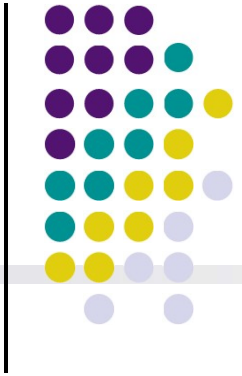
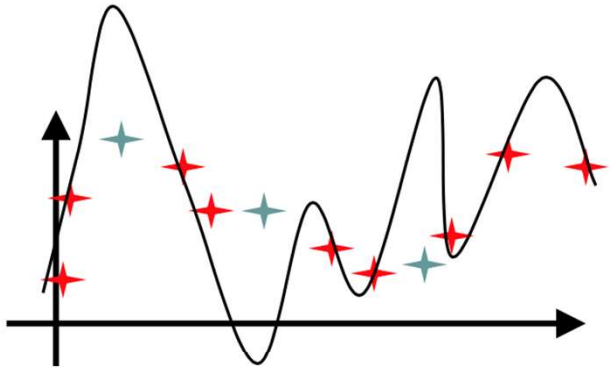
Overfitting and Model Selection

Outline

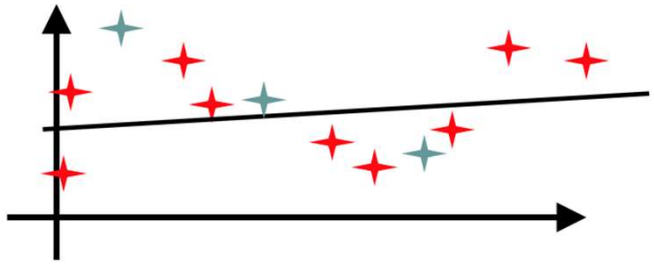


- Overfitting
- Bias-variance decomposition
- Generalization theory and structural risk minimization
- The battle against overfitting

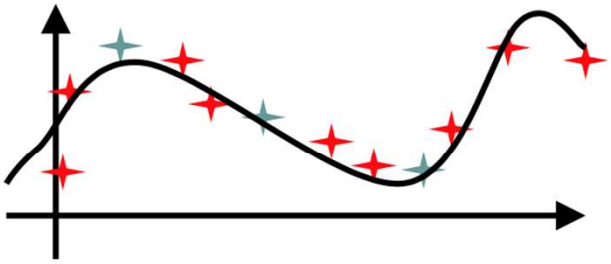
What Is a Good Model ?



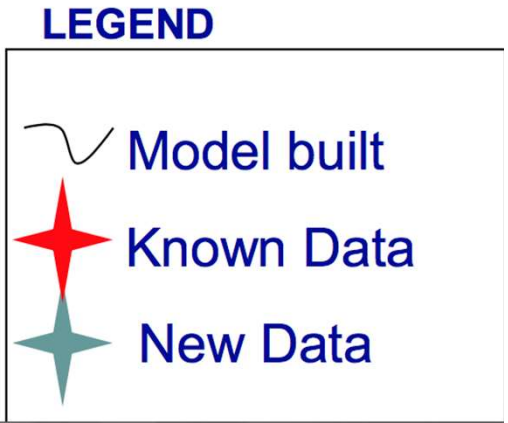
Low Robustness



Low quality /High Robustness

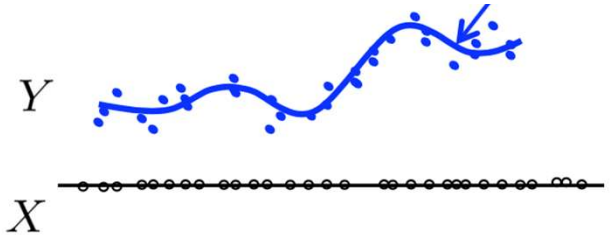


Robust Model



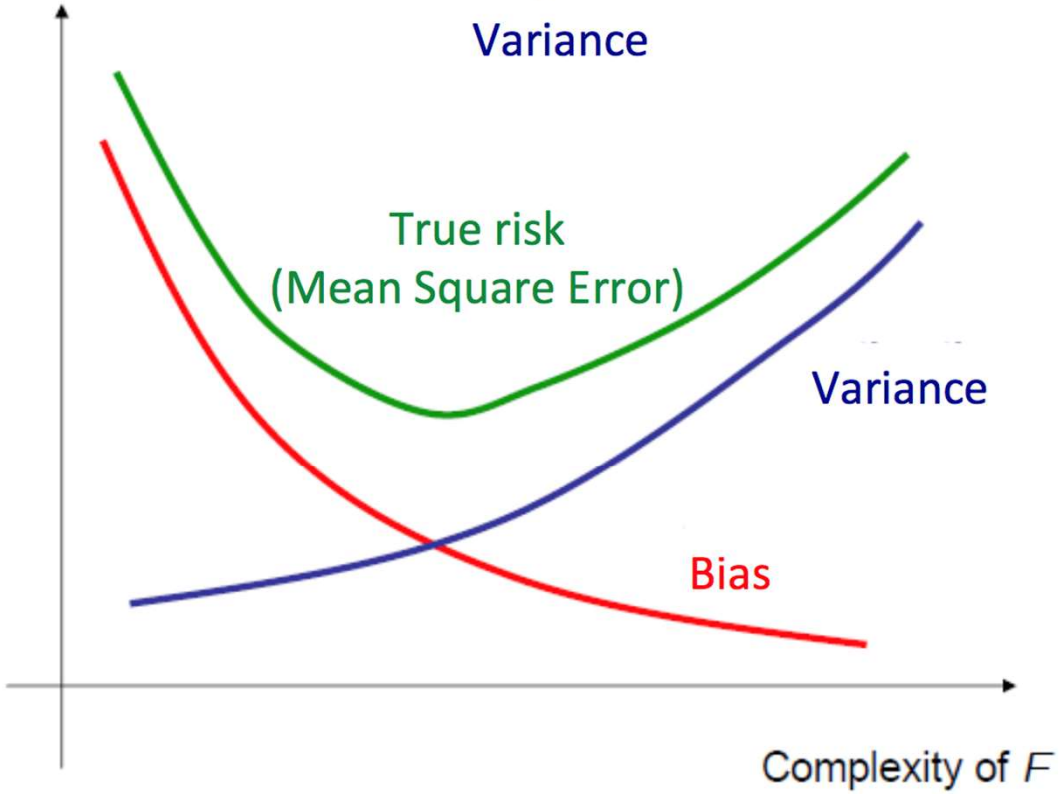
Bias-variance Decomposition

Regression $Y = f^*(X) + \epsilon \qquad \epsilon \sim \mathcal{N}(0, \sigma^2)$

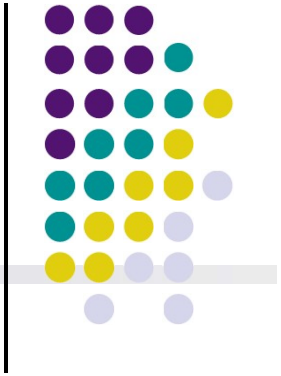


True Risk

$$R(f) = \mathbb{E}[(f(X) - Y)^2] = \underbrace{\mathbb{E}[(f(X) - \mathbb{E}[f(X)])^2]}_{\text{Variance}} + \underbrace{\mathbb{E}[(\mathbb{E}[f(X)] - f^*(X))^2]}_{\text{Bias}} + \underbrace{\sigma^2}_{\text{Bayes error} = R(f^*)}$$

