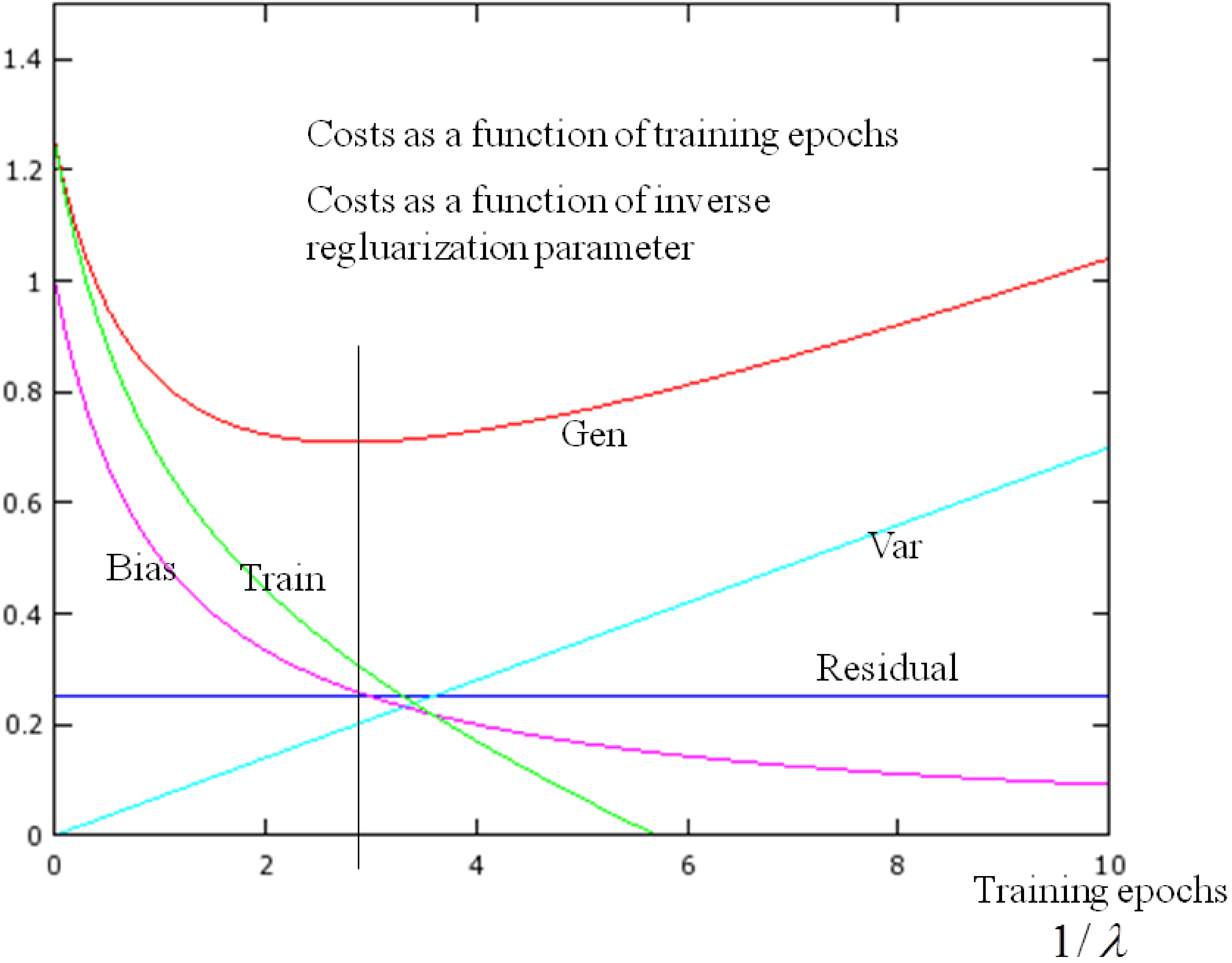
Conceptual Plots

- The next figures show the behavior of bias, variance and residual and mean training and mean test costs

- The complexity is controlled by the number of parameters M, 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
Gen
Var
Train
Bias
Residual
M
0
2
4
6
8
10
0
0.2
0.4
0.6
0.8
1
1.2
1.4



Costs as a function of training epochs

Costs as a function of inverse regluarization parameter

Akaikes Information Criterion (AIC)

- The analysis so far was only valid for models that minimized the squared error. Consider approaches where the log-likelihood is minimized

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

- In other words

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

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

$$AIC/N = 2\left(-\frac{1}{N}\log L + \frac{M}{N}\right) = 2\left(\text{cost}^{l}_{\text{train}}[\hat{\mathbf{w}}|\text{train}, \mathcal{M}] + \frac{M}{N}\right)$$

