Advanced Statistical Topics in Health Research A

Day 3 (November 8, 2023)

Trees and forests

Mark Bech Knudsen (mark.bech.knudsen@sund.ku.dk)

Thomas Alexander Gerds (tag@biostat.ku.dk)

Section of Biostatistics, Department of Public Health

Outline of today's topics

8:15–9:30ish	• Modeling cultures • Model selection • Decision trees	(Thomas Gerds)
9:30–15ish	• From trees to forests • Tuning random forests • Variable importance • Interpretable machine learning tools	(Mark Knudsen)

Software Overview

	Outcome				
Package	Continuous	Binary	Survival	Comp. risks	Method
rpart[1]	X	X	X		Tree
randomForest[2]	X	X			Forest
party[3]	X	X	X		Tree/Forest
randomForestSRC[4]	X	X	X	X	Forest
ranger[5]	X	X	X		Forest

[1] `rpart` Therneau, Atkinson and Ripley

[2] `randomForest` Liaw and Wiener (based on Breiman and Cutler)

[3] `ctree, cforest` Hothorn

[4] `rfsrc` Ishwaran

[5] `ranger` Wright and Ziegler

Targets of analysis: Random forests are used to

- predict individual outcome
- rank and select variables

in particular in high dimensional settings where the number of variables exceeds the number of subjects in the dataset.

Targets of analysis: Random forests are used to

- predict individual outcome
- rank and select variables

in particular in high dimensional settings where the number of variables exceeds the number of subjects in the dataset.

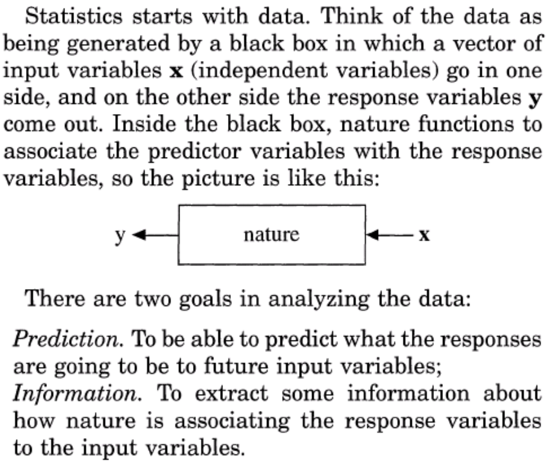
Example: binary outcome Y

$$Y = \begin{cases} 1 & \text{event} \\ 0 & \text{no event} \end{cases}$$

Type	Prediction	Error
Class	c = either 0 or 1	Y==c
Probability	p= value between 0 and 1	$(Y - p)^2$

Random forest can be an alternative to logistic regression

The two cultures



Statistics starts with data. Think of the data as being generated by a black box in which a vector of input variables **x** (independent variables) go in one side, and on the other side the response variables **y** come out. Inside the black box, nature functions to associate the predictor variables with the response variables, so the picture is like this:

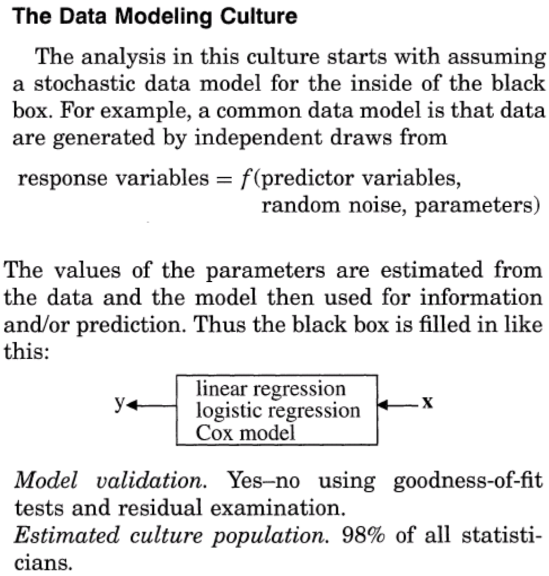
$$\begin{tikzcd} y & \boxed{\text{nature}} \arrow[l] & \mathbf{x} \arrow[l] \end{tikzcd}$$

There are two goals in analyzing the data:

Prediction. To be able to predict what the responses are going to be to future input variables;
Information. To extract some information about how nature is associating the response variables to the input variables.

L. Breiman. Statistical modeling: The two cultures. Statistical Science, 16 (3):199-215, 2001.

The two cultures

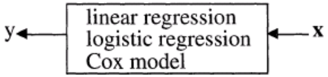


The Data Modeling Culture

The analysis in this culture starts with assuming a stochastic data model for the inside of the black box. For example, a common data model is that data are generated by independent draws from

response variables = f(predictor variables,
random noise, parameters)

The values of the parameters are estimated from the data and the model then used for information and/or prediction. Thus the black box is filled in like this:

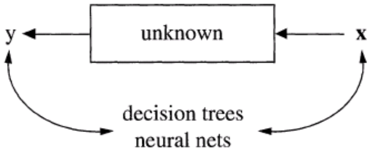


Model validation. Yes–no using goodness-of-fit tests and residual examination.
Estimated culture population. 98% of all statisticians.

The two cultures

The Algorithmic Modeling Culture

The analysis in this culture considers the inside of the box complex and unknown. Their approach is to find a function $f(\mathbf{x})$—an algorithm that operates on $\mathbf{x}$ to predict the responses $\mathbf{y}$. Their black box looks like this:



Model validation. Measured by predictive accuracy.
Estimated culture population. 2% of statisticians, many in other fields.

Example: epo study

Anaemia is a deficiency of red blood cells and/or hemoglobin and an additional risk factor for cancer patients.

Randomized placebo controlled trial[6]: does treatment with epoetin beta – epo – (300 U/kg) enhance hemoglobin concentration level and improve survival chances?

Henke et al. 2006 identified the c20 expression (erythropoietin receptor status) as a new biomarker for the prognosis of locoregional progression-free survival.

[6] Henke et al. Do erythropoietin receptors on cancer cells explain unexpected clinical findings? J Clin Oncol, 24(29):4708-4713, 2006.

Treatment

The study includes 149 [7]head and neck cancer patients with a tumor located in the oropharynx (36%), the oral cavity (27%), the larynx (14%) or in the hypopharynx (23%).

One of the treatments was radiotherapy following

	Resection		
	Complete	Incomplete	No
Placebo	35	14	25
Epo	36	14	25

[7]with non-missing blood values

Outcome

Blood hemoglobin levels were measured weekly during radiotherapy (7 weeks).

Treatment with epoetin beta was defined successful when the hemoglobin level increased sufficiently. For patient i set

$$Y_i = \begin{cases} 1 & \text{treatment successful} \\ 0 & \text{treatment failed} \end{cases}$$

Target

Patient no.	Treatment successful	Predicted probability
1	0	P_1
2	0	P_2
3	1	P_3
4	1	P_4
5	0	P_5
6	1	P_6
7	1	P_7
.	.	.
.	.	.

Predictors

Age	min: 41 y, median: 59 y, max: 80 y
Gender	male: 85%, female: 15%
Baseline hemoglobin	mean: 12.03 g/dl, std: 1.45
Treatment	epo: 50%, placebo 50%
Resection	complete: 48%, incomplete: 19%, no resection: 34%
Epo receptor status	neg: 32%, pos: 68%

Logistic regression

Response: treatment successful yes/no

Factor	OddsRatio	StandardError	CI.95	pValue
(Intercept)	0.00	4.01	–	< 0.0001
Age	0.97	0.03	$[0.91; 1.03]$	0.2807
Sex:female	4.71	0.84	$[0.91; 26.02]$	0.0657
HbBase	3.25	0.27	$[1.99; 5.91]$	< 0.0001
Treatment:Epo	90.92	0.76	$[23.9; 493.41]$	< 0.0001
Resection:Incompl	1.75	0.81	$[0.36; 9.03]$	0.4924
Resection:Compl	4.14	0.69	$[1.13; 17.36]$	0.0395
Receptor:positive	5.81	0.66	$[1.72; 23.39]$	0.0076

Logistic regression

Response: treatment successful yes/no

Factor	OddsRatio	StandardError	CI.95	pValue
(Intercept)	0.00	4.01	–	< 0.0001
Age	0.97	0.03	[0.91; 1.03]	0.2807
Sex:female	4.71	0.84	[0.91; 26.02]	0.0657
HbBase	3.25	0.27	[1.99; 5.91]	< 0.0001
Treatment:Epo	90.92	0.76	[23.9; 493.41]	< 0.0001
Resection:Incompl	1.75	0.81	[0.36; 9.03]	0.4924
Resection:Compl	4.14	0.69	[1.13; 17.36]	0.0395
Receptor:positive	5.81	0.66	[1.72; 23.39]	0.0076

Does that mean everyone should be treated?