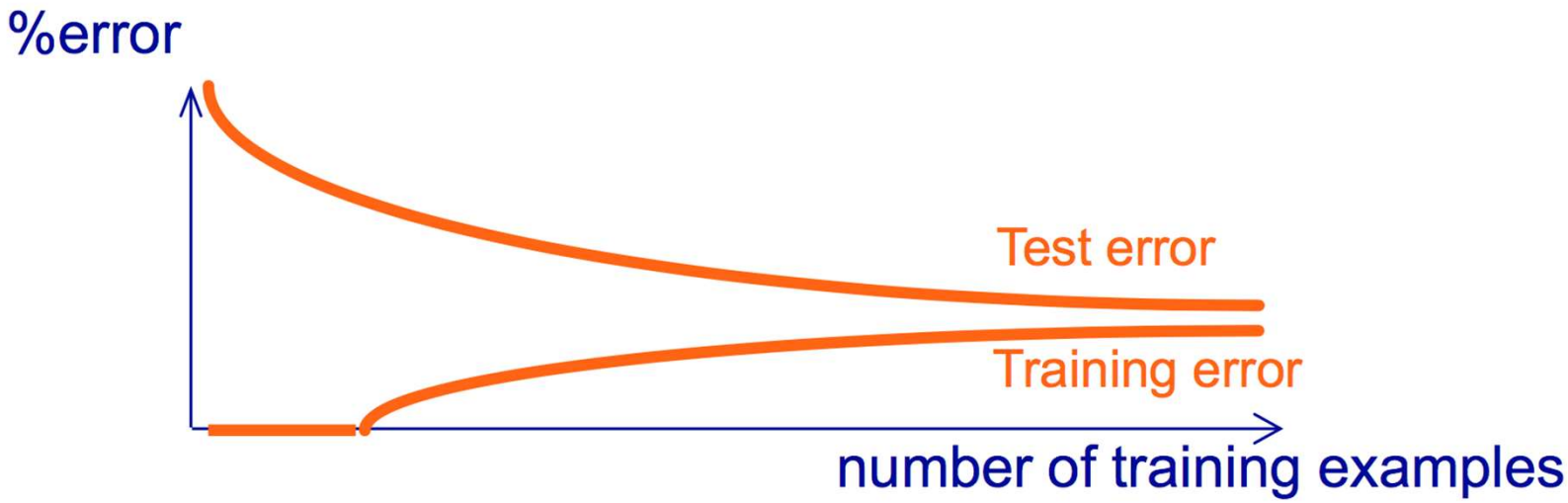


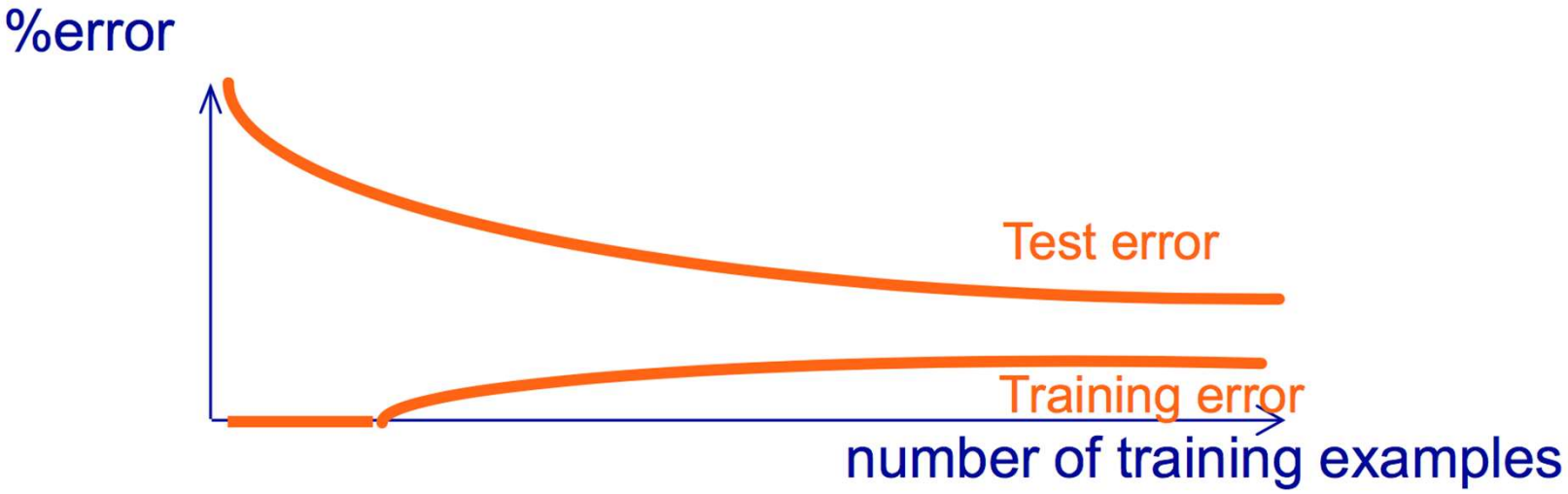
Four Pillars



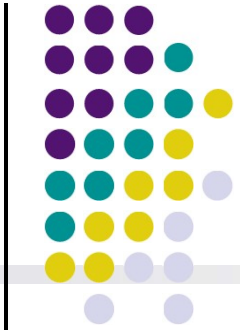
- Consistency (guarantee generalization)
  - Under what conditions will a model be consistent ?
- Model convergence speed (a measure for generalization)
  - How does generalization capacity improve when sample size L grows ?
- Generalization capacity control
  - How to control in an efficient way model 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 ?





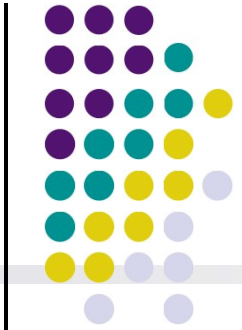
Vapnik Main Theorem



- 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.

