Model Assessment and Selection

Axel Benner
Biostatistics, German Cancer Research Center
INF 280, D-69120 Heidelberg
benner@dkfz.de

Topics

Predictive Ability using Resampling

- Data-Splitting/Cross-Validation
- The Bootstrap

Controlling Model Complexity

- Restriction
- Selection
- Regularization

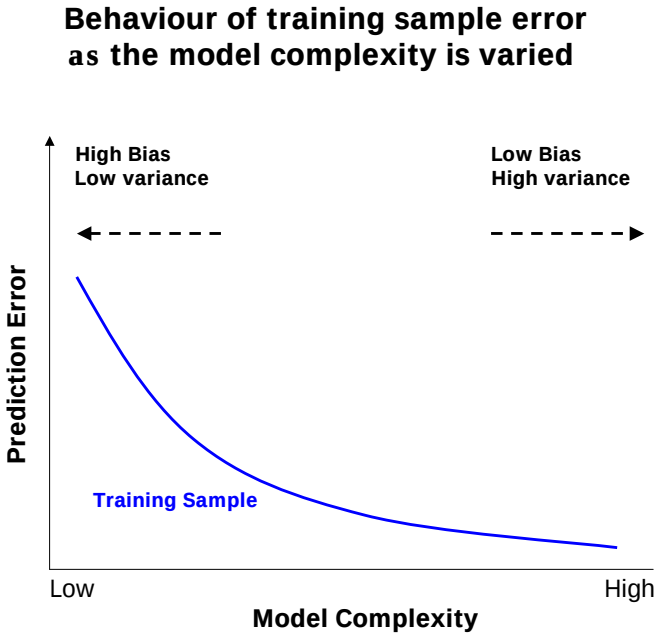
Model: A current approximation to complex relationships

Predictive Accuracy

- Some models are used only for hypothesis testing

- If used for prediction, need to consider accuracy of predictions
- Two major aspects of predictive accuracy that need to be assessed:
  - Reliability or calibration of a model:
    "ability of the model to make unbiased estimates of the outcome"
    ("observed responses agree with predicted responses")
  - Discrimination ability:
    "model is able, through the use of predicted responses, to separate subjects"

Major problem is overfitting

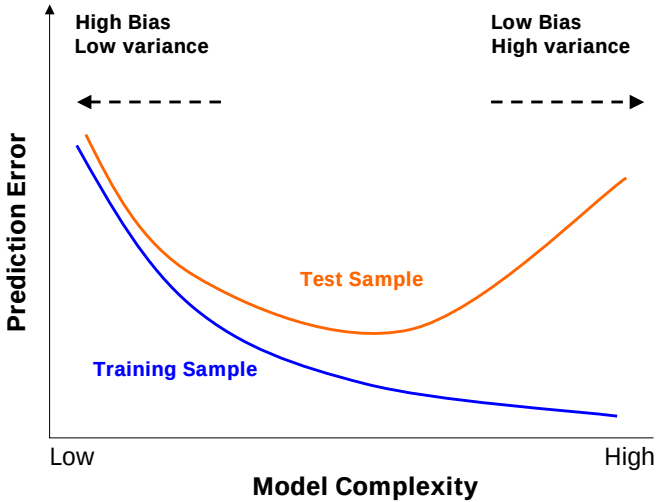


Need: Model assessment/validation to ascertain whether predicted values from the model are likely to accurately predict responses on future subjects or subjects not used to develop the model

- Two modes of validation
  - External:
    Use different sets of subjects for building the model (including tuning) and testing
  - Internal:
    (i) Apparent (evaluate fit on same data used to create fit)
    (ii) Data splitting and its extensions
    (iii) Resampling methods

- Naive approach: Use the entire training data to select our predictor/classifier and estimate the error rate
  - The naive approach has two fundamental problems
    - The final model will overfit the training data. This problem is more pronounced with models that have a large number of parameters.
    - The error rate estimate will be overly optimistic (lower than the true error rate). "In fact, it is not uncommon to have 100% correct classification on training data"
- A much better idea is to split the training data into disjoint subsets or use resampling methods

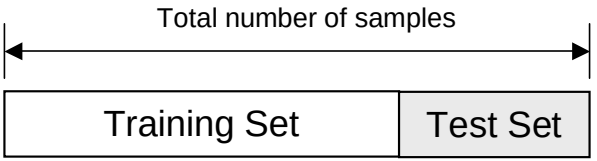
Behaviour of test and training sample error as the model complexity is varied



One-Time Data-Splitting

Split samples into two parts at random with balancing distributions of the response (and predictor variables)

- Training Set: Model development
- Test or Assessment Set: Measure predictive accuracy



Sometimes, a chronological split is used so that the validation is prospective.

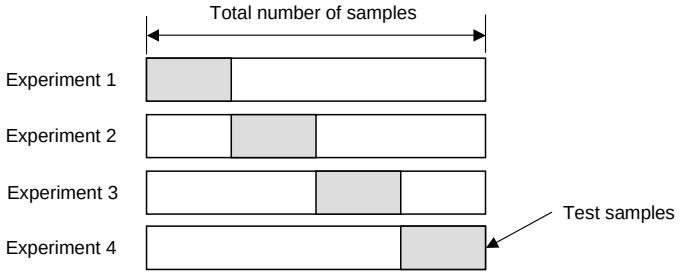
One-Time Data-Splitting

The one-time data-splitting method has two basic drawbacks

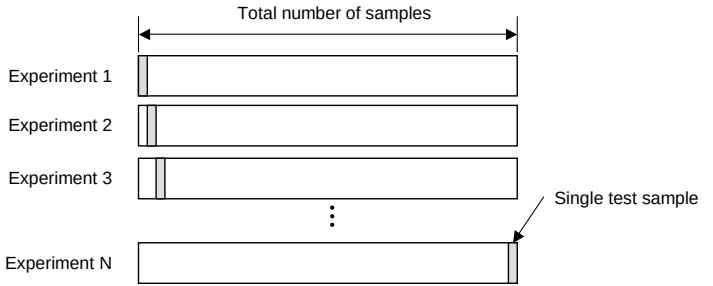
- In problems where we have a sparse data set we may not be able to afford the "luxury" of setting aside a portion of the data set for testing ("significant loss of power")
- The assessment can vary greatly when taking different splits. Since it is a single train-and-test experiment, the estimate of the error rate will be misleading if we happen to get an "unfortunate" split.

Cross-Validation

- K-fold cross-validation



- Leave-One-Out cross-validation



Results are pooled from all test sets to estimate performance of the model (each case is used exactly once).