The model provides information for a single patient

For example: the predicted probability that a 51 year old man with complete tumor resection and baseline hemoglobin level 12.6 g/dl reaches the target hemoglobin level (Y_i=1) is

Epo treatment: 97.4%

Placebo group: 29.2 %

The model provides information for a single patient

For example: the predicted probability that a 51 year old man with complete tumor resection and baseline hemoglobin level 12.6 *g/dl* reaches the target hemoglobin level ($Y_i=1$) is

Epo treatment: 97.4%

Placebo group: 29.2 %

If a similar patient has baseline hemoglobin level 14.8 *g/dl* then the model predicts:

Epo treatment: 99.8%

Placebo group: 84.7 %

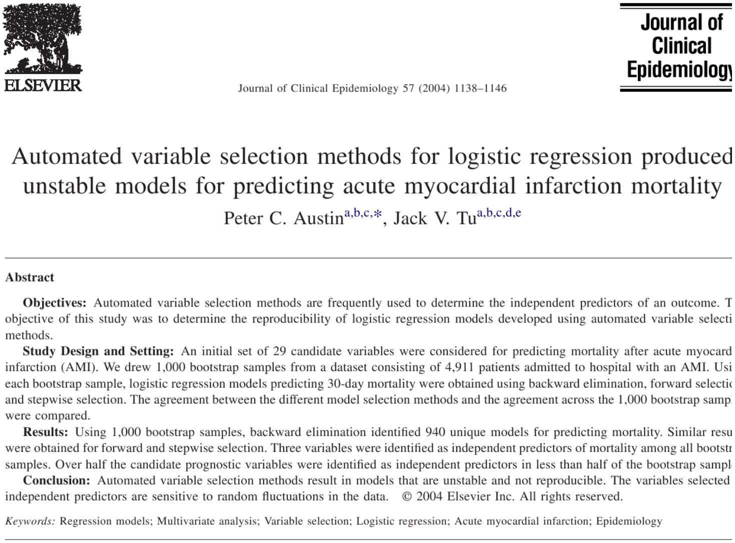
Model selection

Very many different 'logistic regression models' can be constructed by selecting subsets of variables, transformations, and interactions of variables.

"Standard" multiple (logistic) regression works if

- the number of predictors is not too large, and substantially smaller than the sample size
- the decision maker has a-priory knowledge about which variables to put into the model

Ad-hoc model selection algorithms, like automated backward elimination, do not lead to reproducible prediction models.


Journal of Clinical Epidemiology 57 (2004) 1138–1146

Journal of Clinical Epidemiolog

Automated variable selection methods for logistic regression produced unstable models for predicting acute myocardial infarction mortality

Peter C. Austin[a,b,c,*], Jack V. Tu[a,b,c,d,e]

Abstract

Objectives: Automated variable selection methods are frequently used to determine the independent predictors of an outcome. T objective of this study was to determine the reproducibility of logistic regression models developed using automated variable selecti methods.

Study Design and Setting: An initial set of 29 candidate variables were considered for predicting mortality after acute myocard infarction (AMI). We drew 1,000 bootstrap samples from a dataset consisting of 4,911 patients admitted to hospital with an AMI. Usi each bootstrap sample, logistic regression models predicting 30-day mortality were obtained using backward elimination, forward selecti and stepwise selection. The agreement between the different model selection methods and the agreement across the 1,000 bootstrap samp were compared.

Results: Using 1,000 bootstrap samples, backward elimination identified 940 unique models for predicting mortality. Similar resu were obtained for forward and stepwise selection. Three variables were identified as independent predictors of mortality among all bootstr samples. Over half the candidate prognostic variables were identified as independent predictors in less than half of the bootstrap sampl

Conclusion: Automated variable selection methods result in models that are unstable and not reproducible. The variables selected independent predictors are sensitive to random fluctuations in the data. © 2004 Elsevier Inc. All rights reserved.

Keywords: Regression models; Multivariate analysis; Variable selection; Logistic regression; Acute myocardial infarction; Epidemiology

Backward elimination

On full data (n=149):

```
library(rms)
full <- lrm(Y~age+sex+HbBase+Treat+Resection+Receptor,data=Epo)
fastbw(full)
bw <- lrm(Y~sex+HbBase+Treat+Receptor,data=Epo)
```

```
 Deleted   Chi-Sq d.f. P      Residual d.f. P      AIC
 age       1.16   1    0.2807 1.16     1    0.2807 -0.84
 Resection 3.75   2    0.1532 4.92     3    0.1781 -1.08

Approximate Estimates after Deleting Factors

                     Coef   S.E. Wald Z            P
Intercept         -11.257 3.0129 -3.736 0.00018665428
sex=male           -1.672 0.8221 -2.034 0.04195853231
HbBase              1.099 0.2719  4.043 0.00005279348
Treat=Placebo      -3.843 0.6992 -5.496 0.00000003887
Receptor=positive   1.413 0.6355  2.224 0.02615849462

Factors in Final Model

[1] sex      HbBase   Treat    Receptor
```

Backward elimination

On reduced data (n=130):

```
library(rms)
set.seed(17)
Epo17 <- Epo[sample(1:149,replace=FALSE,size=130),]
sub <- lrm(Y~age+sex+HbBase+Treat+Resection+Receptor,
      data=Epo17)
fastbw(sub)
subbw <- lrm(Y~sex+Receptor+HbBase+Treat, data=Epo17)
```

```
 Deleted   Chi-Sq d.f. P      Residual d.f. P      AIC
 age       0.61   1    0.4362 0.61     1    0.4362 -1.39
 Resection 4.81   2    0.0905 5.41     3    0.1440 -0.59

Approximate Estimates after Deleting Factors

                     Coef   S.E. Wald Z           P
Intercept         -11.657 3.2936 -3.539 0.000401291
sex=male           -1.692 0.8643 -1.958 0.050277808
HbBase              1.127 0.2954  3.816 0.000135846
Treat=Placebo      -3.370 0.6971 -4.833 0.000001343
Receptor=positive   1.311 0.6639  1.974 0.048333452

Factors in Final Model

[1] sex      HbBase   Treat    Receptor
```

Predicted chance of treatment success for a new patient

```r
newpatient
```

```
  age  sex HbBase Treat Resection Receptor
1  48 male   10.8   Epo        No negative
```

```r
library(riskRegression)
pfull=predictRisk(full,newdata=newpatient)
pbw=predictRisk(bw,newdata=newpatient)
psubbw=predictRisk(subbw,newdata=newpatient)
# table results
res=cbind(round(100*c(pfull,pbw,psubbw),1))
rownames(res)=c("Full model","BW all data","BW subset")
colnames(res)=c("Predicted chance (%)")
res
```

```
            Predicted chance (%)
Full model                  16.9
BW all data                 24.1
BW subset                   47.8
```

Exercise

Load the Epo data:

```
Epo <- read.csv("http://publicifsv.sund.ku.dk/~helene/
    Epo.csv", stringsAsFactors=TRUE)
```

Epo data set is ready for analysis

- Choose your favorite seed to generate a subsample (n=130) of the Epo data
- Run backward elimination with function rms::fastbw
- Predict the outcome for the following new patient

```
newpatient <- read.csv("http://publicifsv.sund.ku.dk/~
    helene/newpatient", stringsAsFactors=TRUE)
```

- Report the selected variables and the predicted risk

Decision trees

A Conversation of Richard Olshen with Leo Breiman

...

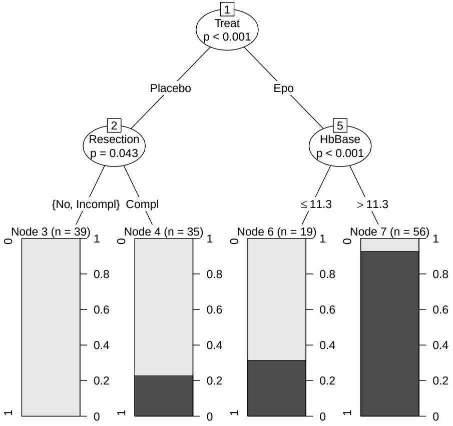
Olshen: *What about arcing, bagging and boosting?*
Breiman: *Okay. Yeah. This is fascinating stuff, Richard. In the last five years, there have been some really big breakthroughs in prediction. And I think combining predictors is one of the two big breakthroughs. And the idea of this was, okay, that suppose you take CART, which is a pretty good classifier, but not a great classifier. I mean, for instance, neural nets do a much better job.*
Olshen: *Well, suitably trained?*
Breiman: **Suitably trained**.
Olshen: *Against an untrained CART?*
Breiman: *Right. Exactly. And I think I was thinking about this. I had written an article on subset selection in linear regression. I had realized then that* **subset selection in linear regression is really a very unstable procedure**. *If you tamper with the data just a little bit, the first best five variable regression may change to another set of five variables. And so I thought, "Okay. We can stabilize this by just perturbing the data a little and get the best five variable predictor. Perturb it again. Get the best five variable predictor and then average all these five variable predictors." And sure enough, that worked out beautifully. This was published in an article in the Annals (Breiman, 1996b).*

...