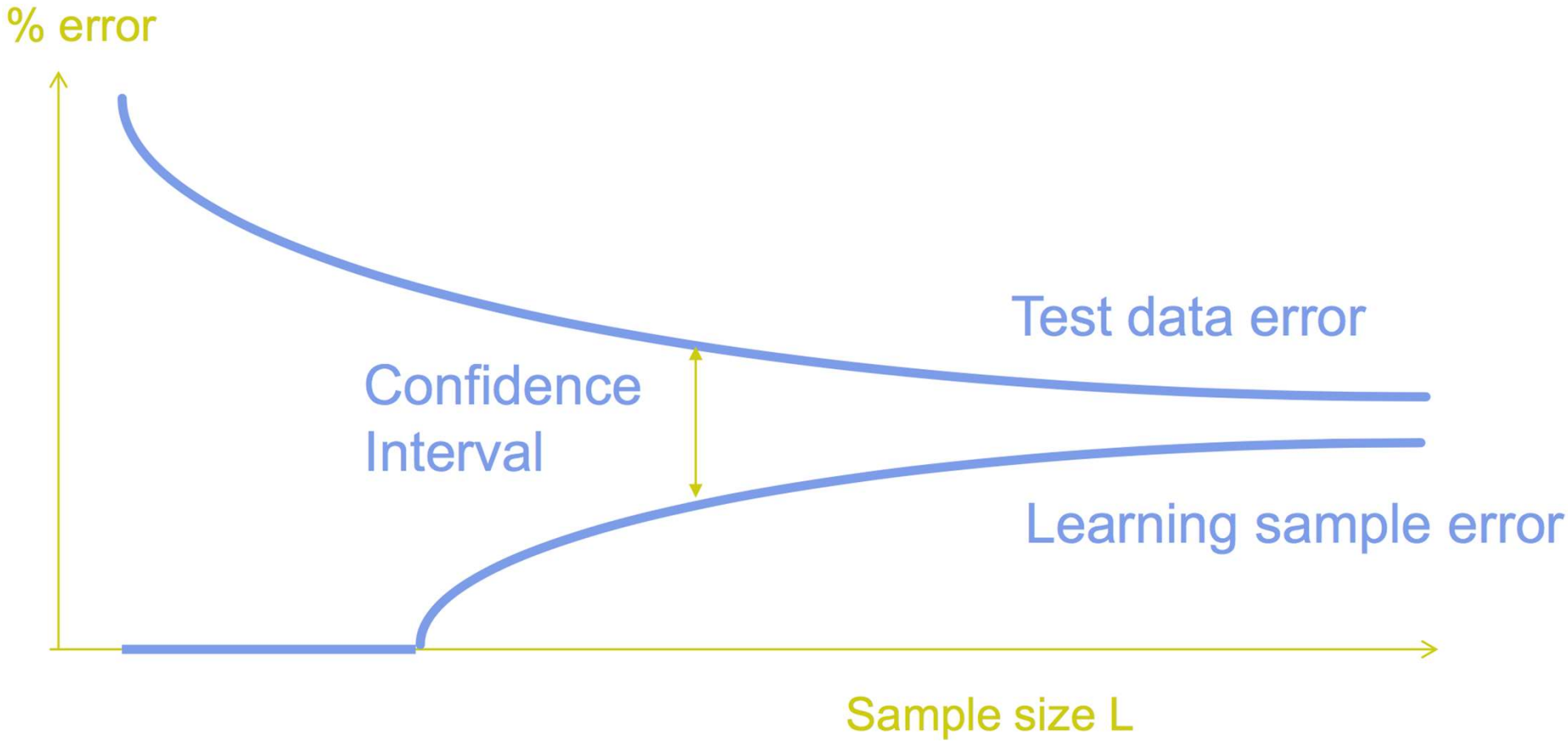
- A finite VC dimension d not only guarantees a generalization capacity (consistency), but to pick h in a family H with finite VC dimension d is the only way to build a model that generalizes.

Model Convergence Speed (Generalization Capacity)

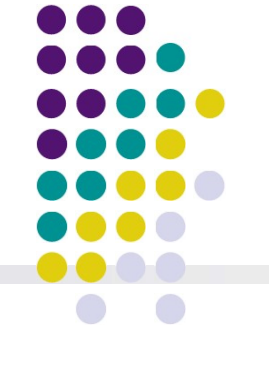


- Q : What is the nature of model error difference between learning data (sample) and test data, for a sample of finite size m ?
- A : This difference is no greater than a limit that only depends on the ratio between VC dimension d of model functions family H, and sample size m, i.e., d/m .
- This statement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



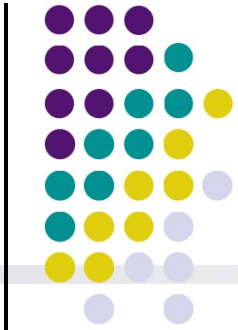
How To Control Model Generalization Capacity



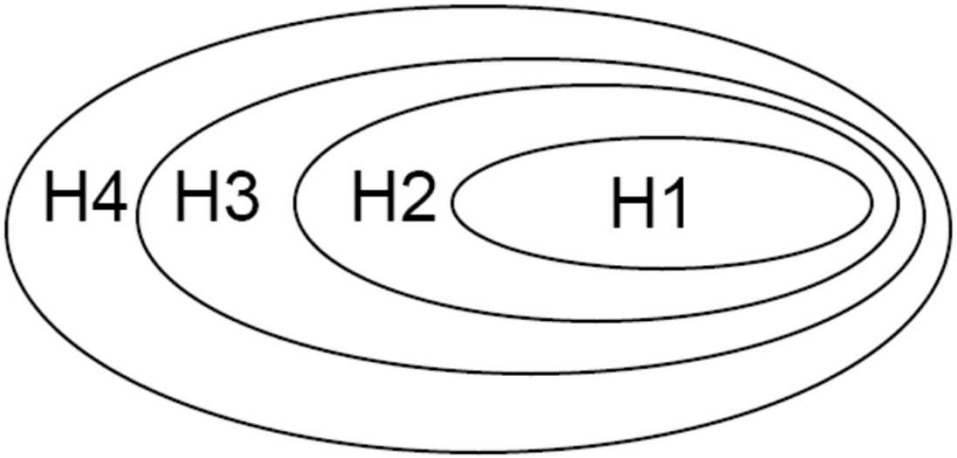
Risk Expectation = Empirical Risk + Confidence Interval

- 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"

Structural Risk Minimization



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

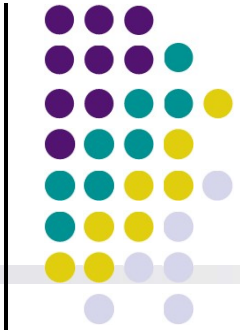


- SRM: choose H to minimize bound on true error !

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

unfortunately a somewhat loose bound…

SRM Strategy (1)



- With probability 1 - δ

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

- 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 ε(h)

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

SRM Strategy (2)

