

# The estimation of prediction error for nonparametric regression: A simulation study.

A comparison of cross-validation, bootstrap and hold-out  
methods to Measure the Prediction Error.

Simone Borra – University of Rome “Tor Vergata”

Agostino Di Ciaccio – University of Rome “Sapienza”



# Aim of this communication

- ❑ We examine and compare the most used non-parametric estimators to evaluate the prediction/generalization error for some nonparametric regression models.
- ❑ We are considering the influence of overfitting, the level of non-linearity, the signal/noise ratio and the sample size on the performance of the estimators.
- ❑ We deal with these specific evaluation problems in the followings paragraphs using an extensive simulation approach without considering any restrictive hypothesis.



# Risk estimation in regression problems

In a regression problem we often intend to predict an outcome  $Y$  given the covariates  $X_j$

Then we need to estimate a model on a sample and estimate the prediction performance of the selected model on the population.

Generally speaking, the following are realistic observational conditions:

- We don't know the data generating distribution
- The sample size is not large
- The relationship between  $Y$  and the covariates is not linear
- The selected model is not the "TRUE" model

# The literature

In the literature there are many contributions on this topic:

- model selection methods (Shao 1993, Zhang 1993, Breiman 1992, Ye 1998, Shen & Ye 2002, Efron 2004)
- asymptotic properties of the estimators (Stone 1977, Burman 1989, Dudoit & van der Laan 2005) .

Few results are available for the properties of estimators considering the (more stable) resampled version of Cross-Validation and Hold-out.

A reliable comparison among different proposals for non-linear models is also missing.

# The Prediction Error

Consider  $Y = f(X_1, X_2, \dots, X_J) + \varepsilon$ , where  $f$  is a general unknown function of the covariates  $\mathbf{X} = X_1, X_2, \dots, X_J$  and  $\varepsilon$  is a random noise with mean zero and variance  $\sigma^2$ .

Take

$$\mu_i = f(x_{i1}, x_{i2}, \dots, x_{iJ}) = E(Y_i | \mathbf{x}_i)$$

the conditional expected value of  $Y$  given the value of covariates.

Chosen an appropriate nonparametric model we could estimate the function  $f(\mathbf{X})$  using a *training-set*  $\mathbf{c} = \{y_i, \mathbf{x}_i\}_1^{n_c}$ , obtaining  $\hat{f}_{\mathbf{c}}$ .

# The Prediction Error /2

We want to know the prediction capability of  $\hat{f}_{\mathbf{c}}$

Chosen a (quadratic) loss function  $L(\cdot)$ ,  
the **True Prediction Error** of a fixed  $\hat{f}_{\mathbf{c}}$  is given by

$$Err = E_{\mathbf{x}_0} E_{Y_0} [L(Y_0, \hat{f}_{\mathbf{c}}(\mathbf{x}_0)) \mid \hat{f}_{\mathbf{c}}]$$

The **Expected True Prediction Error** is obtained not fixing  $\mathbf{c}$  and  $\hat{f}_{\mathbf{c}}$  :

$$\overline{Err} = E_{\mathbf{c}} \{Err\} = E_{\mathbf{c}} E_{\mathbf{x}_0} E_{Y_0} [L(Y_0, \hat{f}_{\mathbf{c}}(\mathbf{x}_0))]$$

Note that  $E_{\mathbf{c}}$  averages on training samples and then on  $\hat{f}_{\mathbf{c}}$ , while  $E_{\mathbf{x}_0}$  considers new test points with  $\hat{f}_{\mathbf{c}}$  fixed.

# Extra/In-sample Error

*True Prediction Error* = *extra-sample error*

A more restrictive definition: the *in-sample error*

The values of the covariates are considered fixed at their observed sample values  $\mathbf{X}_c$  while  $Y$  is random

$$Err_{in} = \frac{1}{n_c} \sum_{i=1}^{n_c} E_{Y_i} \left[ L(Y_i, \hat{f}_c(\mathbf{x}_i)) \mid \hat{f}_c, \mathbf{X}_c \right]$$

This was considered by several authors mainly in a model selection approach (Efron, 1986, 2004; Ye, 1998; Shen & Ye, 2002).