- A model with a smaller AIC is preferred

Comments on AIC

- AIC is equivalent to C_p for Gaussian noise with known noise variance:

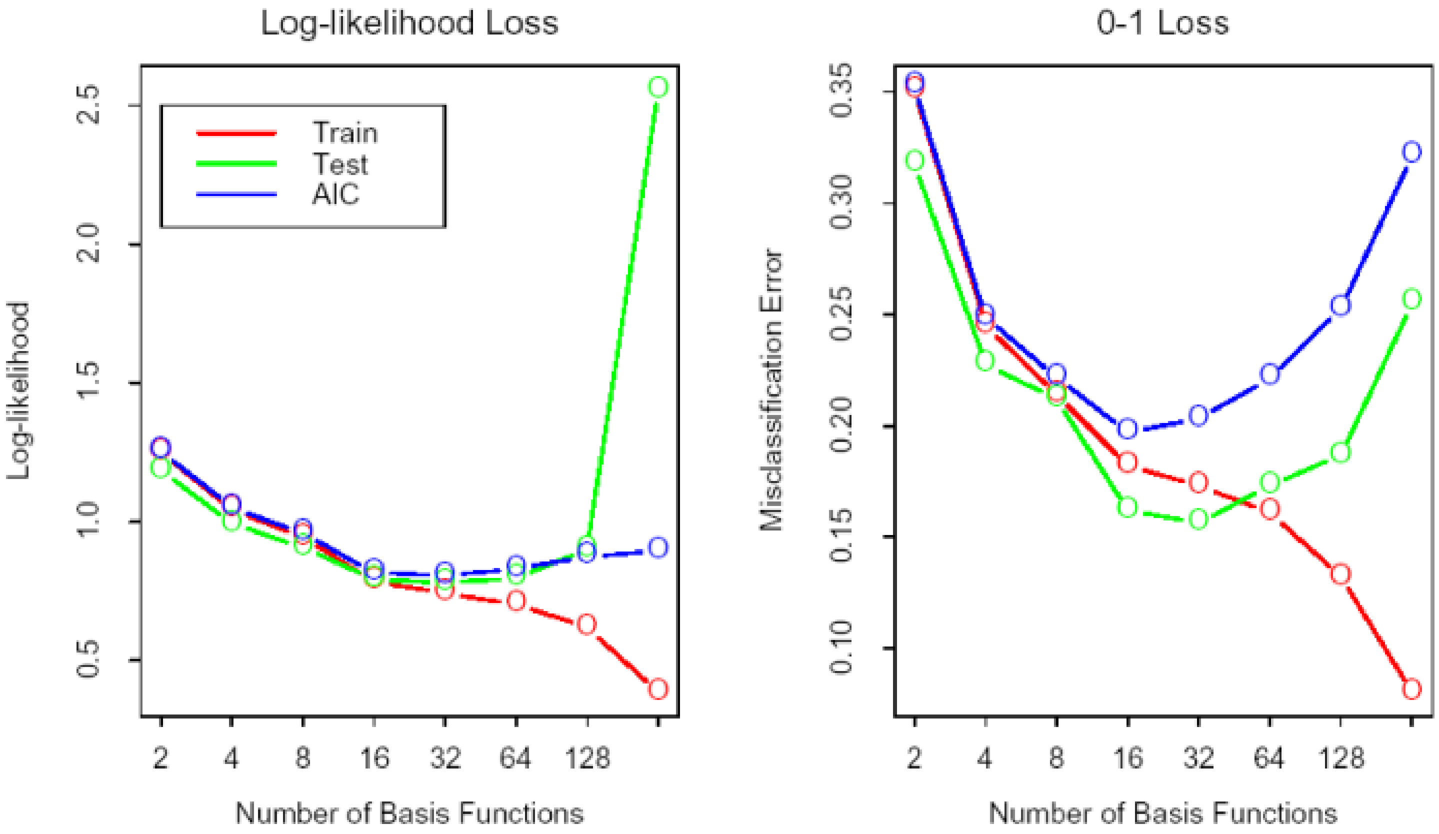
$$AIC/N = 2\frac{1}{2\sigma^2}\frac{1}{N}\sum_{i=1}^{N}(y_i - f(\mathbf{x}_i, \mathbf{w}))^2 + 2\frac{M}{N} = \frac{1}{\sigma^2}C_P$$