Statist. Sci. Volume 16, Issue 2 (2001), 184-198.

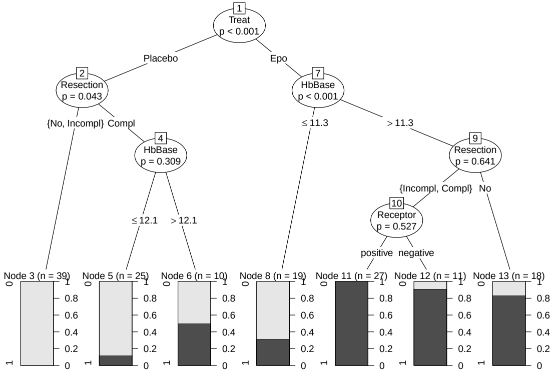
Conditional inference trees are not very deep (by default)

```
library(party)
plot(ctree(Y~age+sex+HbBase+Treat+Resection+Receptor,data=
    Epo))
```



A deeper more greedy tree

```
library(party)
plot(ctree(Y~age+sex+HbBase+Treat+Resection+Receptor,data=
    Epo,controls=ctree_control(mincriterion = .01)))
```



Classification trees

A tree model is a form of recursive partitioning.

It lets the data decide which variables are important and where to place cut-offs in continuous variables.

In general terms, the purpose of the analyzes via tree-building algorithms is to determine a set of splits that permit accurate prediction or classification of cases.

In other words: a tree is a combination of many medical tests.

Roughly, the algorithm works as follows:

1. Find the predictor so that the best possible split on that predictor optimizes some statistical criterion over all possible splits on the other predictors.
2. For ordinal and continuous predictors, the split is of the form $X < c$ versus $X \geq c$.
3. Repeat step 1 within each previously formed subset.
4. Proceed until fewer than k observations remain to be split, or until nothing is gained from further splitting, i.e. the tree is fully grown.
5. The tree is pruned according to some criterion.

Characters of classification trees

- Trees are specifically designed for accurate classification/prediction
- Results have a graphical representation and are easy to interpret
- No model assumptions
- Recursive partitioning can identify complex interactions
- One can introduce different costs of miss-classification in the three

But:

- Trees are not robust against even small perturbations of the data.
- It is quite easy to over-fit the data.
- Trees are weak learners

Random forests

Outline of today's remaining topics

8:15–9:30ish	• Modeling cultures • Model selection • Decision trees	(Thomas Gerds)
9:30–15ish	• From trees to forests • Tuning random forests • Variable importance • Interpretable machine learning tools	(Mark Knudsen)

From trees to forests

Decision trees are nice because:

- They produce results that are easy to interpret
- They require no model assumptions

But:

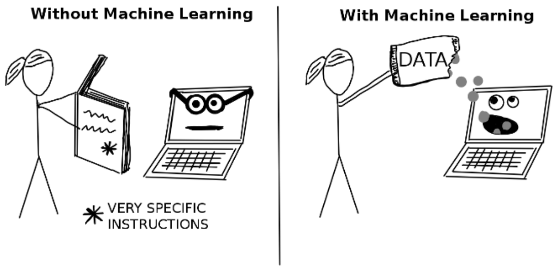
- They easily overfit the data
- Trees are weak learners

A random forest is a machine learning method that combines a large collection of decision trees to construct a strong learner

Machine learning versus classical statistics

When does it make sense to apply "machine learning"?

- Little knowledge of the system we wish to analyze
- No prespecified hypotheses
- Focus on prediction rather than understanding



Why do we need prediction in medical research? (Ex. 1)

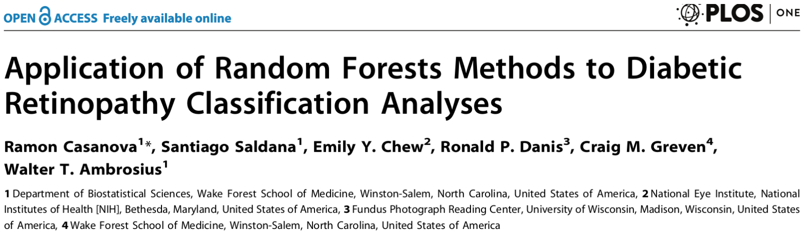
Combined test at 12-week pregnancy scan

- the age of the mother, a blood sample and a measurement of fetus' neck are combined to provide a prediction of the risk of the baby having Down's syndrome, Edwards' syndrome or Patau's syndrome
- those with higher-risk results can have a subsequent diagnostic test that can tell for sure if the baby has Down's syndrome, Edwards' syndrome or Patau's syndrome but can in rare cases cause miscarriage

Why do we need prediction in medical research? (Ex. 2)

Early detection of diabetic retinopathy

- Diabetic retinopathy is a leading cause of blindness
- Diabetic retinopathy may go unnoticed until it is too late for effective treatment
- A prediction model based on fundus photography data can help detect patients with diabetic retinopathy in time for effective therapeutic intervention



OPEN ACCESS Freely available online PLOS ONE

Application of Random Forests Methods to Diabetic Retinopathy Classification Analyses

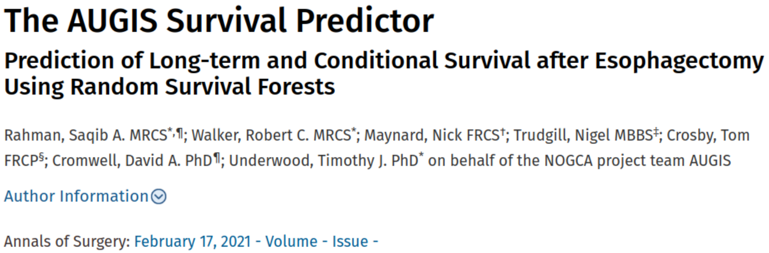
Ramon Casanova[1]*, Santiago Saldana[1], Emily Y. Chew[2], Ronald P. Danis[3], Craig M. Greven[4], Walter T. Ambrosius[1]

1 Department of Biostatistical Sciences, Wake Forest School of Medicine, Winston-Salem, North Carolina, United States of America, **2** National Eye Institute, National Institutes of Health [NIH], Bethesda, Maryland, United States of America, **3** Fundus Photograph Reading Center, University of Wisconsin, Madison, Wisconsin, United States of America, **4** Wake Forest School of Medicine, Winston-Salem, North Carolina, United States of America

Why do we need prediction in medical research? (Ex. 3)

Prediction of long-term survival after esophagectomy

- Esophagectomy is a highly invasive surgical treatment
- A prediction model can combine multiple risk factors to provide personalized survival predictions
- This can further enable identification of high-risk patients for enhanced surveillance and/or treatment intensification



The AUGIS Survival Predictor

Prediction of Long-term and Conditional Survival after Esophagectomy Using Random Survival Forests

Rahman, Saqib A. MRCS[*,¶]; Walker, Robert C. MRCS[*]; Maynard, Nick FRCS[†]; Trudgill, Nigel MBBS[‡]; Crosby, Tom FRCP[§]; Cromwell, David A. PhD[¶]; Underwood, Timothy J. PhD[*] on behalf of the NOGCA project team AUGIS

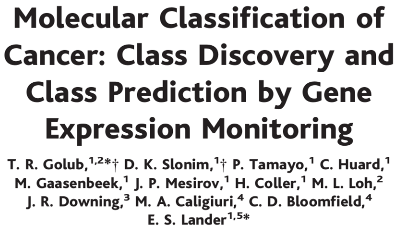
Author Information

Annals of Surgery: February 17, 2021 - Volume - Issue -

Why do we need prediction in medical research? (Ex. 4)

Cancer class classification

- Accurate cancer classification can be used to target specific therapies to distinct tumor types
- A prediction model can be used to provide a data-based classification algorithm based on gene expression monitoring[8]



Molecular Classification of Cancer: Class Discovery and Class Prediction by Gene Expression Monitoring

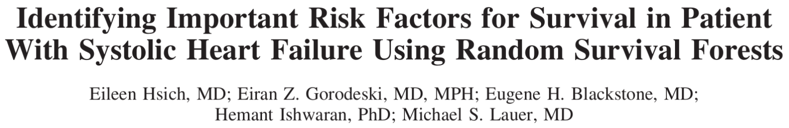
T. R. Golub,[1,2*†] D. K. Slonim,[1†] P. Tamayo,[1] C. Huard,[1] M. Gaasenbeek,[1] J. P. Mesirov,[1] H. Coller,[1] M. L. Loh,[2] J. R. Downing,[3] M. A. Caligiuri,[4] C. D. Bloomfield,[4] E. S. Lander[1,5*]

[8] this is where we will end today, using a random forest ($n = 38$, $p = 3051$)

What else can we use a random forest for? (Ex. 5)

Identifying risk factors for survival in systolic heart failure patients

- Random forests automatically detects non-linear relationships and interactions among variables
- Can be used for variable selection even in settings with many more predictors than observations
- We lose the "usual" statistical inference but can explore effects on prediction with graphical tools



Identifying Important Risk Factors for Survival in Patient With Systolic Heart Failure Using Random Survival Forests

Eileen Hsich, MD; Eiran Z. Gorodeski, MD, MPH; Eugene H. Blackstone, MD; Hemant Ishwaran, PhD; Michael S. Lauer, MD

Random forests as a classical machine learning method

1. Let's try to understand what goes on inside the forest
   - Recap on basic machine learning techniques

2. Applying random forests
   - Hyperparameter selection/tuning

3. Interpretability of random forests[9]
   - Variable importance
   - Partial Dependence Plots (PDPs)

[9] (and machine learning in general)

From trees to forests

From trees to forests

What is a forest[10] ...

