Estimation and decomposition of prediction performance

Is leave-one-out the answer?

Thomas A Gerds
Department of Biostatistics, University of Copenhagen
`tag@biostat.ku.dk`

Joint work with
Mark van de Wiel (Amsterdam)
Jelle Goeman (Leiden)
Hemant Ishwaran (Miami)

Miami, March 01, 2012

If you urgently need information . . .

If you urgently need information ...

For example, to answer the following multiple choice question:

Q: What is bagging?

1. A machine learning ensemble meta-algorithm
2. Searching in a *bag*
3. A special case of model averaging
4. The last name of Leo Breiman's first Ph.D student
5. A short name for bootstrap aggregating

then ...

. . . there are several strategies

Bagging

The results of the k-*nearest neighbor method* can be improved by combining the results of many neighbors, think of *asking the audience* from the well-known tv-show.

More generally, a *weak learner* can be improved by *bagging*[*] .

Random forest[*] combines many decision trees (based on bootstrap) and thereby improves the predictions of a single tree.

[*]Leo Breiman (1996). "Bagging predictors". Machine Learning 24 (2): 123–140

[*]Leo Breiman (2001). "Random Forests". Machine Learning 45 (1), 5-32, Gérard Biau, Luc Devroye, Gábor Lugosi (2008). “Consistency of Random Forests and Other Averaging Classifiers”. Journal of Machine Learning Research 9 2015-2033.

A prediction problem

Outcome:

$$Y = \begin{cases} 1 & \text{positive / disease} \\ 0 & \text{negative / non-disease} \end{cases}$$

Predictors:

$$X = (X^1, X^2, \ldots, X^K)$$

Parameter:

$$\mathrm{P}(Y = 1 | X = x)$$

Data set: $D_n = ((X_1, Y_1), \ldots, (X_n, Y_n)) \qquad \text{iid}$

A survival prediction problem

Time-to-event outcome:

$$Y(t) = I\{T > t\} = \begin{cases} 1 & \text{event-free} \\ 0 & \text{event} \end{cases}$$

Predictors:

$$X = (X^1, X^2, \ldots, X^K)$$

Parameter:

$$\mathrm{P}(Y(t) = 1 | X = x)$$

Data set: $D_n = ((X_1, T_1 \wedge C_1, \Delta_1), \ldots, (X_n, T_n \wedge C_n, \Delta_n)) \qquad \text{iid}$

The object of the talk

A statistical risk prediction model "*m*" is a mapping from individual predictor values X to the probability of an event:

Cox/logistic regression
Support Vector Machines
Bump hunting
Lars and his three cousins
Cart and RandomForests
Local/logic regression

$$X \rightarrow \boxed{\quad} \rightarrow m(X) \approx \mathrm{P}(Y(t)=1|X)$$

The making of a prediction model

A prediction modelling strategy "S" is a mapping from training data

$$D_n = \{(Y_1, X_1), \dots, (Y_n, X_n)\}$$

to the set of prediction models:

$D_n \rightarrow$

Cox/logistic regression
Support Vector Machines
Bump hunting
Lars and his three cousins
Cart and RandomForests
Local/logic regression

$\rightarrow \mathcal{S}(D_n) = M_n$

Predicted risk

The predicted risk based

- on a model obtained with modelling strategy S and data D_n
- for the unknown status Y_{new} of a new patient X_{new}

is the result of applying two mappings:

Building the model $\mathcal{S} : D_n \mapsto \mathcal{S}(D_n) = M_n$

Using the model $X_{new} \mapsto M_n(X_{new}) \in [0, 1]$

Sometimes we want to assess the first, sometimes the second mapping.

Estimation of prediction performance

Per-subject Brier score:

$$\{Y_i - M_n(X_i)\}^2 \qquad \text{(the lower the better)}$$

Population summary – performance of the model

$$\frac{1}{n}\sum_i \{Y_i - M_n(X_i)\}^2$$

or

$$1 - \sqrt{\frac{1}{n}\sum_i \{Y_i - M_n(X_i)\}^2} \qquad \text{(the higher the better)}$$

Estimation of prediction performance

Per-subject Brier score:

$$\{Y_i - M_n(X_i)\}^2 \qquad \text{(the lower the better)}$$

Population summary – performance of the model

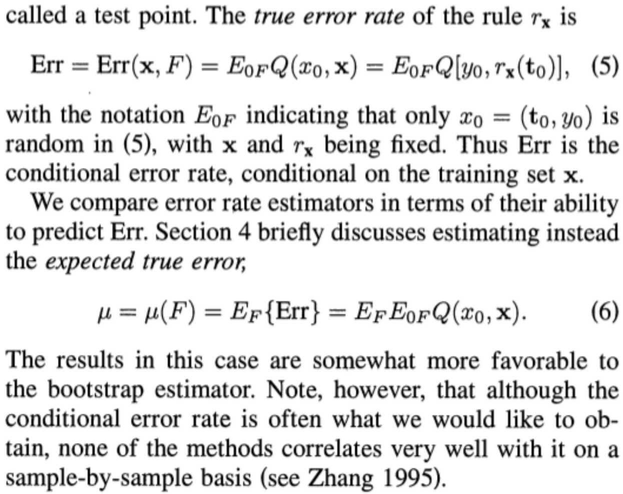
$$\frac{1}{n}\sum_i \{Y_i - M_n(X_i)\}^2$$

or

$$1 - \sqrt{\frac{1}{n}\sum_i \{Y_i - M_n(X_i)\}^2} \qquad \text{(the higher the better)}$$

Which parameter is estimated?

The Efron terminology[1]



called a test point. The *true error rate* of the rule $r_{\mathbf{x}}$ is

$$\mathrm{Err} = \mathrm{Err}(\mathbf{x}, F) = E_{0F} Q(x_0, \mathbf{x}) = E_{0F} Q[y_0, r_{\mathbf{x}}(\mathbf{t}_0)], \quad (5)$$

with the notation E_{0F} indicating that only $x_0 = (\mathbf{t}_0, y_0)$ is random in (5), with $\mathbf{x}$ and $r_{\mathbf{x}}$ being fixed. Thus Err is the conditional error rate, conditional on the training set $\mathbf{x}$.

We compare error rate estimators in terms of their ability to predict Err. Section 4 briefly discusses estimating instead the *expected true error*,

$$\mu = \mu(F) = E_F\{\mathrm{Err}\} = E_F E_{0F} Q(x_0, \mathbf{x}). \quad (6)$$

The results in this case are somewhat more favorable to the bootstrap estimator. Note, however, that although the conditional error rate is often what we would like to obtain, none of the methods correlates very well with it on a sample-by-sample basis (see Zhang 1995).

