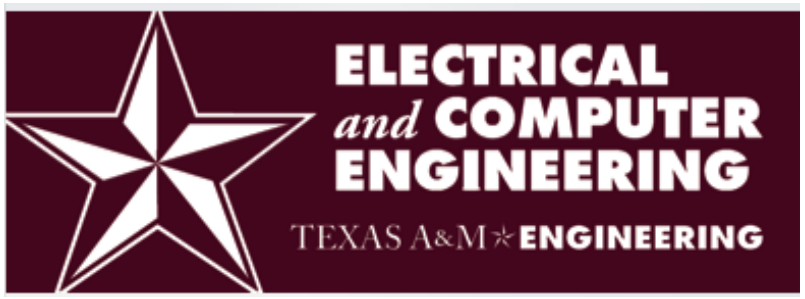
Materials Informatics

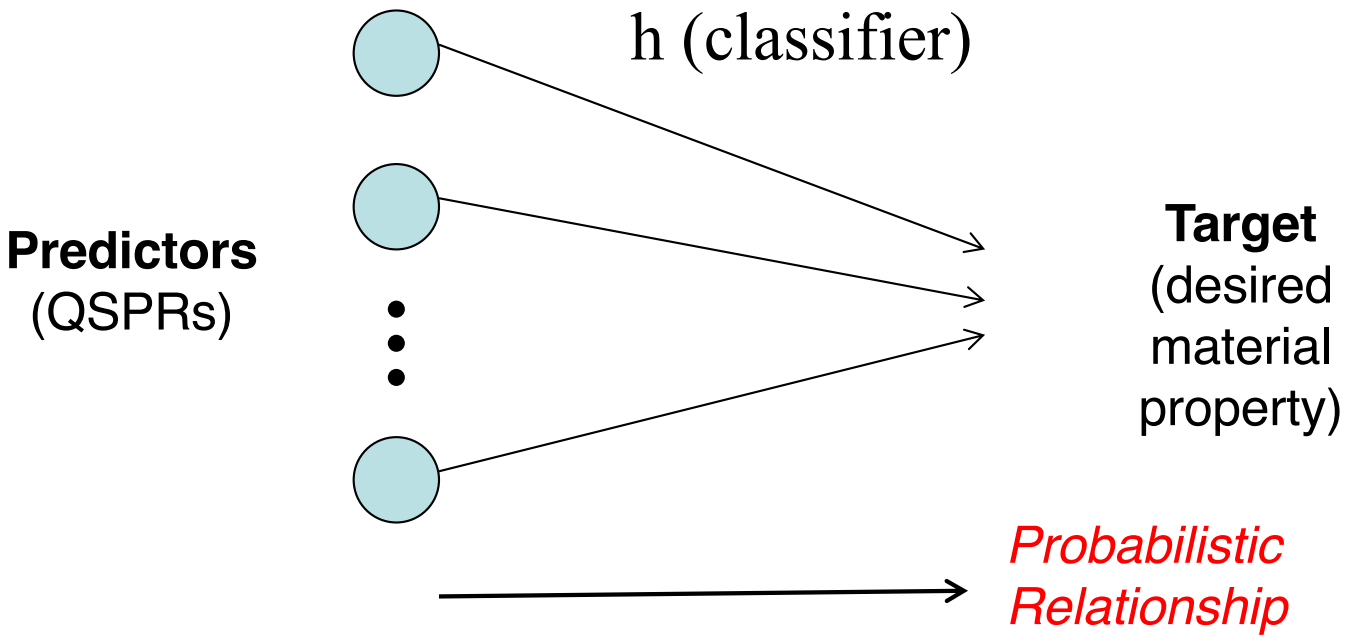
Lecture 5: Basics of Classification

Ulisses Braga Neto

Department of Electrical and Computer Engineering
Texas A&M University



Classifier Design



Variables: $X \in R^p$

Target: $Y \in \{0,1\}$

Joint Distribution: F_{XY}

Classification Error:

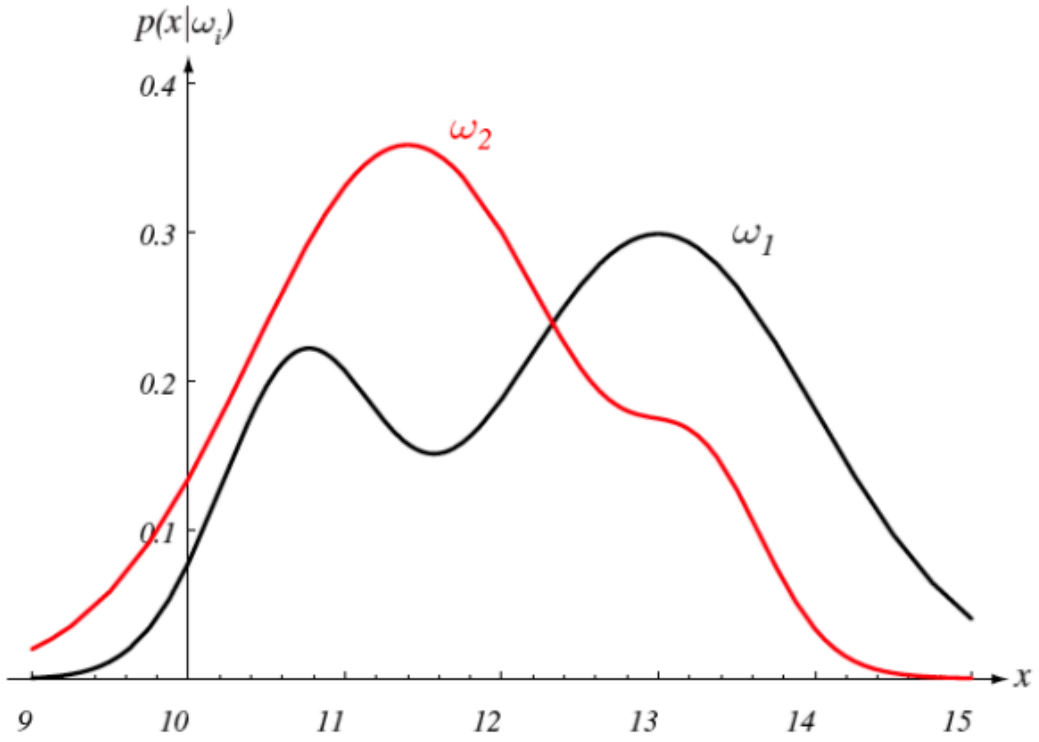
$$\varepsilon[h] = P_{XY}(h(X) \neq Y) = E(|Y - h(X)|)$$

Optimal Classifier

- Every problem has an optimal classifier, called the Bayes classifer.
- The corresponding classification error is called the Bayes error and is usually nonzero.
- To find the Bayes classifier and error one needs to know the joint distribution F_{XY}
- This distribution is usually unknown or only known partially, so one must resort to design sub-optimal classifiers based on training data.

Class-Conditional Densities

The relative frequencies of each label as a function of predictor values are given by the *class-conditional densities* $p(x|Y = i)$, for $i = 0, 1$.



Posterior Probabilities

Using Bayes' theorem, we can start from the prior probabilities and class-conditional densities and find the posterior probability of $Y = i$ given that $X = x$ has been observed, for $i = 0, 1$:

$$P(Y = i|X = x) = \frac{p(x|Y = i)P(Y = i)}{p(x)}$$

$$= \frac{p(x|Y = i)P(Y = i)}{p(x|Y = 0)P(Y = 0) + p(x|Y = 1)P(Y = 1)}$$

where the posterior probability function $\eta : R^d \rightarrow [0, 1]$,

$$\eta(x) = P(Y = 1|X = x)$$

plays a very important role in the sequel.

Classification Error

- The classification error is the "average" conditional classification error:

$$\begin{aligned}\epsilon[\psi] &= P(\psi(X) \neq Y) \\ &= \int_{x \in R^d} P(\psi(X) \neq Y | X = x) p(x)\, dx \\ &= E[P(\psi(X) \neq Y | X)]\end{aligned}$$

Therefore, knowing the error at each point $x \in R^d$ of the feature space, plus the "weight" $p(x)$, is enough to determine the overall classification error.

Class-Specific Error Rates

- We can rewrite the previous equation as:

$$\epsilon[\psi] = (1-c)\epsilon^0[\psi] + c\epsilon^1[\psi]$$

where $c = P(Y=1)$, and

$$\epsilon^0[\psi] = \int_{\{x|\psi(x)=1\}} p(x \mid Y=0)\, dx$$

$$\epsilon^1[\psi] = \int_{\{x|\psi(x)=0\}} p(x \mid Y=1)\, dx$$

are the *class-specific* error rates. Given ψ, these error rates do not depend on the prior probabilities c and $1-c$, while the overall error $\epsilon[\psi]$ clearly does.