A random forest combines the information from a collection of *weak learners* = randomized decision trees

1. Each tree is built on a bootstrap sample of the data
2. Only a small number of randomly selected predictor variables are used to find the best split of each node

The forest predictions are averages over the individual trees

[10] Leo Breiman (2001). "Random Forests". Machine Learning 45 (1), 5-32,

Classical machine learning techniques utilized inside the forest

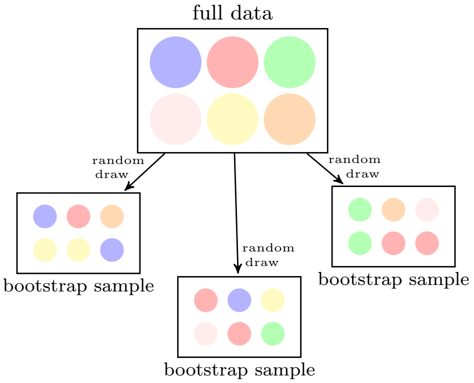
1. Bootstrap sampling
2. Nearest neighbor smoothing
3. Ensemble learning

Classical machine learning techniques utilized inside the forest

1. Bootstrap sampling
2. Nearest neighbor smoothing
3. Ensemble learning

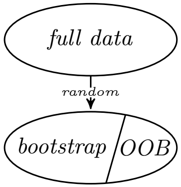
Trees are built on bootstrapped subsamples of the data

The purpose of bootstrapping is to create new pseudo samples, each of which will be used to fit a tree



Trees are built on bootstrapped subsamples of the data

Each time we draw a random bootstrap sample:



inbag : subjects in the bootstrap sample

oob : subjects not in the bootstrap sample

Bootstrapping

```
Epo <- read.csv("http://publicifsv.sund.ku.dk/~helene/
    Epo.csv", stringsAsFactors=TRUE)
```

There are $n = 149$ subjects in the Epo data

```
n <- nrow(Epo)
```

Let's get a bootstrap sample (of same size) of these subjects

Bootstrapping

Everything depends on the seed:

```
set.seed(5)
```

We draw a bootstrap sample of size *n*:

```
bootstrap.sample <- sample(1:n, n, replace=TRUE)
```

Who is included in the bootstrap sample (look at first six)?

```
head(table(bootstrap.sample))
```

```
bootstrap.sample
2 3 4 5 6 8
1 1 3 1 1 3
```

Bootstrapping

Is subject $i = 15$ in this bootstrap sample?

```
15 %in% bootstrap.sample
```

```
[1] TRUE
```

Is subject $i = 15$ inbag or oob (out-of-bag)?

Bootstrapping

Each time a patient is left oob, we can compare the prediction for this patient with the outcome that was observed for sample patient

- model validation (which we get back to)
- variable importance measures (which we get back to)

Classical machine learning techniques utilized inside the forest

1. Bootstrap sampling
2. Nearest neighbor smoothing
3. Ensemble learning

Classical machine learning techniques utilized inside the forest

1. Bootstrap sampling
2. Nearest neighbor smoothing
3. Ensemble learning

A tree is a nearest neighbor method

Say we only have access to two predictors of the Epo dataset:

```
  age HbBase
1  70   10.7
2  68   12.7
3  70   13.4
4  55   12.0
5  69   11.2
6  59   13.5
```

We want to estimate the probability:

$$P(Y = 1 \mid \text{age}, \text{HbBase})$$

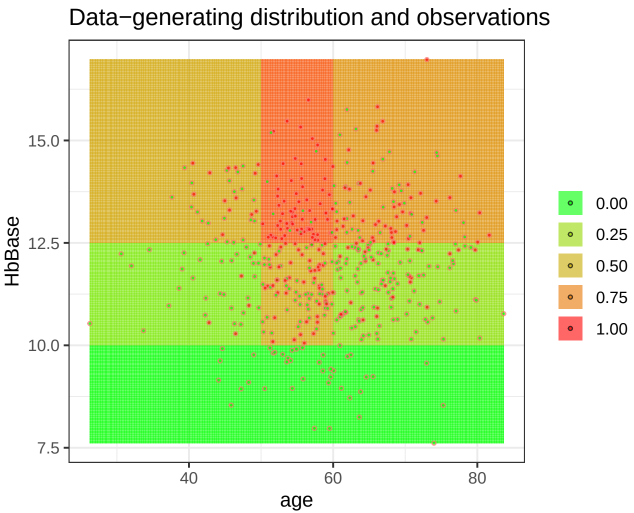
A tree is a nearest neighbor method

In the following, I have cheated

I have simulated data to imitate the Epo data

So I know the true $P(Y = 1 \mid \text{age}, \text{HbBase})$

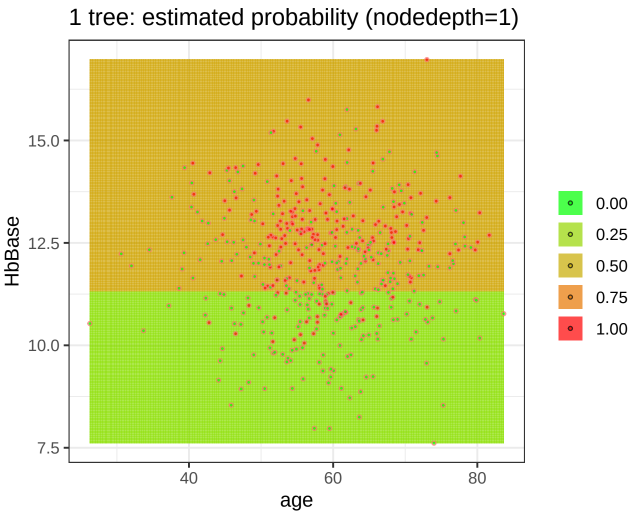
A tree is a nearest neighbor method



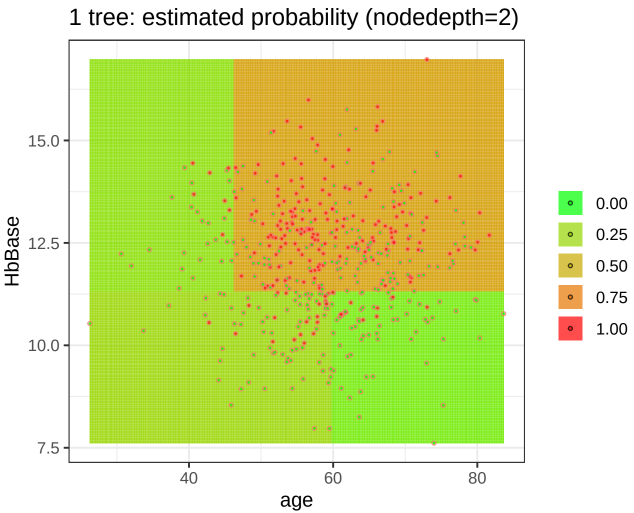
A tree is a nearest neighbor method

Grow a tree to fit $P(Y = 1 \mid \text{age}, \text{HbBase})$

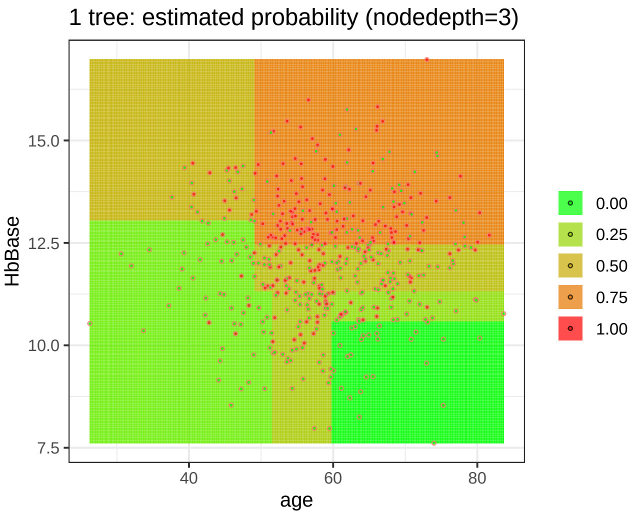
A tree is a nearest neighbor method



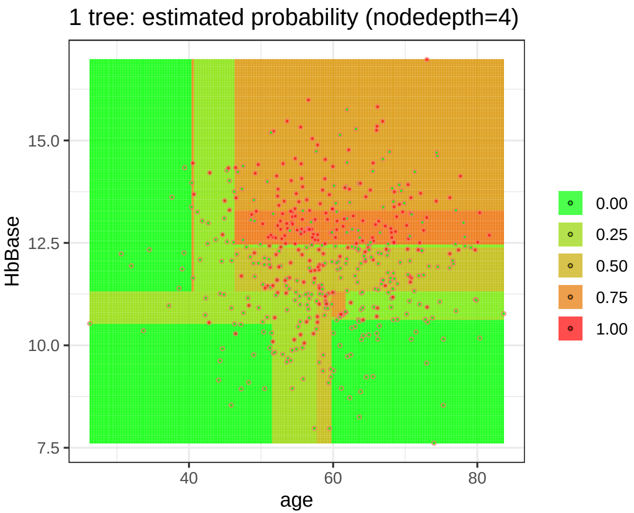
A tree is a nearest neighbor method



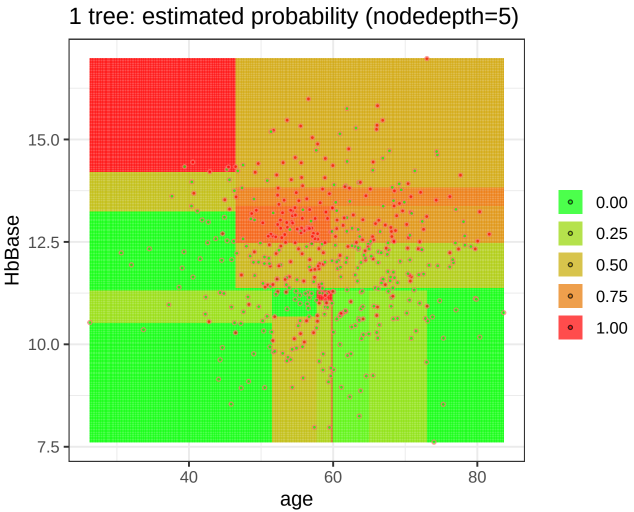
A tree is a nearest neighbor method



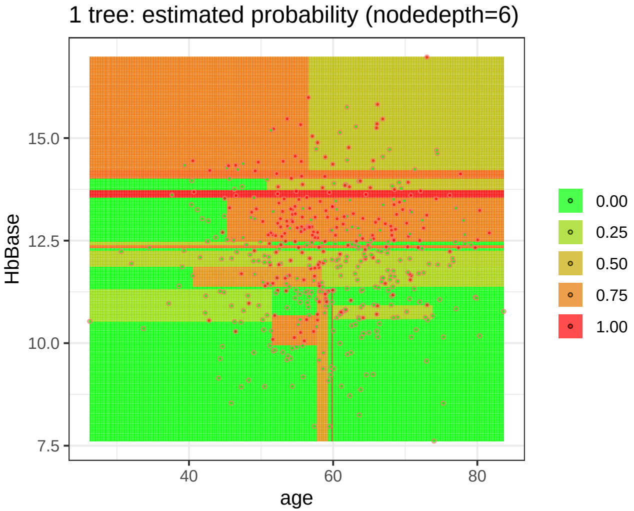
A tree is a nearest neighbor method



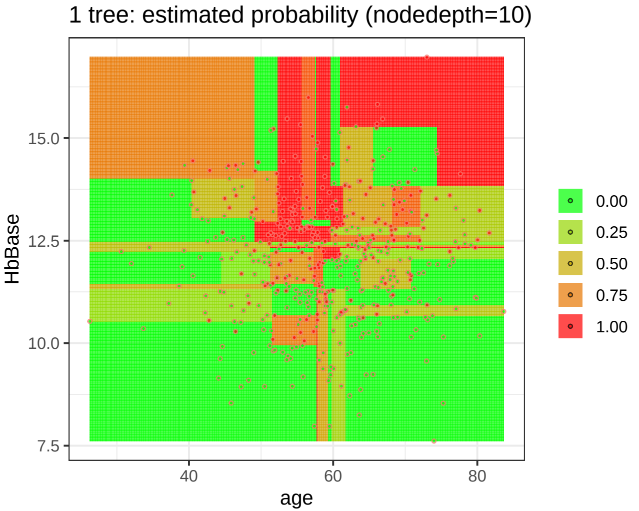
A tree is a nearest neighbor method



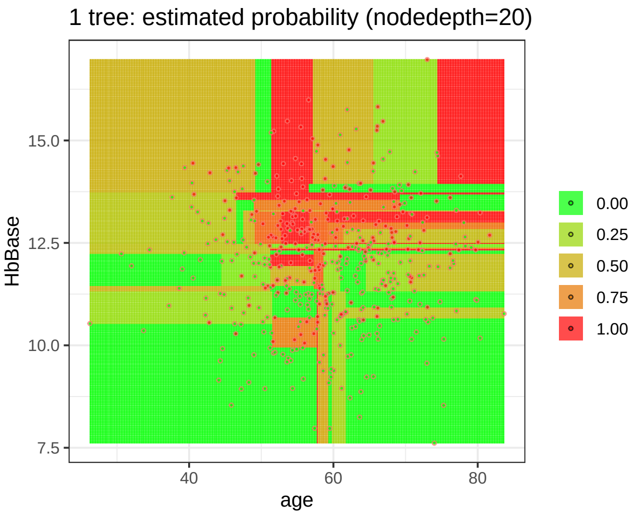
A tree is a nearest neighbor method



A tree is a nearest neighbor method



A tree is a nearest neighbor method



Classical machine learning techniques utilized inside the forest

1. Bootstrap sampling
2. Nearest neighbor smoothing
3. Ensemble learning

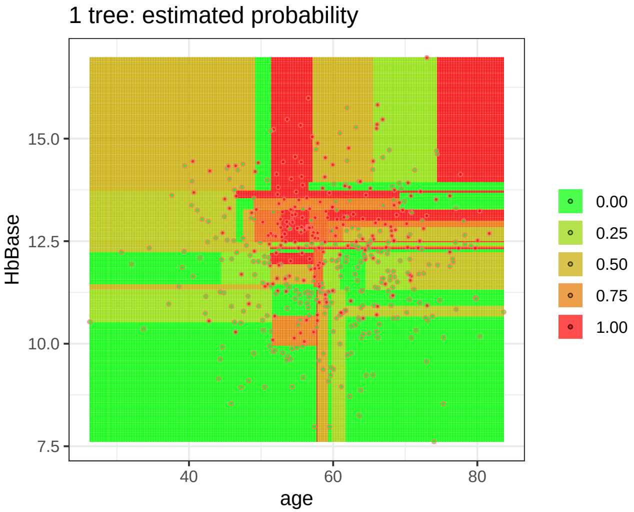
Classical machine learning techniques utilized inside the forest

1. Bootstrap sampling
2. Nearest neighbor smoothing
3. Ensemble learning

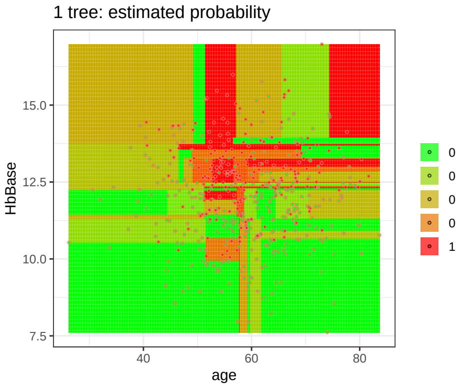
A forest is a weighted nearest neighbor method

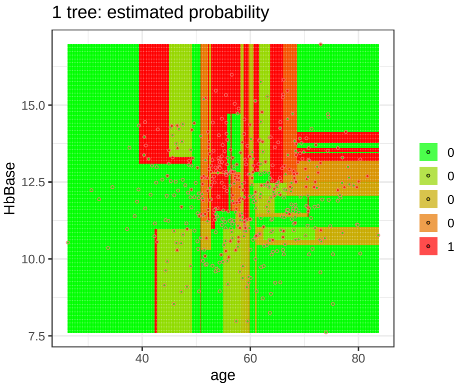
A forest takes the nearest neighbors from each tree (new tree, new seed, new bootstrap sample) to define "weighted nearest neighbors"