- The expression

$$\frac{AIC}{2N} = \left(\text{cost}^l_{\text{train}}[\widehat{\mathbf{w}}|\text{train}, \mathcal{M}] + \frac{M}{N}\right)$$

estimates the log-likelihood cost on new data

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


Log-likelihood Loss

0-1 Loss

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 $E_{\text{train}}(\widehat{\mu}))$ and we add and subtract μ (true parameter). Then,

$$E_{\text{train}}E_x(\widehat{\mu}-x)^2 = E_{\text{train}}E_x\left[(\widehat{\mu} - E_{\text{train}}(\widehat{\mu})) + (E_{\text{train}}(\widehat{\mu}) - \mu) + (\mu - x)\right]^2$$

- One gets

$$E_{\text{train}}E_x(\widehat{\mu} - x)^2 = \text{Bias}^2 + \text{Var} + \text{Rest}$$

$$\text{Rest} = E_x(x - \mu)^2$$

$$\text{Bias} = E_{\text{train}}(\widehat{\mu}) - \mu$$

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

Proof: Bias-Variance Decomposition (cont'd)

$$E_{\text{train}}E_x(\widehat{\mu}-x)^2 = E_{\text{train}}E_x\left[(\widehat{\mu}-E_{\text{train}}(\widehat{\mu}))+(E_{\text{train}}(\widehat{\mu})-\mu)+(\mu-x)\right]^2$$

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

$$E_{\text{train}}E_x[(\widehat{\mu}-E_{\text{train}}(\widehat{\mu}))(E_{\text{train}}(\widehat{\mu})-\mu)] = E_{\text{train}}[(\widehat{\mu}-E_{\text{train}}(\widehat{\mu}))(E_{\text{train}}(\widehat{\mu})-\mu)]$$

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

$$E_{\text{train}}E_x[(\widehat{\mu}-E_{\text{train}}(\widehat{\mu}))(\mu-x)] = E_{\text{train}}[\widehat{\mu}-E_{\text{train}}(\widehat{\mu})]E_x[\mu-x] = 0\times 0 = 0$$

$$E_{\text{train}}E_x[(E_{\text{train}}(\widehat{\mu})-\mu)(\mu-x)] = E_{\text{train}}[E_{\text{train}}(\widehat{\mu})-\mu]E_x[\mu-x] = \text{Bias}\times 0 = 0$$

Adding Variances

- When we generalize to several variables (or even functions) it might seem odd that we can simply add the variances. But note the difference
- For the the variance of the sum we have

$$Var(x + y) = Var(x) + Var(y) - 2Cov(x, y)$$

- For the sum of the variances we have (using $E(a + b) = E(a) + E(b)$)

$$E((x - E(x))^2 + (y - E(y))^2) = E(x - E(y))^2 + E(y - E(y))^2$$

$$= Var(x) + Var(y)$$

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

- Recall from a previous lecture that $P(\mathbf{w}|D,\mathcal{M})$ is a Gaussian with mean $\mathbf{w}_{MAP}$ and

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

- Thus at $\mathbf{w}_{MAP}$ the exponent is zero and we are left with

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

$$= -\frac{M}{2}\log(2\pi) - \frac{1}{2}\log\det\text{cov}(\mathbf{w}|D,\mathcal{M})$$

- Thus,

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

$$+\frac{M}{2}\log(2\pi) + \frac{1}{2}\log\det\text{cov}(\mathbf{w}|D,\mathcal{M})$$

- For large N, one can approximate

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

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

Thus

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

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

- If we consider models with different number of parameters M, then $\log P(D|w_{MAP}) + \log P(w_{MAP})$ might produce a larger value (better fit) for the model with the larger M. But for the larger model, we subtract a larger $M \log N$, so we obtain a compromise between both terms at the optimum

Laplace Approximation of the Marginal Likelihood

- By simplifying the previous equation and using the ML (Maximum Likelihood) estimate instead of the MAP estimate one obtains

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

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

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

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

Bayesian Information Criterion (BIC)

- We get

$$\frac{BIC}{2N} = \text{cost}^{l}_{\text{train}}[\hat{\mathbf{w}}|\text{train}, \mathcal{M}] + \frac{1}{2}\frac{M}{N}\log N$$

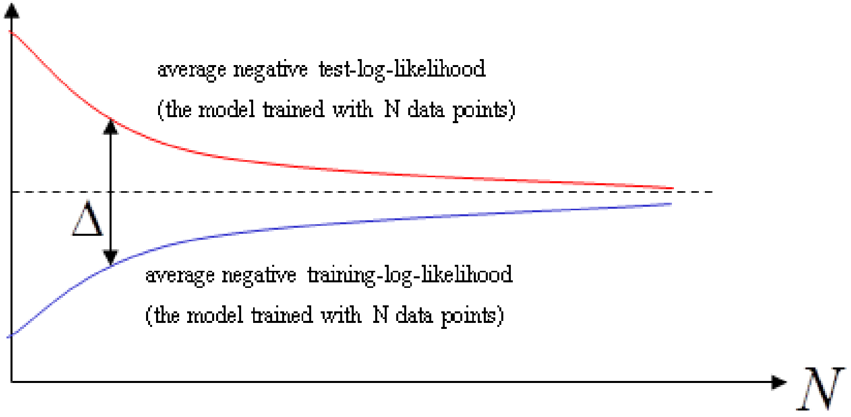
  Compare

$$\frac{AIC}{2N} = \text{cost}^{l}_{\text{train}}[\hat{\mathbf{w}}|\text{train}, \mathcal{M}] + \frac{M}{N}$$