# Predictors

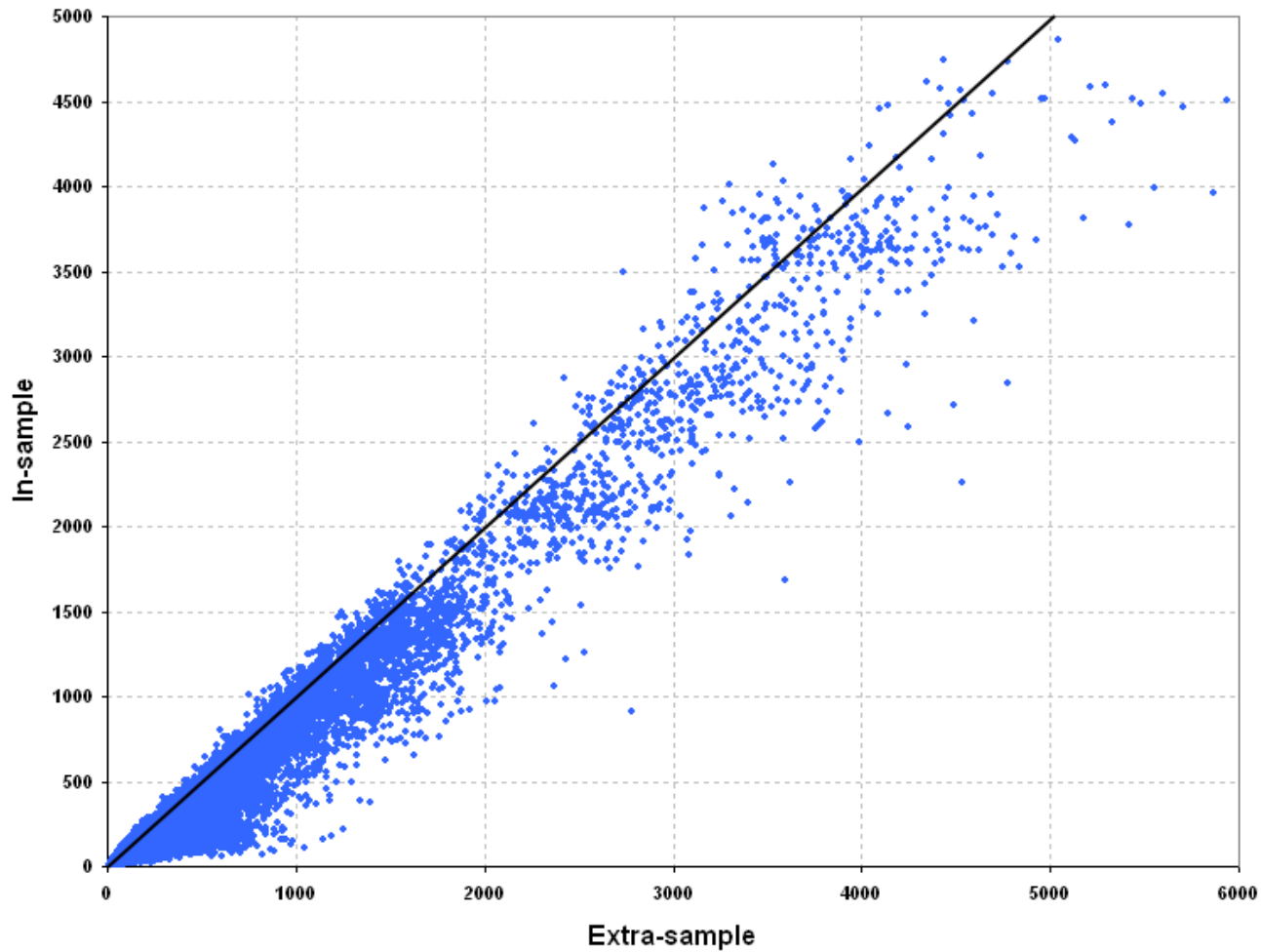
In our simulation study we considered the following nonparametric predictors  $\hat{f}_c$  :

**TREE:** Regression Tree (CART)  
(min # of cases for terminal nodes= 2 or 10, no pruning)

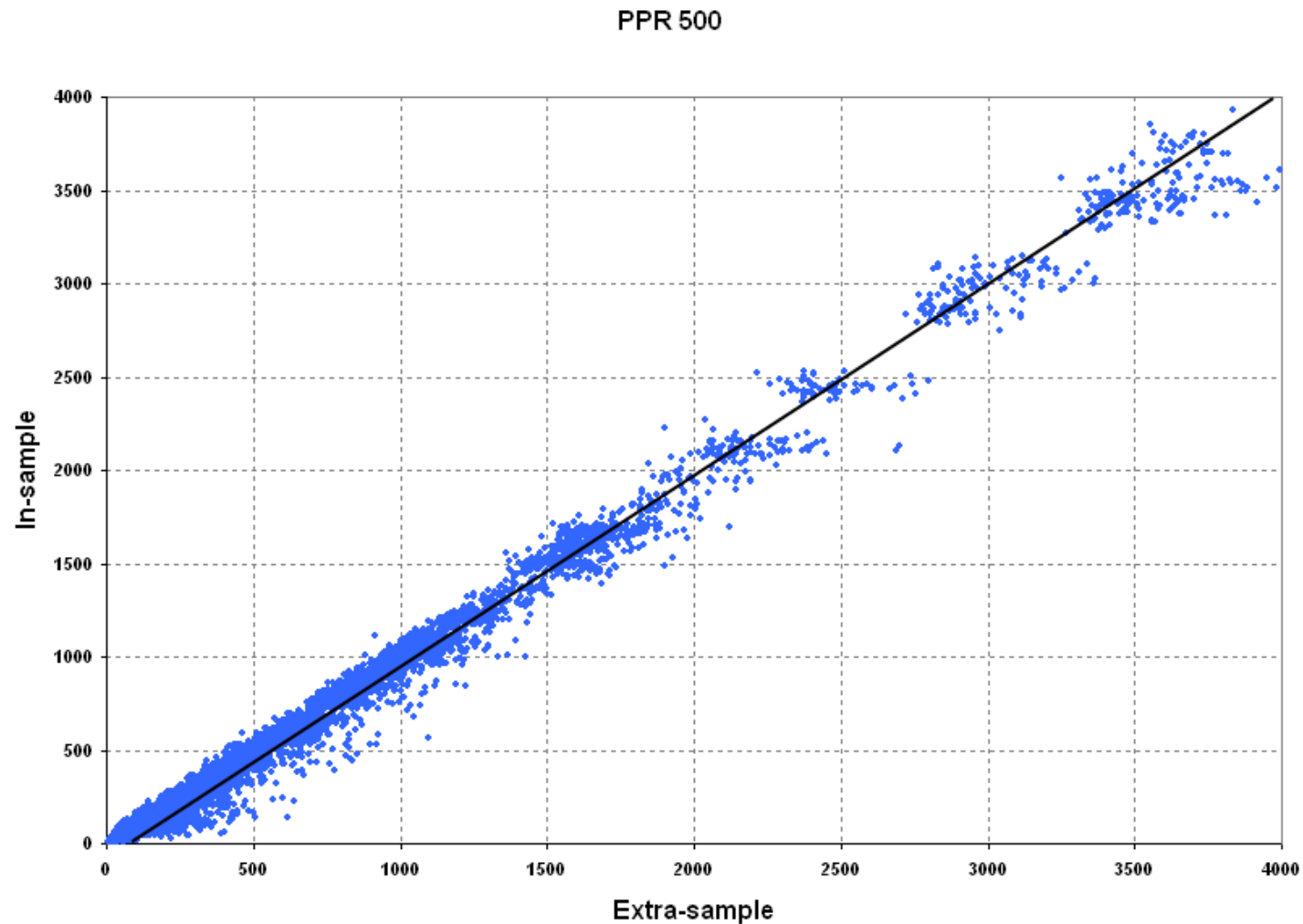
**PPR:** Projection Pursuit Regression (3 nonparametric scatterplot smoothers)