[1]Efron & Tibshirani (1997)

Conditional prediction performance of a model

Define the **conditional performance** of the model M_n by

$$\begin{aligned}\mathcal{P}(M_n) &= \mathrm{E}_{Y,X}\left[\{Y - \mathcal{S}(D_n)(X)\}^2 \mid D_n\right] \\ &= \mathrm{E}_{Y,X}\left[\{Y - M_n(X)\}^2 \mid D_n\right]\end{aligned}$$

Conditional prediction performance of a model

Define the **conditional performance** of the model M_n by

$$\begin{aligned}\mathcal{P}(M_n) &= \mathrm{E}_{Y,X}\left[\{Y - \mathcal{S}(D_n)(X)\}^2 \mid D_n\right] \\ &= \mathrm{E}_{Y,X}\left[\{Y - M_n(X)\}^2 \mid D_n\right]\end{aligned}$$

$\mathcal{P}(M_n)$ measures the performance of the model built with **this training data set**.

Expected prediction performance of a strategy

Define the **expected performance** of the strategy $\mathcal{S}$ as

$$\begin{aligned}\rho(\mathcal{S}, n) &= \mathrm{E}_{D_n}\left(\mathrm{E}_{Y,X}\left[\{Y - \mathcal{S}(D_n)(X)\}^2 \mid D_n\right]\right)\\ &= \mathrm{E}_{D_n}\left(\mathrm{E}_{Y,X}\left[\{Y - M_n(X)\}^2 \mid D_n\right]\right)\end{aligned}$$

Expected prediction performance of a strategy

Define the **expected performance** of the strategy $\mathcal{S}$ as

$$\begin{aligned}\rho(\mathcal{S}, n) &= \mathrm{E}_{D_n}\left(\mathrm{E}_{Y,X}\left[\{Y - \mathcal{S}(D_n)(X)\}^2 \mid D_n\right]\right)\\ &= \mathrm{E}_{D_n}\left(\mathrm{E}_{Y,X}\left[\{Y - M_n(X)\}^2 \mid D_n\right]\right)\end{aligned}$$

The parameter $\rho(\mathcal{S}, n)$ measures the performance of the **average model** obtained with this strategy at training sample size n.

Decomposition of the expected prediction performance

Using notation

$$m_n(x) = \mathrm{E}_{D_n}\{S(D_n)(x)\}$$

for the average model of strategy $\mathcal{S}$ at sample size n, we get[†]

$$\rho(\mathcal{S}, n) = \underbrace{\mathrm{E}_{X,Y}\left[\{Y - m_n(X)\}^2\right]}_{\text{Model accuracy}} + \underbrace{\mathrm{E}_{D_n}\left[\mathrm{E}_X\{M_n(X) - m_n(X)\}^2\right]}_{\text{Model uncertainty.}}$$

Note: the model accuracy part equals the conditional performance of the average model: $\mathcal{P}(m_n)$.

[†] $\mathrm{E}_{X,Y}\mathrm{E}_{D_n}\{\mathcal{S}(D_n)(X) - m_n(X)\} = 0$

What do we want to estimate?

Situation 1:

The best model should be implemented for practical usage.

We are not interested in the variability of the models that *could have been build on other data bases which do not exist*.

We are interested in the conditional prediction performance $\mathcal{P}(M_n)$.

What do we want to estimate?

Situation 1:

The best model should be implemented for practical usage.

We are not interested in the variability of the models that *could have been build on other data bases which do not exist*.

We are interested in the conditional prediction performance $\mathcal{P}(M_n)$.

Situation 2:

We have developed a new modelling strategy and the aim is to assess how well it works and whether it can compete with alternative strategies.

We are interested in the expected prediction performance $\rho(\mathcal{S}, n)$.

Example: Risk prediction in medicine

A patient needs to know:

- Am I diseased? (current status)
- Will I develop the disease? (future status)
- Should I stop smoking?
- Do I really need chemotherapy?

The community wants a risk prediction model

A basic researcher wants a biologically plausible model

A statistician wants a widely applicable strategy

Estimation design (I)

The data are split once into independent **training and validation/test** data sets.

Advantages:

- large sample theory should work for growing test set size

Disadvantages:

- the prediction model cannot be optimal.
- the results may heavily depend on how the data are split.
- does not include the model variability (not without additional efforts)

Estimation design (II)

The data are repeatedly split into training and validation/test data sets (**cross-validation**).

Advantages:

- includes the model variability, because all modelling steps are repeated in each training set.
- provides estimates of the expected performance

Disadvantages:

- many variants exist (k-fold cv, leave-one-out cv, leave-one-out bootstrap, .632+ bootstrap)
- requires that the strategy can be programmed
- may be time consuming

Strategy I: Logistic regression

Predictions are obtained by substituting X_{new} for X in

$$\mathrm{P}(Y = 1|X) = \text{expit}\{\hat{\beta}^0 + \hat{\beta}^1\,\hat{f}^1(X^1) + \cdots + \hat{\beta}^p\,f^k(X^K)\}$$

The data D_n are used

- to estimate the regression coefficients $\hat{\beta}^k$
- to select variables ($\hat{f}^k = 0$), e.g. using backward elimination or LASSO
- to transform variables (e.g. $\hat{f}^k = \log$, or restricted cubic splines)

Simulated data

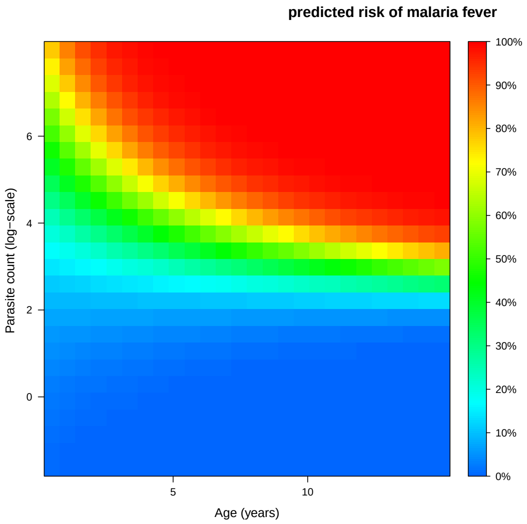
```
expit <- function(x){exp(x)/(1+exp(x))}

malariaData <- function(N){
  Age <- runif(N,.5,15)
  Parasites <- rnorm(N,mean=3.5-0.03*Age)
  LP <- -3.5-.3*Age +.55*Parasites+0.15*Age*Parasites
  Fever <- rbinom(N,1,expit(LP))
  data.frame(Fever,Age,Parasites)
}
```

Simulated data

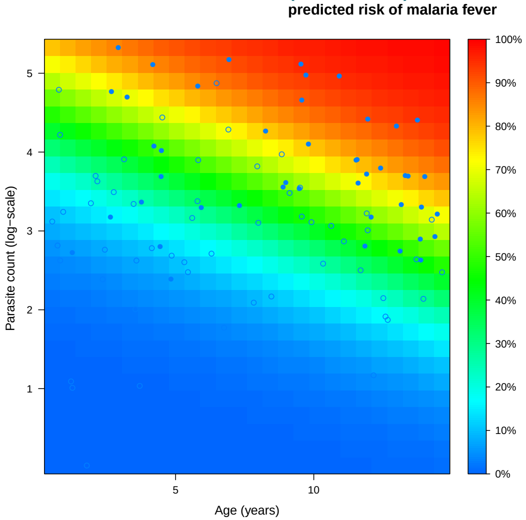
```
set.seed(37)
Malaria <- malariaData(100)
head(Malaria)
Fever Age        Parasites
    0 2.198393  5.059277
    1 7.518431  4.240693
    0 9.942502  2.119874
    0 1.491752  4.093514
    1 5.796818  3.923598
    0 3.746604  2.958425
```