Class-Specific Error Rates

- Suppose ψ is used as a *test* to distinguish "positive" cases (class 1) from "negative" cases (class 0).
- Then $\epsilon^0[\psi]$ and $\epsilon^1[\psi]$ are called the test's *false positive* and *false negative* error rates, respectively.
- One also defines the test's *sensitivity* and *specificity* as

$$\text{sensitivity} = 1 - \epsilon^1[\psi] = \int_{\{x|\psi(x)=1\}} p(x \mid Y = 1)\, dx$$

$$\text{specificity} = 1 - \epsilon^0[\psi] = \int_{\{x|\psi(x)=0\}} p(x \mid Y = 0)\, dx$$

Optimal Classifier

- (DGL Theorem 2.1) The classifier with minimal error is

$$\psi^*(x) = \arg\max_i P(Y = i|X = x) = I_{\eta(x)>\frac{1}{2}} .$$

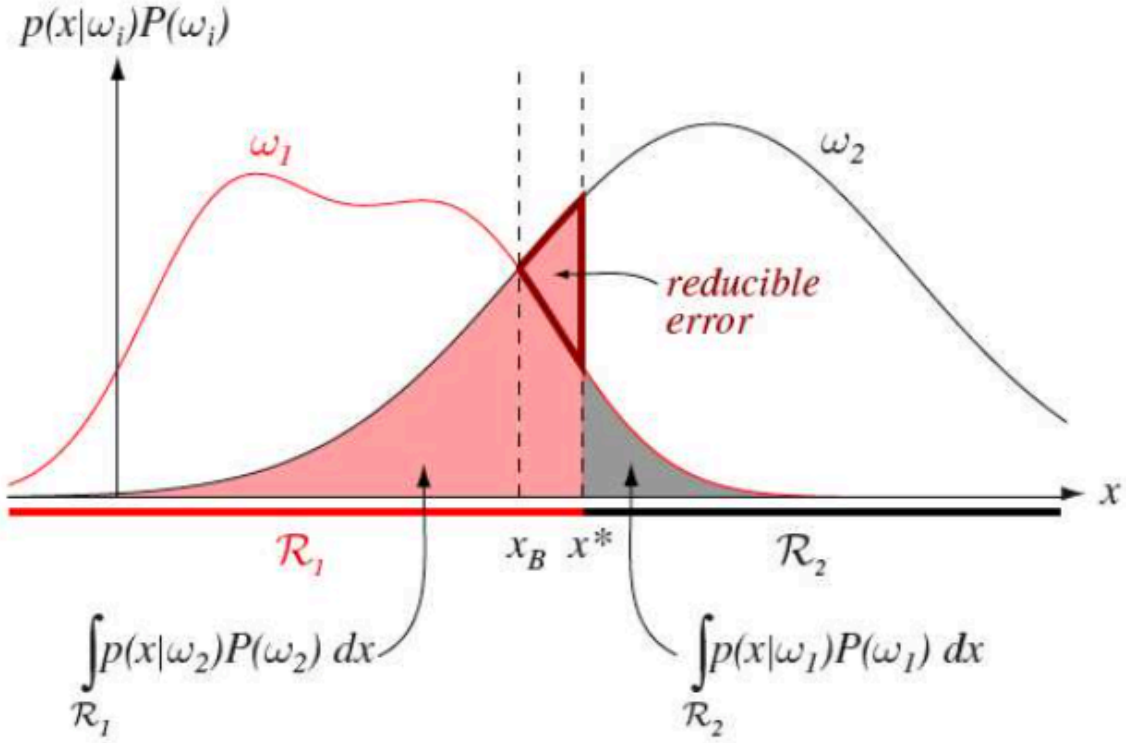
This is the MAP (Maximum A-Posteriori) classifier, more commonly known as the *Bayes classifier*.

Optimal Classifier

- By Bayes theorem, we have, equivalently,

$$\psi^*(x) = \arg\max_i p(x|Y=i)P(Y=i)$$

- Graphical interpretation:



Bayes Error

- The error of the Bayes classifier $\epsilon^* = \epsilon[\psi^*]$ is a fundamental quantity in PR, known as the *Bayes error*.
- Note that the Bayes classifier is given by

$$\psi^*(x) = \begin{cases} 1\,, & \eta(x) > 1-\eta(x) \quad (\Leftrightarrow \eta(x) > \frac{1}{2}) \\ 0\,, & \eta(x) \leq 1-\eta(x) \quad (\Leftrightarrow \eta(x) \leq \frac{1}{2}) \end{cases}$$

- Therefore

$$\epsilon^* = \int_{\{x|\eta(x)<1-\eta(x)\}} \eta(x)p(x)\,dx + \int_{\{x|\eta(x)\geq 1-\eta(x)\}} (1-\eta(x))p(x)\,dx$$

$$= E[\min\{\eta(X), 1-\eta(X)\}]$$

Bayes Error - II

- Using the identity

$$\min\{a, 1-a\} = \frac{1}{2} - \frac{1}{2}|2a-1|, \quad 0 \le a \le 1$$

It follows that

$$\epsilon^* = \frac{1}{2} - \frac{1}{2}E[|2\eta(X)-1|]$$

- In particular, we always have $\epsilon^* \le \frac{1}{2}$.

Discriminant Functions

- A classifier can be specified through a set of *discriminant functions* $\{g_i(x)|i = 0, 1, \ldots, c-1\}$ as:

$$\psi(x) = \arg\max_i g_i(x)$$

- The $i-th$ *decision region* is determined by

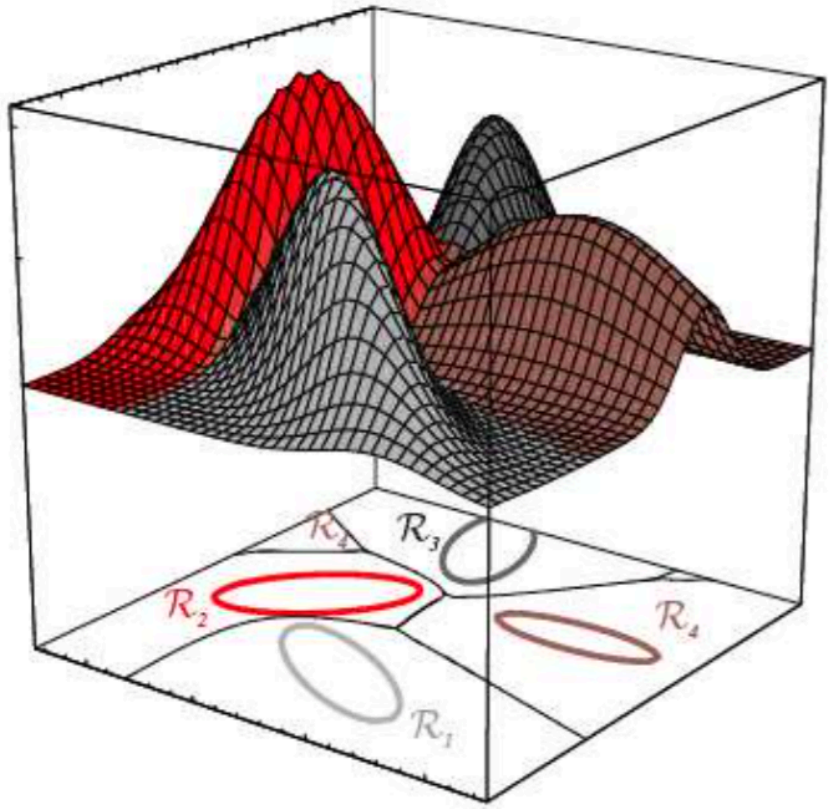
$$g_i(x) > g_j(x)\,, \quad \text{for all } i \neq j$$

The loci of ties among largest discriminant functions determine the *decision surfaces*.

- A set of discriminant functions determines a unique classifier, but the converse is not true: the same classifier can be determined by multiple sets of discriminant functions.

Discriminant Functions - II

Graphical Example:



Discriminant Functions - III

- In the two-category case, we can define a single discriminant function

$$g(x) = g_1(x) - g_0(x)$$

In which case the classifies is determined by

$$g(x) > 0 \Rightarrow \psi(x) = 1$$
$$g(x) \leq 0 \Rightarrow \psi(x) = 0$$

- For example, for the Bayes classifier

$$g(x) = P(Y = 1|X = x) - P(Y = 0|X = x)$$
$$g(x) = \ln \frac{p(x|Y = 1)}{p(x|Y = 0)} + \ln \frac{p(Y = 1)}{p(Y = 0)}$$

Gaussian Case

- Consider the case where the class-conditional densities are multivariate Gaussian densities:

$$p(x|Y=i) \sim N_d(\mu_i, \Sigma_i)\,, \quad i = 0, 1, \ldots, c-1$$

In other words,

$$p(x|Y=i) \;=\; (2\pi)^{-\frac{d}{2}} |\Sigma_i|^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(x-\mu_i)^T \Sigma_i^{-1}(x-\mu_i)\right)$$

for $i = 0, 1, \ldots, c-1$.

- This is a case of great interest in engineering and science.