(Aggregate) Prediction Error

General notation:

- Let y denote the response variable and x the covariate vector.
- Let y_+ and x_+ denote response and covariate values for a new case.
- Measure the prediction error by loss function: $c(y_+, \hat{y}_+)$
- Prediction rule: $\hat{y}_+ = \mu(x_+, \hat{F})$
  where $\hat{F}$ is the empirical distribution function sampled from distribution F
- (Aggregate) prediction error

$$D = D(F, \hat{F}) = E[c(Y_+, \mu(X_+, \hat{F}))|\hat{F}]$$

$\longrightarrow$ Use an estimate of D, say $\Delta = \Delta(F)$

Misclassification error (two groups)

- Suppose a response y which is equal 1 or 0.
- The prediction rule $\mu(x_+, \hat{F})$ is an estimate of $P(Y_+ = 1|x_+)$ for a new case (x_+, y_+).
- Set $\hat{y}_+ = 1$ if $\mu(x_+, \hat{F}) \geq 0.5$ and $\hat{y}_+ = 0$ otherwise.
- If misclassifications costs are equal, the misclassification loss function is

$$c(y_+, \hat{y}_+) = \begin{cases} 1, & y_+ \neq \hat{y}_+ \\ 0, & \text{otherwise} \end{cases}$$

- The aggregate prediction error D is then the overall misclassification rate, equal to the proportion of cases where y_+ is wrongly predicted.

- Apparent error (resubstitution error):

  Use the same data for prediction which was used for fitting the model

$$\Delta_{app} = D(\hat{F}, \hat{F}) = \frac{1}{n} \sum_{i=1}^{n} c(y_i, \mu(x_i, \hat{F}))$$

  Δ_{app} *underestimates the true* Δ *("it is downwardly biased")*

- Leave-one-out Cross-Validation

  Training sets of size $(n-1)$ are taken and prediction rule is tested for a single observation:

$$\Delta_{cv} = \frac{1}{n}\sum_{i=1}^{n} c(y_i, \mu(x_i, \hat{F}_{-i}))$$

  where $\hat{F}_{-i}$ represents the data excluding the i-th case.

  Note the small bias of leave-one-out cv:
  "It differs from Δ by terms of order n^{-2} (whereas the apparent error differs by terms of order n^{-1})".

- K-fold Cross-Validation

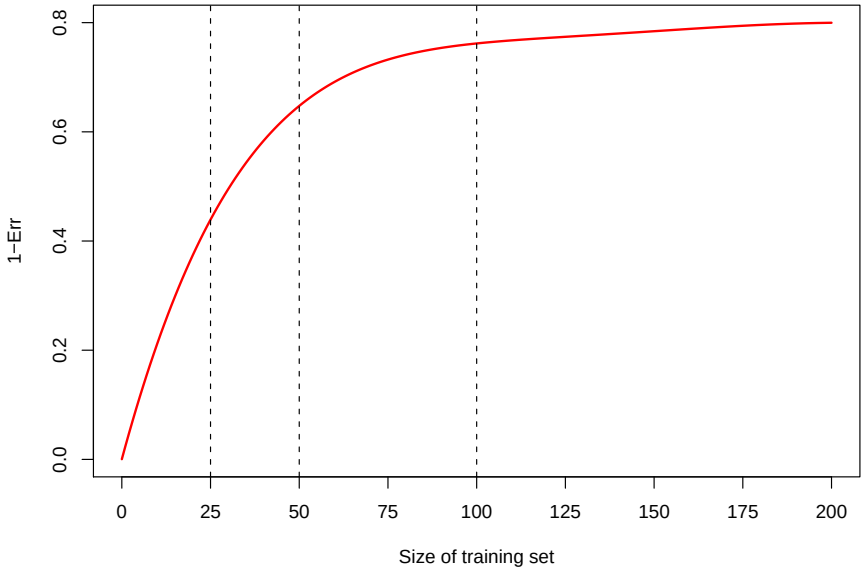
  Since the n training sets are so similar to one another this can make $\hat{\Delta}_{cv}$ too variable

  $\rightarrow$ leave out groups of observations; especially K disjoined groups

$$\Delta_{cv,K} = \frac{1}{n}\sum_{i=1}^{n} c(y_i, \mu(x_i, \hat{F}_{-k(i)}))$$

  where $\hat{F}_{-k(i)}$ represents the data excluding the group containing the i-th case.

- K-fold Cross-Validation: Training set size bias



Hypothetical learning curve: The performance of the predictor improves as the training set size increases to about 100 observations.

Increasing this number further brings only a small benefit.

- Leave-one-out vs. K-fold Cross-Validation

	Advantages	Disadvantages
Leave-One-Out (K=n)	Effective use of data Nearly unbiased estimate of the prediction error	Computationally expensive High variance
K-fold	Lower variance than Leave-One-Out	Training set size bias Overestimates prediction error

- K-fold Cross-Validation (cont)
  - Good strategy: Take $K = min(\sqrt{n}, 10)$
    A size of at least $\sqrt{n}$ should perturb the data sufficiently to give small variance.
  - Problem: increasing bias (especially if K is small)!
    "overestimation of the prediction error depending on the training sample size"

    $\rightarrow$ Reduce bias by adjustment:

    Denote by $\hat{F}_{-k}$ the data with the k-th group omitted, $k = 1, ..., K$, and let p_k denote the proportion of the k-th group in the data set.
    $\rightarrow$

$$\Delta_{acv,K} = \Delta_{cv,K} \; + \; D(\hat{F}, \hat{F}) - \sum_{k=1}^{K} p_k D(\hat{F}, \hat{F}_{-k})$$

K-fold Adjusted Cross-Validation

1. Fit the regression model to all cases, calculate predictions $\hat{y}_i$ from that model, and average the values of $c(y_i, \hat{y}_i)$ to get $D(\hat{F}, \hat{F})$.
2. Choose group sizes $m_1, ..., m_K$ such that $m_1 + ... + m_K = n$.
3. For $k = 1, ..., K$
   (a) choose C_k by sampling m_k times without replacement from {1,2,...,n} minus elements chosen for previous C_is
   (b) fit the regression model to all data except cases $i \in C_k$
   (c) calculate new predictions $\hat{y}_i = \mu(x_i, \hat{F}_{-k})$ for $i \in C_k$
   (d) calculate predictions $\hat{y}_{ki} = \mu(x_i, \hat{F}_{-k})$ for all i; then
   (e) average the n values $c(y_i, \hat{y}_{ki})$ to give $D(\hat{F}, \hat{F}_{-k})$.
4. Average the n values of $c(y_i, \hat{y}_i)$ using $\hat{y}_i$ from step 3(c) to give $\hat{\Delta}_{cv,K}$.
5. Calculate $\Delta_{acv,K} = \Delta_{cv,K} + D(\hat{F}, \hat{F}) - \sum_{k=1}^{K} p_k D(\hat{F}, \hat{F}_{-k})$ with $p_k = m_k/n$.

Drawbacks of Cross-Validation

- Leave-one-out cv: may have large variance

- K-fold cv: may have large bias, depending on the choice of the number of observations to be hold out from each fit.

  If the learning curve has a considerable slope at the given training set size, 5 or 10-fold cv will strongly overestimate the true prediction error.

Estimate Prediction Error (Bootstrap)

The bootstrap estimate of the prediction error is

$$\hat{\Delta} = \Delta(\hat{F}) = E(D(\hat{F}, \hat{F}^{*b}))$$

where $\hat{F}^{*b}$ denotes a bootstrap sample $(x_1^{*b}, y_1^{*b}), \ldots, (x_n^{*b}, y_n^{*b})$ of the original data.