A forest is a weighted nearest neighbor method

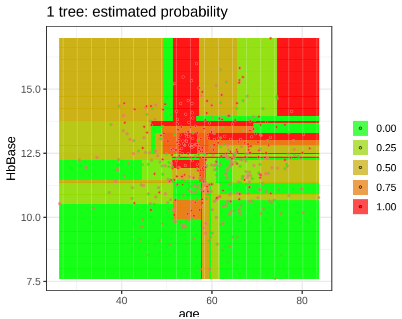


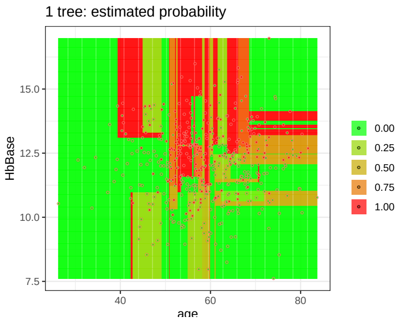
A forest is a weighted nearest neighbor method

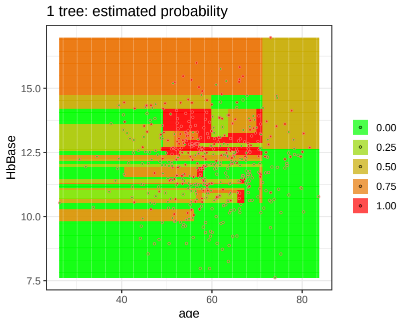


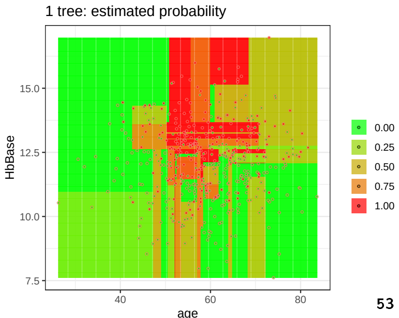


A forest is a weighted nearest neighbor method





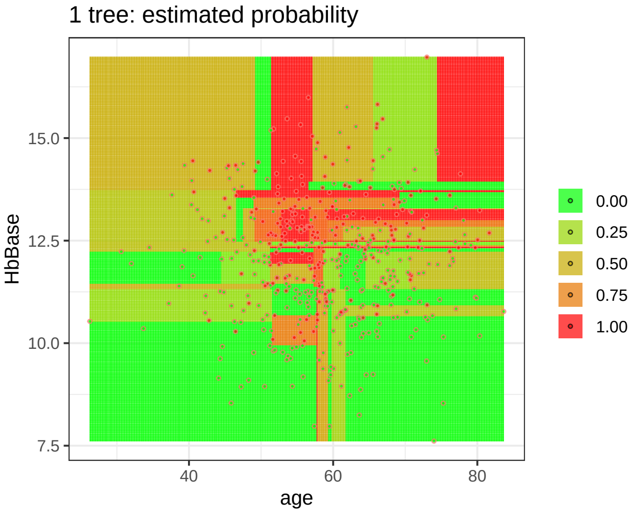




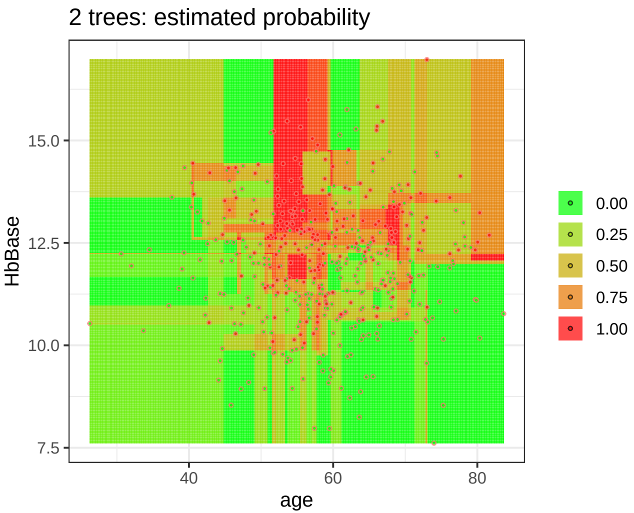
A forest is a weighted nearest neighbor method

Now, combine trees to fit $P(Y = 1 \mid \text{age}, \text{HbBase})$

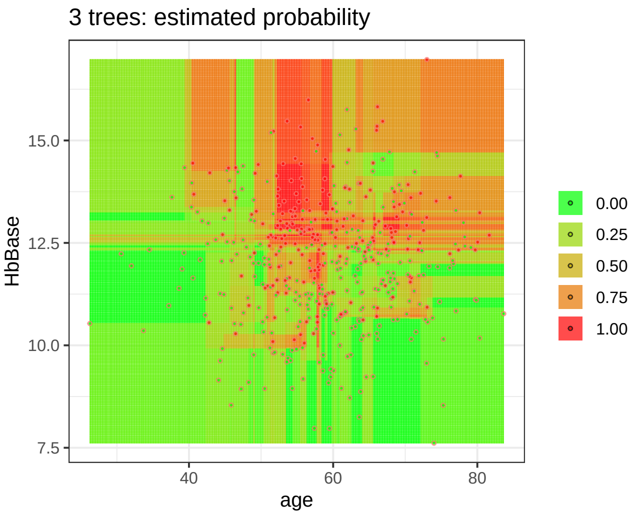
A forest is a weighted nearest neighbor method



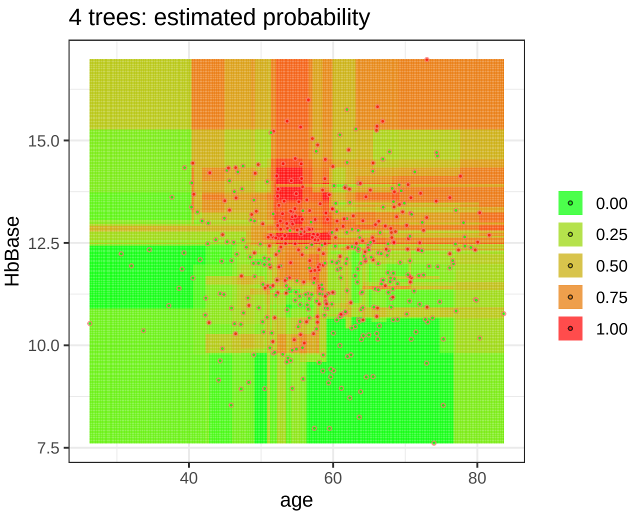
A forest is a weighted nearest neighbor method



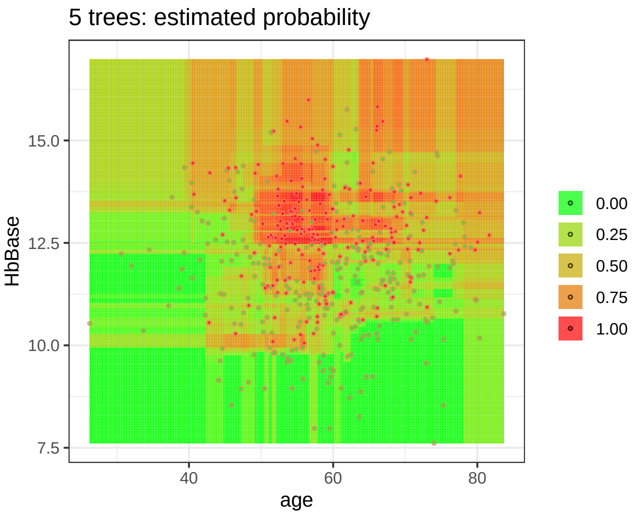
A forest is a weighted nearest neighbor method



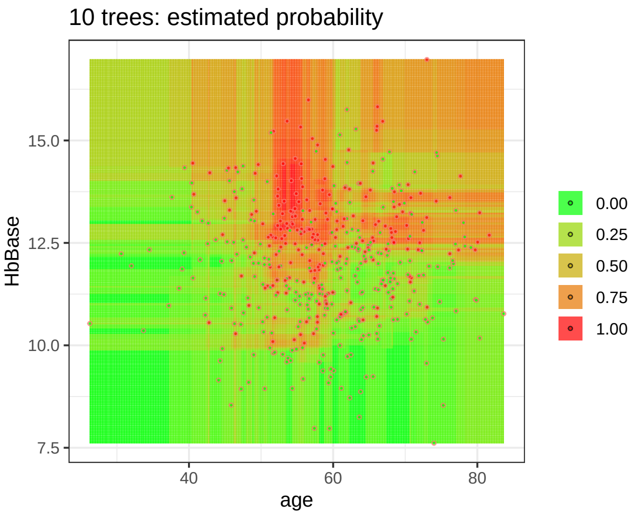
A forest is a weighted nearest neighbor method