Equal-Variance Case

- If we can assume that the covariance matrices are equal to each other

$$\Sigma_i = \Sigma\,, \quad i = 0, 1$$

We can show that the optimal discriminant (and thus the optimal decision boundary) is linear:

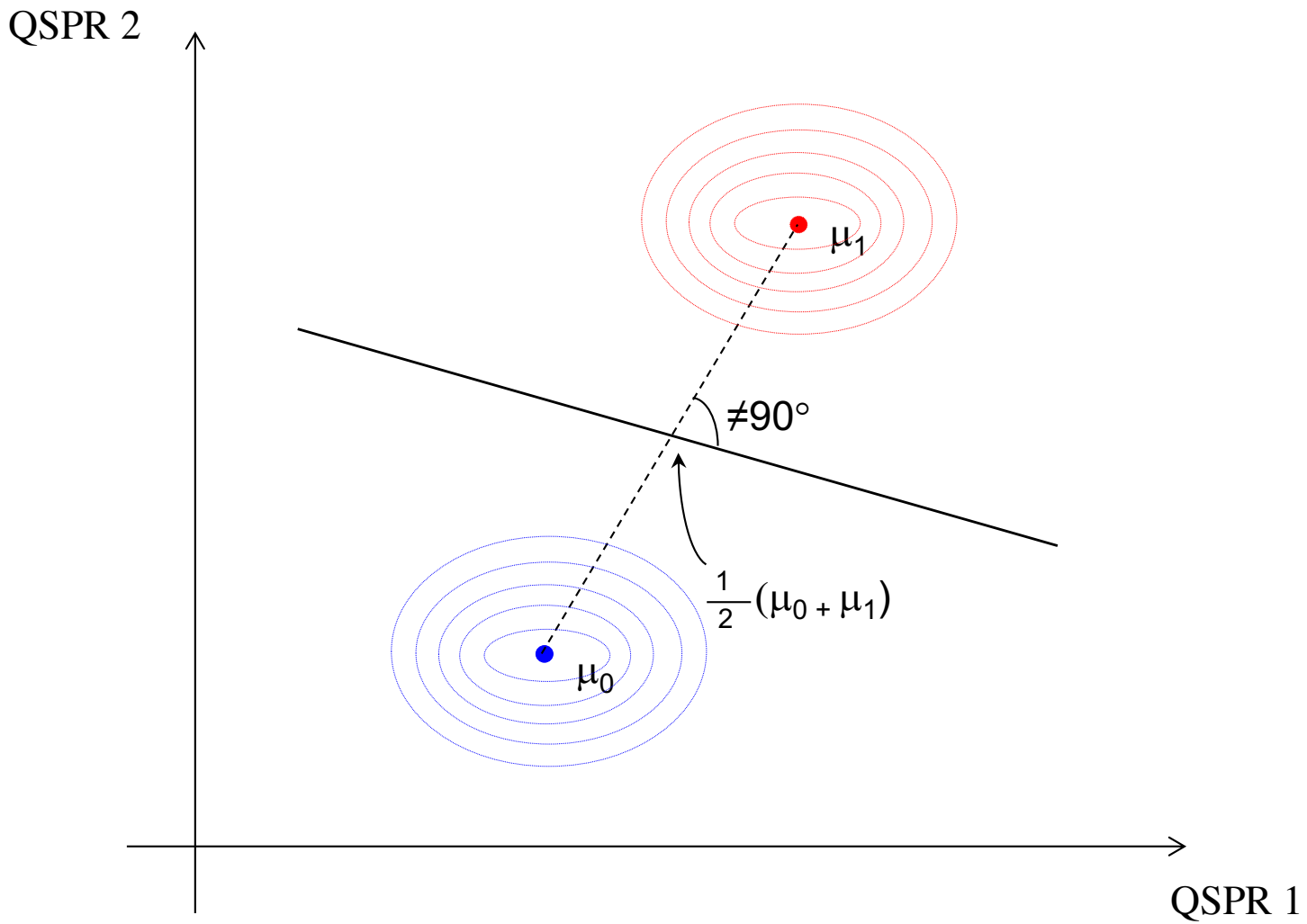
$$g(x) = a^T x + b = 0$$

where

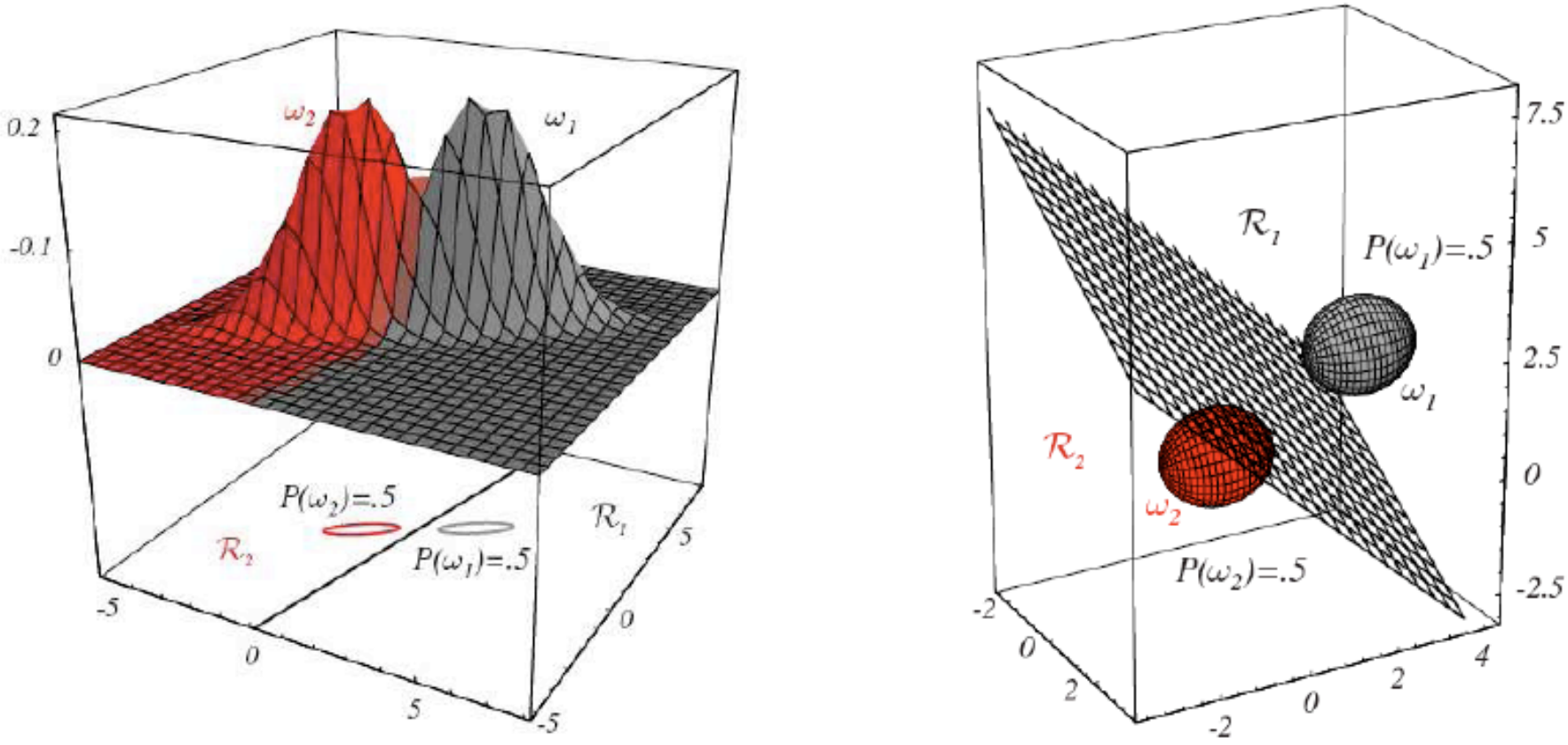
$$a = \Sigma^{-1}(\mu_1 - \mu_0)$$

$$b = -\frac{1}{2}(\mu_1 - \mu_0)^T \Sigma^{-1}(\mu_1 + \mu_0)$$

Example



Example - II



From R. Duda, P. Hart and D. Stork, Pattern Classification, 2nd ed., John Wiley & Sons, 2001.

Optimal Classification Error

- The optimal classifier in the Gaussian equal-variance case is a hyperplane, and its error can be shown to be given by

$$\epsilon^* = \Phi\left(-\frac{1}{2}\delta\right)$$

where Φ is the cdf of a standard Gaussian and δ is the Mahalanobis distance between classes:

$$\delta^2 = (\mu_1 - \mu_0)^T \Sigma^{-1} (\mu_1 - \mu_0)$$

Sample Data

- In practice, the feature-label distribution F_{XY} is unknown, and so the Bayes classifier is unknown.
- What is available instead is a *sample* from F_{XY}:

$$S_n = \{(X_1, Y_1), \ldots, (X_n, Y_n)\}$$

where the (X_i, Y_i) are independent and identically distributed (i.i.d.), with $(X_i, Y_i) \sim F_{XY}$.

- The *sample size* n is a deterministic parameter, while

$$n_0 = \sum_{i=1}^{n} I_{Y_i=0} \quad \text{and} \quad n_1 = \sum_{i=1}^{n} I_{Y_i=1}$$

are binomial *random variables* with parameters $(n, 1-c)$ and (n, c), respectively.

Sample Data - II

- In *separate sampling*, the data are sample from each population separately.