Now the prediction rule is fitted to these data resulting in predictions $\mu(x_i, \hat{F}^{*b})$ of y_i.

Using a loss function $c(\cdot)$ $\hat{\Delta}$ is then approximated by

$$\hat{\Delta}_b \;=\; \frac{1}{B}\sum_{b=1}^{B}\frac{1}{n}\sum_{i=1}^{n} c(y_i, \mu(x_i, \hat{F}^{*b}))$$

derived by fitting the model on a set of bootstrap samples, and comparing its predictions with the original data.

Problem: Bootstrap sample act as training sample, and original training set act as test set.

Both samples have observations in common

$\longrightarrow$ overoptimistic estimate due to overfitting

$\longrightarrow$ underestimates the error

- Alternative 1: Leave-one-out bootstrap estimate of prediction error

$$\hat{\Delta}_{bcv} = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{|B_{-i}|}\sum_{b\in B_{-i}} c(y_i, \mu(x_i, \hat{F}^{*b}))$$

B_{-i} is set of indices that does not contain observation i and $|B_{-i}|$ is the size of this set.

Note that $|B_{-i}|/B$ is approximately equal to $e^{-1} = 0.368$

$\hat{\Delta}_{bcv}$ is a bootstrap smoothing of the leave-one-out cv.

$\longrightarrow$ overfitting no problem, but (like cv) bias by training set size.

$\longrightarrow$ possibly overestimates error rate.

Example: 6 bootstrap samples

original data	1	2	3	4	5
bootstrap sample 1	**1**	**1**	**3**	**4**	**4**
bootstrap sample 2	1	2	2	3	5
bootstrap sample 3	**1**	**3**	**3**	**3**	**4**
bootstrap sample 4	**3**	**4**	**4**	**5**	**5**
bootstrap sample 5	2	2	3	4	4
bootstrap sample 6	1	1	2	4	5

Now bootstrap samples 1,3,and 4 do not include observation 2.
And so we get: $B_{-2} = \{1, 3, 4\}$ with $|B_{-2}| = 3$.

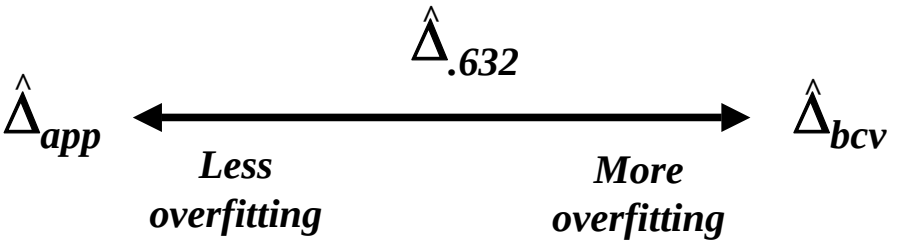
- Alternative 2: ".632" bootstrap estimate of prediction error

$$\hat{\Delta}_{.632} \;=\; .368\hat{\Delta}_{app} \;+\; .632\hat{\Delta}_{bcv}$$

where $\hat{\Delta}_{app}$ is the apparent error estimate

pulls leave-one-out down toward training error

$\longrightarrow$ may underestimate the error in overfitting situations



- Problem: $\hat{\Delta}_{.632}$ can break down in overfitted situation

  $\rightarrow$ take into account amount of overfitting.

  $\rightarrow$ put relatively more weight on $\hat{\Delta}_{bcv}$

  $\rightarrow$ ".632+" bootstrap estimate of prediction error

  This estimate was proposed by Efron & Tibshirani (JASA, 1997) for highly overfit rules like nearest neighbors.

- Alternative 3: ".632+" bootstrap estimate of prediction error

$$\hat{\Delta}_{.632+} \;=\; (1-\hat{w})\cdot\hat{\Delta}_{app} \;+\; \hat{w}\cdot\hat{\Delta}_{bcv}$$

where the weight w is given by

$$\hat{w} \;=\; \frac{.632}{1-.368\hat{R}}\,.$$

and where

$$\hat{R} \;=\; \frac{\hat{\Delta}_{bcv}-\hat{\Delta}_{app}}{\hat{\gamma}-\hat{\Delta}_{app}}$$

is the "relative overfitting rate".

γ denotes the "no information error rate" that would apply if input and output are independent

$$\hat{\gamma} = \frac{1}{n^2}\sum_{i=1}^{n}\sum_{i'=1}^{n} c(y_i, \mu(x_{i'}, \hat{F}^{*b}))$$

- For classification problems γ is estimated by

$$\hat{\gamma} = \sum_{i=1}^{g} p_i(1 - q_i)$$

  where g is the number of different classes, p_i is the proportion of the original cases from the ith class (= prior probability of class i), and q_i is the proportion assigned to the ith class by the prediction rule (= posterior probability of class i).

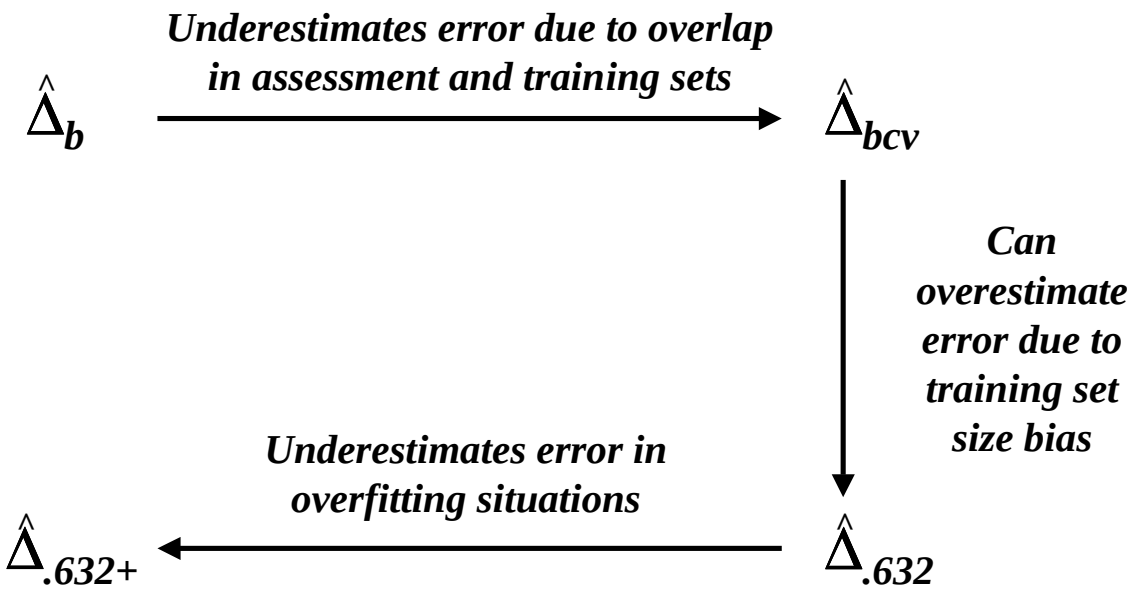
- $\hat{R}$ may have to be truncated to not fall outside $[0, 1]$.

- w varies from 0.632 ($\hat{R} = 0$) to 1.0 ($\hat{R} = 1$).