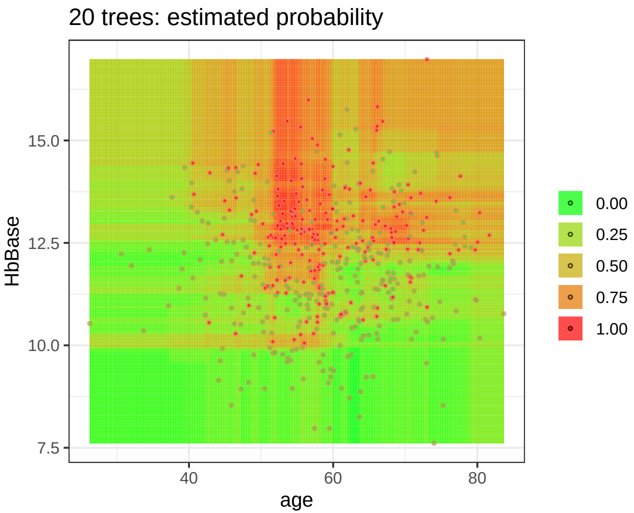
A forest is a weighted nearest neighbor method



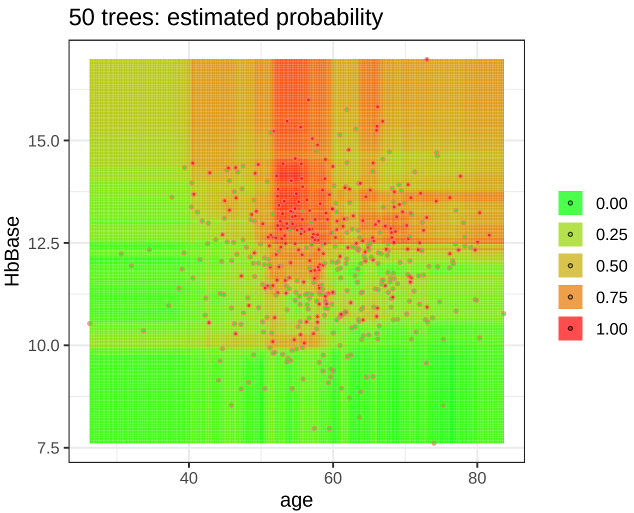
A forest is a weighted nearest neighbor method



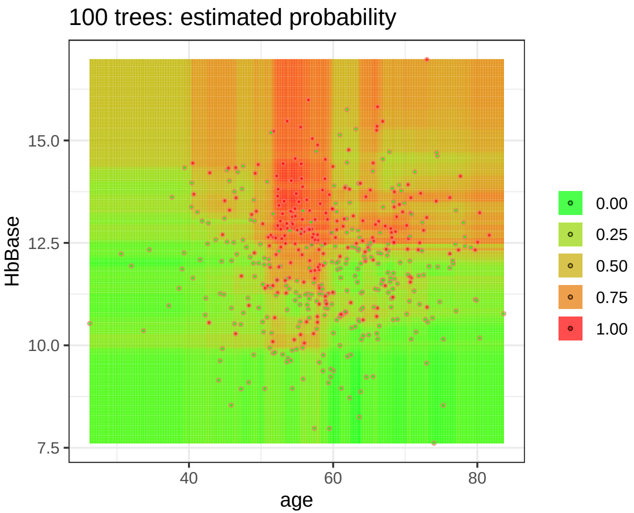
A forest is a weighted nearest neighbor method



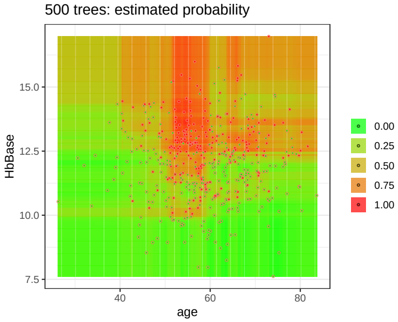
A forest is a weighted nearest neighbor method

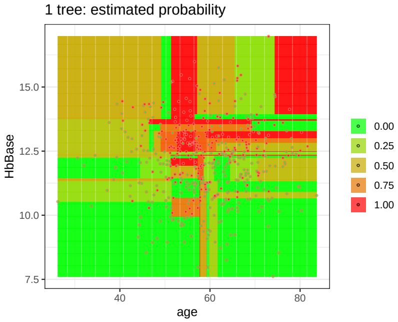


A forest is a weighted nearest neighbor method



A forest is a weighted nearest neighbor method





Random trees and forests in R

Load the package:

```
library("randomForestSRC")
```

We will start by using the software to grow single trees

```
tree1 <-
    rfsrc(Y~age+sex+HbBase+Treat+Resection,
      Epo, # data
      ntree=1, # only 1 tree!
      seed=1)  # the result depends on seed
```

Random trees and forests in R

Prediction and the oob prediction, e.g., for individual $i = 89$:

```
tree1$predicted[89]
```

```
[1] 0.7777778
```

```
tree1$predicted.oob[89]
```

```
[1] NA
```

... individual $i = 89$ was inbag

Exercise: From trees to forests

In this exercise, we will use the `rfsrc()` function from the `randomForestSRC` package to grow single trees

- The point is to assess stability of tree and forest predictions

The exercise is described in **day3-practical.pdf**

- **Exercise 1**: From trees to forests

Exercise: From trees to forests (result plot)

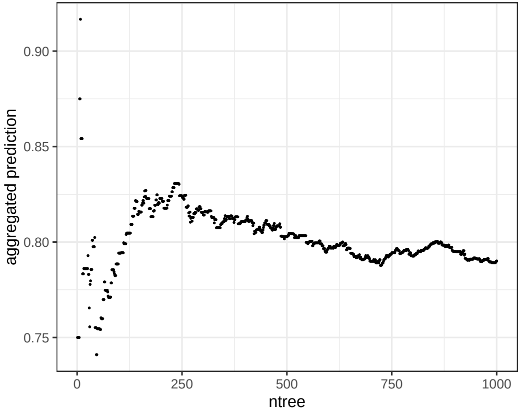
```
M <- 1000
pred <- rep(0, M)

for (ii in 1:M) {
    tree1 <- rfsrc(Y~age+sex+HbBase+Treat+Resection,
             Epo, ntree=1, seed=ii)
    pred[ii] <- tree1$predicted.oob[25]
}


pred.mean <- sapply(1:M, function(ii) {
    mean(na.omit(pred[1:ii]))
})
```

Exercise: From trees to forests (result plot)

```r
plot(pred.mean)
```



Exercise: From trees to forests (remark)

Here we produced the forest prediction ourselves across an increasing number of trees

In real life we use the implementation in R to do this automatically. Here we use 1000 trees by specifying the argument `ntree=1000`:

```
rf1 <- rfsrc(Y~age+sex+HbBase+Treat+Resection,
         Epo, ntree=1000, seed=5)
```

This gives us directly the forest prediction:

```
rf1$predicted.oob[25]
```

```
[1] 0.7528273
```

Prediction accuracy

Measuring and comparing performances of machine learning models

Predictive accuracy

Combined test at 12-week pregnancy scan

- accurate prediction is important to avoid recommending unneeded invasive subsequent diagnostic test

Early detection of diabetic retinopathy

- accurate prediction is important to discover as many patients as possible in time for effective treatment

Prediction of long-term survival after esophagectomy

- accurate prediction is important to correctly identify and attend to as many high-risk patients as possible

Predictive accuracy

Patient no.	Treatment successful	Predicted probability
1	0	P_1
2	0	P_2
3	1	P_3
4	1	P_4
5	0	P_5
6	1	P_6
7	1	P_7
.	.	.
.	.	.

Prediction error and predictive accuracy

Prediction error is measured in terms of some distance[11] between:

1) the observed outcome: Y_i

2) and the predicted probability: $\hat{P}_i = \hat{P}(Y_i = 1 \mid \text{age}_i, \text{HbBase}_i, \ldots)$

One example of a loss function is the squared error loss:

$$\mathscr{L}(Y_i, \hat{P}_i) = (Y_i - \hat{P}_i)^2$$

[11] Measured in terms of a *loss function*

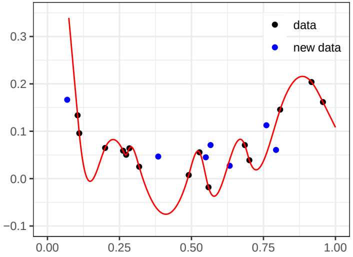
Prediction error and predictive accuracy

Machine learning 101

To measure the prediction error correctly, we cannot train the model and assess the model on the same data

Prediction error and predictive accuracy

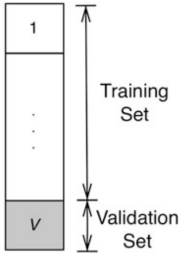
Overfitting happens when a model learns the detail and noise in the data too well so that it negatively impacts the performance of the model on new data



Evaluating a model on the same data results in overfitting.