Risk plot: data generating model: n=100,000



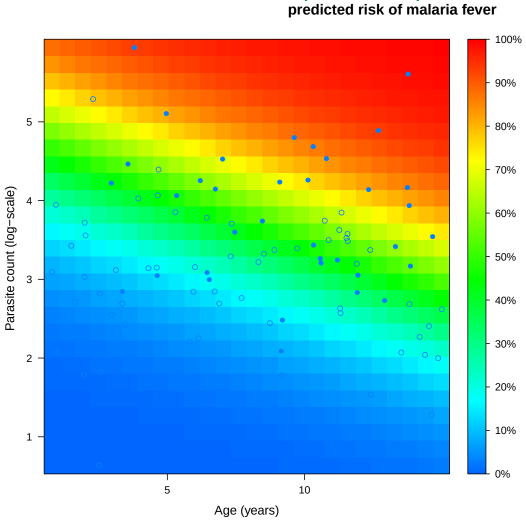
```
predict(lrm(fever~Age+Parasites+Age*Parasites,data=Malaria)
```

Risk plot: logistic regression (n=100)



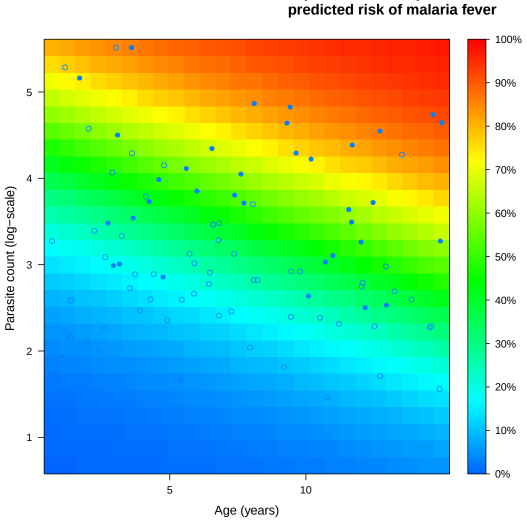
seed: 137
predict(lrm(fever age+parasite,data=Malaria))

Risk plot: logistic regression (n=100)



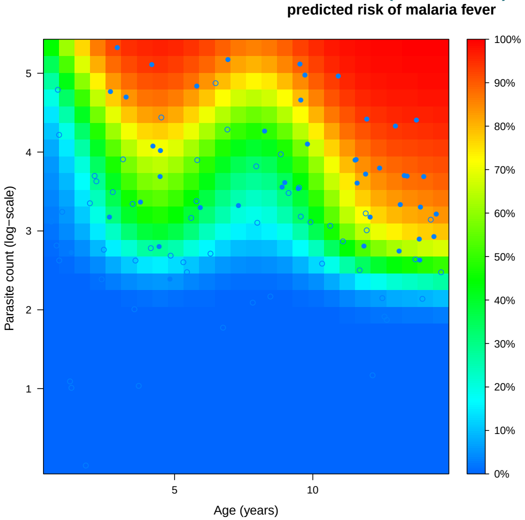
seed: 88
predict(lrm(fever age+parasite,data=Malaria))

Risk plot: logistic regression (n=100)



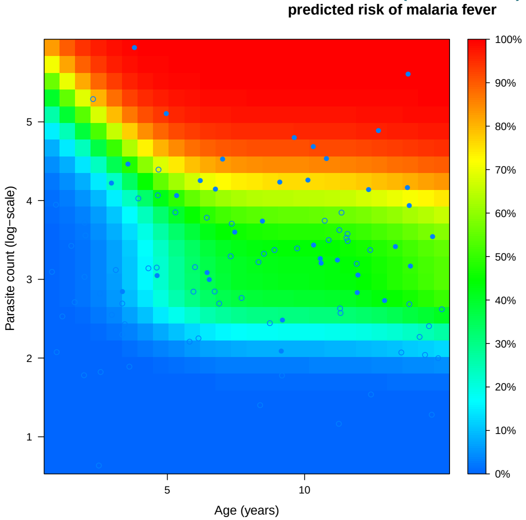
seed: 22
predict(lrm(fever age+parasite,data=Malaria))

Risk plot: logistic spline regression (n=100)



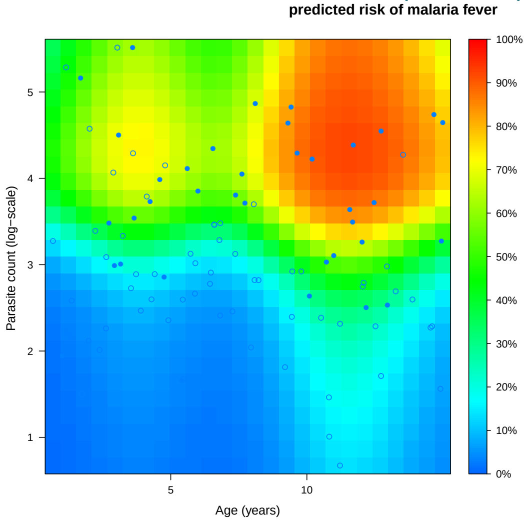
seed: 137
predict(lrm(Fever rcs(Age)+rcs(Parasites),data=Malaria))

Risk plot: logistic spline regression (n=100)



seed: 88
predict(lrm(Fever rcs(Age)+rcs(Parasites),data=Malaria))

Risk plot: logistic spline regression (n=100)



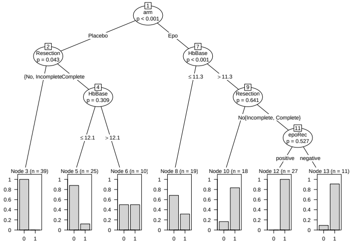
seed: 22

predict(lrm(Fever rcs(Age)+rcs(Parasites),data=Malaria))

Strategy II: classification tree

A classification tree divides the data D_n into terminal nodes. Predictions are obtained by averaging the corresponding terminal node $\mathcal{T}(x)$:

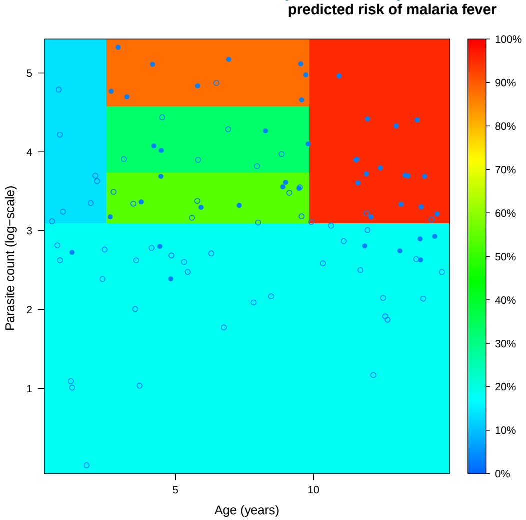
$$\mathrm{P}(Y = 1 | X = x) = \frac{1}{|\mathcal{T}(x)|} \sum_{j \in \mathcal{T}(X)} Y_j$$



