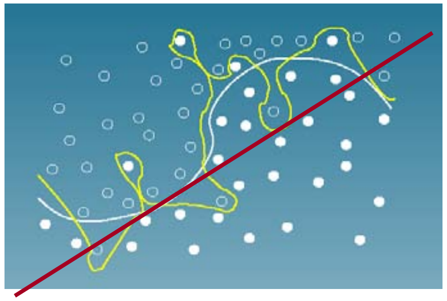
Machine Learning

Overfitting and Model Selection

Eric Xing



Lecture 6, August 13, 2010

Reading:

Outline

- Overfitting
  - kNN
  - Regression

- Bias-variance decomposition
- Generalization Theory and Structural Risk Minimization

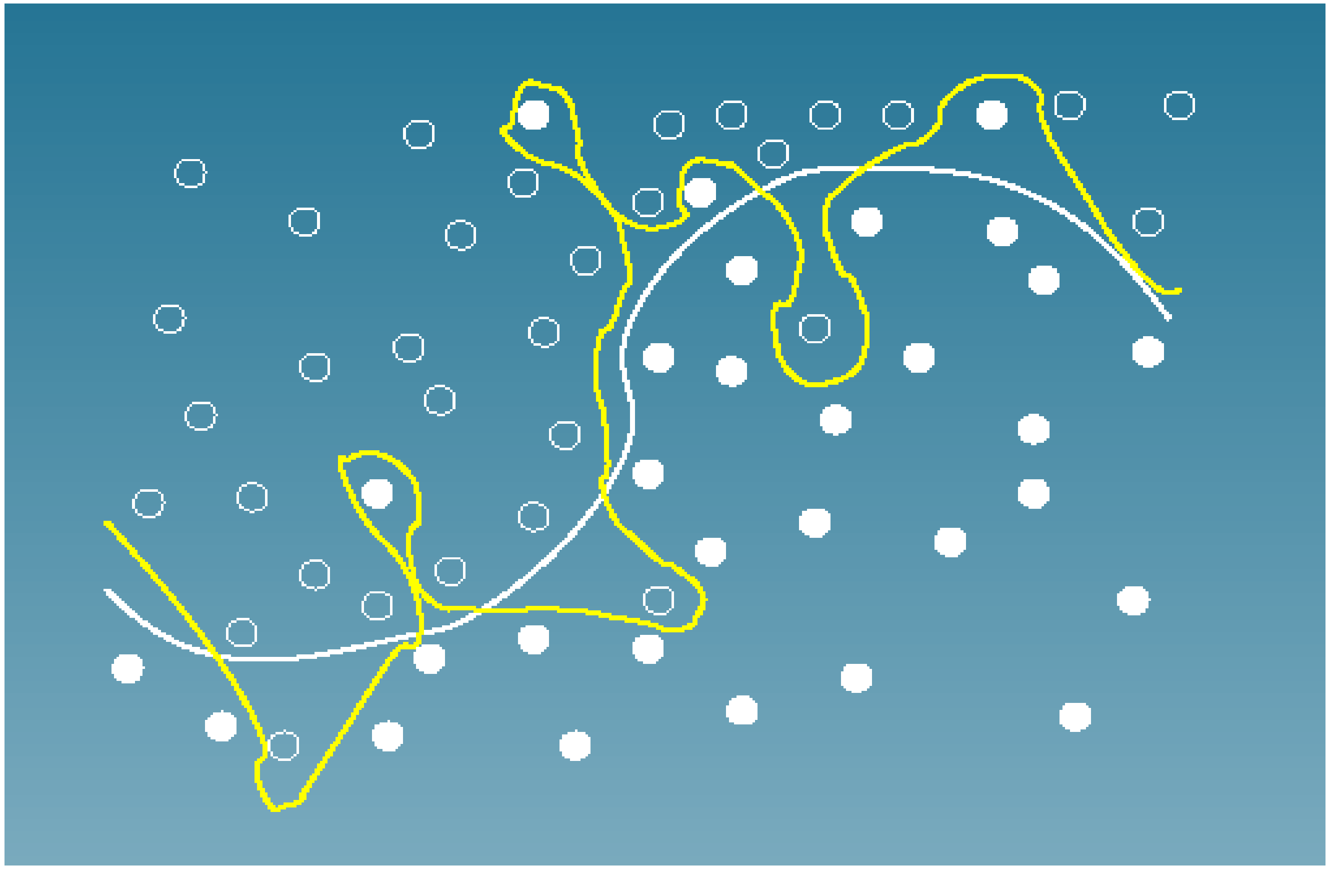
- The battle against overfitting:

  each learning algorithm has some "free knobs" that one can "tune" (i.e., heck) to make the algorithm generalizes better to test data.

  But is there a more principled way?

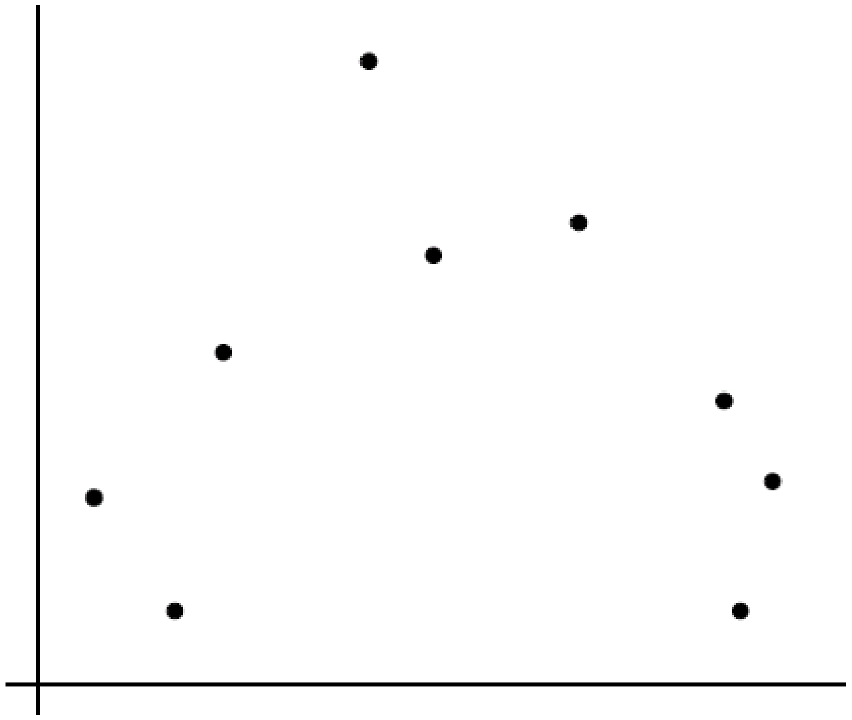
  - Cross validation
  - Regularization
  - Feature selection
  - Model selection --- Occam's razor
  - Model averaging

Overfitting: kNN



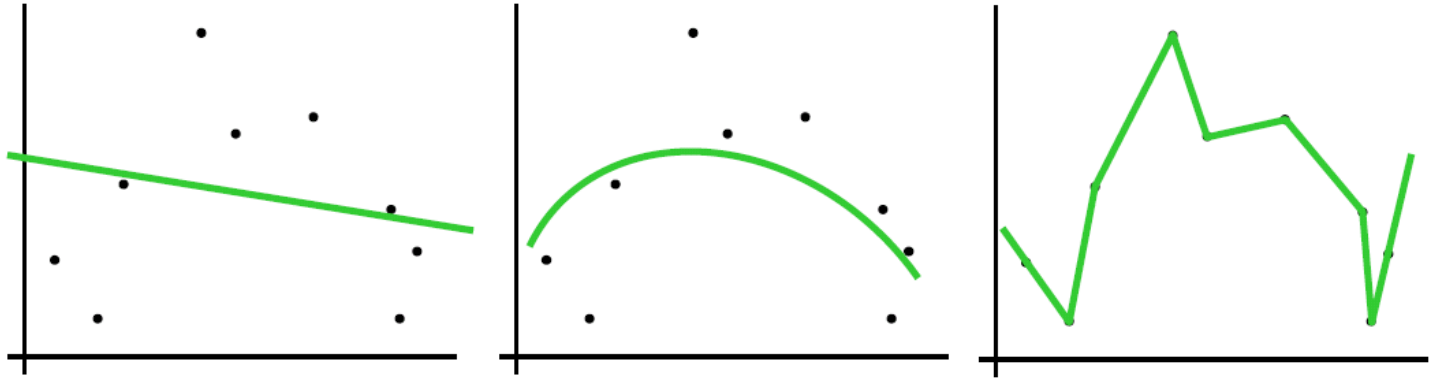
Another example:

- Regression

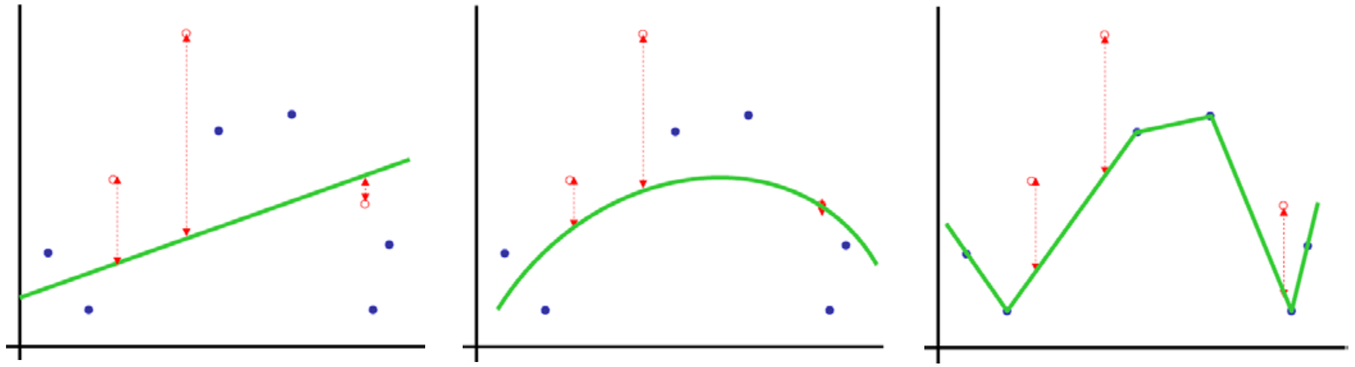


Overfitting, con'd

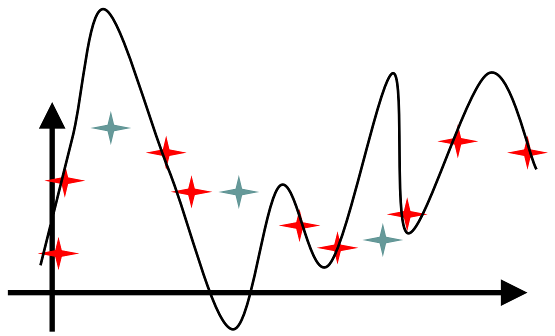
- The models:



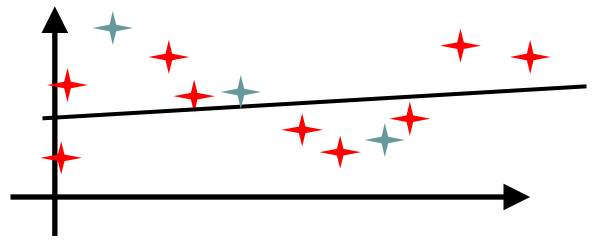
- Test errors:



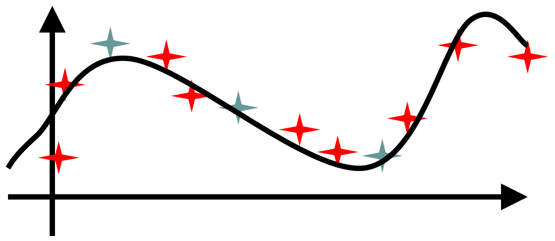
What is a good model?



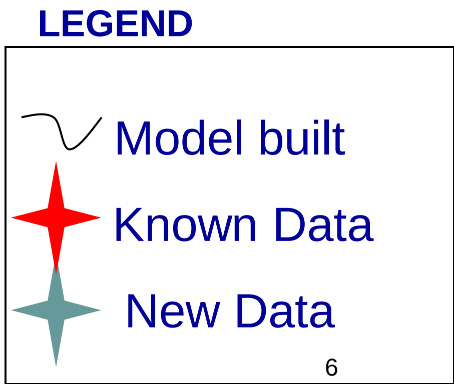
Low Robustness



Low quality /High Robustness

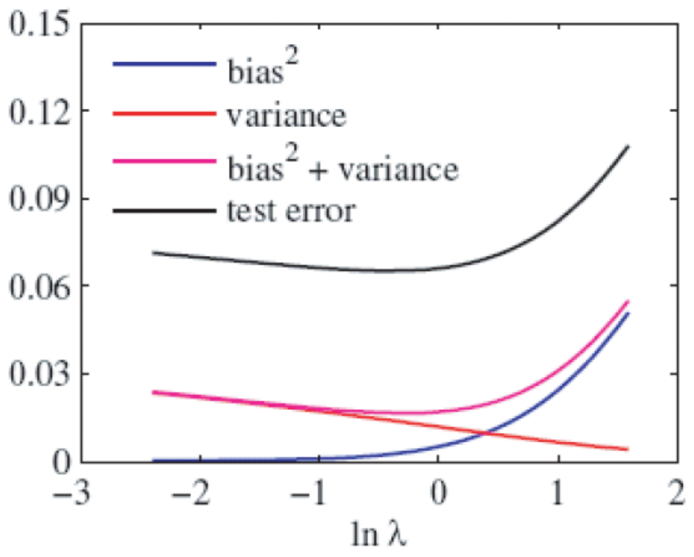


Robust Model



Bias-variance decomposition

- Now let's look more closely into two sources of errors in an functional approximator:



- Let $h(x) = E[t|x]$ be the **optimal** predictor, and $y(x)$ our actual predictor:

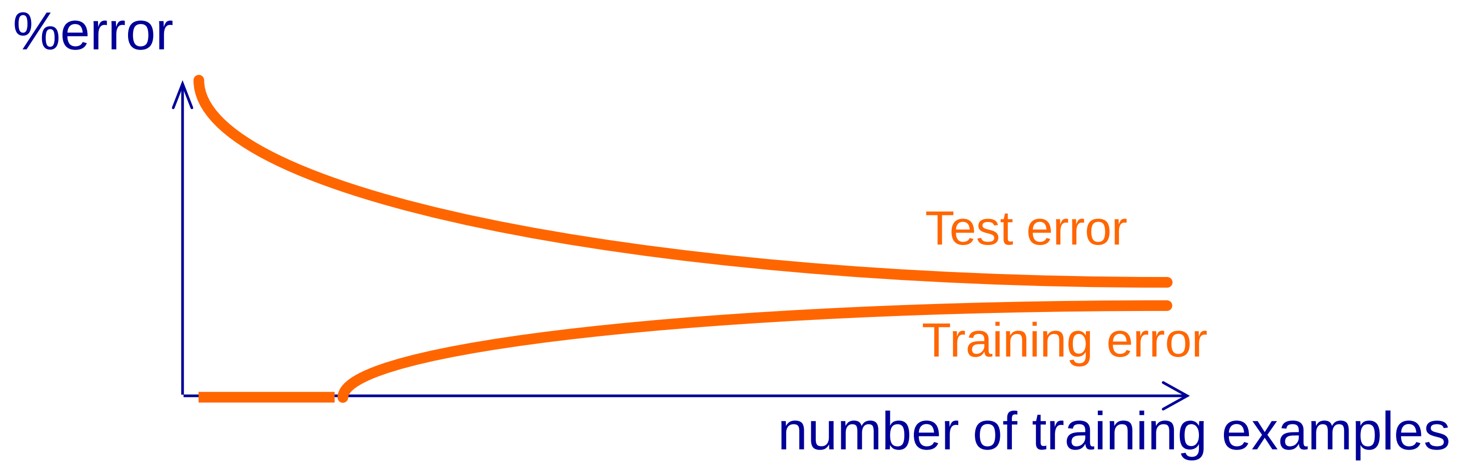
$$E_D\left[\left(y(x;D)-h(x)\right)^2\right]=\left(E_D\left[y(x;D)\right]-h(x)\right)^2+E_D\left[\left(y(x;D)-E_D\left[y(x;D)\right]\right)^2\right]$$