- In this case, the labels $Y_1, \ldots, Y_n$ are not i.i.d. and

$$n_0 = \sum_{i=1}^{n} I_{Y_i=0} \quad \text{and} \quad n_1 = \sum_{i=1}^{n} I_{Y_i=1}$$

are *deterministic* parameters chosen prior to sampling.

Classification Error

- Two kinds of error are of interest here. The first is the familiar classification error of the designed classifier:

$$\epsilon_n = P(\psi_n(X) \neq Y | S_n)$$

This is called the *conditional error* or *true error*.

- The conditional error is a function of the random data S_n, and therefore it is a random variable if the value of S_n is not given. The second kind of error of interest is the expected value of ϵ_n over all sample sets S_n:

$$\mu_n = E[\epsilon_n] = P(\psi_n(X) \neq Y)$$

This is called the *unconditional error* or *expected error*.

Consistency

- Consistency has to do with the natural requirement that, as the number of samples increases to infinity, classification error should in some sense converge to the Bayes error.
- The classification rule Ψ_n is said to be (weakly) consistent if

$$\epsilon_n \to \epsilon^* \quad \text{in probability}$$

whereas it is said to be strongly consistent if

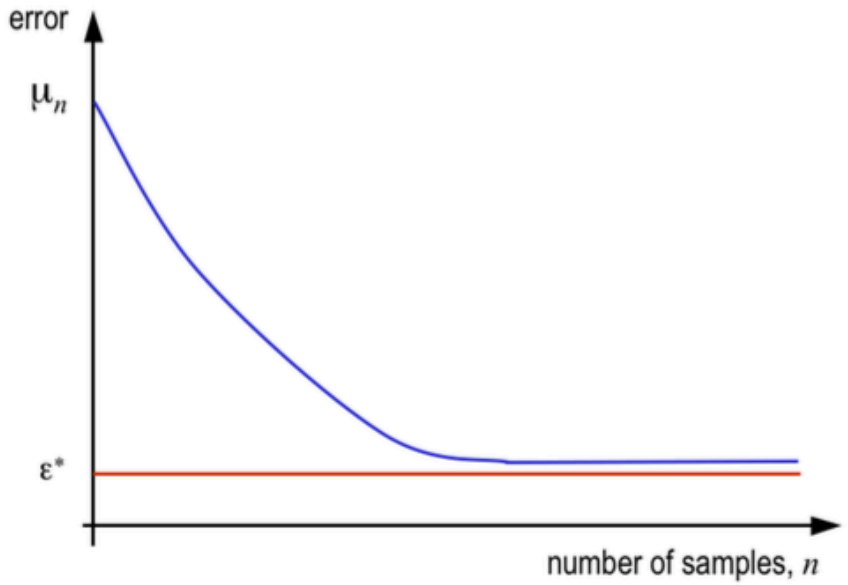
$$\epsilon_n \to \epsilon^* \quad \text{with probability 1}$$

Consistency - II

- The following result relates (weak) consistency to the expected error.
- Theorem: Ψ_n is weakly consistent if and only if

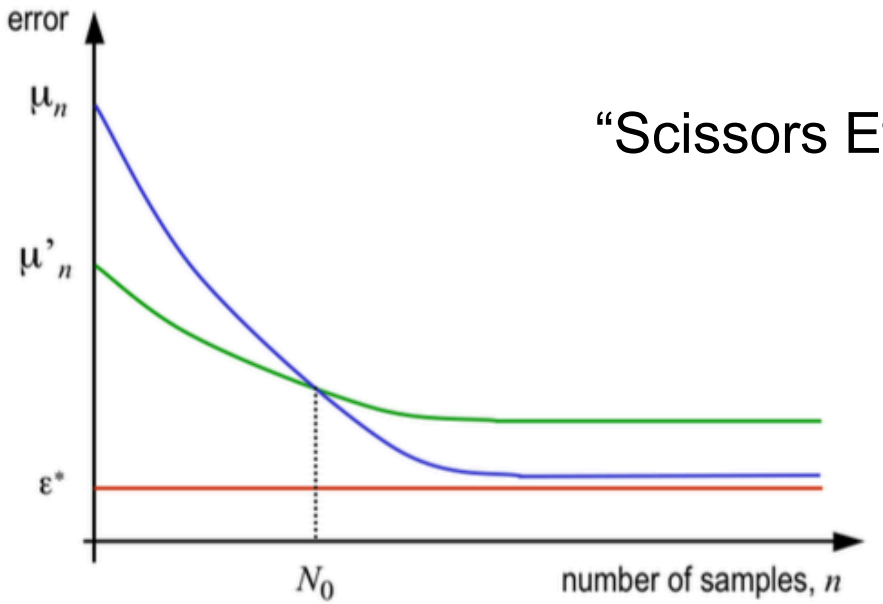
$$E[\epsilon_n] \rightarrow \epsilon^*$$

Note that this is ordinary convergence of real numbers.



Consistency - III

- A word of caution: a non-consistent classification rule may still be useful, in fact, it may be better than a consistent one, in *small-sample* scenarios.
- In the example below, the non-consistent classification rule is better than the consistent one for $n < N_0$.



"Scissors Effect"

Linear Discriminant Analysis

- LDA uses the Gaussian assumption to estimate the optimal classifier, using the sample means and sample covariance matrices. The discriminant is linear:

$$g_n(x) = a_n^T x + b_n = 0$$

where

$$a_n = \hat{\Sigma}^{-1}(\hat{\mu}_1 - \hat{\mu}_0)$$

$$b_n = -\frac{1}{2}(\hat{\mu}_1 - \hat{\mu}_0)^T \hat{\Sigma}^{-1}(\hat{\mu}_1 + \hat{\mu}_0)$$

Linear Discriminant Analysis - II

- Here

$$\hat{\mu}_0 = \frac{1}{n_0}\sum_{i=1}^{n} X_i I_{Y_i=0} \qquad \hat{\mu}_1 = \frac{1}{n_1}\sum_{i=1}^{n} X_i I_{Y_i=1}$$

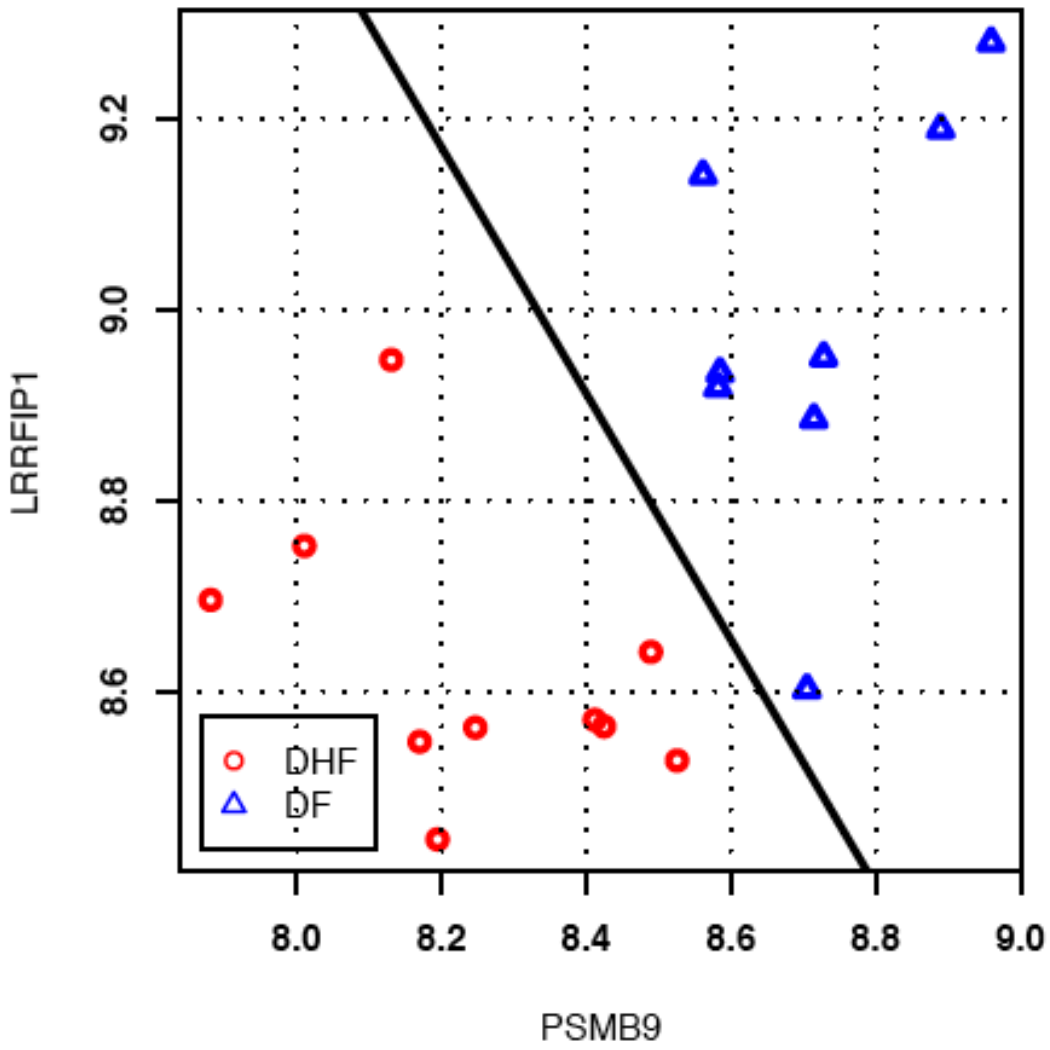
are the sample means, and

$$\hat{\Sigma} = \frac{1}{n-2}\sum_{i=1}^{n}\left[(X_i-\hat{\mu}_0)(X_i-\hat{\mu}_0)^T I_{Y_i=0} + (X_i-\hat{\mu}_1)(X_i-\hat{\mu}_1)^T I_{Y_i=1}\right]$$

is the pooled sample covariance matrix.