The data D_n are used

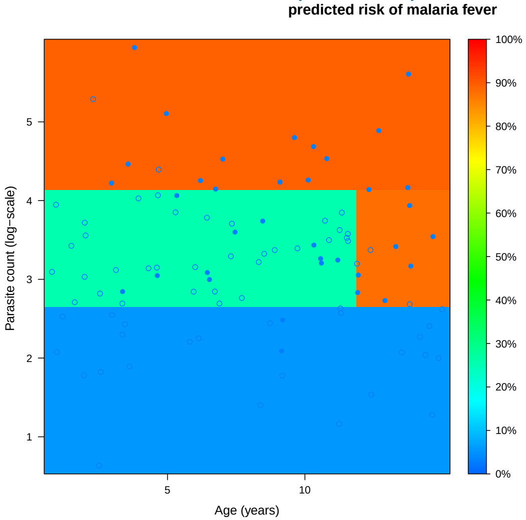
- to grow the trees using splitting and stopping criteria
- to prune the trees
- to estimate the event probability in the terminal nodes

Risk plot: classification tree (n=100)



seed: 137
predict(rpart(fever age+parasite,data=Malaria))

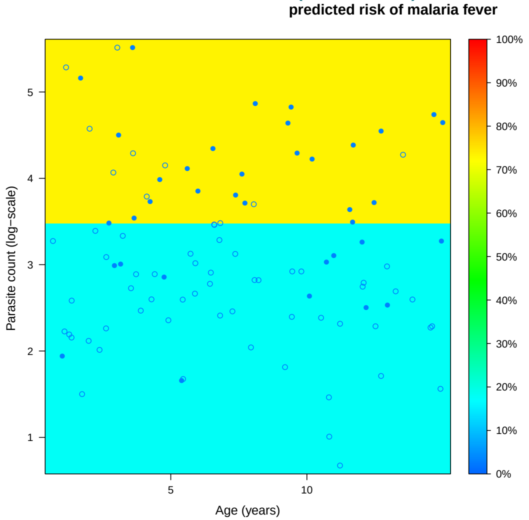
Risk plot: classification tree (n=100)



seed: 88

predict(rpart(fever age+parasite,data=Malaria))

Risk plot: classification tree (n=100)



seed: 22

predict(rpart(fever age+parasite,data=Malaria))

Strategy III: random forest

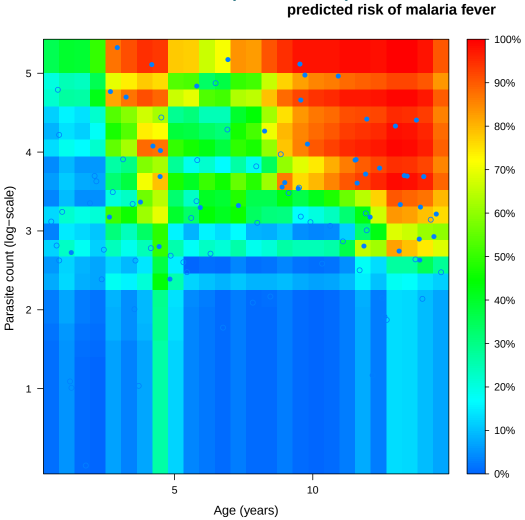
Classification trees are grown in K bootstrap samples. Predictions are obtained by averaging the corresponding terminal nodes:

$$\mathrm{P}(Y = 1|X = x) = \frac{1}{K}\sum_{t=1}^{K}\frac{1}{|\mathcal{T}(x)|}\sum_{j\in\mathcal{T}(x)} Y_j$$

The data D_n are used

- to construct bootstrap data sets
- to grow the classification trees
- to predict the event probabilities in the terminal nodes

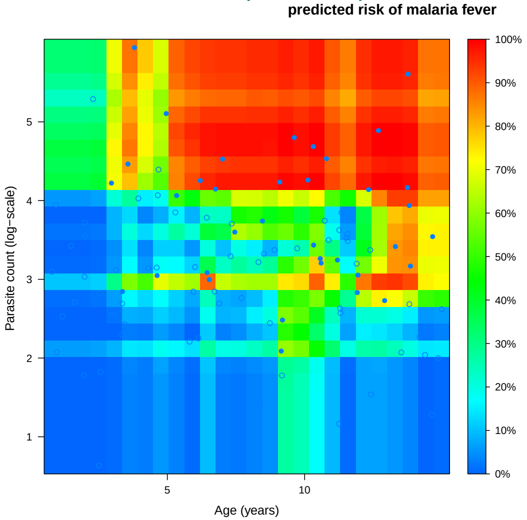
Risk plot: random forest (n=100)



seed: 137

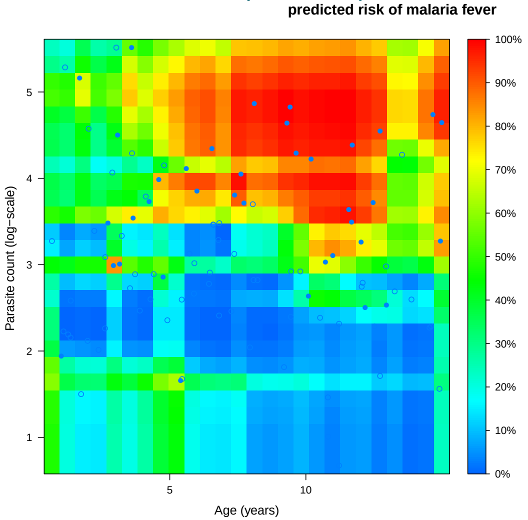
predict(randomForest(fever age+parasite,data=Malaria))

Risk plot: random forest (n=100)



seed: 88
predict(randomForest(fever age+parasite,data=Malaria))

Risk plot: random forest (n=100)



seed: 22
predict(randomForest(fever age+parasite,data=Malaria))

Estimation of performance

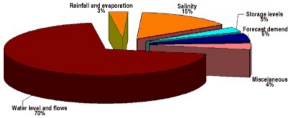
The aim: we want to compare the prediction performance of the four modelling strategies.

The dilemma:

We have only one data set, we want to use all the data to build the best possible model, and if we would have independent validation data, then we would rather merge them with the training data.

Solution: cross-validation!?

Repeatedly take some of the data away to estimate the performance. Leave these data out when building the model.



But how many?

Leave-one-out cross-validation

Leave-one-out (jackknife) estimate:

$$LOOCV(M_n) = \frac{1}{n} \sum_{i \in D_n} \left\{ Y_i - \mathcal{S}(D^{-i}_{n-1})(X_i) \right\}^2$$

where $D^{-i}_{n-1} = \{D_n \text{ without } (X_i, Y_i)\}$.

Leave-one-out cross-validation

Leave-one-out (jackknife) estimate:

$$LOOCV(M_n) = \frac{1}{n} \sum_{i \in D_n} \left\{ Y_i - \mathcal{S}(D_{n-1}^{-i})(X_i) \right\}^2$$

where $D_{n-1}^{-i} = \{D_n \text{ without } (X_i, Y_i)\}$.