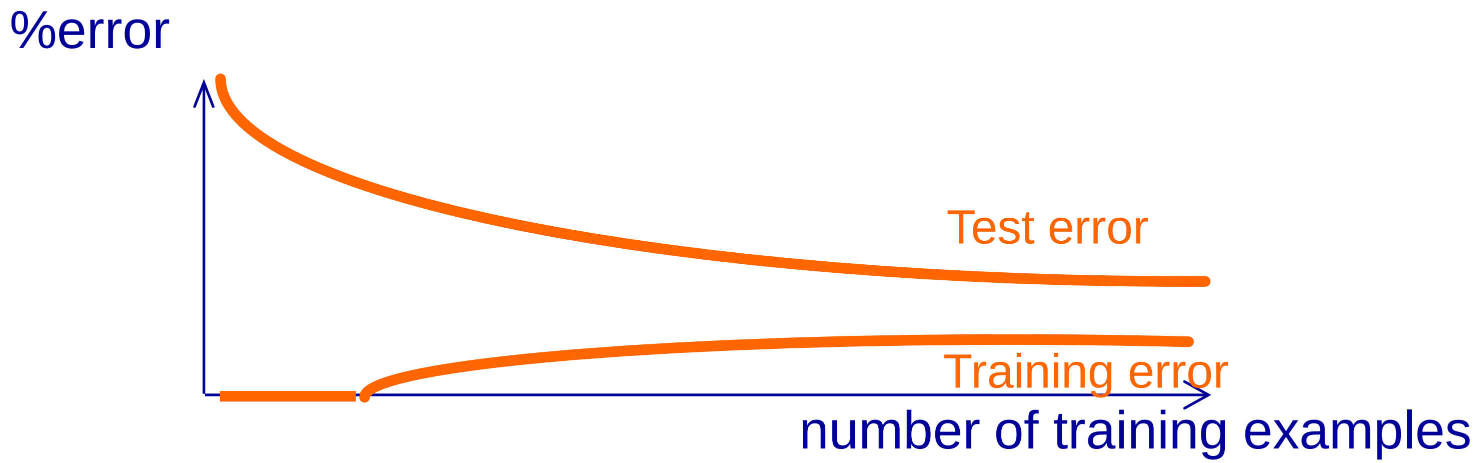
  - expected loss = (bias)^2 + variance + noise

Four Pillars for SLT

- Consistency (guarantees generalization)
  - Under what conditions will a model be consistent ?
- Model convergence speed (a measure for generalization)
  - How does generalization capacity improve when sample size L grows?
- Generalization capacity control
  - How to control in an efficient way model generalization starting with the only given information we have: our sample data?
- A strategy for good learning algorithms
  - Is there a strategy that guarantees, measures and controls our learning model generalization capacity ?

Consistent training?





Vapnik main theorem

- Q : Under which conditions will a learning model be consistent?
- A : A model will be consistent if and only if the function *h* that defines the model comes from a family of functions *H* with finite VC dimension *d*

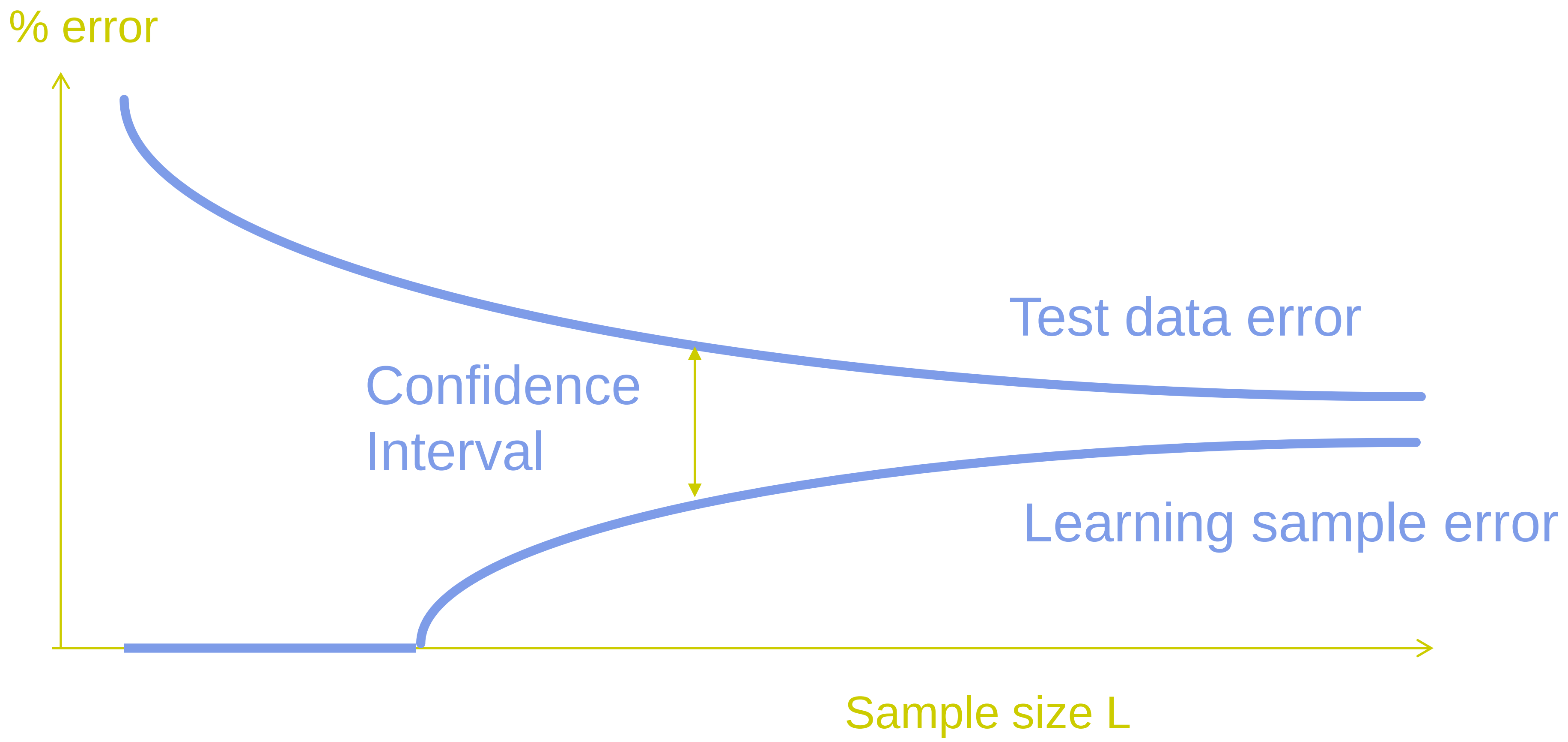
- A finite VC dimension d not only guarantees a generalization capacity (consistency), but to pick *h* in a family *H* with finite VC dimension *d* is the only way to build a model that generalizes.

Model convergence speed (generalization capacity)

- Q : What is the nature of model error difference between learning data (sample) and test data, for a sample of finite size *m*?
- A : This difference is no greater than a limit that only depends on the ratio between VC dimension *d* of model functions family *H*, and sample size *m*, i.e., *d/m*

This statement is a new theorem that belongs to Kolmogorov-Smirnov way for results, i.e., theorems that do not depend on data's underlying probability law.