- $\hat{\Delta}_{.632+}$ puts more weight on the bootstrap leave-one-out error $\hat{\Delta}_{bcv}$ when the amount of overfitting ($\hat{\Delta}_{bcv} - \hat{\Delta}_{app}$) is large.

- Use $\hat{\Delta}_{.632+}$ if the prediction rule is overfit due to feature selection.

Bootstrap overview



- $\hat{\Delta}_{bcv}$ is based on $\approx .632 \times n$ of the original cases and closely agrees with the "half-sample" cross-validation and therefore it is upwardly biased.

  $\rightarrow$ Use ".632" estimator (Efron, 1983) to correct for the upward bias in $\hat{\Delta}_{bcv}$ with the downward bias in the apparent error estimate.

- In gene expression studies the prediction rule is an overfit formed from a large number of genes relative to the number of cases.

  $\rightarrow$ Use ".632+" estimator (Efron & Tibshirani, 1997) to put relatively more weight on $\hat{\Delta}_{bcv}$ (preferably in overfit situations like feature selection).

Misclassification error (two groups)

- The prediction $\mu(x_+, \hat{F})$ and the measure of error $c(y_+, \hat{y}_+)$ are not continuous functions of the data.

  $\rightarrow$
  bootstrap methods for estimating D or its expected value Δ are superior to cross-validation methods, in terms of variability.

Variable/Gene Selection

- Model/variable selection implies that there is some likelihood of a "true" model,

  some pre-specified variables have zero association with response Y

- Need to perform gene selection preceding the predictive modelling

  $\longrightarrow$ e.g. eliminate variables whose distributions are too narrow.

Variable/Gene Selection (cont)

- Gene filtering is helpful, but

  estimating the error rate after variable selection leads to biased estimates of the prediction error
  $\rightarrow$ overstating importance of variables which are retained in the model.

- Make sure that you are cross-validating the experiment that you have carried out,
  in particular, if you are selecting genes, rather than working with known genes, you must cross-validate the gene selection process as well.

- There are many examples with low classification error rates which do not cross-validate properly (model/gene selection was not validated).

Ambroise & McLachlan (PNAS, 2002): Selection bias in gene extraction on the basis of microarray gene-expression data

> ... it seems that the selection method and the number of selected genes are more important than the classification method for constructing a reliable prediction rule.
>
> ... it is important to correct for the selection bias in estimating the prediction error for a rule formed by using a subset of genes selected from a very large set of available genes.

Prediction error in gene selection situations

Example: The expression set Huang.RE which is discussed in THE LANCET (2003) 361:1590-1596. The data contains microarrays of 52 women with breast cancer of whom 34 did not experience a recurrence of the tumour during a 3 years time period.

For simplicity select 1000 most variable probe sets (e.g. by largest variability) for the exercises (data frame `mydata`)

```
library(affy)
sd.exp <- apply(exprs(Huang.RE),1,sd)
index <- order(sd.exp,decreasing=TRUE)[1:1000]

mydata <- data.frame(t(exprs(Huang.RE)[index,]),
                     Recurrence=as.factor(pData(Huang.RE)$Recurrence))
```

Now we select probe sets by comparing their univariate p-values of a two-sample t-test with a pre-specified level of 0.05 and train a LDA using the selected probe sets only (function `mymod`).

```
mymod <- function(formula, data, level = 0.05) {
  sel <- which(lapply(data, function(x) {
     if (!is.numeric(x))
          return(1)
     else return(t.test(x ~ data$Recurrence)$p.value)
  }) < level)
  sel <- c(which(colnames(data) %in% "Recurrence"), sel)
  mod <- lda(formula, data = data[, sel])
  function(newdata) {
  predict(mod, newdata = newdata[, sel])$class
 }
}
```

The **.632+ bootstrap** estimate of the prediction error using B=25 bootstrap samples gives a misclassification rate of 0.27.

```
library(ipred)
set.seed(71003)
errorest(Recurrence ~ ., data=mydata, model=mymod, estimator="632plus",
                         est.para=control.errorest(nboot=25))
```

```
errorest.data.frame(formula=Recurrence ~ ., data=mydata,
    model=mymod, estimator="632plus", est.para=control.errorest(nboot=25))

        .632+ Bootstrap estimator of misclassification error
        with 25 bootstrap replications

Misclassification error:  0.2705
```

Define a gene expression set of 1000 genes with no association to the response

```
set.seed(63321)
mydata <- data.frame(matrix(rnorm(52*1000),52,1000),
                     Recurrence=as.factor(pData(Huang.RE)$Recurrence))
```

1. Select genes by individual t tests (selection level 0.05), perform a **lda** using the selected subset and compute estimate of the misclassification error (ignoring the selection process)

```
sel <- which(lapply(mydata, function(x) {
     if (!is.numeric(x)) return(1)
     else return(t.test(x ~ mydata$Recurrence)$p.value)
}) < 0.05)
sel <- c(which(colnames(mydata) %in% "Recurrence"), sel)
mypredict.lda <- function(object, newdata) {
 predict(object, newdata = newdata)$class
}
errorest(Recurrence ~ ., data = mydata[, sel],
 model = lda, estimator = "632plus", predict = mypredict.lda)
```

```
Call:
errorest.data.frame(formula = Recurrence ~ ., data = mydata[,
    sel], model = lda, predict = mypredict.lda, estimator = "632plus")

        .632+ Bootstrap estimator of misclassification error
        with 25 bootstrap replications

Misclassification error:  0.1005
```

2. Now repeat the error estimation taking into account the gene selection by individual t tests (using 25 bootstrap samples)

```
errorest(Recurrence ~ ., data=mydata, model=mymod,
    estimator="632plus", est.para=control.errorest(nboot=25))
```

```
Call:
errorest.data.frame(formula = Recurrence ~ ., data = mydata,
    model = mymod, estimator = "632plus",
    est.para = control.errorest(nboot = 25))

        .632+ Bootstrap estimator of misclassification error
        with 25 bootstrap replications

Misclassification error:  0.3447
```

3. Finally repeat the error estimation taking into account the gene selection by individual t tests (using 7-fold cross validation)

```
errorest(Recurrence ~ ., data=mydata, model=mymod,
      estimator="cv", est.para=control.errorest(k=7))

Call:
errorest.data.frame(formula = Recurrence ~ ., data = mydata,
    model = mymod, estimator = "cv", est.para = control.errorest(k = 7))

        7-fold cross-validation estimator of misclassification error

Misclassification error:  0.4231
```

Note: The true misclassification rate is $1 - 34/52 = 0.346$.

Result of this example:
Ignoring the selection process results in an error estimate of 10%. The **.632+ bootstrap** estimate of 0.345 is nearly correct, while the **7-fold cross-validation** overestimates the error.