Advantages:

- $M_n(x) = S(D_n)(x) \approx S(D_{n-1}^{-i})(x)$. (for most strategies)
- does not depend on the random seed

Disadvantage:

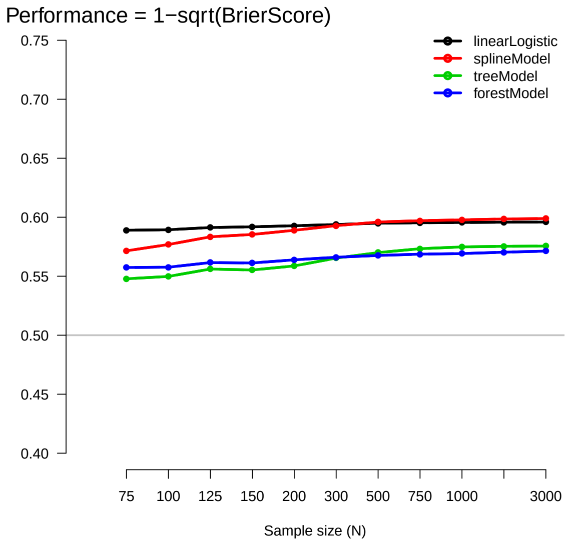
- The strategy has to be applied n times.
- Bias? Variance? For which parameter?

Learning curves

To investigate the performance of the LOOCV estimate for estimating the expected performance, we approximate the learning curves of different strategies (logistic, spline, tree, forest) as follows:

1. Draw 1000 training (malaria) data sets for each sample size (n=75, 100, ..., 3000)
2. Fit the strategies in each data set and save the models
3. Draw a large validation data set (N=200,000)
4. Predict the validation data based on all the models
5. Compute the Brier score performance
6. For each sample size average the 1000 Brier score performances

Learning curves:



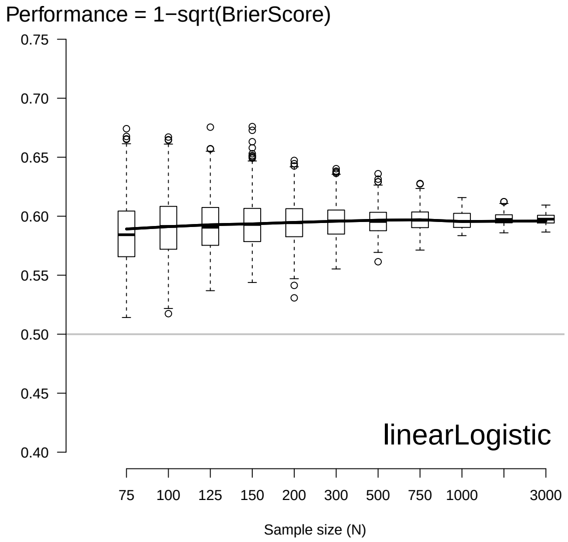
Performance of the LOOCV estimate

For each sample size repeat leave-one-out cross-validation in each of the 1000 training data sets.

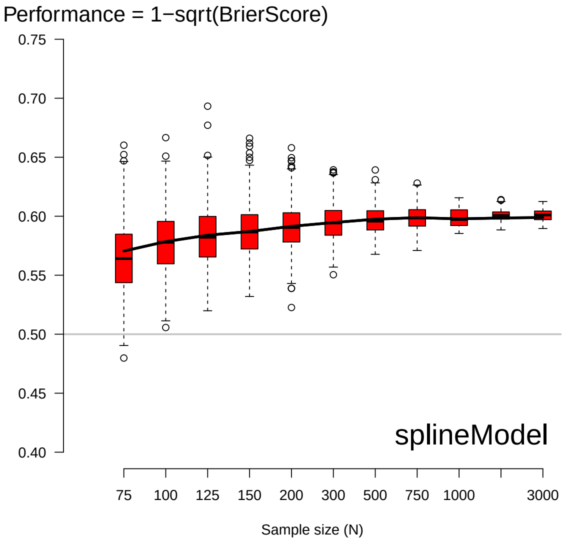
```
...
Brier(trainedModels,data=simulated,splitMethod="loocv")
...
```

For each modelling strategy this yields 1000 LOOCV estimates of prediction performance which can be summarized using boxplots and compared to the corresponding learning curve.

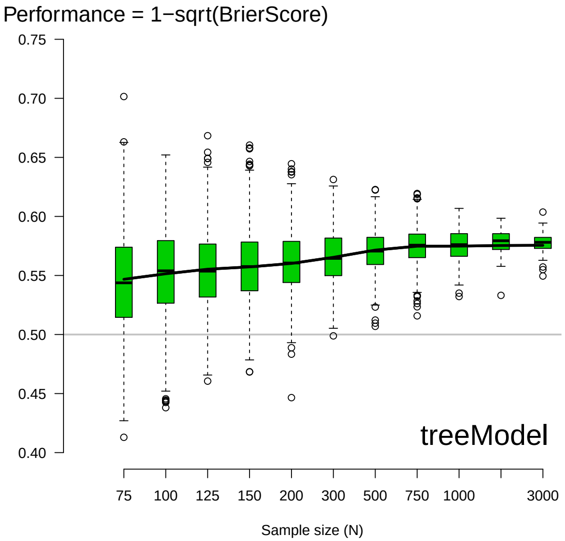
Performance of leave-one-out cv performance



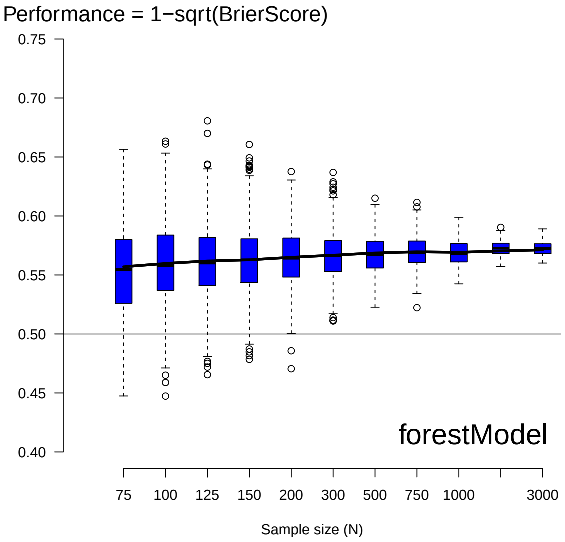
Performance of leave-one-out cv performance



Performance of leave-one-out cv performance



Performance of leave-one-out cv performance



Bootstrap cross-validation

Draw many ($B \geq 1000$) bootstrap training data sets $D_b^{\texttt{train}}$ either with replacement of size $m = n$ or without replacement of size $m < n$.

In each step use the left-out data to compute the performance, then average.

$$\text{BootCV}(\mathcal{S}) = \frac{1}{B}\sum_{b=1}^{B}\frac{1}{n_b}\sum_{i \notin D_b^{\texttt{train}}}\left\{Y_i - M_b^{\texttt{train}}(X_i)\right\}^2$$

where $M_b^{\texttt{train}} = \mathcal{S}(D_b^{\texttt{train}})$.