Model convergence speed



How to control model generalization capacity

Risk Expectation = Empirical Risk + Confidence Interval

- To minimize Empirical Risk alone will not always give a good generalization capacity: one will want to minimize the sum of Empirical Risk and Confidence Interval

- What is important is not the numerical value of the Vapnik limit, most often too large to be of any practical use, it is the fact that this limit is a non decreasing function of model family function “richness”

Empirical Risk Minimization

- With probability $1-\delta$, the following inequality is true:

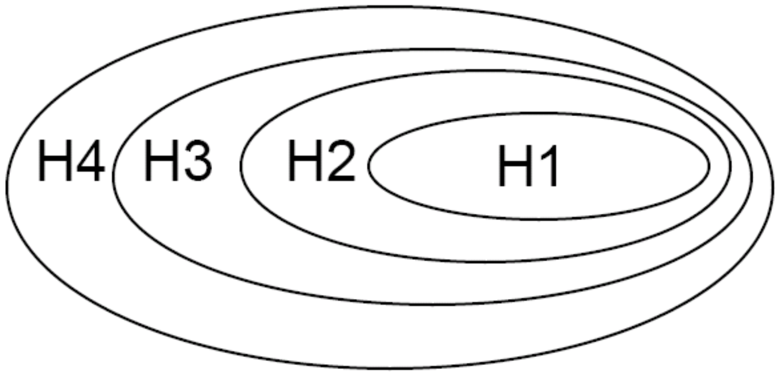
$$\int \left(y - f(x, w^0)\right)^2 dP(x, y) < \frac{1}{m}\sum_{i=1}^{m}\left(y_i - f(x_i, w^0)\right)^2 + \sqrt{\frac{d\left(\ln(2m/d)+1\right) - \ln\delta}{m}}$$

- where w^0 is the parameter w value that minimizes Empirical Risk:

$$E(W) = \frac{1}{m}\sum_{i=1}^{m}\left(y_i - f(x_i, w)\right)^2$$

Structural Risk Minimization

- Which hypothesis space should we choose?
- Bias / variance tradeoff



- SRM: choose H to minimize bound on true error!

$$\epsilon(h) \leq \hat{\epsilon}(h) + O\left(\sqrt{\frac{d}{m}\log\frac{m}{d} - \frac{1}{m}\log\delta}\right)$$

unfortunately a somewhat loose bound...

SRM strategy (1)

- With probability $1\text{-}\delta$,

$$\epsilon(h) \leq \hat{\epsilon}(h) + O\left(\sqrt{\frac{d}{m}\log\frac{m}{d} - \frac{1}{m}\log\delta}\right)$$

- When m/d is small (d too large), second term of equation becomes large

- SRM basic idea for strategy is to minimize simultaneously both terms standing on the right of above majoring equation for $\varepsilon(h)$

- To do this, one has to make *d* a controlled parameter

SRM strategy (2)

- Let us consider a sequence $H_1 < H_2 < .. < H_n$ of model family functions, with respective growing VC dimensions

  $$d_1 < d_2 < .. < d_n$$

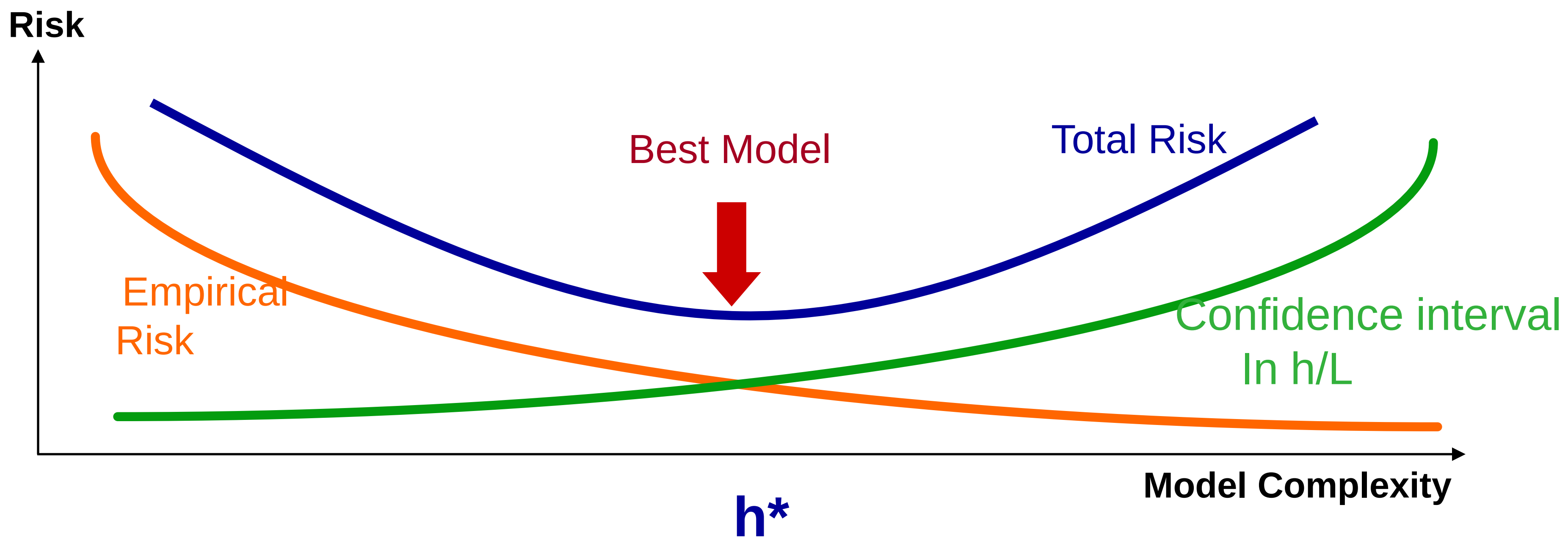
- For each family H_i of our sequence, the inequality

  $$\epsilon(h) \leq \hat{\epsilon}(h) + O\left(\sqrt{\frac{d}{m}\log\frac{m}{d} - \frac{1}{m}\log\delta}\right)$$

  is valid
  - That is, for each subset, we must be able either to compute *d*, or to get a bound on *d* itself.
- SRM then consists of finding that subset of functions which minimizes the bound on the actual risk.

SRM strategy (3)

SRM : find i such that expected risk $\varepsilon(h)$ becomes minimum, for a specific $d^*=d_i$, relating to a specific family H_i of our sequence; build model using h from H_i



Putting SRM into action: linear models case (1)

- There are many SRM-based strategies to build models:

- In the case of linear models

$$y = w^T x + b,$$

one wants to make ||w|| a controlled parameter: let us call H_C the linear model function family satisfying the constraint:

$$||w|| < C$$

Vapnik Major theorem:

When C decreases, $d(H_C)$ decreases

$$||x|| < R$$

Putting SRM into action: linear models case (2)

- To control ||w||, one can envision two routes to model:
  - *Regularization/Ridge Regression*, ***ie min. over w and b***

    $RG(w,b) = S\{(y_i - <w|x_i> - b)^2 \mid i=1,..,L\} + \lambda \, ||w||^2$

  - *Support Vector Machines (SVM)*, ***ie solve directly an optimization problem (classif. SVM, separable data)***

    Minimize $||w||^2$,

    with (y_i= +/-1)

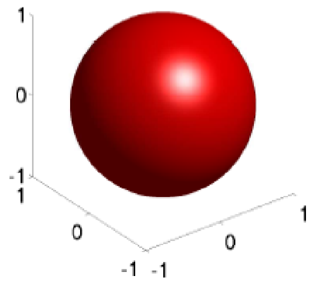
    and $y_i(<w|x_i> + b) >= 1$ for all i=1,..,L

Regularized Regression

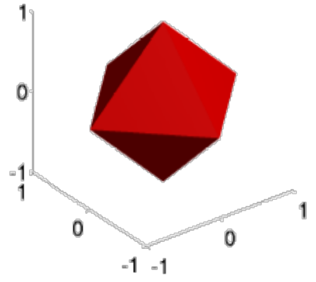
- Recall linear regression: $\mathbf{y} = \mathbf{X}^T\theta + \epsilon$

$$\begin{aligned} \theta^* &= \arg\max_\theta (\mathbf{y} - \mathbf{X}^T\theta)^T(\mathbf{y} - \mathbf{X}^T\theta) \\ &= \arg\max_\theta ||\mathbf{y} - \mathbf{X}^T\theta||^2 \end{aligned}$$