From: Ambroise & McLachlan (PNAS, 2002)

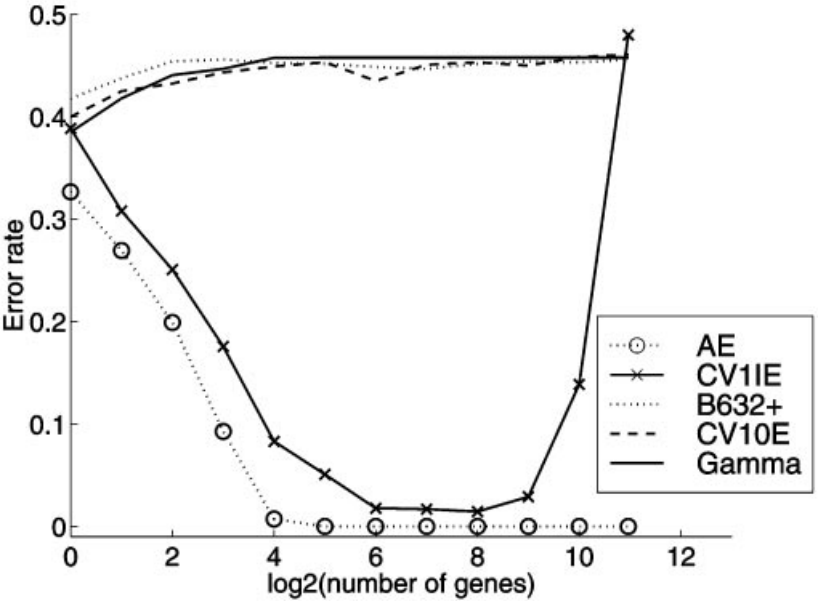
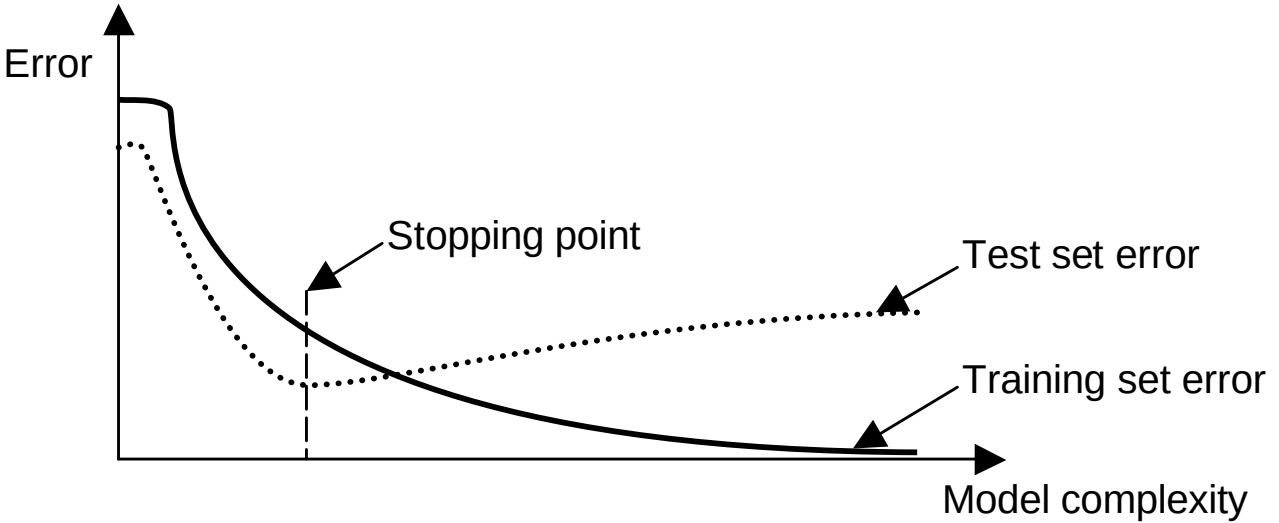


Fig. 5. Error rates of the SVM rule averaged over 20 noninformative samples generated by random permutations of the class labels of the colon tumor tissues.


Error
Stopping point
Test set error
Training set error
Model complexity

Separate Test and Validation Sets

- The error rate estimate of the final model on test data will be biased since the test set is used to select the final model
- If model selection and error estimates are to be computed simultaneously, we need an additional validation set:
  - Training set:
    Used to fit the model
  - Test (or assessment set):
    Used to tune the parameters of a predictor
  - Validation set:
    A set of cases used only to assess the performance of a fully-trained predictor.

  After assessing the final model with the validation set, YOU MUST NOT further tune the model.