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

  $d_1 < d_2 < \ldots < d_n$

- For each family H_i of our sequence, the inequality

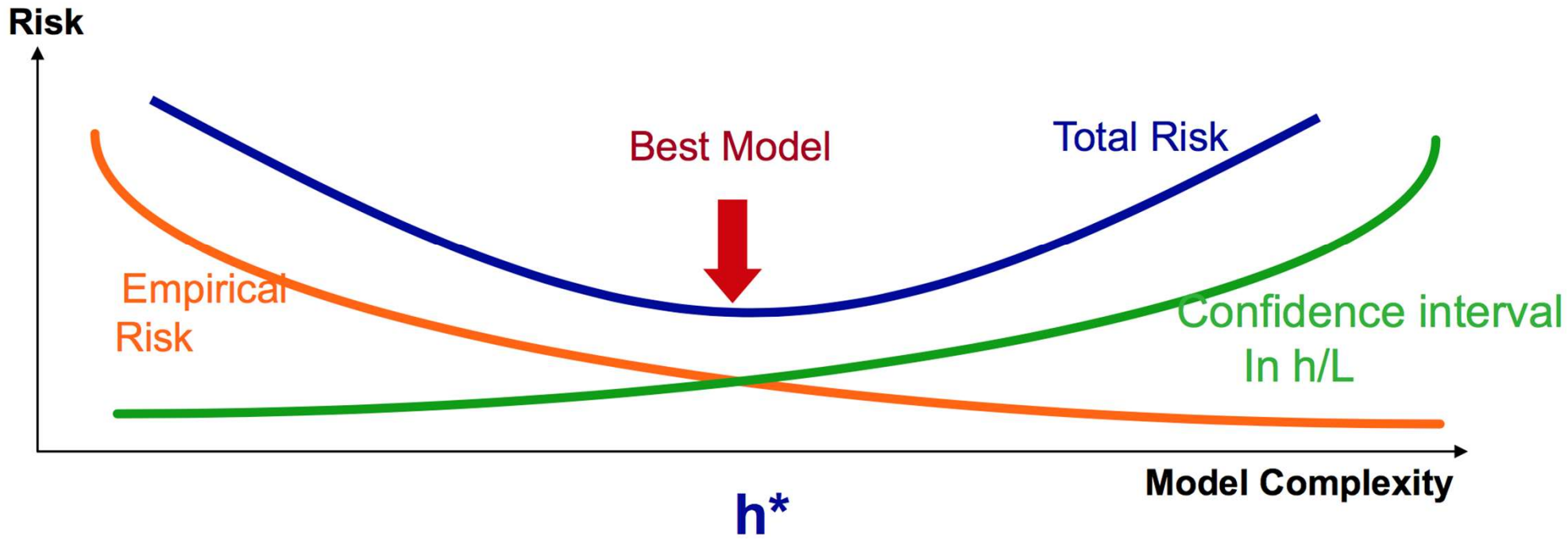
$$\varepsilon(h) \leq \hat{\varepsilon}(h) + O\left(\sqrt{\frac{d}{m}log\frac{m}{d} - \frac{1}{m}log\delta}\right)$$ is valid

  - 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

Putting SRM Into Action: Linear Models Case (2)

- To control ||w||, one can envision two routes to model:

  - Regularization / Ridge Regression, ***i.e. min. over w and b***

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

  - Support Vector Machines (SVM), ***i.e. 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: $y = X^T\theta + \varepsilon$

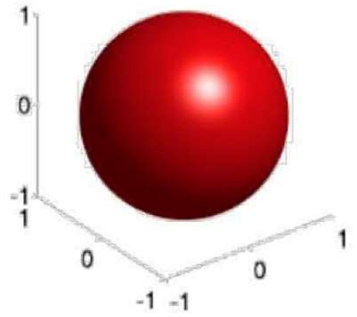
$$\theta^* = \underset{\theta}{argmin}(y - X^T\theta)^T(y - X^T\theta)$$
$$= \underset{\theta}{argmin}\,||y - X^T\theta||^2$$

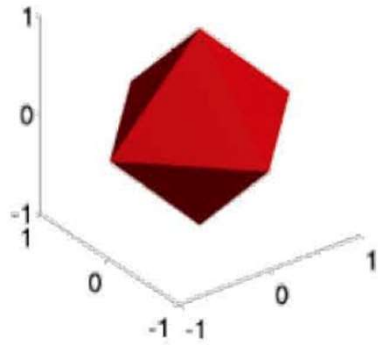
- Regularized LR:
  - L2-regularized LR:

$$\theta^* = \underset{\theta}{argmin}\,||y - X^T\theta||^2 + \lambda||\theta||$$

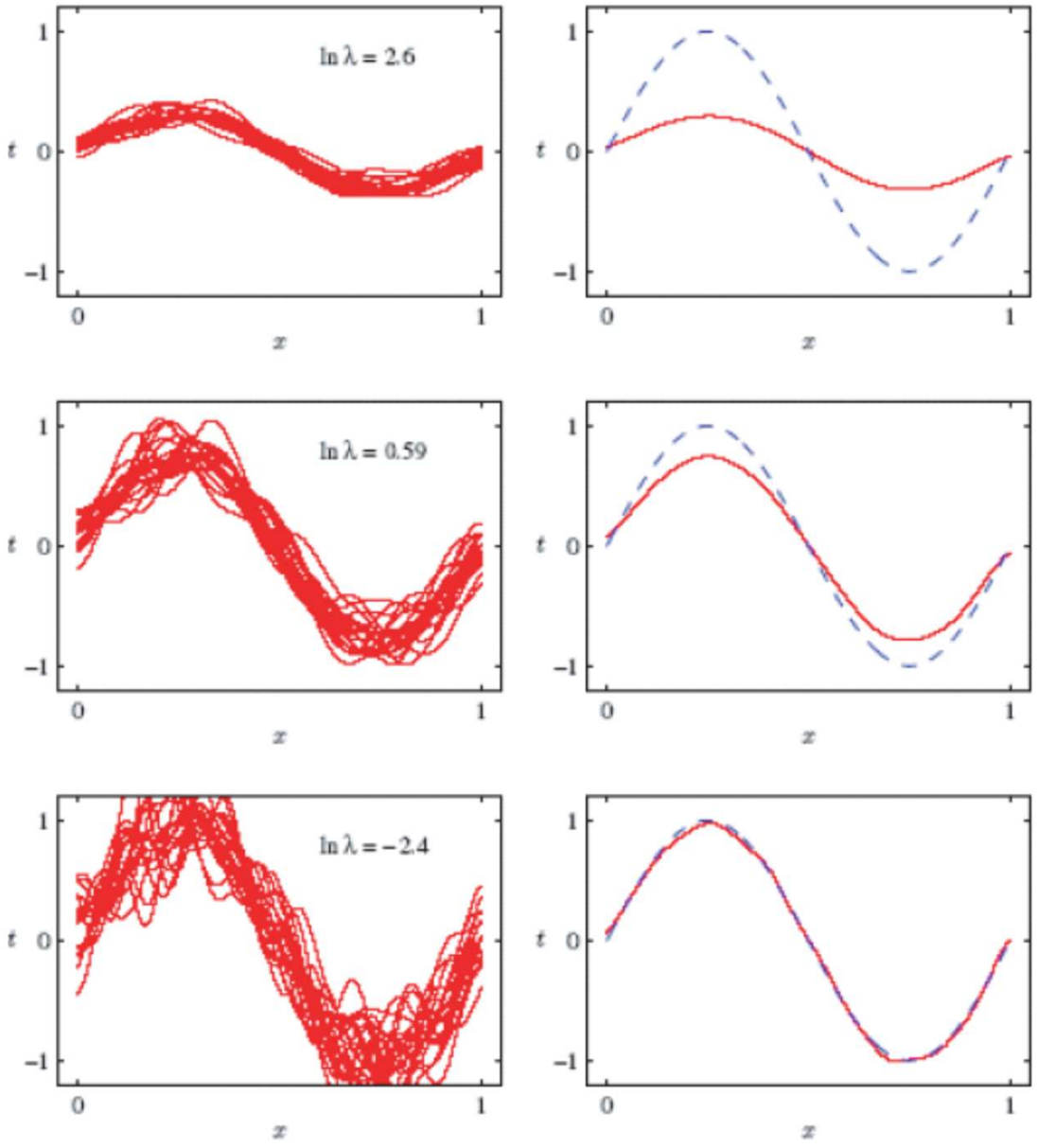
  - L1-regularized LR:

$$\theta^* = \underset{\theta}{argmin}\,||y - X^T\theta||^2 + \lambda|\theta|$$



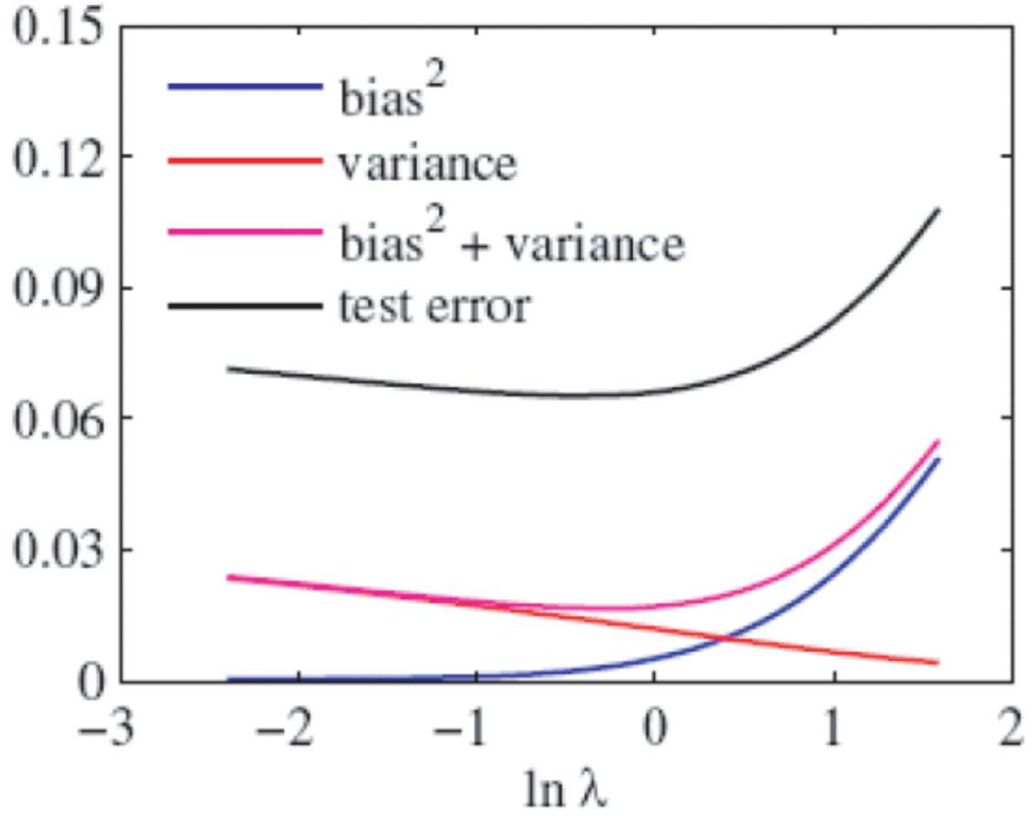


Bias-Variance Tradeoff



- λ is a "regularization" terms, 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 to 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 F = { M_1, M_2, …, M_l }, find Mi ∈ F s.t.

$$M_i = \underset{M \in F}{argmax}\, J(D, M)$$

1. Cross Validation

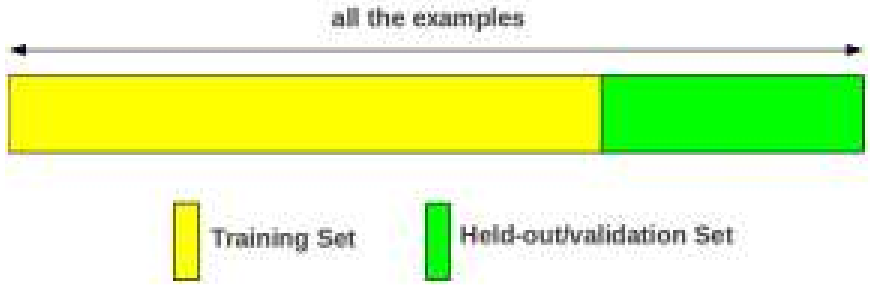
- We are given training data D and test D_{test}, and we would like to fit this data with a model $p_i(x;\theta)$ from the family F (e.g., LR), which is indexed by i and parameterized by θ.

- Hold-out cross-validation/simple cross validation

- K-fold cross-validation

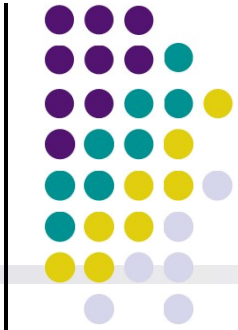
Held-out Data

- Set aside a fraction α (say 10%-20%) of the training data
- This part becomes our held-out data
  Other names: validation/development data



- Remember: Held-out data is NOT the test data
- Train each model using the remaining training data
- Evaluate error on the held-out data
- Choose the model with the smallest held-out error
- Problems:
  - Wastes training data, so typically used when we have plenty of training data
  - Held-out data may not be good if there was an unfortunate split → repeated random subsampling

K-fold Cross Validation



- Create K equal sized partitions of the training data
- Each partition has N/K examples
- Train using K − 1 partitions, validate on the remaining partition
- Repeat the same K times, each with a different validation partition

- Finally, choose the model with smallest average validation error
- Usually K is chosen as 10 or 5

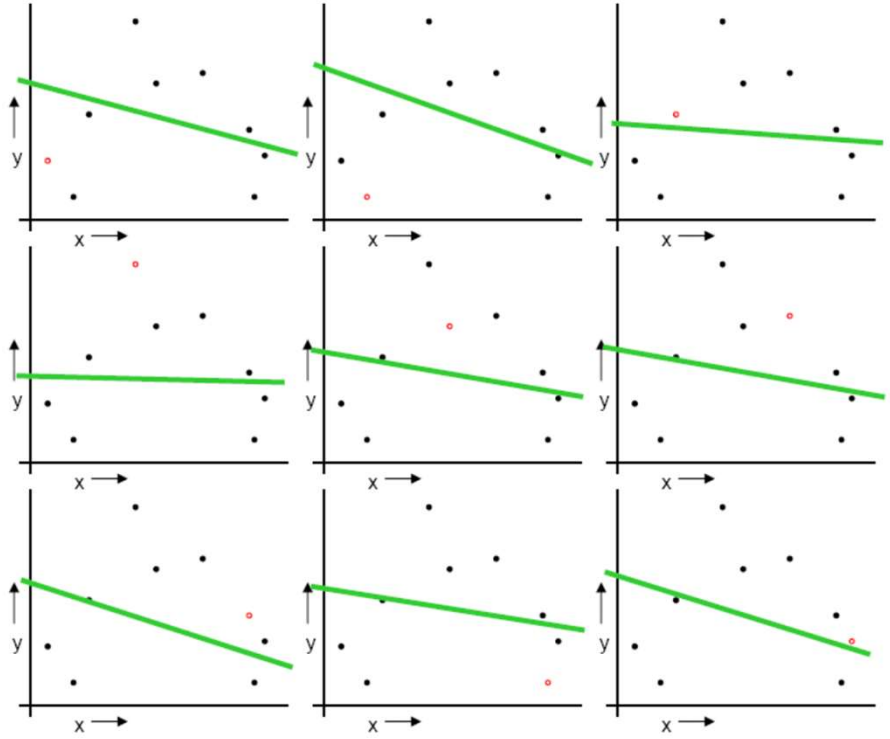
Leave-One-Out (LOO) Cross-Validation

- Special case of K-fold CV when K = N (number of training examples)
- Each partition is now an example
- Train using N − 1 examples, validate on the remaining example
- Repeat the same N times, each with a different validation example