Leave-one-out Bootstrap

Draw many ($B \geq 1000$) bootstrap training data sets $D_b^{\texttt{train}}$ either with replacement of size $m = n$ or without replacement of size $m < n$.

In each step use the left-out data to compute the performance, then average.

$$\text{LOOBOOT}(\mathcal{S}) = \frac{1}{n} \sum_{i=1}^{n} \frac{1}{K_i} \sum_{b: i \notin D_b^{\texttt{train}}} \left\{ Y_i - M_b^{\texttt{train}}(X_i) \right\}^2$$

where $M_b^{\texttt{train}} = \mathcal{S}(D_b^{\texttt{train}})$.

Leave-one-out Bootstrap

In each step use the left-out data to compute the performance, then average.

$$\mathrm{BootCV}(\mathcal{S}) = \frac{1}{B}\sum_{b=1}^{B}\frac{1}{n_b}\sum_{i\notin D_b^{\texttt{train}}}\left\{Y_i - M_b^{\texttt{train}}(X_i)\right\}^2$$

$$\mathrm{LOOBOOT}(\mathcal{S}) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{K_i}\sum_{b:i\notin D_b^{\texttt{train}}}\left\{Y_i - M_b^{\texttt{train}}(X_i)\right\}^2$$

where $M_b^{\texttt{train}} = \mathcal{S}(D_b^{\texttt{train}})$.

Leave-one-out Bootstrap

In each step use the left-out data to compute the performance, then average.

$$\mathrm{BootCV}(\mathcal{S}) = \frac{1}{B}\sum_{b=1}^{B}\frac{1}{n_b}\sum_{i\notin D_b^{\texttt{train}}}\left\{Y_i - M_b^{\texttt{train}}(X_i)\right\}^2$$

$$\mathrm{LOOBOOT}(\mathcal{S}) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{K_i}\sum_{b:i\notin D_b^{\texttt{train}}}\left\{Y_i - M_b^{\texttt{train}}(X_i)\right\}^2$$

where $M_b^{\texttt{train}} = \mathcal{S}(D_b^{\texttt{train}})$.

Advantages

- includes more model variability
- should be more stable

Disadvantages

- underestimates the performance at n
- depends on the random seed unless B is huge

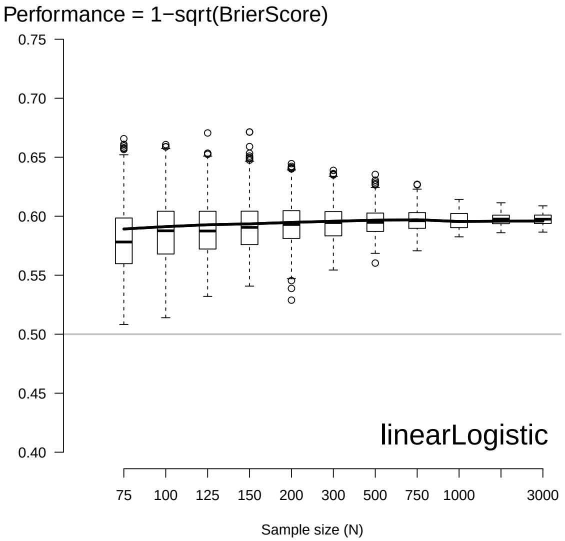
Performance of the BootCV estimate

For each sample size repeat bootstrap cross-validation based on 1000 subsampling bootstrap sets in each of the 1000 simulated training data sets.

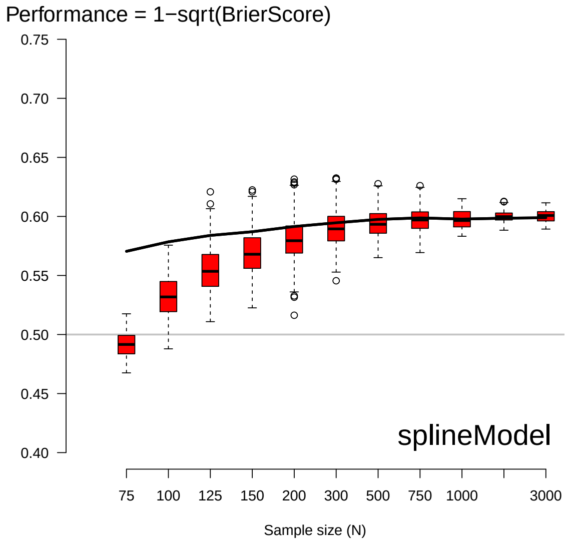
```
...
Brier(trainedModels,data=simulated,splitMethod="BootCV",
      B=1000,M=round(.632*N))
...
```

For each modelling strategy this yields 1000 BootCV estimates of prediction performance which can be summarized using boxplots and compared to the corresponding learning curve.

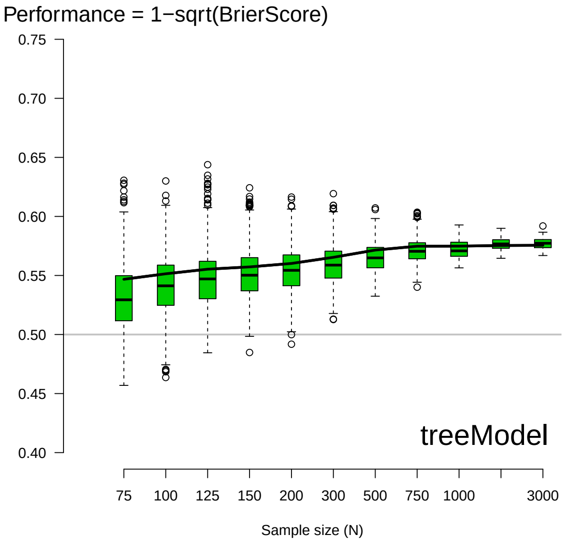
Performance of BootstrapCrossValidation performance



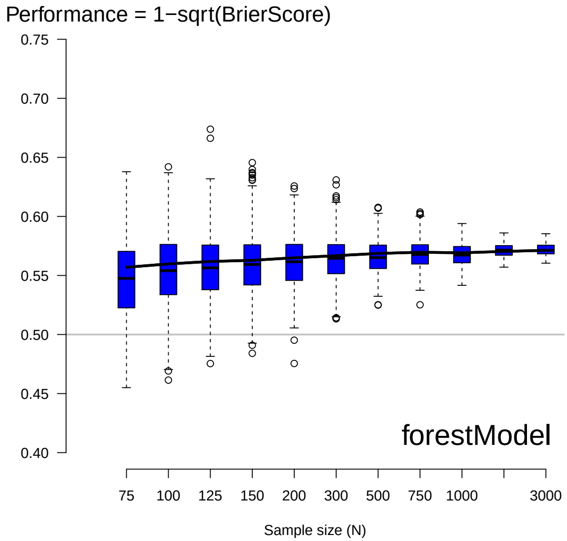
Performance of BootstrapCrossValidation performance