**NN:** Neural Networks  
(1 hidden layer with 8 knots, 35% validation set, max 800 epochs, feedforward backpropagation, transfer function=tansig)

# Extra-sample VS in-sample error. PPR 120



# Extra-sample in-sample error. PPR 500



# Apparent Error and Optimism

The simplest estimator is the **Apparent Error (AE)** defined by

$$err = \frac{1}{n_c} \sum_{i=1}^{n_c} L(y_i, \hat{f}_c(\mathbf{x}_i))$$

The (average) **optimism** is usually defined as the expected difference between  $Err_{in}$  and  $err$  on new training data, having random  $Y$  but fixing  $\mathbf{X}$  at the observed sample values (Efron, 1986, Tibshirani & Knight 1999):

$$op = E_Y[Err_{in} - err | \mathbf{X}_c]$$

$\hat{f}_c$  change for each training sample:  $E_Y$  averages with respect to training samples which have fixed  $\mathbf{X}_c$  and random  $Y$ .

Optimism is typically positive: *Apparent error* is usually biased downward as an estimate of the (*in-sample*) *prediction error*.

# Apparent Error and Optimism

For a quadratic loss and a general function  $f(\mathbf{X})$  we obtain (Efron, 1986):

$$E_y[Err_{in}] = E_y[err] + op = E_y[err] + \frac{2}{n_c} \sum_i \text{cov}_y(\hat{\mu}_i, y_i)$$

the covariance penalty term added to the apparent error indicates the influence of a point  $y_i$  on its estimate  $\hat{\mu}_i$ .

In literature there are some proposals to estimate the covariance term based on bootstrap method:

Ye(1998, 2002), Breiman (1992), Shen and Ye (2002), Efron (2004)

# Extra sample error estimators

The list of estimators considered is the following:

- $err$ , the apparent error;
- $err^{CV}$ , the Repeated 10-fold Cross Validation estimator, with 10 random start;
- $err^{CV*}$ , the same as  $err^{CV}$  but with the Burman's correction;
- $err^{RHO}$ , the Repeated Hold-Out estimator, with 100 subsampling;
- $err^{LOO}$ , the Leave-one-out Cross Validation estimator;
- $err^{B632}$ , the Bootstrap .632 estimator, with 100 subsampling;
- $err^{B632+}$ , the Bootstrap .632+ estimator, with 100 subsampling;
- $err^{PB}$ , the Parametric Bootstrap estimator, with 100 subsampling;

# Cross-validation

Split the sample in  $K$  subset and  $m=n/K$ .

Indicate  $\mathbf{c}^h$  = the training-set without the  $h$ -th subset  $t_h$

The **K-fold CV-estimator** is defined as the average error on the  $K$  analyses:

$$err^{CV} = \frac{1}{K} \sum_{h=1}^K \frac{1}{m} \sum_{j \in t_h} L(y_j, \hat{f}_{\mathbf{c}^h}(\mathbf{x}_j))$$

If  $K = n_c$ , we obtain the "Leave-One-Out" C.V. (LOO).

**LOO** is an almost unbiased estimator of  $Err$  but it has high variability producing non-reliable estimates (Efron, 1983).

**K-fold C.V.** has lower variability but is biased upwards, especially for small samples.

# Corrected Cross-validation estimator

Burman (1989) showed

$$E(\text{err}^{CV} - \text{Err}) \approx c_0(K-1)^{-1}n^{-1}$$

For  $K=n$  the right-side term of the expression is  $O(n^{-2})$ , but when  $K$  is small this term is not necessarily very small. Moreover the constant term  $c_0$  is of the same order of the number of parameters being estimated and so CV may give a poor estimate of  $\text{Err}$  if the number of parameters is large. Therefore Burman introduced the **corrected cross-validation CV\***:

$$\text{err}^{CV^*} = \text{err}^{CV} + \text{err} - \bar{e}^+$$

Where  $\bar{e}^+ = \frac{1}{K} \sum_j e_j^+$  and  $e_j^+$  is obtained considering  $K-1$  folds for estimation and the whole sample for test.