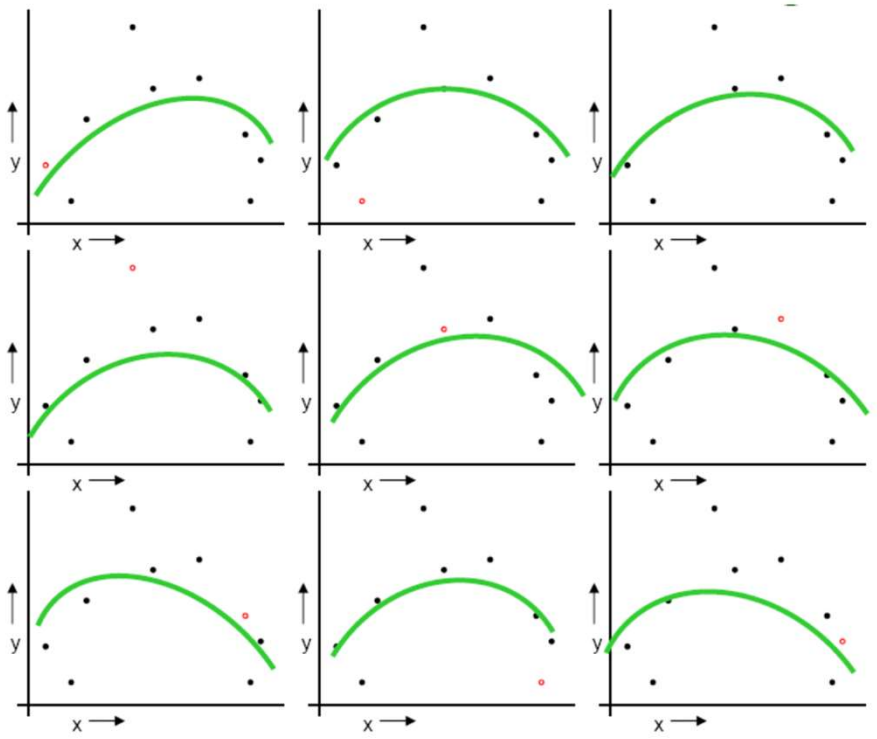
- Finally, choose the model with smallest average validation error
- Can be expensive for large N. Typically used when N is small

Example:

- When K = N, the algorithm is known as Leave-One-Out-Cross-Validation (LOOCV)

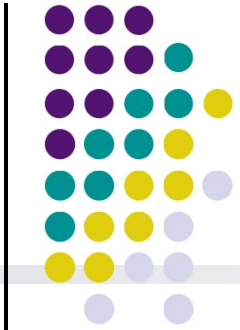


MSELOOCV(M_1)=2.12



MSELOOCV(M_2)=0.962

Practical Issues for CV



- How to decide the values for K and α (held-out ratio)
  - Commonly used K = 10 and α = 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

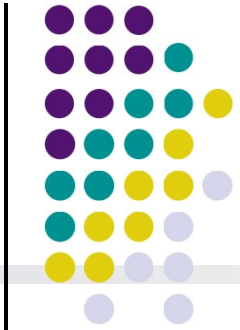
- Maximum-likelihood estimates are not always the best (James and Stein showed a counter example in the early 60's)
- Alternative: we "regularize" the likelihood objective (also known as penalized likelihood, shrinkage, smoothing, etc.), by adding to it a penalty term:

$$\hat{\theta}_{shrinkage} = \underset{\theta}{argmin}[l(\theta; D) + \lambda||\theta||]$$

where $\lambda > 0$ 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 L_1 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(θ|D) since θ does not have a probability distribution. Instead one can only write P(D|θ).
  - 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(θ|D) or P(f(θ)|D) for some function f.
  - One estimates parameters by computing P(θ|D) using Bayes rule:

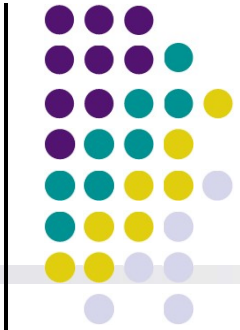
$$p(\theta|D) = \frac{p(D|\theta)p(\theta)}{p(D)}$$

Bayesians vs. Frequentists

You are no good when sample is small

You give a different answer for different priors

Bayesian Interpretation of Regularization



- 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

$$l(\theta) = nlog\frac{1}{\sqrt{2\pi}\sigma} - \frac{1}{\sigma^2}\frac{1}{2}\sum_{i=1}^{n}(y^{(i)} - \theta^T x^{(i)})^2$$

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

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

  - What is the posterior distribution of θ ?

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

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

Bayesian Interpretation of Regularization, cont.

- The posterior distribution of θ

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

- This leads to a new objective

$$I_{MAP}(\theta; D) = -\frac{1}{2\sigma^2}\frac{1}{2}\sum_{i=1}^{n}(y^{(i)} - \theta^T x^{(i)})^2 - \frac{1}{\tau^2}\frac{1}{2}\sum_{k-1}^{K}{\theta_k}^2$$

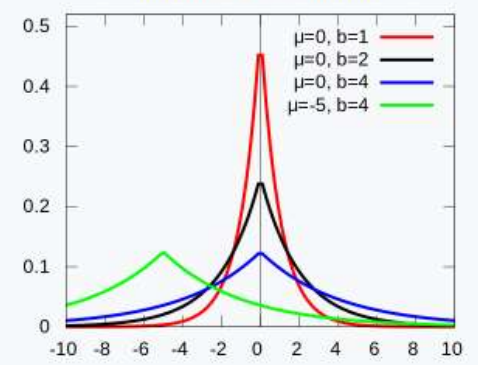
$$= l(\theta; D) + \lambda||\theta||$$

  - This is L_2 regularized LR ! — a MAP estimation of θ
  - What about L_1 regularized LR !

- How to choose λ
  - cross-validation !

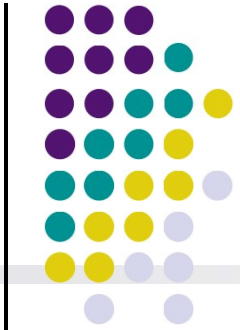
Laplace

Probability density function



Parameters	μ location (real) $b > 0$ scale (real)
Support	$\mathbb{R}$
PDF	$\frac{1}{2b}\exp\left(-\frac{\lvert x-\mu\rvert}{b}\right)$

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.

- Selecting a useful subset from all the features

Why Feature Selection

- Some algorithms scale (computationally) poorly with increased dimension
- Irrelevant features can confuse some algorithms
- Redundant features adversely affect regularization
- Removal of features can increase (relative) margin (and generalization)
- Reduces data set and resulting model size

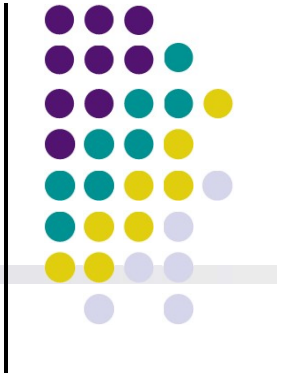
Note: Feature Selection is different from Feature Extraction
The latter transforms original features to get a small set of new features

- More on feature extraction when we cover Dimensionality Reduction

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.