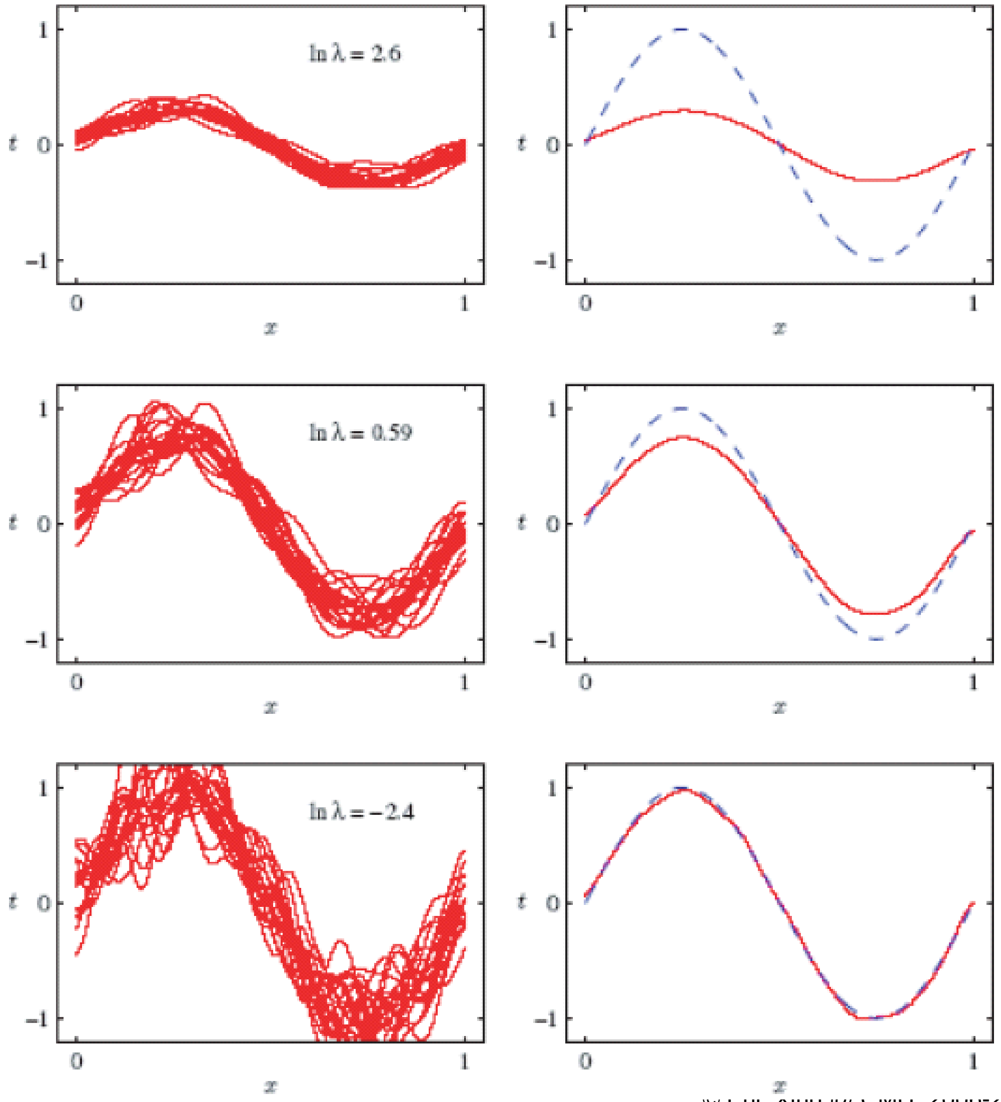
- Regularized LR:
  - L2-regularized LR: $\theta^* = \arg\max_\theta ||\mathbf{y} - \mathbf{X}^T\theta||^2 + \lambda||\theta||$



  - L1-regularized LR: $\theta^* = \arg\max_\theta ||\mathbf{y} - \mathbf{X}^T\theta||^2 + \lambda|\theta|$

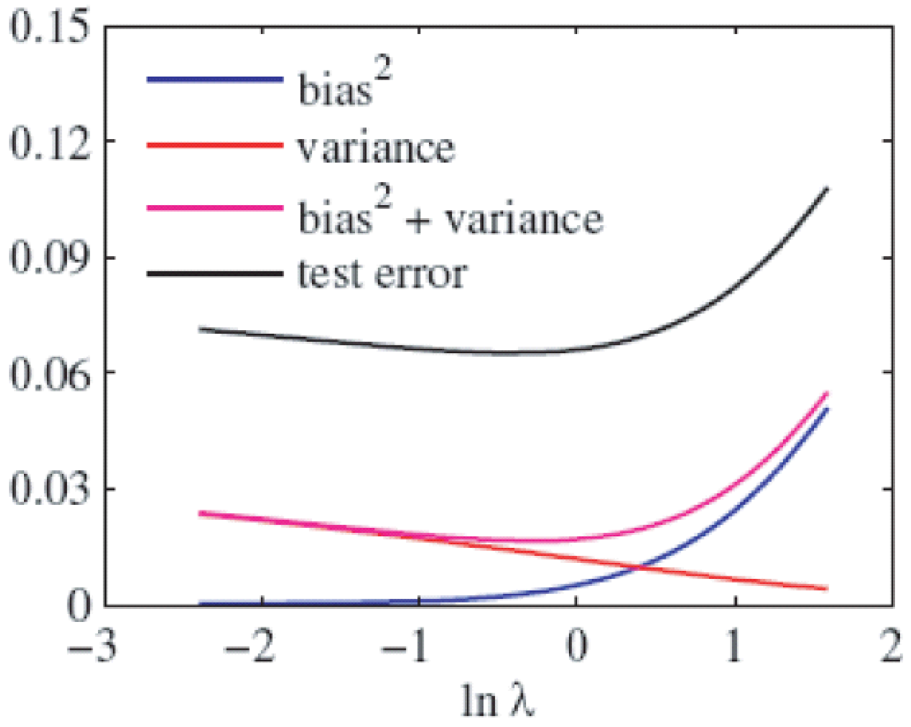


Bias-variance tradeoff



- λ is a "regularization" terms in LR, the smaller the λ, is more complex the model (why?)
  - Simple (highly regularized) models have low variance but high bias.
  - Complex models have low bias but high variance.
- You are inspecting an empirical average over 100 training set.
- The actual E_D can not be computed

Bias^2+variance vs regularizer



- Bias^2+variance predicts (shape of) test error quite well.
- However, bias and variance cannot be computed since it relies on knowing the true distribution of x and t (and hence $h(x) = E[t|x]$).

The battle against overfitting

Model Selection

- Suppose we are trying select among several different models for a learning problem.
- Examples:

  1. polynomial regression

  $$h(x;\theta) = g(\theta_0 + \theta_1 x + \theta_2 x^2 + \ldots + \theta_k x^k)$$

  - Model selection: we wish to **automatically** and **objectively** decide if k should be, say, 0, 1, . . . , or 10.

  2. locally weighted regression,

  - Model selection: we want to automatically choose the bandwidth parameter τ.

  3. Mixture models and hidden Markov model,

  - Model selection: we want to decide the number of hidden states

- The Problem:
  - Given model family $\mathscr{F} = \{M_1, M_2, \ldots, M_I\}$, find $M_i \in \mathscr{F}$ s.t.

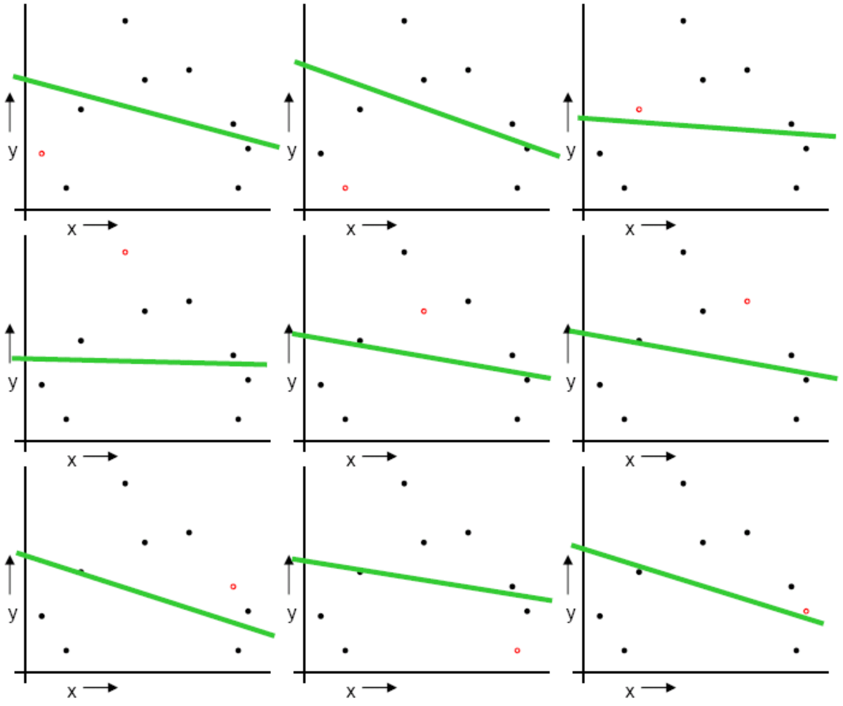
  $$M_i = \arg\max_{M \in \mathscr{F}} J(D, M)$$