# Hold-out estimators

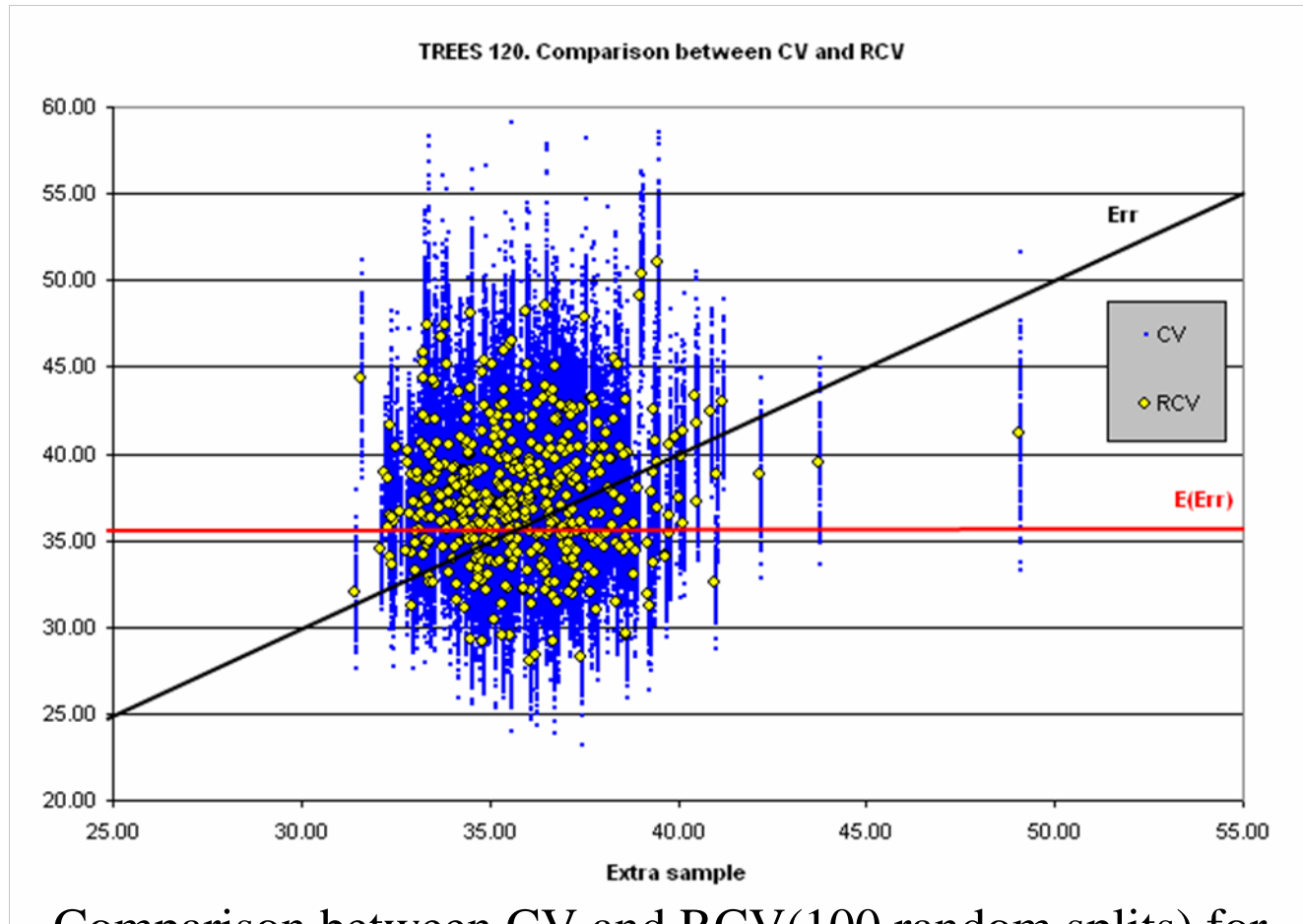
The **Hold-out** estimator **(HO)** is similar to a cross-validation estimator. It is obtained randomly splitting the sample in a training set to estimate the model and a test set to evaluate the prediction error.

Usually, the dimension of the training set is  $2/3$  of the sample.

As for Cross Validation, the Hold-out estimate depends on the random partition.

To overcome this problem, it is useful to introduce the **Repeated CV estimators (RCV)**

# Repeated CV estimators



Comparison between CV and RCV(100 random splits) for Regression Trees: 500 samples of size 120 from one data generating distribution, with respect to extra-sample Error.

# Nonparametric Bootstrap estimators

Given  $K$  subsamples,  $\{b_1, b_2, \dots, b_K\}$ , with replacement, the model is estimated on each subsample  $b_j$ ,  $err_j^{Bt}$  is calculated considering only the cases of the sample not included in  $b_j$ .

The **Leave-one-out bootstrap** estimator of  $Err$  is given by:

$$err^{Bt} = \frac{1}{K} \sum_{j=1}^K err_j^{Bt}$$

the **.632 bootstrap** estimator (Efron 1983) **B632**:

$$err^{B632} = 0.632 \cdot err^{Bt} + 0.368 \cdot err$$

The **.632+ bootstrap** estimator is given by (**B632+**):

$$err^{B632+} = \hat{w} \cdot err^{Bt} + (1 - \hat{w}) \cdot err$$

The weight  $\hat{w}$  ranges from 0.632 (no-overfitting) to 1 (severe overfitting).