Example

Classification of Dengue Fever



Error of the LDA Classifier

- It can be shown that the classification error of the estimated LDA classifier is given by

$$\epsilon_n = \frac{1}{2}\left[\Phi\left(\frac{a_n^T\mu_0 + b_n}{\sqrt{a_n^T\Sigma a_n}}\right) + \Phi\left(-\frac{a_n^T\mu_1 + b_n}{\sqrt{a_n^T\Sigma a_n}}\right)\right]$$

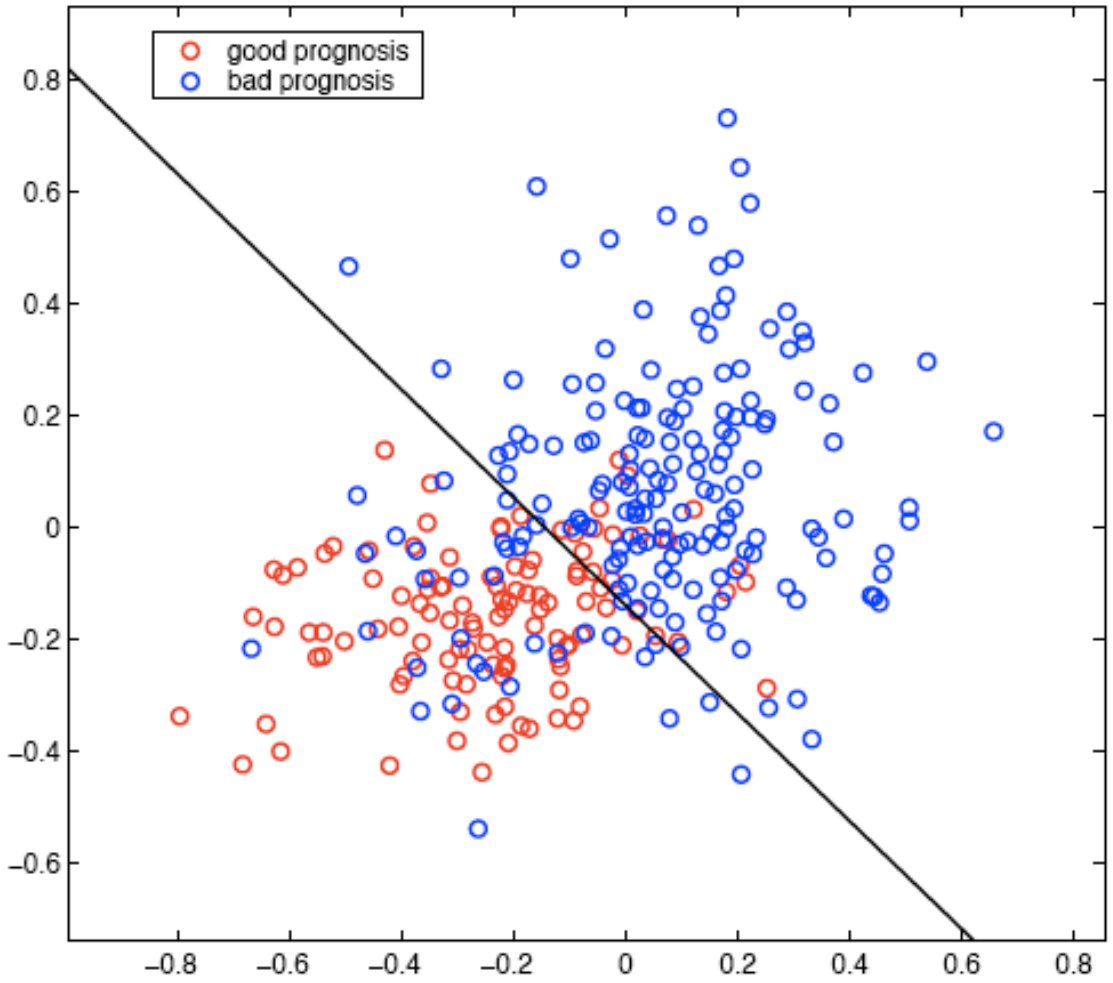
Error Estimation

- How does one estimate the classification error (the generalization error) of a given designed classifier?
- We do not know the true distribution of the data, but are given only training data, or (rarely) testing data.
- Error estimation is involved in classifier design itself (implicitly) and in feature selection.
- Error estimation has to do with the epistemological question: is scientific knowledge possible?

Apparent Error

- Also called resubstitution, it is the proportion of errors committed on the training data.
- It is very simple and fast to compute, and shows little variability.
- Its main disadvantage is that it is usually (but not always) optimistic on average, that is, on average it gives an error estimate smaller than the true classification error.
- In statistics, we say that such an estimator is (optimistically) biased.

Example



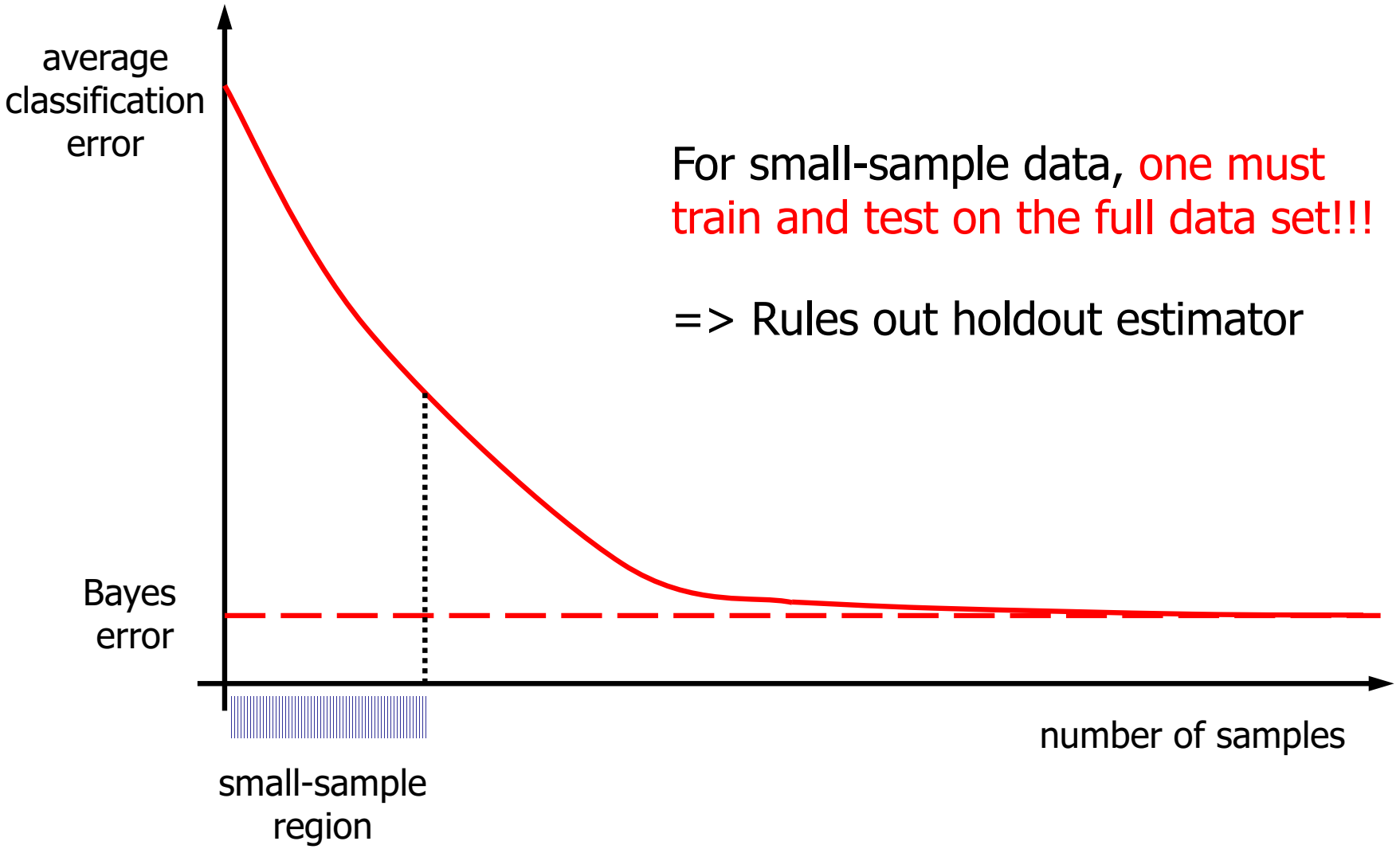
$$\hat{\epsilon}_n^r = \frac{\text{errors committed}}{\text{number of points}}$$

$$= \frac{52}{295} = 17.6\%$$

Test Sample

- To obtain an unbiased estimator, that is, one that on average is neither optimistic or pessimistic, the best option is to use a test set: an independent set of samples, which is not used in classifier design.
- The problem is to have such a set of samples available. If the number of test samples is small, the test sample estimator (also called holdout estimator) will be too variable.
- Unbiasedness is useless if variance is too large.