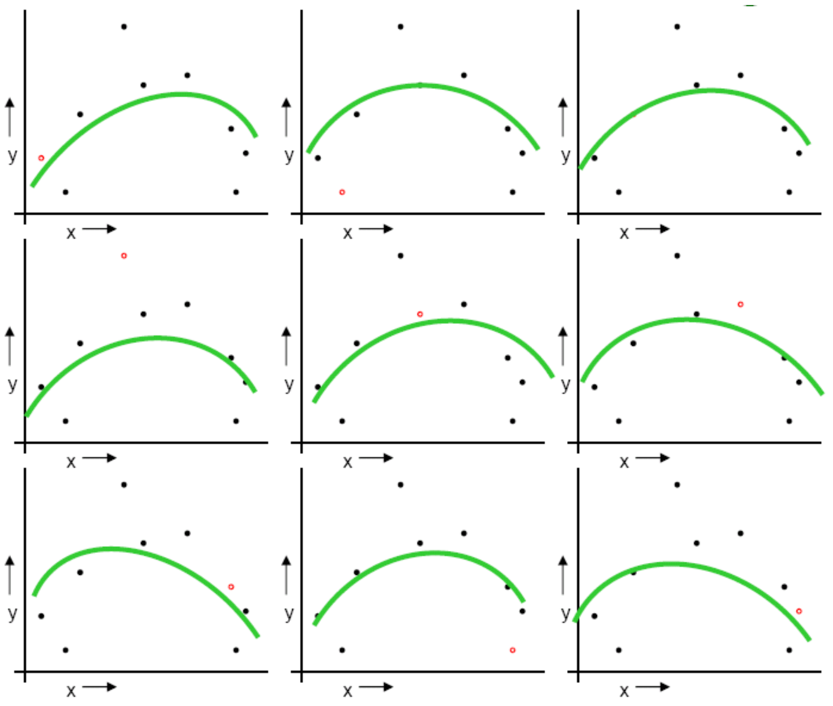
1. Cross Validation

- We are given training data D and test data D_{test}, and we would like to fit this data with a model $p_i(x;\theta)$ from the family $\mathcal{F}$ (e.g, an LR), which is indexed by i and parameterized by θ.
- K-fold cross-validation (CV)
  - Set aside αN samples of D (where $N = |D|$). This is known as the held-out data and will be used to evaluate different values of i.
  - For each candidate model i, fit the optimal hypothesis $p_i(x;\theta^*)$ to the remaining $(1-\alpha)N$ samples in D (i.e., hold i fixed and find the best θ).
  - Evaluate each model $p_i(x|\theta*)$ on the held-out data using some pre-specified risk function.
  - Repeat the above ***K* times**, choosing a **different** held-out data set each time, and the scores are averaged for each model $p_i(.)$ over all held-out data set. This gives an estimate of the risk curve of models over different i.
  - For the model with the lowest risk, say $p_{i*}(.)$, we use all of D to find the parameter values for $p_{i*}(x;\theta^*)$.

Example:

- When $\alpha = 1/N$, the algorithm is known as Leave-One-Out-Cross-Validation (LOOCV)





MSELOOCV(M_1)=2.12

MSELOOCV(M_2)=0.962

Practical issues for CV

- How to decide the values for K and α
  - Commonly used $K = 10$ and $\alpha = 0.1$.
  - when data sets are small relative to the number of models that are being evaluated, we need to decrease α and increase K
  - K needs to be large for the variance to be small enough, but this makes it time-consuming.
- Bias-variance trade-off
  - Small α usually lead to low bias. In principle, *LOOCV* provides an almost unbiased estimate of the generalization ability of a classifier, especially when the number of the available training samples is severely limited; but it can also have high variance.
  - Large α can reduce variance, but will lead to under-use of data, and causing high-bias.
- One important point is that the test data D_{test} is never used in CV, because doing so would result in overly (indeed dishonest) optimistic accuracy rates during the testing phase.

2. Regularization

- Maximum-likelihood estimates are not always the best (James and Stein showed a counter example in the early 60's)
- Alternative: we "regularize" the likelihood objective (also known as penalized likelihood, shrinkage, smoothing, etc.), by adding to it a penalty term:

$$\hat{\theta}_{\text{shrinkage}} = \arg\max_{\theta} \left[l(\theta; D) + \lambda \|\theta\| \right]$$

  where λ>0 and $||\theta||$ might be the L_1 or L_2 norm.

- The choice of norm has an effect
  - using the L_2 norm pulls directly towards the origin,
  - while using the L1 norm pulls towards the coordinate axes, i.e it tries to set some of the coordinates to 0.
  - This second approach can be useful in a feature-selection setting.

Recall Bayesian and Frequentist

- Frequentist interpretation of probability
  - Probabilities are objective properties of the real world, and refer to limiting relative frequencies (e.g., number of times I have observed heads). Hence one cannot write *P*(Katrina could have been prevented|*D*), since the event will never repeat.
  - Parameters of models are *fixed, unknown constants*. Hence one cannot write $P(\theta|D)$ since θ does not have a probability distribution. Instead one can only write $P(D|\theta)$.
  - One computes point estimates of parameters using various *estimators*, $\theta^*= f(D)$, which are designed to have various desirable qualities when *averaged over future data D* (assumed to be drawn from the "true" distribution).
- Bayesian interpretation of probability
  - Probability describes degrees of belief, not limiting frequencies.
  - Parameters of models are *hidden variables*, so one can compute $P(\theta|D)$ or $P(f(\theta)|D)$ for some function *f*.
  - One estimates parameters by computing $P(\theta|D)$ using Bayes rule:

$$p(\theta|D) = \frac{p(D\,|\,\theta)\,p(\theta)}{p(D)}$$

Bayesian interpretation of regulation

- Regularized Linear Regression
  - Recall that using squared error as the cost function results in the LMS estimate
  - And assume iid data and Gaussian noise, LMS is equivalent to MLE of θ

$$l(\theta) = n \log \frac{1}{\sqrt{2\pi}\sigma} - \frac{1}{\sigma^2}\frac{1}{2}\sum_{i=1}^{n}(y_i - \theta^T \mathbf{x}_i)^2$$

  - Now assume that vector θ follows a normal prior with 0-mean and a diagonal covariance matrix

$$\theta \sim N(0, \tau^2 I)$$

  - What is the posterior distribution of θ?

$$p(\theta|D) \propto p(D,\theta)$$

$$= p(D|\theta)\,p(\theta) = \left(2\pi\sigma^2\right)^{-n/2} \exp\left\{-\frac{1}{2\sigma^2}\sum_{i=1}^{n}\left(y_n - \theta^T x_i\right)^2\right\} \times C \exp\left\{-(\theta^T\theta/2\tau^2\right\}$$

Bayesian interpretation of regulation, con'd

- The posterior distribution of θ

$$p(\theta|D) \propto \exp\left\{-\frac{1}{2\sigma^2}\sum_{i=1}^{n}\left(y_n - \theta^T x_i\right)^2\right\} \times \exp\left\{-\theta^T\theta/2\tau^2\right\}$$

- This leads to a now objective

$$\begin{aligned} l_{MAP}(\theta; D) &= -\frac{1}{2\sigma^2}\frac{1}{2}\sum_{i=1}^{n}(y_i - \theta^T \mathbf{x}_i)^2 - \frac{1}{\tau^2}\frac{1}{2}\sum_{k=1}^{K}\theta_k^2 \\ &= l(\theta; D) + \lambda\|\theta\| \end{aligned}$$

  - This is L_2 regularized LR! --- a MAP estimation of θ
  - What about L_1 regularized LR! (homework)

- How to choose λ.
  - cross-validation!