# Parametric Bootstrap

To evaluate the prediction capability of a model we could estimate the optimism and then add it to the apparent error.

- 1) Estimate the model  $\hat{\boldsymbol{\mu}} = m(\mathbf{y})$  and  $\hat{\sigma}^2$
- 2) Generate, from the bootstrap density  $\hat{\mathbf{f}} = N(\hat{\boldsymbol{\mu}}, \hat{\sigma}^2 \mathbf{I})$  and for each value  $\mathbf{x}_i$  of the sample,  $B$  observations and estimates

$$y_i^*, \hat{\mu}_i^* = m(y_i^*).$$

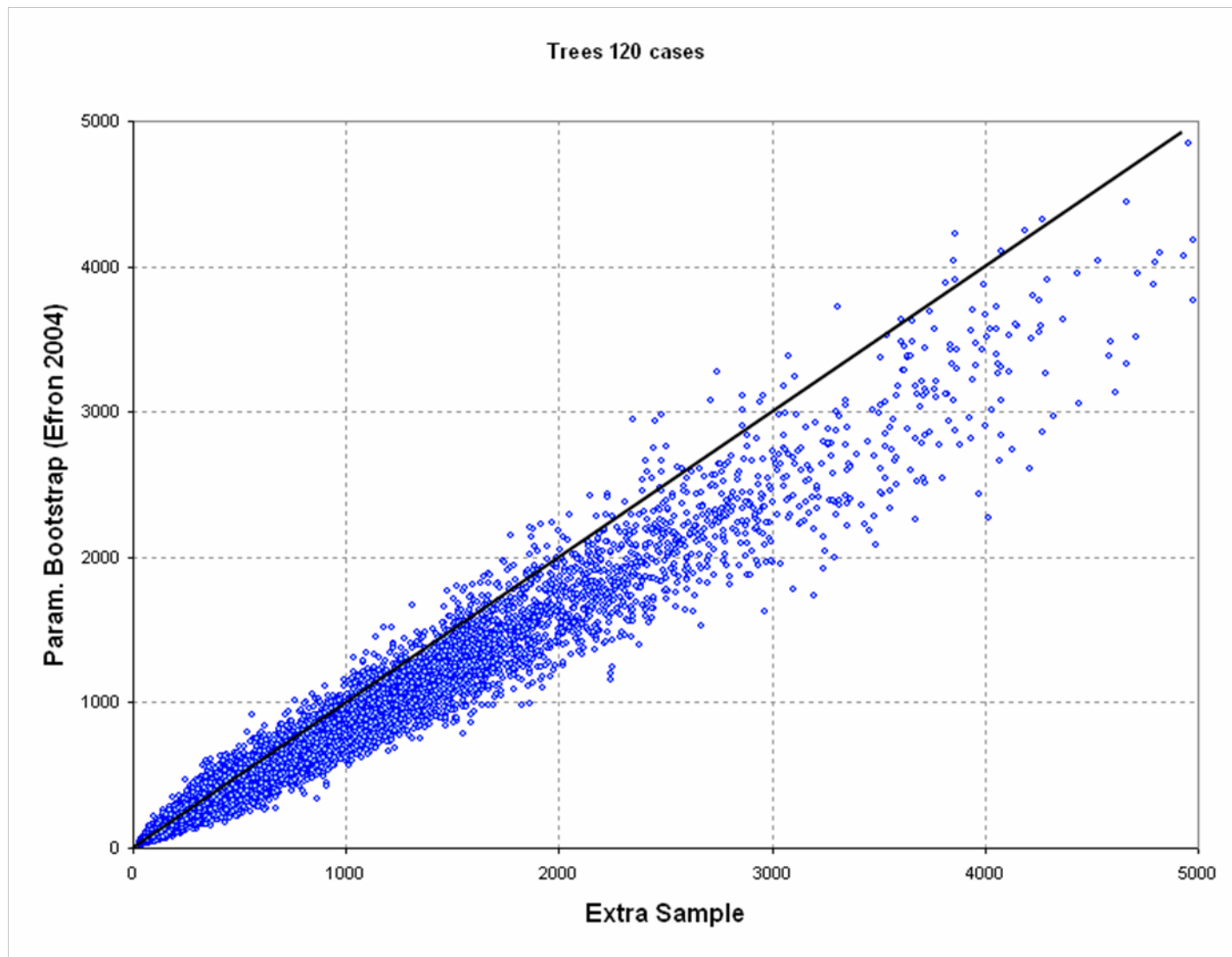
- 3) Estimate  $\text{cov}_{\mathbf{y}}(y_i, \hat{\mu}_i)$ :

$$\text{cov}_i = \sum_{b=1}^B \hat{\mu}_i^* (y_i^{*b} - \bar{y}_i^*) / (B-1) \text{ with } \bar{y}_i^* = \sum_b \frac{y_i^{*b}}{B}.$$