- $\frac{1}{2}\frac{M}{N}\log N$ is an estimate of the difference between the mean test likelihood and the mean 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

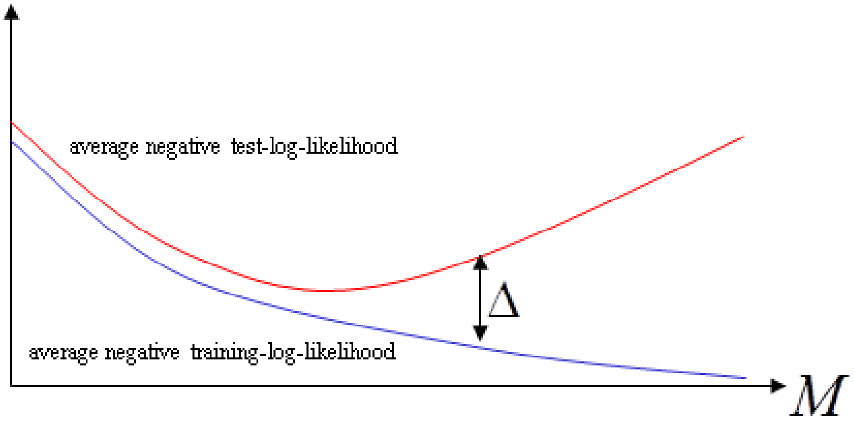
Comparison: AIC and BIC



Schätzung von D:

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

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



With the number of data points N, Δ decreases, qith increasing complexity M,

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*) Learning (Leslie Valiant)

Function Classes

- Data is generated according to some distribution $P(\mathbf{x}, y)$. This distribution is fixed but otherwise arbitrary.

- In SLT theory one considers functions $f(x)$ out of a class of measurable functions $\mathcal{M}$. Example: $\mathcal{M}$ is the class of all linear classifiers with M parameters and $f(x)$ is one of those. We will write again $f(\mathbf{x}, \mathbf{w}, \mathcal{M})$

- SLT does not assume that the best possible function $f_{true}(x)$ is contained in $\mathcal{M}$

SLT Bounds

- Let's assume a random sample of $P(\mathbf{x}, y)$, i.e. train
- SLT considers the difference between $\text{cost}^m_{P(\mathbf{x},y)}[\mathbf{w}, \mathcal{M}]$ and $\text{cost}^m_{\text{train}}[\mathbf{w}, \mathcal{M}]$, for any $\mathbf{w}$
- We now consider binary classification without noise (i.e., classes are in principal separable)
- Consider the function, for which this difference is maximum for a given train and then average over all train of size N

SLT Bounds (cont'd)

- SLT shows that

$$P_{\text{train}}\left(\sup_{f\in\mathcal{M}}\left|\text{cost}^{m}_{P(\mathbf{x},y)}[\mathbf{w},\mathcal{M}]-\text{cost}^{m}_{\text{train}}[\mathbf{w},\mathcal{M}]\right|>\epsilon\right)\leq bound$$

- Model selection is performed on $\text{cost}^{m}_{\text{train}}(f(\widehat{w}|\text{train}))+bound$

- Different bounds have been proven and they depend on the so-called VC-dimension of the model class. The VC-dimension can be infinite. For linear classifiers $dim_{VC}=M$, 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 $N>dim_{VC}$.

SLT Discussion

- The only assumption is that $P(\mathbf{x}, y)$ is fixed, but it can be arbitrarily complex. In particular, the optimal function does not need to be in the class $\mathcal{M}$ of functions under consideration
- The bounds are often very conservative and the observed generalization costs are often much smaller than the obserbed ones

Conclusion

- Machine learning focusses on generalization costs and traditionally not as much on parameter estimation (exceptions: Bioinformatics, Biomedicine)

- Empirical model selection is most often used. In each publication, test set performance needs to be reported

- Frequentist approaches typically estimate $E_{\text{train}}\text{cost}^{m}_{P(\mathbf{x},y)}[\widehat{\mathbf{w}}|\text{train}, \mathcal{M}]$. 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 optimal function does not need to be included in the class of observed functions

- 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