Small-Sample Problem



For small-sample data, one must train and test on the full data set!!!

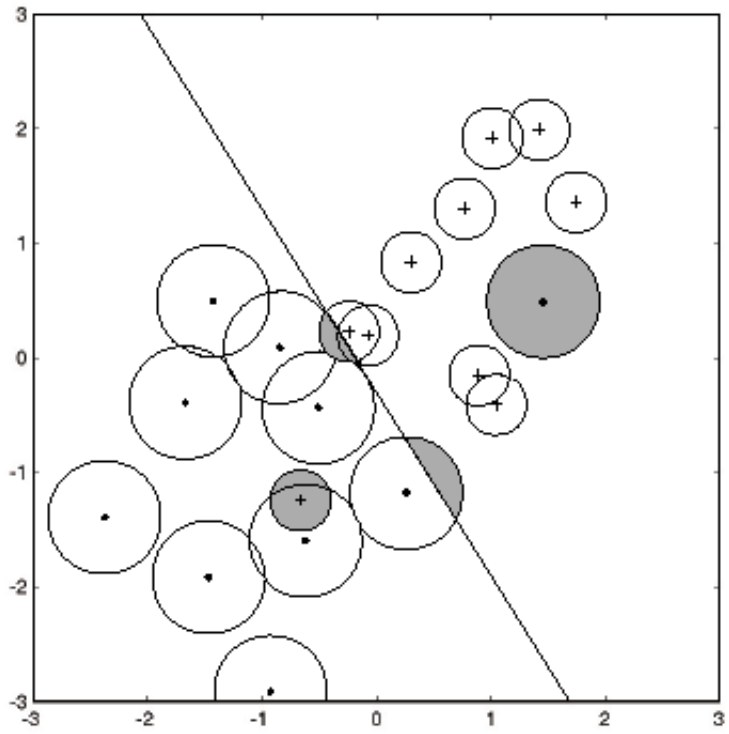
=> Rules out holdout estimator

Data-Efficient Error Estimators

- Several alternatives exist that use the training data itself to estimate classification error:
  - Apparent error
  - Cross-Validation
  - Bootstrap error estimators
  - Bolstered error estimators
- Of course, avoiding optimistic bias is always a concern, but variance and speed are also issues to consider.

Bolstered Resubstitution

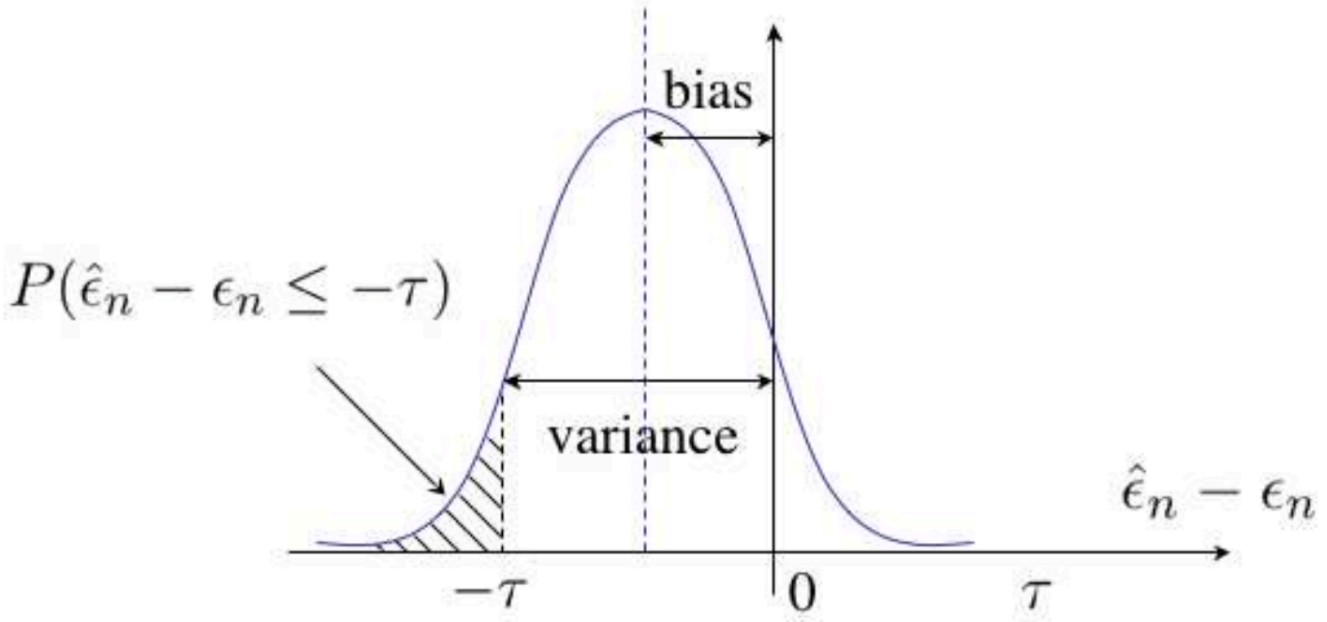
Example: Uniform circular bolstering kernels



Error estimate = shaded area divided by n

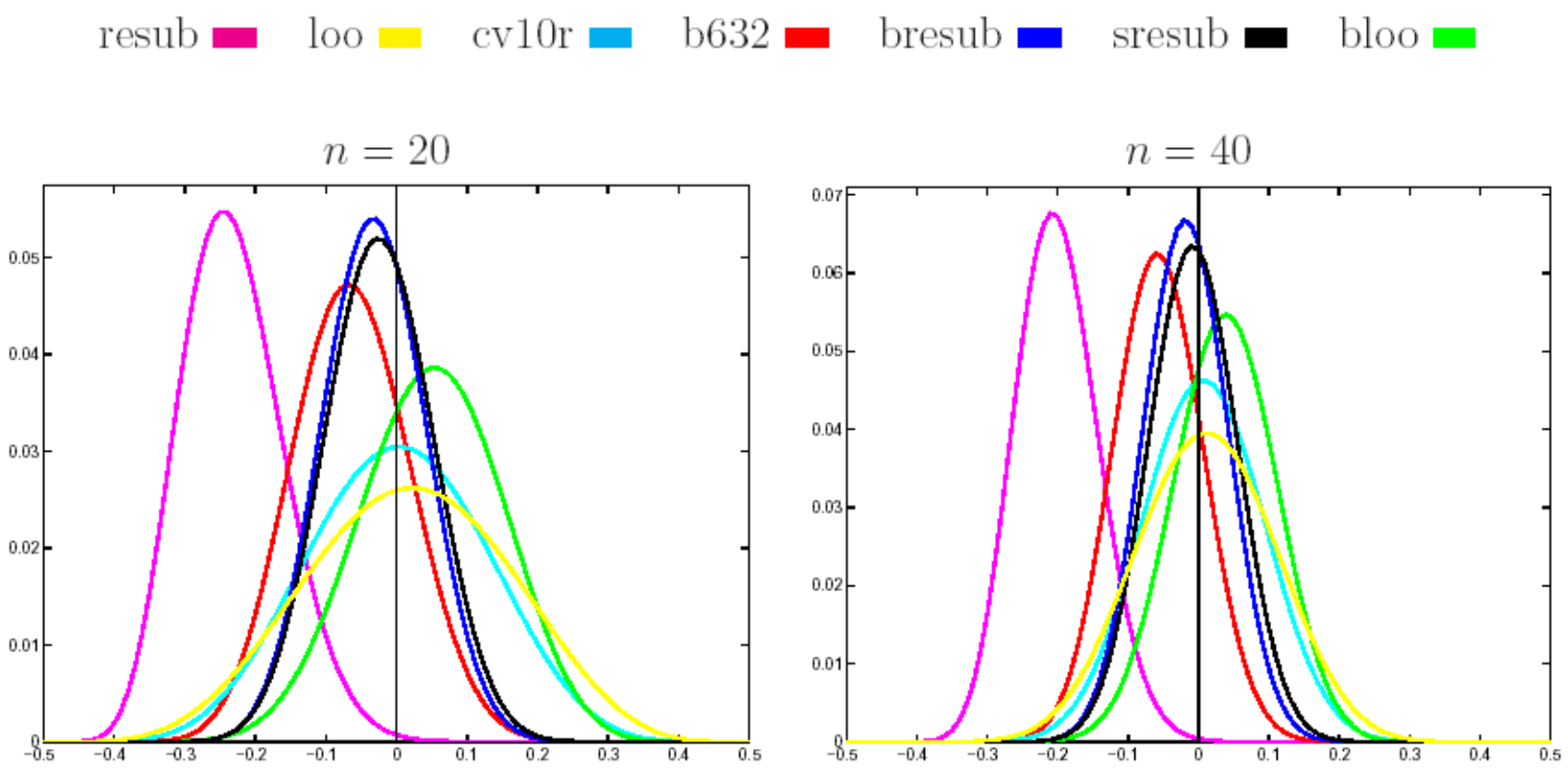
Comparison of Error Estimators

Of particular interest is the quantity $\hat{\epsilon}_n - \epsilon_n$, called the *deviation*. The distribution of this random variable is called the *deviation distribution*.