The **parametric bootstrap estimator** (Efron 2004):

$$\text{err}^{PB} = \text{err} + \frac{2}{n_c} \sum_i \text{cov}_i$$

# Parametric Bootstrap



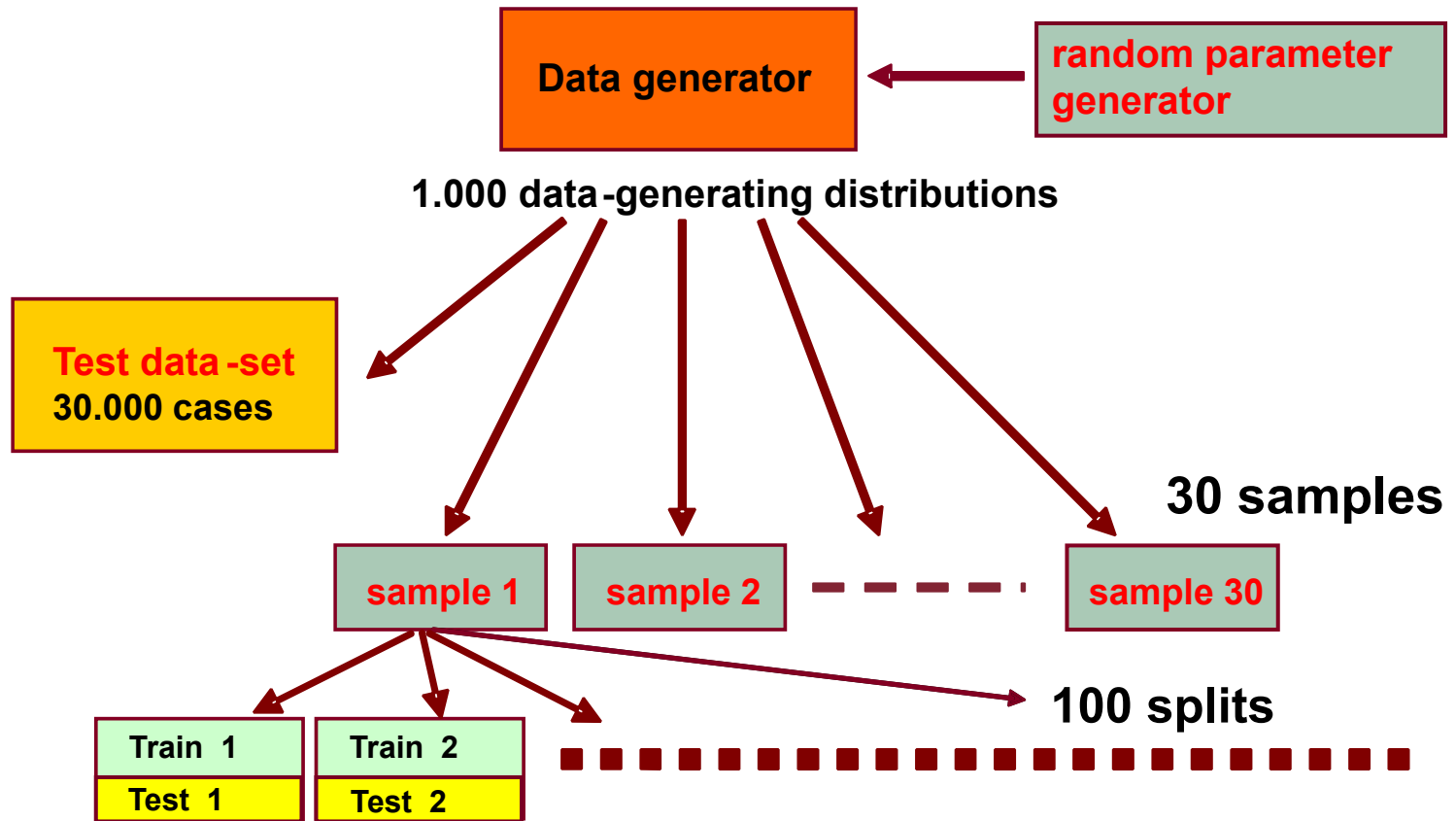
# The Simulation

1000 data generating distributions, in which  $Y = f(X_1, X_2, X_3, X_4) + \varepsilon$

$$f(\mathbf{X}) = a \left( c_0 + \sum_{j=1}^4 c_j X_j \right) + \sum_{j=1}^4 \beta_j (X_j - \bar{X}_j)^2$$

- $X_j$  ( $j=1, \dots, 4$ ) is the  $j$ -th explanatory variable generated from a Beta distribution with parameters  $(g_j, t_j)$ , where the latter are drawn from a Uniform distribution in the interval  $[2, 10]$ ;
- $a$ ,  $c_0$ ,  $c_j$  and  $\beta_j$  ( $j=1, \dots, 4$ ) are randomly drawn from Uniform distributions in the intervals  $[0, 4]$ ,  $[-5, 5]$ ,  $[-5, 5]$  and  $[-50, 50]$ , respectively;
- $\varepsilon$  is the noise generated from a Gaussian distribution  $N(0, \sigma_\varepsilon^2)$ .

# The Simulation (repeated hold out)



# The simulations

## Two different simulations:

### For **PPR** and **TREE**:

1000 generating distribution having a random signal to noise ratio  $r_{s/n}$  between 0.5 and 4.5. The signal account for, approximately and depending on the generating distribution, between 20% and 95% of the total variability of  $Y$ .