Performance of BootstrapCrossValidation performance



Performance of BootstrapCrossValidation performance



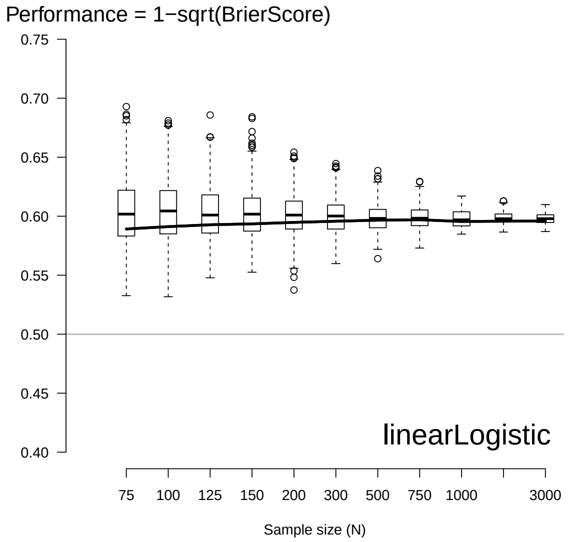
Apparent performance

The apparent performance (aka re-substitution performance) is obtained by validating the model in the training data.

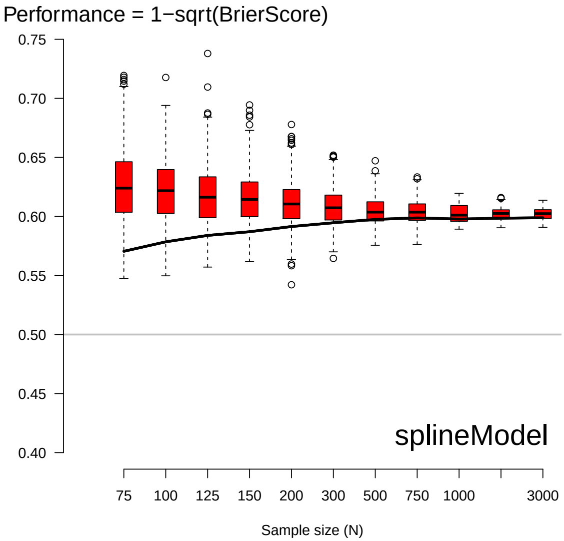
$$\mathrm{App}(\mathcal{S}) = \frac{1}{n} \sum_{i \in D_n} \{Y_i - M_n(X_i)\}^2$$

Disadvantage: Overestimates the prediction performance

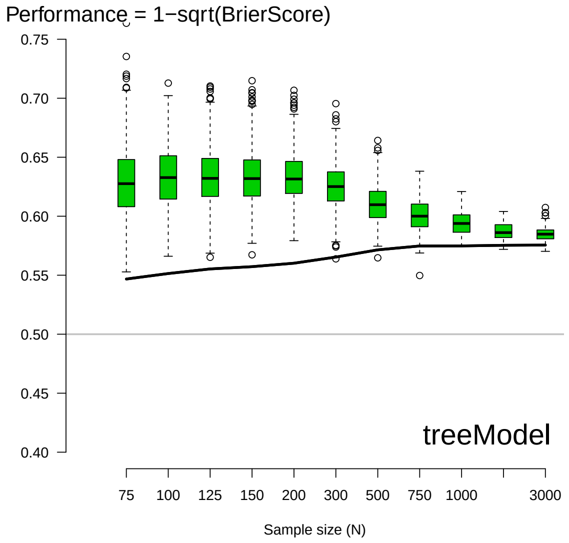
Apparent performance



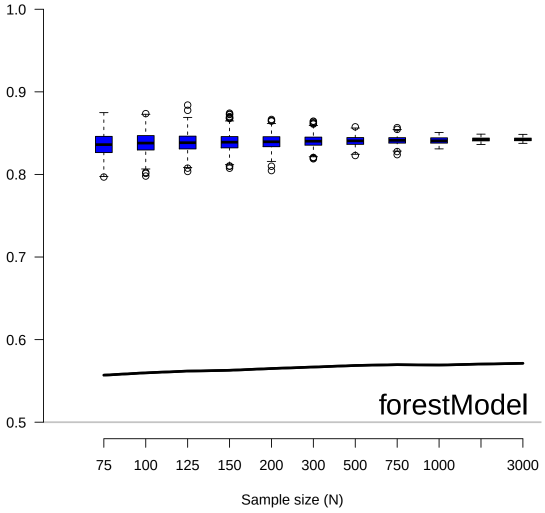
Apparent performance



Apparent performance



Apparent performance



The .632+ bootstrap estimate

Two problems:

1. The Bootstrap cross-validation estimate **underestimates** the expected performance
2. the apparent performance **overestimates** the expected performance.

One solution:

$$\text{BootCV} + \omega(\text{AppErr} - \text{BootCV})$$

The .632+ bootstrap estimate

The no-information performance assesses the overfitting by permutation

$$\text{NoInf} = \sum_{j=1}^{n} \sum_{i=1}^{n} \frac{\{Y_i - M_n(X_j)\}^2}{n^2}$$

Following Efron & Tibshirani . . .

$$\hat{\omega}_{632^+} = .632 / \left(1 - .368 \frac{\text{App} - \text{BootCV}}{\text{App} - \text{NoInf}}\right)$$

This yields the so-called .632$^+$ bootstrap estimator:

$$\text{Boot}.632^+ = \text{BootCV} + \hat{\omega}_{632^+}(\text{App} - \text{BootCV})$$

The .632+ bootstrap estimate

Following Efron & Tibshirani . . .

$$\hat{\omega}_{632^+} = .632 / \left(1 - .368 \frac{\text{App} - \text{BootCV}}{\text{App} - \text{NoInf}}\right)$$

This yields the so-called $.632^+$ bootstrap estimator:

$$\text{Boot}.632^+ = \text{BootCV} + \hat{\omega}_{632^+}(\text{App} - \text{BootCV})$$