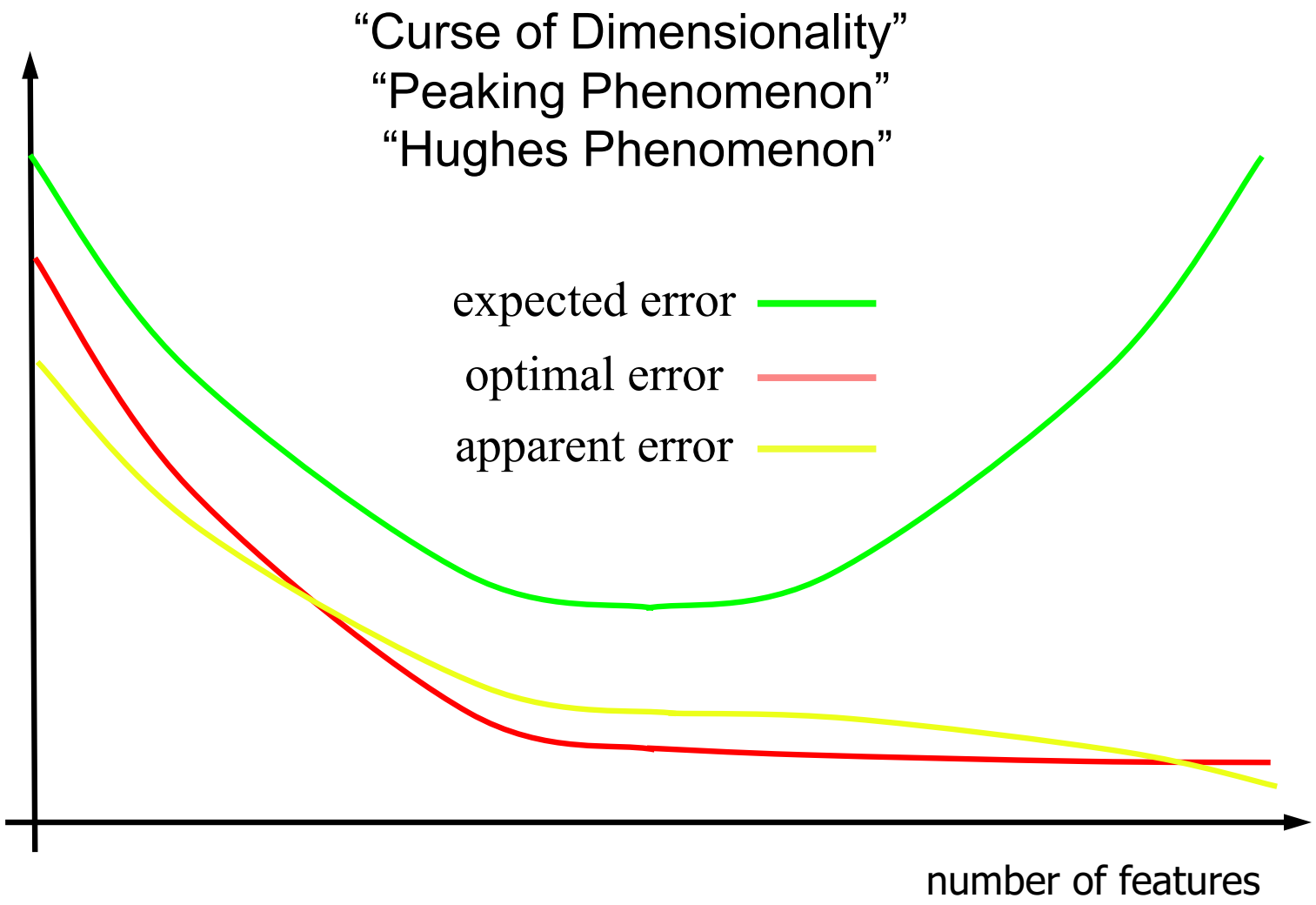


Comparison of Error Estimators

Deviation distribution for 5 genes and a tree classifier

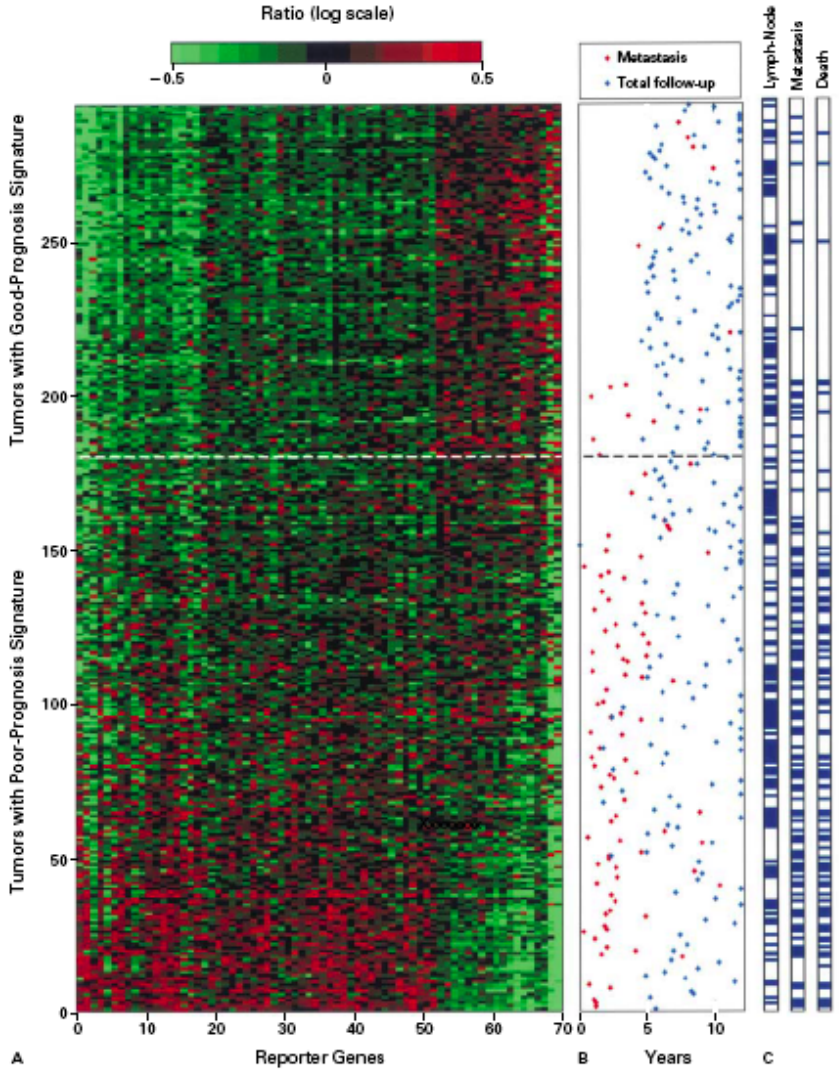


Feature Selection



Feature selection can improve accuracy.