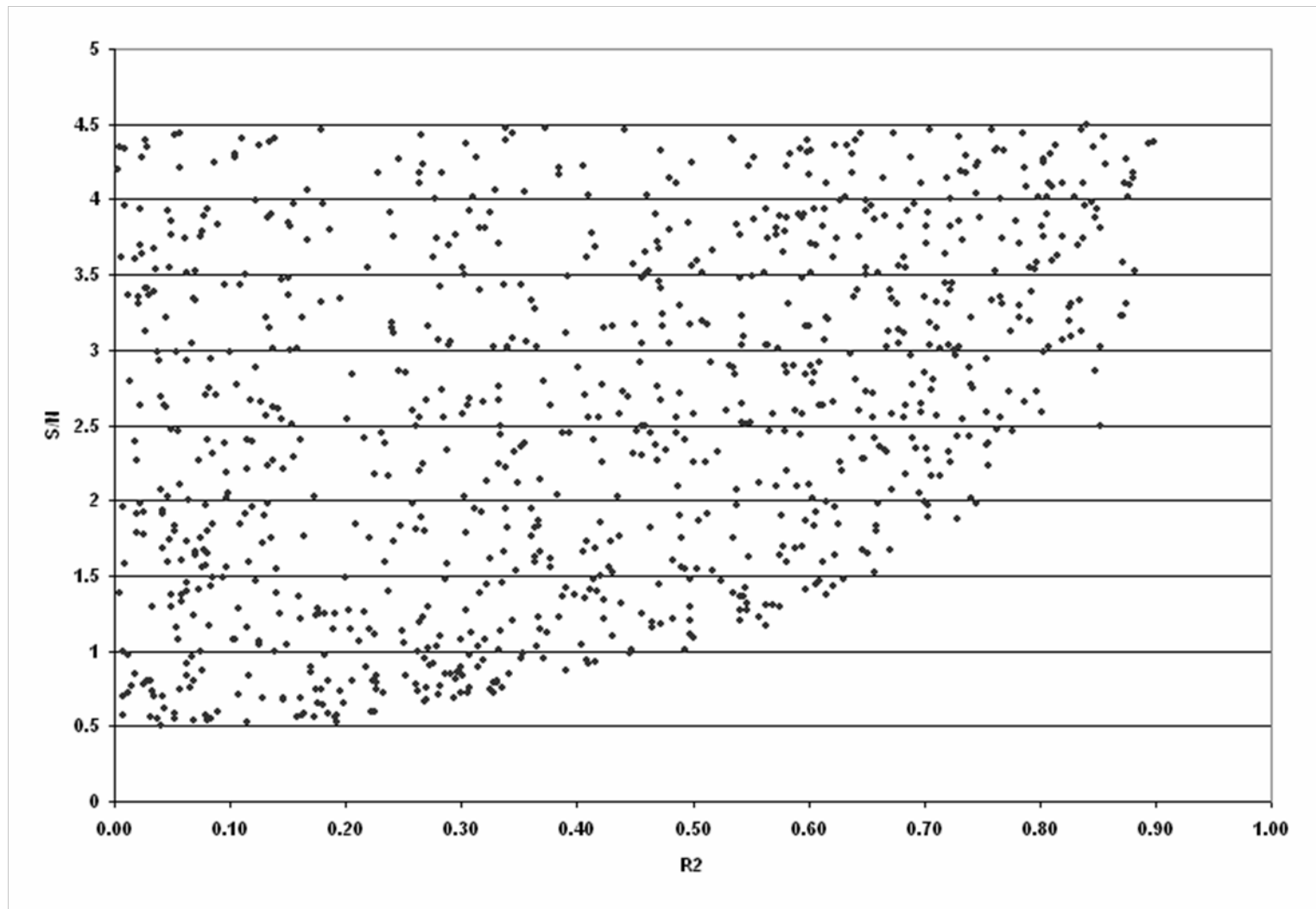
- **PPR 120**: Projection Pursuit Regression with sample size 120.
- **PPR 500**: Projection Pursuit Regression with sample size 500.
- **TREE 120**: Regression Tree with sample size 120.
- **TREE 500**: Regression Tree with sample size 500.

### For **NN**:

100 generating distribution having a random signal to noise ratio  $r_{s/n}$  between 0.5 and 4.5.

- **NN 120**: Neural Network with sample size 120.
- **NN 500**: Neural Network with sample size 500.

# Generating distributions



# Comparing bias and variability

$$rb_h = \frac{1}{K} \sum_{j=1}^K \frac{E\hat{r}_{jh} - Err_{jh}}{E\hat{r}_{jh} + Err_{jh}} \quad (\text{relative bias for } h\text{-th distrib.})$$

$$\overline{arb} = \frac{1}{H} \sum_{h=1}^H |rb_h| \quad (\text{mean absolute relative bias})$$

For each distribution, we computed the values:

$$v = \sqrt{\frac{1}{K} \sum_{j=1}^K (E\hat{r}_j - Err_j)^2}$$

An overall measure can be obtained by

$$rse_h = \frac{v_h}{\text{var}(Err_h)} \quad (\text{relative root squared error for } h\text{-th distrib.})$$