Topics

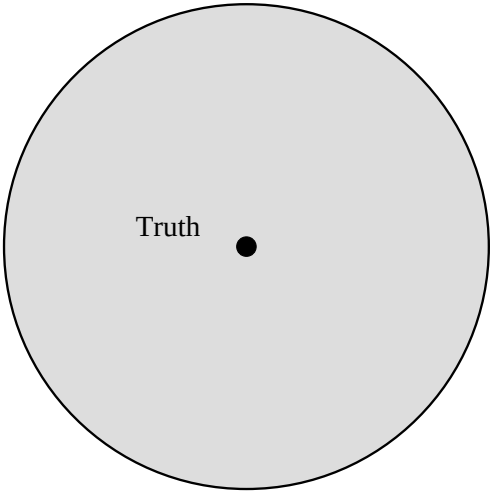
Predictive Ability using Resampling

- Data-Splitting/Cross-Validation
- The Bootstrap

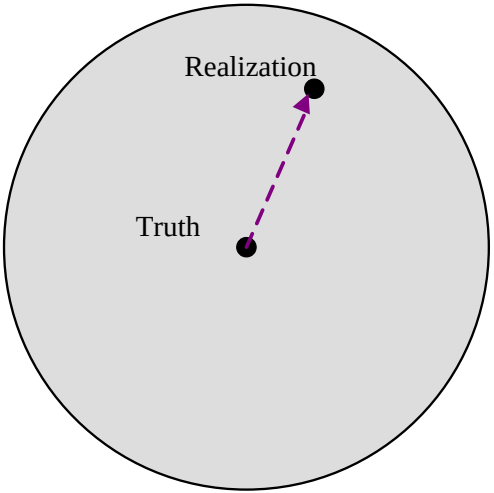
Controlling Model Complexity

- Restriction
- Selection
- Regularization

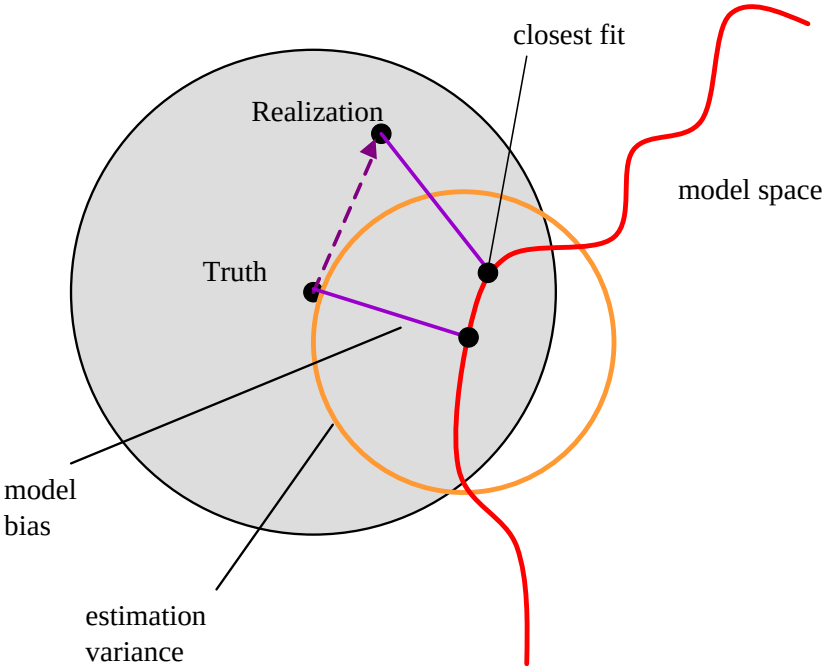
Controlling the complexity of the model



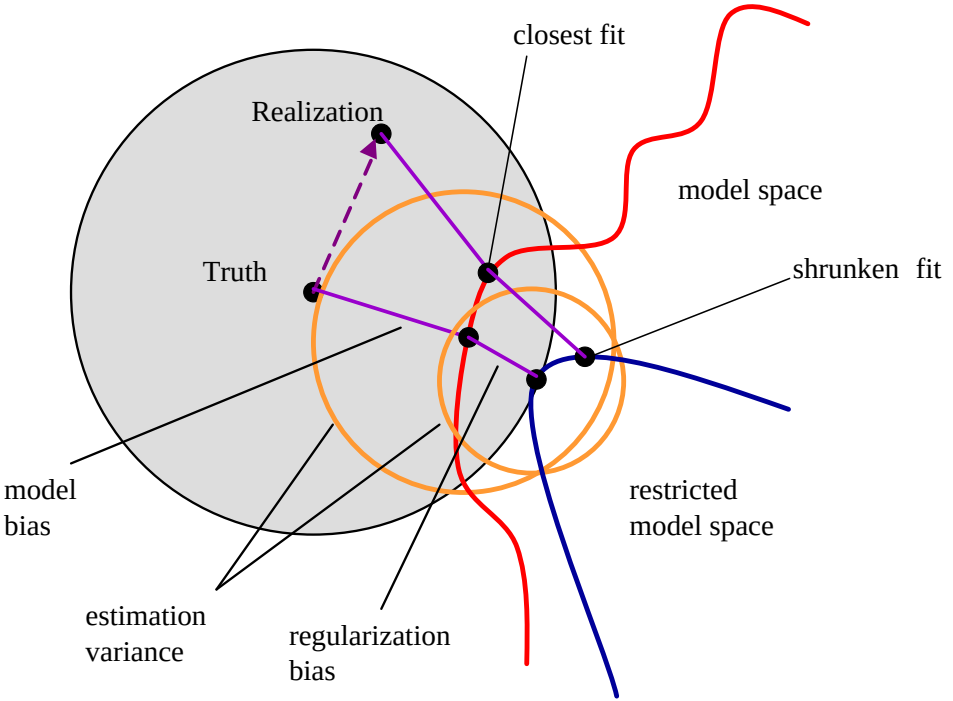
Hastie, Tibshirani, Friedman, 2001



Hastie, Tibshirani, Friedman, 2001



Hastie, Tibshirani, Friedman, 2001



Hastie, Tibshirani, Friedman, 2001

Controlling the complexity of the model

- **Restriction methods**

  The class of functions of the input variables defining the model is limited.

  Example:
  Allow only linear combinations of given basis functions h_{jm}

$$f(X) = \sum_{j=1}^{G} f_j(X_j) = \sum_{j=1}^{G} \sum_{m=1}^{M_j} \beta_{jm} h_{jm}(X_j)$$

  h_{jm} is the m^{th} basis function of the j^{th} input variable.

  The size of the model is limited by the number M_j of basis functions used for the G components f_j.

Controlling the complexity of the model

- **Selection methods**

  Include only those basis functions h_{jm} that contribute 'significantly' to the fit of the model.

  Examples:

  – Variable selection methods

  – Stagewise greedy approaches like boosting

- **Regularization methods**

  Restrict the coefficients of the model.

  Example: Ridge regression

Penalized Regression

- Maximizing the log likelihood can result in fitting noise in the data.
- A shrinkage approach will often result in estimates of the regression coefficients that, while biased, are lower in mean squared error and are more close to the true parameters.
- A good approach to shrinkage is penalized maximum likelihood estimation (le Cessie & van Houwelingen, 1990).

  From the log-likelihood $logL$ a so-called 'penalty' is subtracted, that discourages regression coefficients to become large.

  $\rightarrow$ penalized log likelihood:

$$logL - \lambda \cdot p(\beta)$$

  $p(\beta)$ penalty function, λ non-negative penalty factor.

Penalized Regression

- Often used penalty function:
  Quadratic regularization (ridge regression)

$$p(\beta) = \frac{1}{2}\sum_{j=1}^{G}\beta_j^2$$

- Harrell (2001):
  Use scaling constants $s_1, s_2, ..., s_G$, chosen to make $s_j\beta_j$ unitless

$$p(\beta) = \frac{1}{2}\sum_{j=1}^{G}(s_j\beta_j)^2$$

Choice of λ

- AIC (Akaike's Information Criterion):

$$-2logLik + 2d.f.$$

For a given λ the effective number of parameters being estimated is reduced because of shrinkage $\rightarrow$ effective degrees of freedom (cp. Gray, 1992)

$$d.f. = trace(I(\beta^p) \cdot V(\beta^p))$$

β^p is penalized MLE; I is information matrix ignoring the penalty and V is covariance matrix computed by inverting the information matrix that included the second derivatives with respect to β in the penalty.

- Alternative: Cross validation (and smoothing on the pairs $\{\lambda, \text{predictive accuracy}\}$).