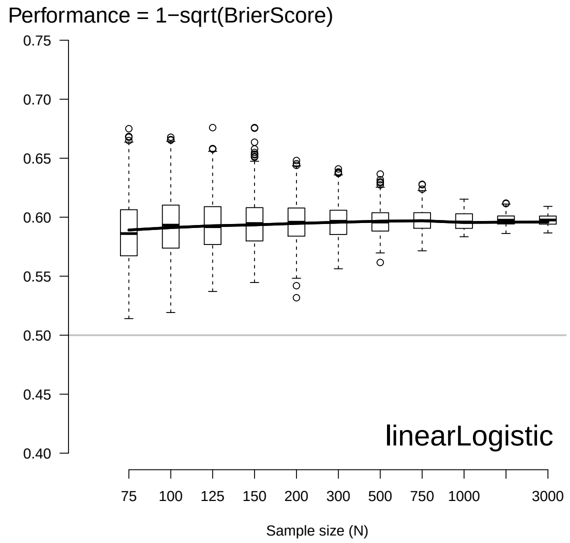
Advantages

- improves on App and BootCV

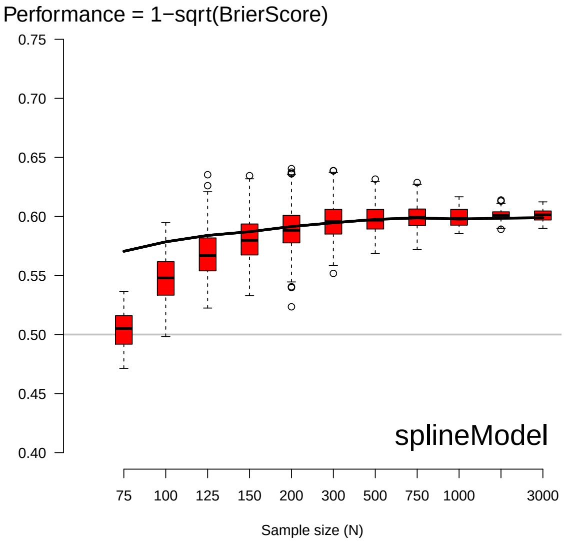
Disadvantages

- approximation does not work for all strategies . . .

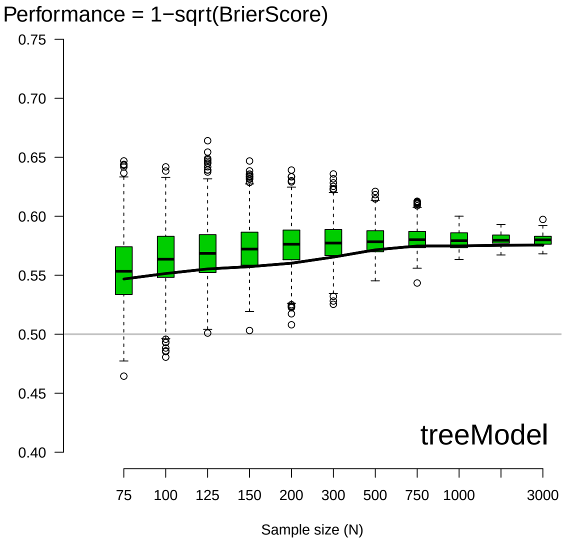
The .632+ bootstrap estimate



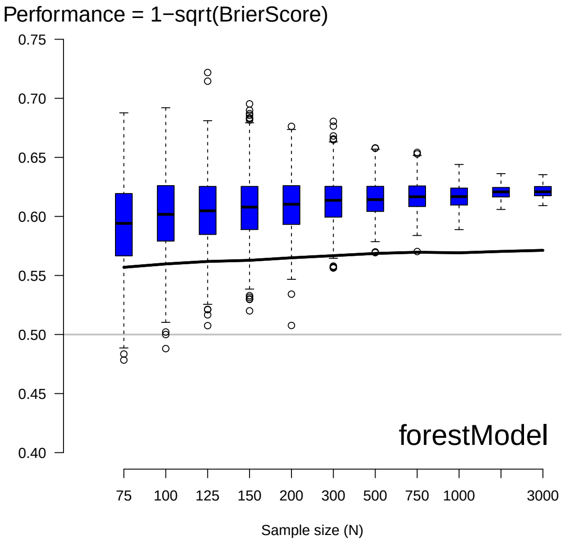
The .632+ bootstrap estimate



The .632+ bootstrap estimate



The .632+ bootstrap estimate



Conditional performance

For **each training data set** compute the absolute distance between the estimated performance (LOOCV, BOOTCV) and the conditional performance (huge validation data set).

Absolute conditional bias in %:

	Logistic		Spline		Forest		Tree	
N	LOOCV	BootCV	LOOCV	BootCV	LOOCV	BootCV	LOOCV	BootCV
75	2.49	2.67	3.23	6.48	3.22	2.95	4.49	3.21
100	2.27	2.27	2.77	3.3	3.18	2.89	3.39	2.57
125	1.98	1.95	2.11	2.67	2.59	2.37	3.02	2.5
150	1.47	1.47	1.61	1.9	1.9	1.74	2.94	2.01
200	1.53	1.49	1.7	1.6	2.02	1.82	2.32	1.88
300	1.14	1.15	1.21	1.19	1.54	1.53	1.89	1.53
500	0.97	1	0.93	0.97	1.32	1.22	1.48	1.24
750	0.77	0.76	0.78	0.77	1.09	1.05	1.37	1.01
1000	0.64	0.62	0.65	0.64	0.92	0.8	1.23	0.86
2000	0.45	0.46	0.45	0.44	0.57	0.51	0.79	0.58
3000	0.43	0.43	0.43	0.42	0.52	0.46	0.63	0.47

Conclusions from the simulation study

- The leave-one-out cross-validation estimate has a small bias for the conditional performance but a large variance
- The leave-one-out bootstrap reduces the variance and has comparable bias for the conditional performance if the model uncertainty is low.
- The Efron & Tibshirani .632+ method has been developed for the misclassification error. It does not work for the Brier score for all strategies.

Decomposition of the leave-one-out bootstrap estimate

$$LOOBOOT(\mathcal{S}, m) = \underbrace{\frac{1}{n}\sum_{i=1}^{n}\frac{1}{K_i}\sum_{b:i\notin D_b^{\texttt{train}}}\left\{Y_i - m_{K_i}^{\texttt{train}}(X_i)\right\}^2}_{\text{Estimated model accuracy}}$$

$$+ \underbrace{\frac{1}{n}\sum_{i=1}^{n}\frac{1}{K_i}\sum_{b:i\notin D_b^{\texttt{train}}}\left\{M_b^{\texttt{train}}(X_i) - m_{K_i}^{\texttt{train}}(X_i)\right\}^2}_{\text{Estimated model uncertainty}}$$

where

$$m_{K_i}^{\texttt{train}}(X_i) = \frac{1}{K_i}\sum_{b:i\notin D_b^{\texttt{train}}} M_b^{\texttt{train}}(X_i).$$

is the average prediction at X_i of the bootstrap training models which did not include i.

Interpretation of the decomposition

(I) The estimated model accuracy could be a good (consistent?) estimate of the conditional performance of the average model . . . if we hope that the performance of the current model is similar to that of the average model.

(II) The estimated model uncertainty could be used to compare modelling strategies in how confident they are about the predictions, at fixed X_{new}, and also across the population:

$$\text{Confidence score} = 1 - \sqrt{\text{Estimated model uncertainty}}$$

Note: For most strategies there is no explicit (asymptotic) variance formula, but the (subsampling) bootstrap should work.

Summary & discussion

- A statistical risk prediction is usually the result of two mappings:
  1. a modelling strategy selects a model
  2. the model predicts the probability of an event
- The expected prediction performance can be decomposed into model accuracy and model uncertainty
- The mathematical properties of some commonly applied performance estimates are not very clear. How to choose size of bootstrap subsamples? [1]
- Many authors (including myself) were not very clear with respect to which parameter is estimated
- Confidence scores may be useful as a *model free* measure of model uncertainty for comparing strategies.

[1] The bootstrap 632+ estimate likes random forests.

That's all Folks!