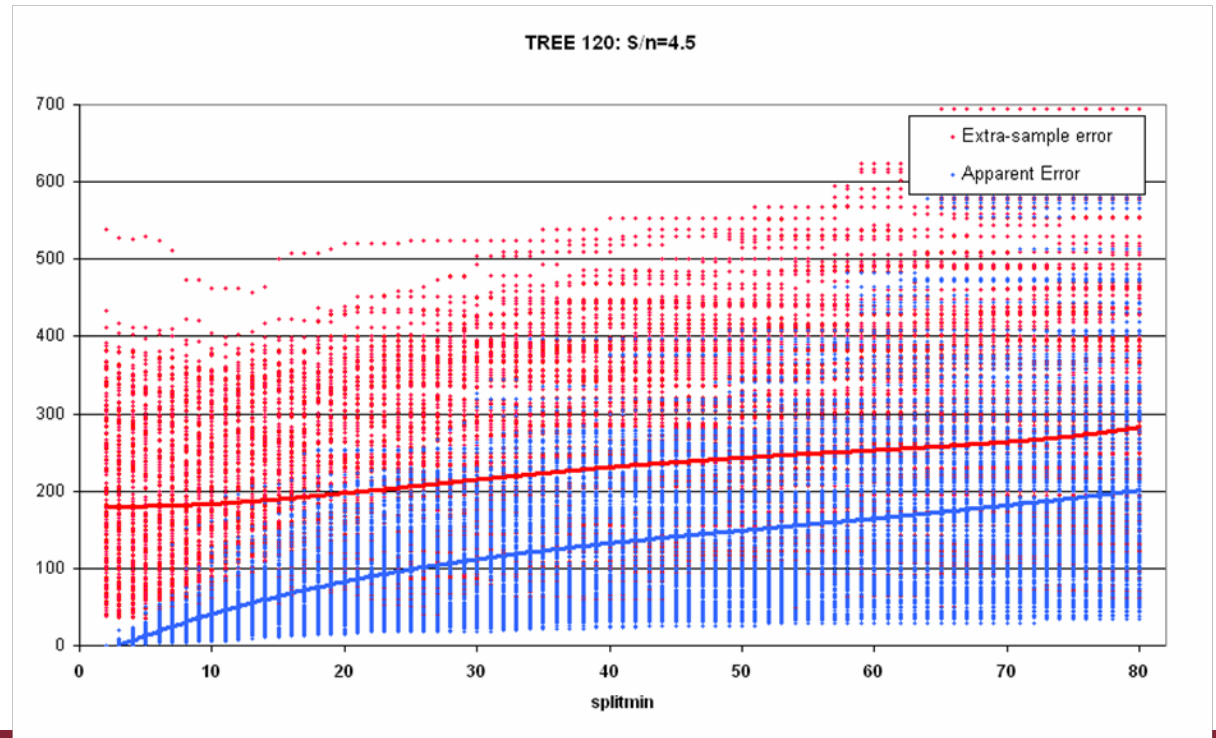
$$\overline{rse} = \frac{1}{H} \sum_{h=1}^H rse_h \quad (\text{mean relative root squared error})$$

# Comparing the estimators

Mean on 1000 generating distributions

	Mean of extra-sample error	Mean of apparent error	Ratio
PPR 120	395.22	141.33	0,358
PPR 500	262.54	209.78	0,799
TREE 120	462.37	91.37	0,208
TREE 500	340.14	125.97	0,384
NN 120	444,31	189,36	0,435
NN 500	151,98	121,53	0,810

} Overfit (no pruning)



## Regression Trees.

	$\overline{arb}$ sample size 120	$\overline{arb}$ sample size 500	$\overline{rse}$ sample size 120	$\overline{rse}$ sample size 500
RCV	<b>0.014</b>	<b>0.010</b>	<b>1.658</b>	<b>1.828</b>
RCV*	0.043	0.030	1.733	2.149
PB	0.074	0.045	1.956	2.485
B632	0.109	0.080	2.286	3.486
B632+	0.056	0.055	1.732	2.669
LOO	<b>0.018</b>	<b>0.008</b>	1.856	2.038
RHO	0.037	0.034	1.807	2.341

## Projection Pursuit

	$\overline{arb}$ sample size 120	$\overline{arb}$ sample size 500	$\overline{rse}$ sample size 120	$\overline{rse}$ sample size 500
RCV	0.030	0.019	1.181	1.315
RCV*	<b>0.022</b>	<b>0.008</b>	<b>1.091</b>	<b>0.962</b>
PB	0.062	0.012	1.183	<b>0.945</b>
B632	0.124	0.074	1.563	1.773
B632+	0.182	0.082	2.185	1.961
LOO	<b>0.023</b>	0.013	1.260	1.323
RHO	0.122	0.045	1.674	1.562

## Neural Networks

	$\overline{arb}$ sample size 120	$\overline{arb}$ sample size 500	$\overline{rse}$ sample size 120	$\overline{rse}$ sample size 500
RCV	0.164	0.070	1.274	1.279
RCV*	<b>0.109</b>	<b>0.037</b>	<b>0.923</b>	<b>0.783</b>
PB	0.123	<b>0.034</b>	<b>0.944</b>	<b>0.707</b>
B632	0.134	0.057	1.051	1.040
B632+	0.149	0.058	1.190	1.062
LOO				
RHO	0.247	0.097	2.159	1.972

# Conclusions

- ❑ The level of overfitting and signal/noise ratio influence the performance of the estimators
- ❑ The comprehensive best performance was obtained by RCV\*
- ❑ With no-overfitting PB has similar performance of RCV\*
- ❑ The Leave-one out is not always the less biased estimator and it has relevant variability and heavy computational time
- ❑ Resampled estimators RCV, RCV\*, RHO have lower variability than the simple estimators CV, CV\*, HO
- ❑ Stability of the model and sample size influence the results
- ❑ Non-parametric B632 and B632+ have a poor performance, Parametric Bootstrap is better (in particular with large sample and stable models)