3. Feature Selection

- Imagine that you have a supervised learning problem where the number of features d is very large (perhaps d >>#samples), but you suspect that there is only a small number of features that are "**relevant**" to the learning task.

- VC-theory can tell you that this scenario is likely to lead to high generalization error – the learned model will potentially overfit unless the training set is fairly large.

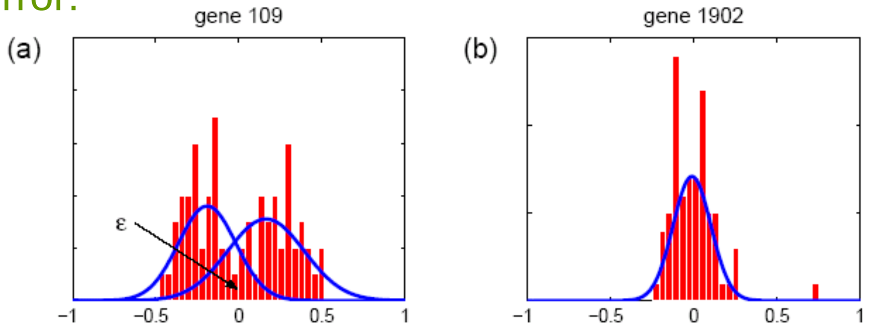
- So lets get rid of useless parameters!

How to score features

- How do you know which features can be pruned?
  - Given labeled data, we can compute some simple score $S(i)$ that measures how informative each feature x_i is about the class labels y.
  - Ranking criteria:
    - Mutual Information: score each feature by its mutual information with respect to the class labels

$$MI(x_i, y) = \sum_{x_i \in \{0,1\}} \sum_{y \in \{0,1\}} p(x_i, y) \log \frac{p(x_i, y)}{p(x_i) p(y)}$$

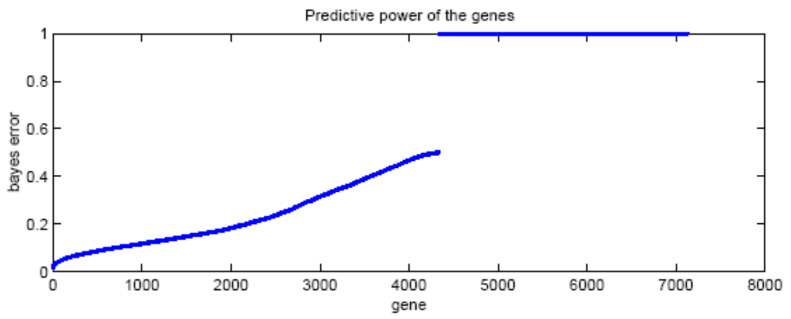
    - Bayes error:



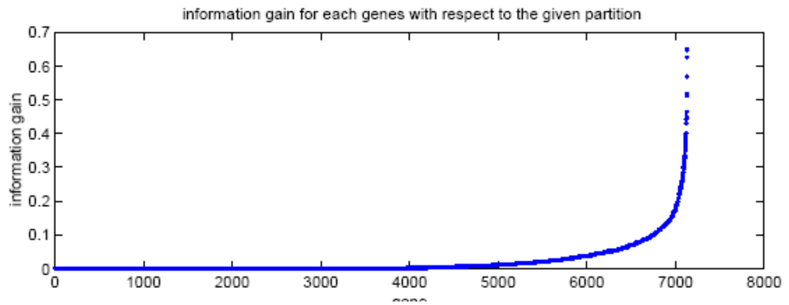
    - Redundancy (Markov-blank score) …
  - We need estimate the relevant p()'s from data, e.g., using MLE

Feature Ranking

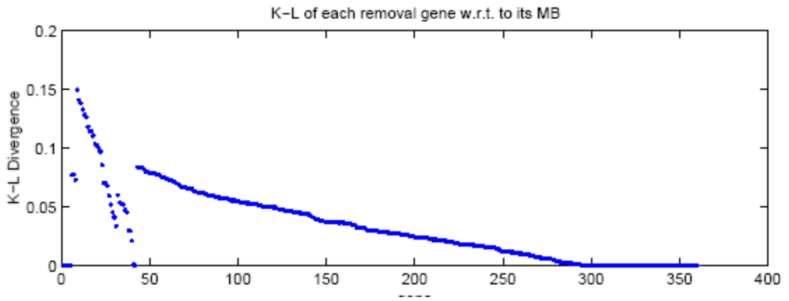
- Bayes error of each gene



- information gain for each genes with respect to the given partition



- KL of each removal gene w.r.t. to its MB

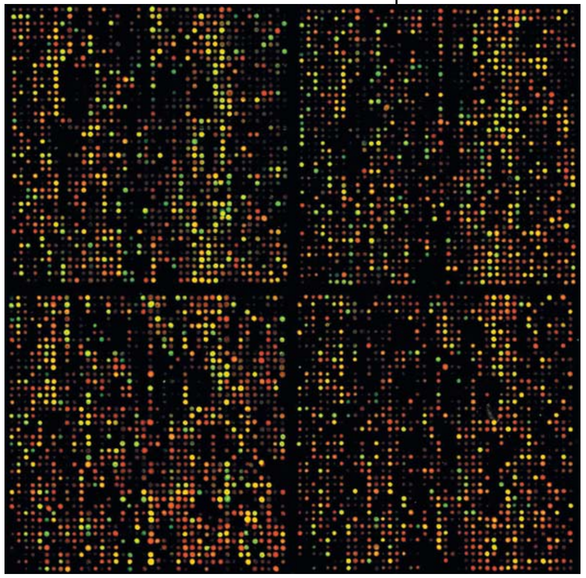


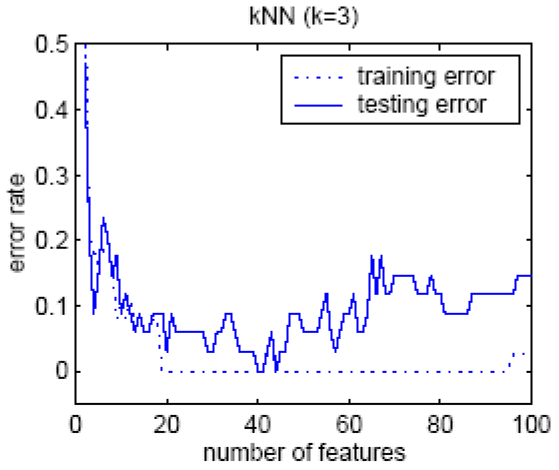
Feature selection schemes

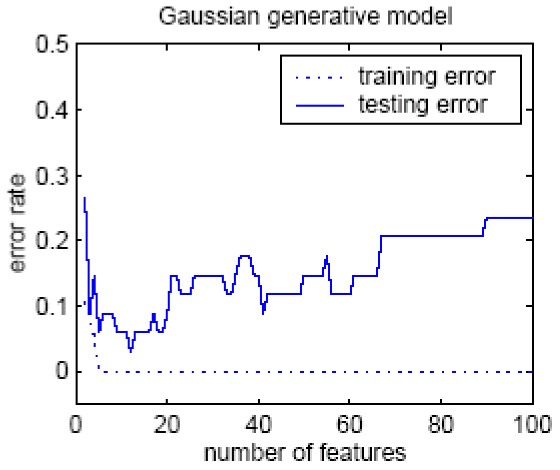
- Given n features, there are 2^n possible feature subsets (why?)
- Thus feature selection can be posed as a model selection problem over 2^n possible models.
- For large values of n, it's usually too expensive to explicitly enumerate over and compare all 2^n models. Some heuristic search procedure is used to find a good feature subset.
- Three general approaches:
  - Filter: i.e., direct feature ranking, but taking no consideration of the subsequent learning algorithm
    - add (from empty set) or remove (from the full set) features one by one based on $S(i)$
    - Cheap, but is subject to local optimality and may be unrobust under different classifiers
  - Wrapper: determine the (inclusion or removal of) features based on performance under the learning algorithms to be used. See next slide
  - Simultaneous learning and feature selection.
    - E.x. L_1 regularized LR, Bayesian feature selection (will not cover in this class), etc.