Applications

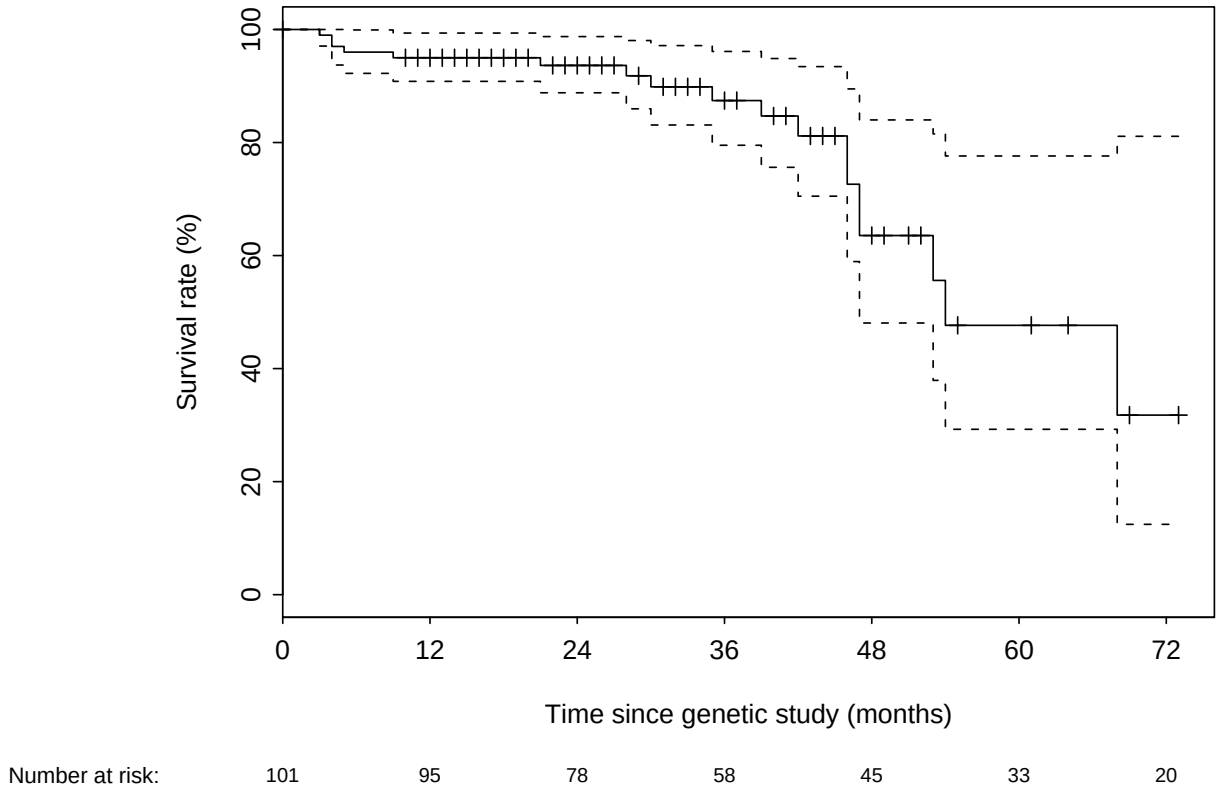
- **Data**
  Blood samples of a subset of 101 patients, randomly selected from a study of 325 B-cell chronic lymphocytic leukemia (B-CLL) patients
  - Affymetrix Human Genome U95 array (version 1 and 2):
    Gene expression values for 12600 probe sets
  - Array normalization:
    Robust Multichip Average (RMA) procedure (Irizarry et al., 2003).

  Median follow up was 28 months since date of the genetic study.

  A prognostic survival model was build using clinical data, molecular cytogenetic data and microarray measurements.

  For demonstration purposes we use only genes (i.e. probe sets) as input variables, thus ignoring cytogenetic and clinical features.

Survival since date of genetic study of 101 B-CLL patients.



Median survival 54 months (18 patients died).

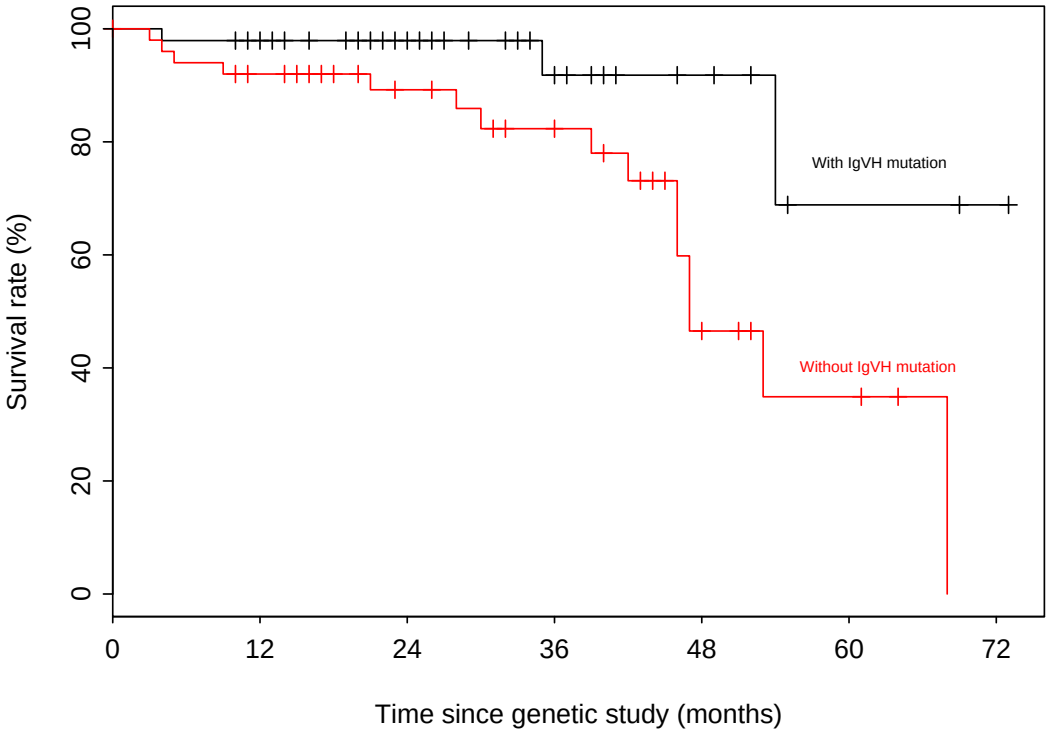
Classification

- Immunoglobulin variable heavy chain (IgVH) gene mutation status is a strong prognostic marker for survival of patients with B-CLL Patients without gene mutation of the IgVH region $\rightarrow$ worse prognosis.

- Task: Discriminate observations into the two categories characterized by the IgVH mutation status by using gene expression data. The outcome variable of interest is $y = 1$ if no IgVH mutation was observed and zero otherwise.

- For two of the patients no information on the IgVH mutation status was available.

Survival since date of genetic study of 99 B-CLL patients according to their IgVH mutation status.



Number at risk:

With IgVH mutation	48	45	31	14	6	2	1
Without IgVH mutation	51	42	28	20	7	3	0

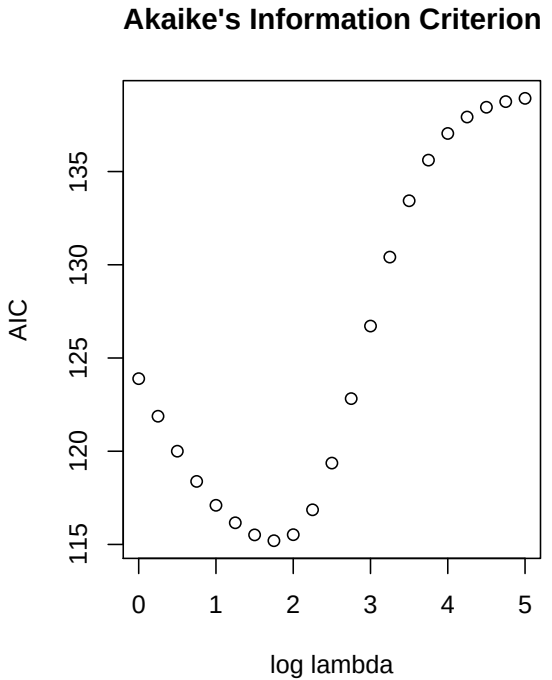
Penalized Regression ($p >> n$)