How to Score Features

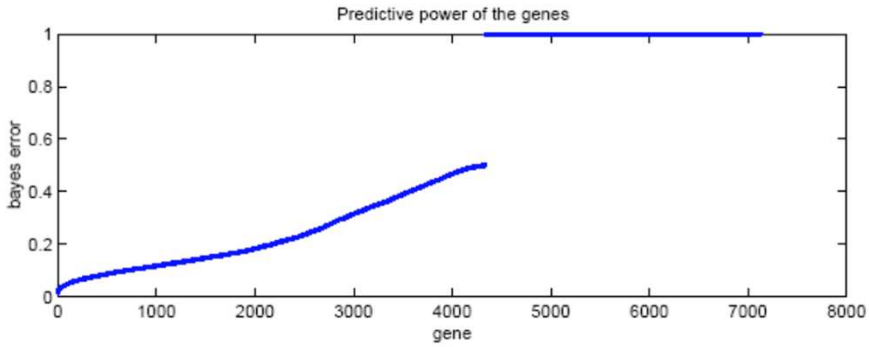


- How do you know which features can be pruned ?
  - Given labeled data, we can compute some simple score S(i) that measures how informative each feature x_i is about the class labels y.
  - Ranking criteria:
    - Mutual Information: score each feature by its mutual information with respect to the class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

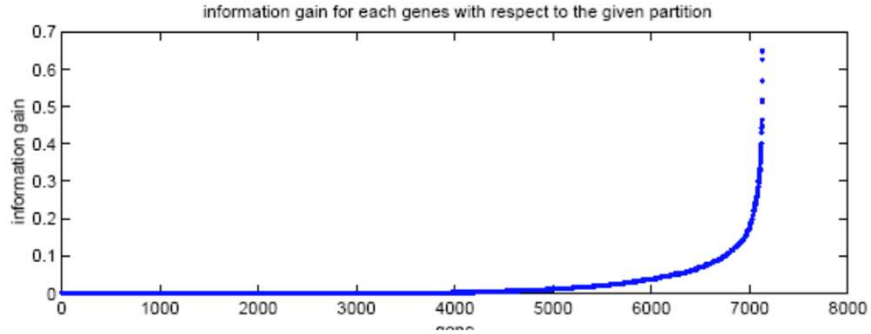
$$MI(x_i, y) = \sum_{x_i \in \{0,1\}} \sum_{y \in \{0,1\}} p(x_i, y) log \frac{p(x_i, y)}{p(x_i)p(y)}$$

Feature Ranking

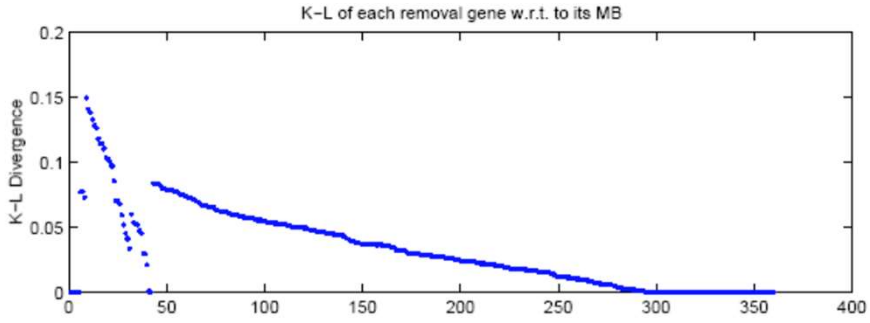
- Bayes error of each gene



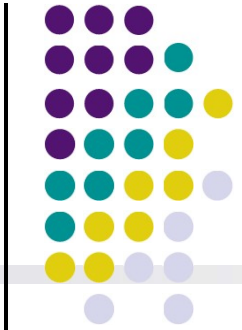
- Information gain for each genes with respect to the given partition



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

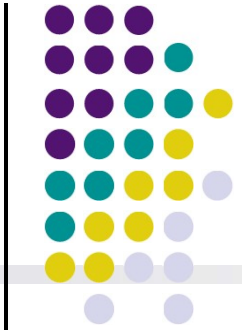


Wrapper Method



- Two types: Forward Search and Backward Search
- Forward Search
  - Start with no features
  - Greedily include the most relevant feature
  - Stop when selected the desired number of features
- Backward Search
  - Start with all the features
  - Greedily remove the least relevant feature
  - Stop when selected the desired number of features
- Inclusion/Removal criteria uses cross-validation

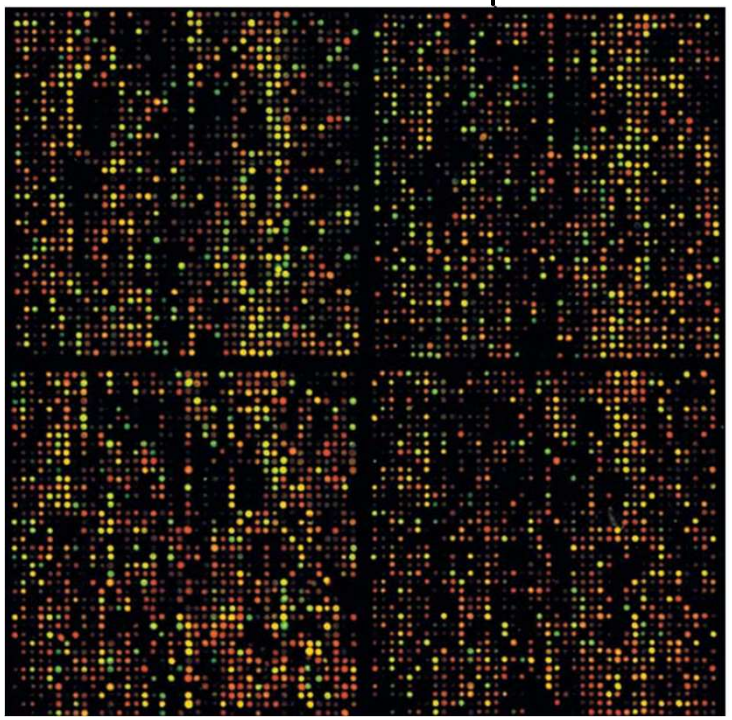
Wrapper Method

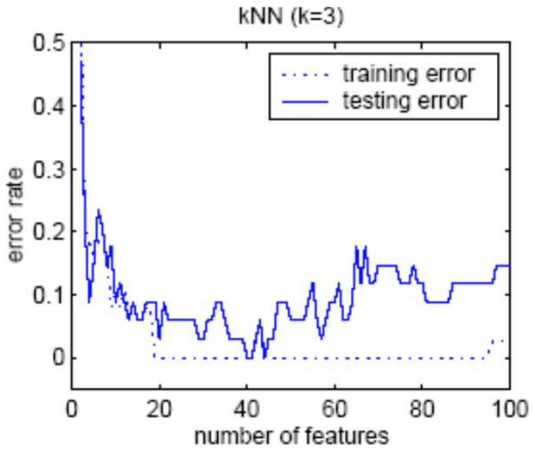


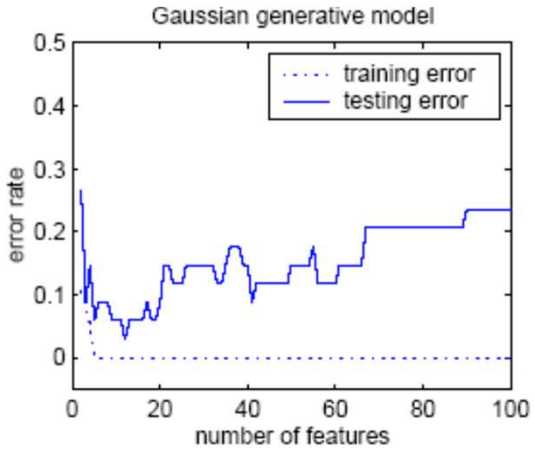
- Forward Search
  - Let F = {}
  - While not selected desired number of features
  - For each unused feature f:
  - Estimate model's error on feature set F ∪ f (using cross-validation)
  - Add f with lowest error to F
- Backward Search
  - Let F = {all features}
  - While not reduced to desired number of features
  - For each feature f ∈ F:
  - Estimate model's error on feature set F\f (using cross-validation)
  - Remove f with lowest error from F