Gene-Expression Example

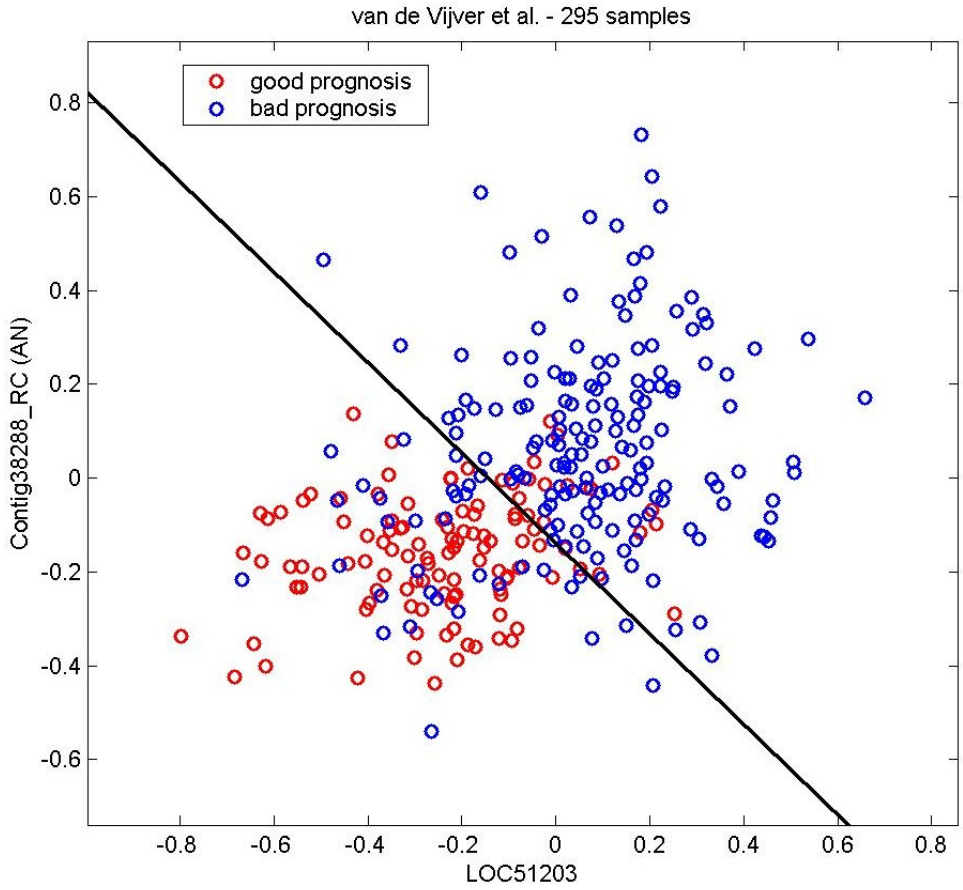


van de Vijver, *et al*. (2002) "A gene-expression signature as apredictor of survival in breast cancer." *New England Journal of Medicine*, Vol. 347, 1999–2009.

Originallly published 70-gene signature:

independent test-set error = 68/180 = 37.7%

More Accurate 2-Gene Classifier



U.M. Braga-Neto, Fads and Fallacies in the Name of Small-Sample Microarray Classification. IEEE Signal Processing Magazine, Special Issue on Signal Processing Methods in Genomics and Proteomics, Vol. 24, No. 1, January 2007, pp. 91-99.

Error ≈ 52/295 = 17.6%

Other Reasons for Feature Selection

- The objective may be to obtain an inexpensive diagnostic/prognostic kit to be applied for mass screening in hospital settings.
- Closely related is the issue that, in deployment of the diagnostic/prognostic methodology, it may not be technologically feasible or practical to measure a large number of variables.
- Scientists prefer to work with small sets of features that can be validated with traditional lab assays, and that can be helpful in uncovering the molecular mechanisms of disease.

Some Heuristics

Dimensionality reduction will generally involve loss of information. One typically wants to reduce the number of features in such a way that this loss is minimized. Some heuristics for this are:

- Features that are functions of other features should be discarded.
- Features that are nearly constant (small-variance) should be discarded.
- Features strongly correlated with Y should be retained.
- Features weakly correlated with Y (i.e., “noisy features”) should be discarded.

Exhaustive Search

- Let $J(A) = J(X^A, Y)$ be a class-separability criterion associated with A. The feature selection problem is to find A^* such that

$$A^* = \arg \max_{|A|=d} J(A)$$

- Since this is a finite problem, the optimal solution is guaranteed to be reached by *exhaustive search*: compute $J(A)$ for all possible subsets $A \subset \{1, \ldots, p\}$ of size d and pick the maximum. The number of subsets to be evaluated is clearly:

$$m = \binom{p}{d} = \frac{p!}{d!(p-d)!}$$

This number can be astronomical for even modest p and d (e.g., $p = 100$ and $d = 10$ give $m > 10^{13}$).

Filter vs. Wrapper Feature Selection

- The ultimate objective of feature selection is to provide a feature vector with which to design a classifier via a classification rule.
- If the criterion $J(A)$ is indepedent of the classification rule, the method is said to be a *filter approach*.
- Otherwise, the method is said to be a *wrapper approach*. Usually in this case one has as criterion: $J(X^A, Y) = 1 - \epsilon_n(X^A, Y)$ or, in practice, $J(X^A, Y) = 1 - \hat{\epsilon}_n$, where $\hat{\epsilon}_n$ is an *error estimator* for $\epsilon_n(X^A, Y)$, based on the training data S_n.
- The true performance of the selected feature set and classifier has to be assessed either with knowledge of F_{XY} or by means of a large independent test set.

Sub-Optimal Feature Selection

- In applications where the initial number of features is in the order of 1000's and 10000's, exhaustive searching is impractical.

- Sub-optimal (i.e., non-exhaustive) feature selection algorithms are based on heuristics to avoid the exponential complexity of having to search the entire space.

- However, the Cover-Van Campenhout Theorem implies that in the worst-case, the exponential complexity cannot really be avoided.

Sub-Optimal Feature Selection

- There are a number of fast feature selection algorithms that execute sub-optimal searches.
  - Best Individual d Features
  - Sequential Forward Search
  - Sequential Backward Search
  - Plus-l Take-r Search
  - Generalized Sequential Forward Search
  - Generalized Sequential Backward Search
  - Generalized Plus-l Take-r Sequential Search
  - Floating Search

Top Individual Features

- This is the simplest method: just compute $J(X_i, Y)$ for each individual original feature X_i, and pick the d features with largest J.
- This is an intuitive heuristic, but it can fail badly, as it ignores multivariate relationships.
- For a simple theoretical counter-example, consider the case where the best d features are equal: $X'_1 = \cdots X'_d$
- This method is nevertheless quite common. It is often based on the correlation of individual features with Y or scores such as the t-score or rank-sum score.
- Surprisingly, in small-sample cases, such simple filter methods can outperform more complex wrapper approaches.

Toussaint's Counter-Example

Even if all p original features are uncorrelated, the result of best-individual method can be very bad.

This surprising fact is shown, in the case of the Bayes error, by the following result.

(Thm 32.2 DGL) Let $p = 3$. There is a distribution of (X, Y) such that X_1, X_2 and X_3 are conditionally-independent given Y and

$$\epsilon^*(\{1\}) < \epsilon^*(\{2\}) < \epsilon^*(\{3\})$$

But such that

$$\epsilon^*(\{1,2\}) > \epsilon^*(\{1,3\}) > \epsilon^*(\{2,3\})$$

Therefore, the best 2 individual features form the worst 2-feature set, and the worst 2 individual features form the best 2-feature set.

Sequential Forward Search

Sequential methods generally outperform best individual feature selection (except in small-sample cases).

- Sequential Forward Search (bottom-up search):
  - Let $X_{(0)} = \emptyset$.
  - Given the current feature set $X_{(k)}$, the criterion $J(X_{(k)} \cup X_i, Y)$ is evaluated for each $X_i \notin X_{(k)}$ and the X_i^* that maximizes this is added to the feature set: $X_{(k+1)} = X_{(k)} \cup X_i^*$.
  - Stop if $k = d$ or if no improvement is possible.

This has the disadvantage that once a feature is added, it is "frozen" in place, i.e. it can never be removed from the working feature set.

Sequential Backward Search

- Sequential Backward Search (top-down search):
  - Let $X_{(0)} = X$.
  - Given the current feature set $X_{(k)}$, the criterion $J(X_{(k)} \setminus X_i, Y)$ is evaluated for each $X_i \in X_{(k)}$ and the X_i^* that minimizes the drop

$$J(X_{(k)}, Y) - J(X_{(k)} \setminus X_i, Y)$$

    is removed from the feature set: $X_{(k+1)} = X_{(k)} \setminus X_i^*$.
  - Stop at $k = d$.

The main disadvantage of this method is that feature sets of high dimensionality have to be considered. If the criterion J involves the classification error (e.g. wrapper feature selection), then this method is impractical for large p.

Generalized Sequential Search

- Generalized Sequential Forward Search: This is a generalization of sequential forward search, where at each stage, all combinations Z_j of r features not in the current feature set $X_{(k)}$ are considered, and the group Z_j^* that maximizes $J(X_{(k)} \cup Z_j, Y)$ is added: $X_{(k+1)} = X_{(k)} \cup Z_j^*$.

- Generalized Sequential Backward Search: This is a generalization of sequential forward search, where at each stage, all combinations Z_j of r features in the current feature set $X_{(k)}$ are considered, and the group Z_j^* that minimizes the drop $J(X_{(k)}, Y) - J(X_{(k)} \setminus Z_j, Y)$ is removed: $X_{(k+1)} = X_{(k)} \setminus Z_j^*$.

Plus-l Take-r Search

- This allows back-tracking in the sequential search.
- If $l > r$ this is a bottom-up search. At each stage, l features are added to the current feature set using SFS and then r features are removed using SBS.
- If $r > l$ this is a top-down search. At each stage, r features are removed from the current feature set using SBS and then l features are added using SFS.
- Generalized Plus-l Take-r Search: This uses GSFS and GSBS instead of SFS and SBS, respectively.

Floating Search

- This can be considered a development of the Plus-l Take-r Search method, where the values of l and r are allowed to vary, i.e., "float," at different stages of the feature selection process.
- The advantage of this method is that one is allowed to backtrack in an "optimal" sense.
- There is a bottom-up version (SFFS) and a top-down version (SFBS).