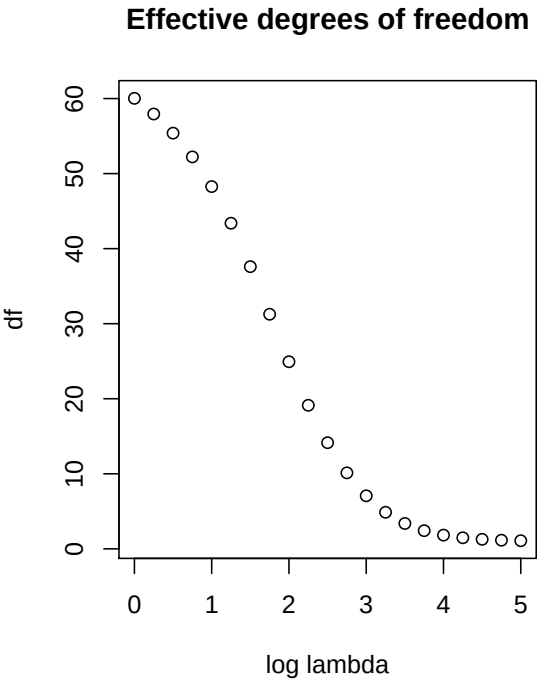
- The dimension of the resulting systems of equations is of the size of numbers of genes

- Can be reduced using singular value decomposition to a size corresponding to the (much smaller) number of observations (Eilers et al., 2001).

Binomial log-likelihood loss and L_2-penalty

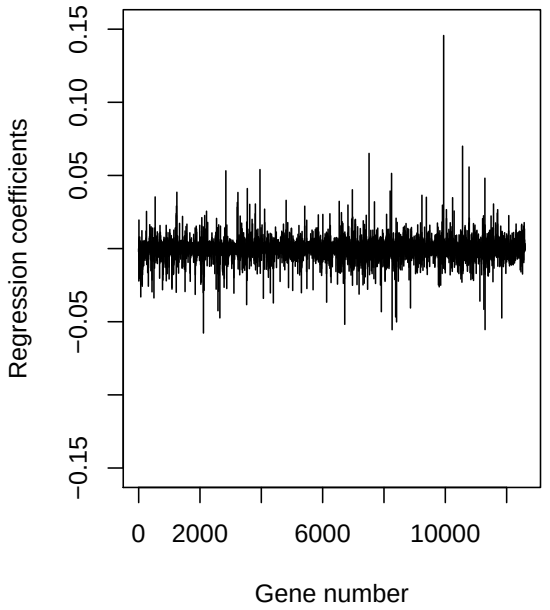
$$logL_\lambda \;=\; \sum_{i=1}^{n} y_i log(p_i) + \sum_{i=1}^{n}(1-y_i)log(1-p_i) \;+\; \tfrac{1}{2}\lambda \sum_{j=1}^{G} \beta_j^2$$

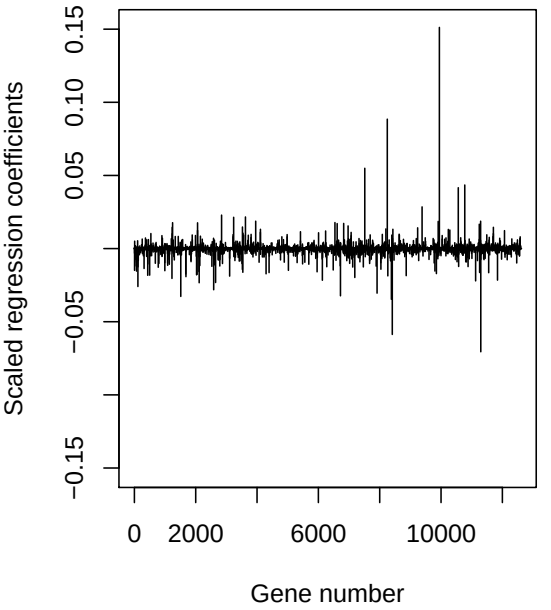


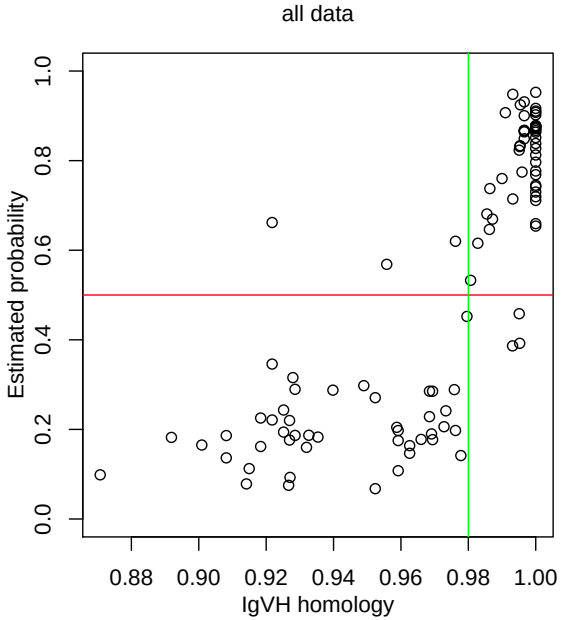


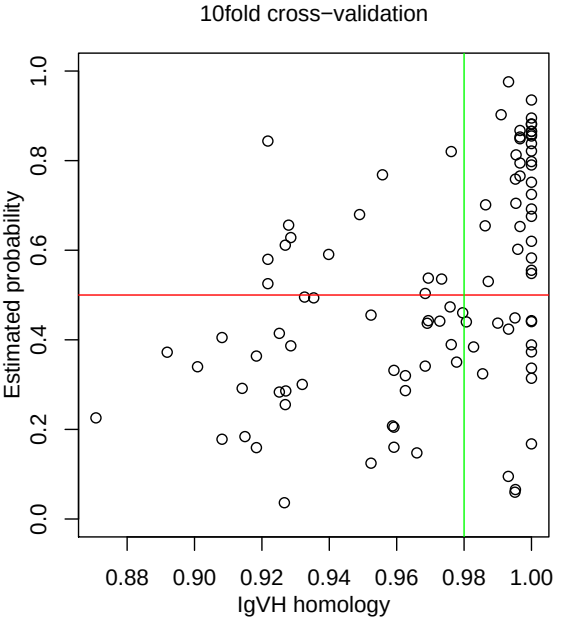
Minimum AIC at $\log(\lambda)$=1.75 (effective d.f.=31.26)

(Unscaled and scaled) parameter estimates









		IgVH mutation	
		no	yes
All data	no	45	3
	yes	3	48

		IgVH mutation	
		no	yes
10-fold cv	no	35	16
	yes	13	35

References

- Ambroise C, McLachlan GJ (2002). Selection bias in gene extraction on the basis of microarray gene-expression data. Proc. Natl. Acad. Sci. USA, 99, 6562-6566.
- Davison AC, Hinkley DV (1997). Bootstrap Methods and Their Applications. Cambridge University Press.
- Harrell FE (2001). Regression Modeling Strategies. Springer-Verlag.
- Hastie T, Tibshirani R, Friedman JH (2001). The Elements of Statistical Learning. Springer-Verlag.

R Packages

- Prediction: `ipred`, `pamr`, `Design`.
- Bootstrap: `boot`, `Design`.