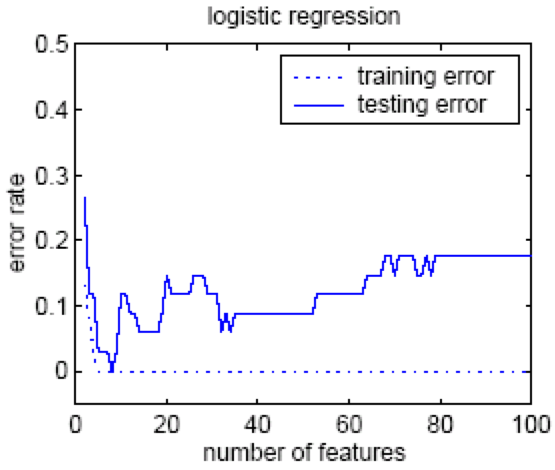
Case study [Xing et al, 2001]

- The case:
  - 7130 genes from a microarray dataset
  - 72 samples
  - 47 type I Leukemias (called ALL) and 25 type II Leukemias (called AML)
- Three classifier:
  - kNN
  - Gaussian classifier
  - Logistic regression





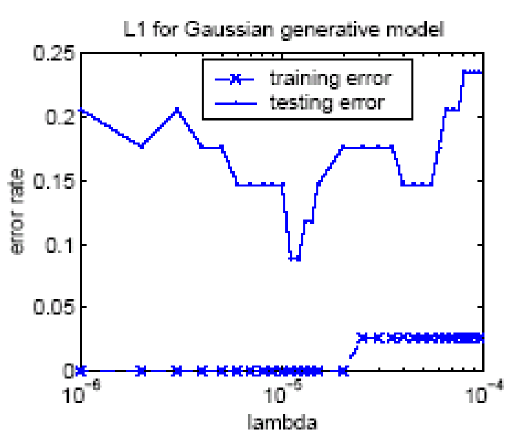


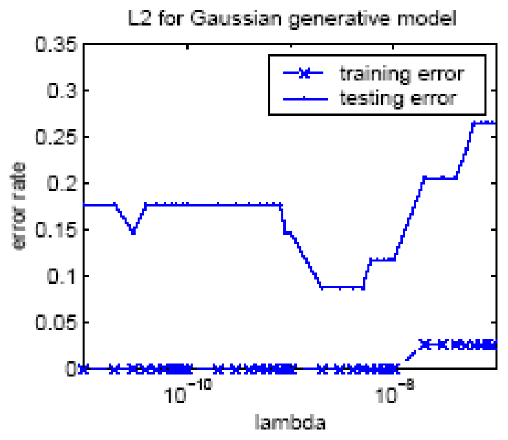


Regularization vs. Feature Selection

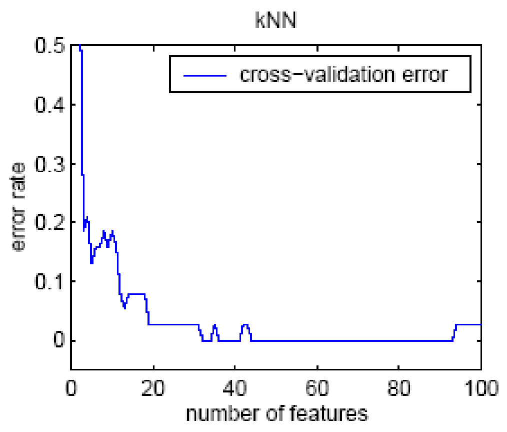
- Explicit feature selection often outperform regularization

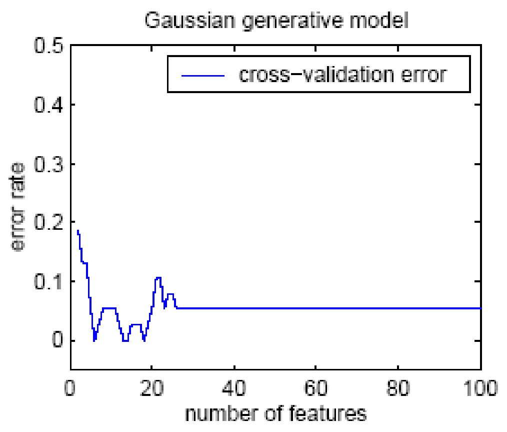
regression

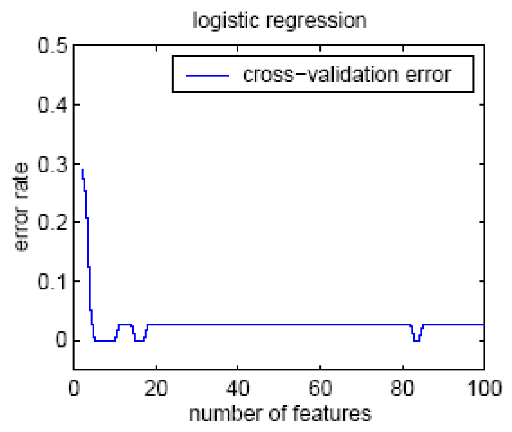




Feature Selection







4. Information criterion

- Suppose we are trying select among several different models for a learning problem.

- The Problem:
  - Given model family $\mathcal{F} = \{M_1, M_2, \ldots, M_I\}$, find $M_i \in \mathcal{F}$ s.t.

$$M_i = \arg\max_{M \in \mathcal{F}} J(D, M)$$

- We can design J that not only reflect the predictive loss, but also the amount of information M_k can hold

Model Selection via Information Criteria

- Let $f(x)$ denote the truth, the underlying distribution of the data
- Let $g(x,\theta)$ denote the model family we are evaluating
  - $f(x)$ does not necessarily reside in the model family
  - $\theta_{\mathrm{ML}}(y)$ denote the MLE of model parameter from data y
- Among early attempts to move beyond Fisher's *Maliximum Likelihood* framework, **Akaike** proposed the following information criterion:

$$E_y\left[D\left(f \,\|\, g(x \mid \theta_{ML}(y)\right)\right]$$

which is, of course, intractable (because $f(x)$ is unknown)

AIC and TIC

- AIC (An information criterion, not **Akaike** information criterion)

$$A = \log\, g(x \mid \hat{\theta}(y)) - k$$

  where k is the number of parameters in the model

- TIC (Takeuchi information criterion)

$$A = \log\, g(x \mid \hat{\theta}(y)) - \mathrm{tr}(I(\theta_0)\Sigma)$$

  where

$$\theta_0 = \arg\min\, D(f \,\|\, g(\cdot \mid \theta)) \qquad I(\theta_0) = -E_x\left[\frac{\partial^2 \log\, g(x \mid \theta)}{\partial\theta\partial\theta^T}\right]\Bigg|_{\theta=\theta_0} \qquad \Sigma = E_y\left(\hat{\theta}(y) - \theta_0\right)\left(\hat{\theta}(y) - \theta_0\right)^T$$

  - We can approximate these terms in various ways (e.g., using the bootstrap)
  - $\mathrm{tr}(I(\theta_0)\Sigma) \approx k$

5. Bayesian Model Averaging

- Recall the Bayesian Theory: (e.g., for date *D* and model *M*)

$$P(M|D) = P(D|M)P(M)/P(D)$$

  - the **posterior** equals to the **likelihood** times the **prior**, up to a constant.

- Assume that *P*(*M*) is uniform and notice that *P*(*D*) is constant, we have the following criteria:

$$P(D|M) = \int_\theta P(D|\theta, M)P(\theta|M)d\theta$$

- A few steps of approximations (you will see this in advanced ML class in later semesters) give you this:

$$P(D|M) \approx \log P(D|\hat{\theta}_{ML}) - \frac{k}{2}\log N$$

  where N is the number of data points in D.

Summary

- Structural risk minimization
- Bias-variance decomposition
- The battle against overfitting:
  - Cross validation
  - Regularization
  - Feature selection
  - Model selection --- Occam's razor
  - Model averaging
    - The Bayesian-frequentist debate
    - Bayesian learning (weight models by their posterior probabilities)