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



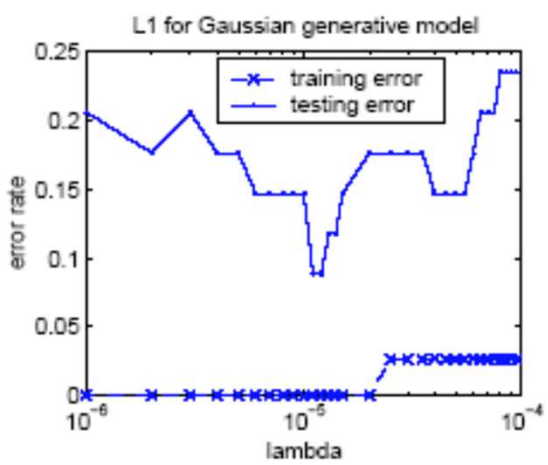


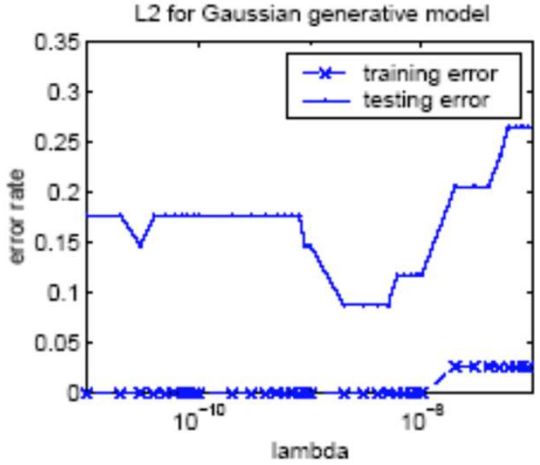


Regularization VS. Feature Selection

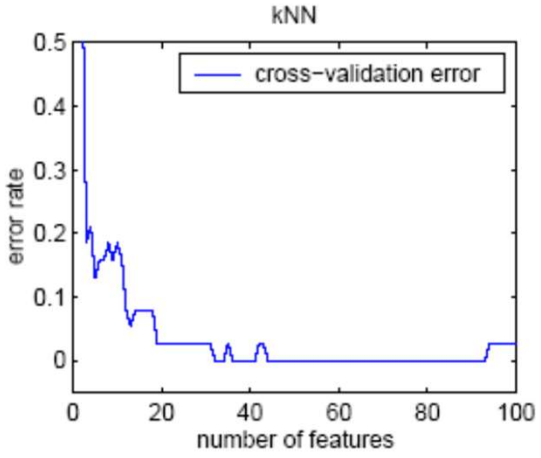
- Explicit feature selection often outperform regularization

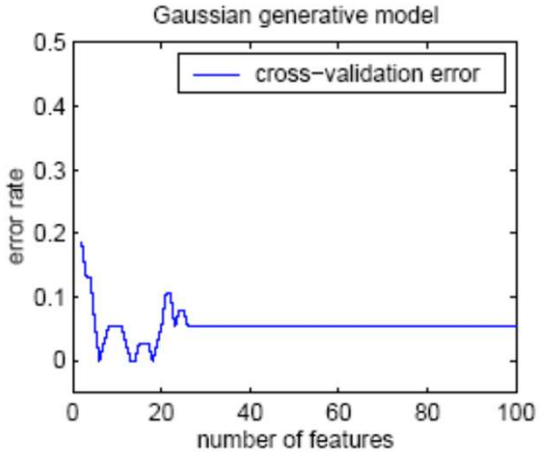
regression

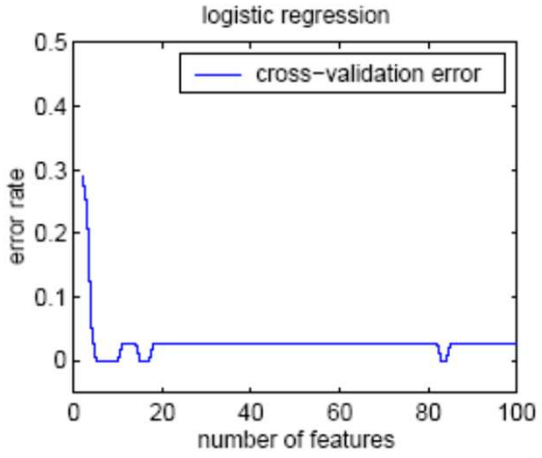




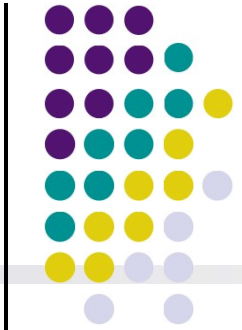
Feature Selection







4. Information Criterion



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

- The problem:
  - Given model family F = { M_1, M_2, …, M_l }, find $M_i \in F$ s.t.

$$M_i = \underset{M \in F}{argmax}\, J(D, M)$$

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

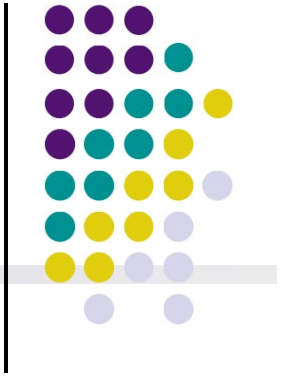
Model Selection via Information Criteria

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

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

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

AIC and BIC



- Akaike Information Criteria (AIC)

$$AIC = 2k - 2\log(L)$$

- Bayesian Information Criteria (BIC)

$$BIC = k\log(N) - 2\log(L)$$

k: # of model parameters

L: maximum value of the model likelihood function

- Applicable for probabilistic models (when likelihood is defined)
- AIC/BIC penalize model complexity
  - .. as measured by the number of model parameters
  - BIC penalizes the number of parameters more than AIC
- Model with the lowest AIC/BIC will be chosen
- Can be used even for model selection in unsupervised learning

5. Bayesian Model Averaging

- Recall the Bayesian Theory: (e.g., for data 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:

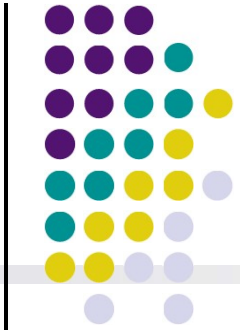
$$P(D|M) = \int_{\theta} P(D|\theta, M)P(\theta|M)d\theta$$

- A few steps of approximations 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)