Prediction error and predictive accuracy

To do it correctly, we can use sample splitting:



1. I create and fit my model on the training data: $\hat{P}^{\text{train}}$
2. I check the quality of my model on the validation data
   - Average of $\mathscr{L}(Y_i, \hat{P}_i^{\text{train}})$ in validation sample

Prediction error and predictive accuracy

Let's compare the predictions from 1 tree to those from a forest of 100 trees

Fix seed:

```
set.seed(5)
```

Take 10 % of original data to be our validation set:

```
val.set <- sample(1:n, n/10, replace=FALSE)
```

The rest comprise our training data:

```
train.set <- (1:n)[!(1:n) %in% val.set]
```

Prediction error and predictive accuracy

Fit 1 tree on the training data:

```
tree1.train <-
    rfsrc(Y~age+sex+HbBase+Treat+Resection,
      Epo[train.set,],
      ntree=1, seed=1)
```

Fit a forest of 100 trees on the training data:

```
forest.train <-
    rfsrc(Y~age+sex+HbBase+Treat+Resection,
      Epo[train.set,],
      ntree=100, seed=1)
```

Prediction error and predictive accuracy

Predict from the tree model on the validation set:

```
tree1.val <- predict(tree1.train,
                newdata=Epo[val.set,],
                type="response")$predicted
```

Predict from the forest model on the validation set:

```
forest.val <- predict(forest.train,
                newdata=Epo[val.set,],
                type="response")$predicted
```

Prediction error and predictive accuracy

We define the loss function:

```
loss.fun <- function(Y, Phat) mean((Y-Phat)^2)
```

Now we can compare performance:

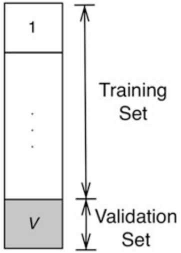
```
print(rbind(
    "1 tree " = loss.fun(Epo[val.set, ]$Y, tree1.val),
    "forest " = loss.fun(Epo[val.set, ]$Y, forest.val))
    )
```

```
             [,1]
1 tree  0.1443149
forest  0.0726101
```

Which one seems to perform best?

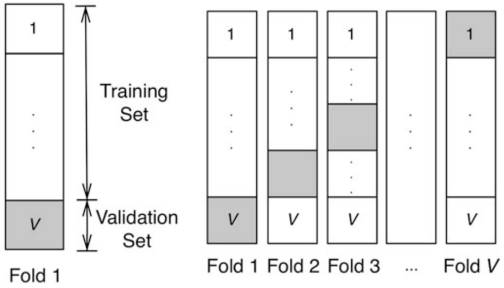
Prediction error and predictive accuracy

In practice, the splitting of data is not done once



Prediction error and predictive accuracy

In practice, the splitting of data is not done once

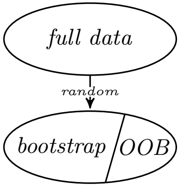


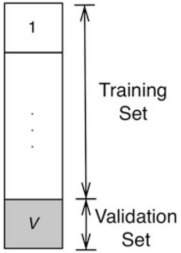
... but several times. This is called V-fold cross-validation

Prediction error and predictive accuracy

The same technique is used inside the forest!

- The oob prediction for patient i only uses the trees built on boostrap samples where patient i was left oob





- The oob prediction error is estimated by:

$$\widehat{\text{error}}_{\text{oob}} = \frac{1}{n}\sum_{i=1}^{n}\mathscr{L}(Y_i, \hat{P}_i^{\text{oob}})$$

Prediction error and predictive accuracy

```
rfsrc(Y~age+sex+HbBase+Treat+Resection,
      Epo, ntree=100, seed=1)
```

```
                         Sample size: 149
                     Number of trees: 100
           Forest terminal node size: 5
       Average no. of terminal nodes: 12.85
No. of variables tried at each split: 2
              Total no. of variables: 5
       Resampling used to grow trees: swor
    Resample size used to grow trees: 94
                            Analysis: RF-R
                              Family: regr
                      Splitting rule: mse *random*
       Number of random split points: 10
                % variance explained: 58.37
                          Error rate: 0.1
```

Prediction error and predictive accuracy

Consider the oob predicted errors for a number of different forests:

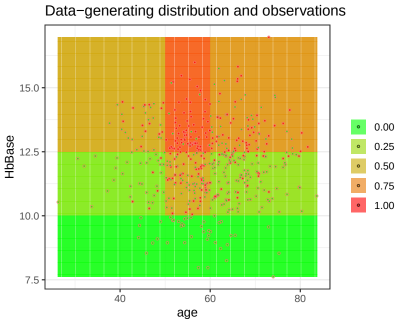
```
                          [,1]
1 tree performance   0.2455492
forest (5 trees)     0.1528393
forest (10 trees)    0.1254213
forest (50 trees)    0.1075489
forest (100 trees)   0.1034025
forest (150 trees)   0.1064458
forest (200 trees)   0.1060658
forest (500 trees)   0.1054694
forest (1000 trees)  0.1039286
```

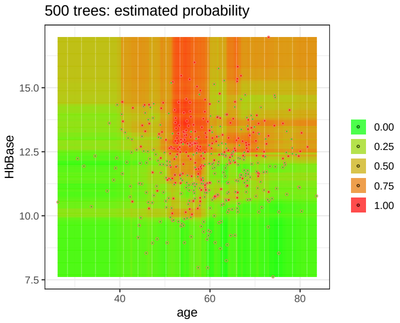
Picking the random forest model

Hyperparameter tuning

Picking the random forest model

The random forest algorithm automatically detects nonlinear effects, complex interactions, . . .



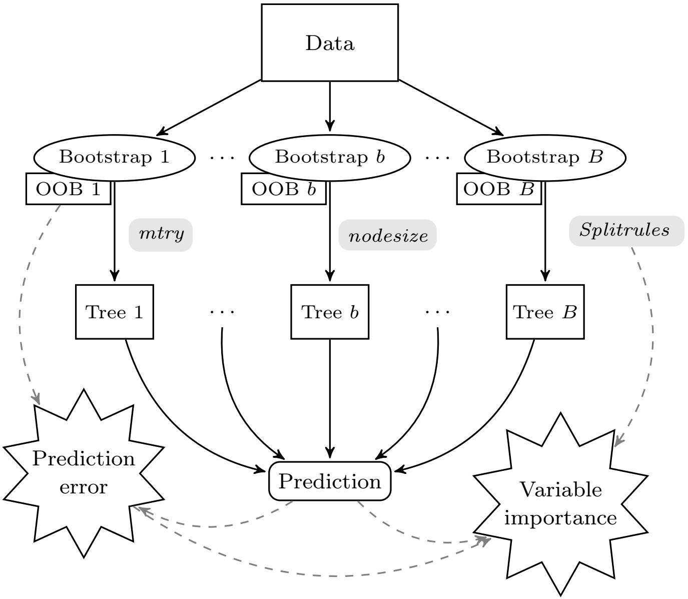


Picking the random forest model

But the algorithm involves some choices: hyperparameters!

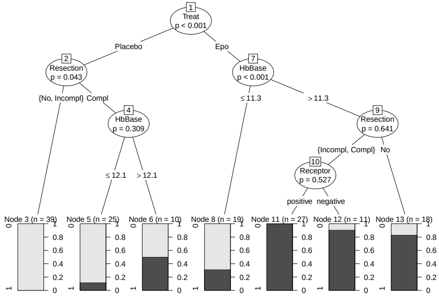
- These can be tuned and lead to different results
- These can be tuned to optimize predictive performance

There are a lot of different choices involved in the algorithm



Hyperparameters of the random forest

`ntree`	the number of trees
`mtry`	only `mtry` randomly selected predictor variables are used to find the best split ("split-variable randomization")
`nodesize`	is connected to the depth of each tree; it specifies the minimum number of observations that must be remain to perform a split



Applying a random forest

```
library("randomForestSRC")
```

```
rfsrc(formula,
      dataset,
      seed     = 5,
      ntree    = 1000, # how many trees
      mtry     = 3,    # number of randomly selected
                       # variables as candidates for
                       # splitting a node
      nodesize = 5)    # how many unique data points
                       # in each terminal node
```

Applying a random forest

We fit a random forest model on the Epo data with 1000 trees, mtry=3 and nodesize=3:

```
rf1 <- rfsrc(Y~age+sex+HbBase+Treat+Resection,
         Epo,
         seed = 5,
         nodesize = 3,
         mtry = 3,
         ntree = 1000)
```

Applying a random forest

Look at predictions (here first 5):

```
rf1$predicted.oob[1:5]
```

```
[1] 0.029569892 0.042219020 0.971616712 0.984539768 0.00493
```

The oob prediction for patient 25:

```
rf1$predicted.oob[25]
```

```
[1] 0.7713155
```

Compare to what was observed for this patient:

```
Epo[25, "Y"]
```

```
[1] 1
```

Predictions on new data

```
newpatient
```

```
  age  sex HbBase Treat Resection Receptor
1  48 male   10.8   Epo        No negative
```

Predictions on new data

Make predictions for this patient:

```
rf.pred.new <- predict(rf1, newdata=newpatient)
```

```
rf.pred.new$predicted
```

```
[1] 0.5763333
```

Exercise: Get predictions for `newpatient`

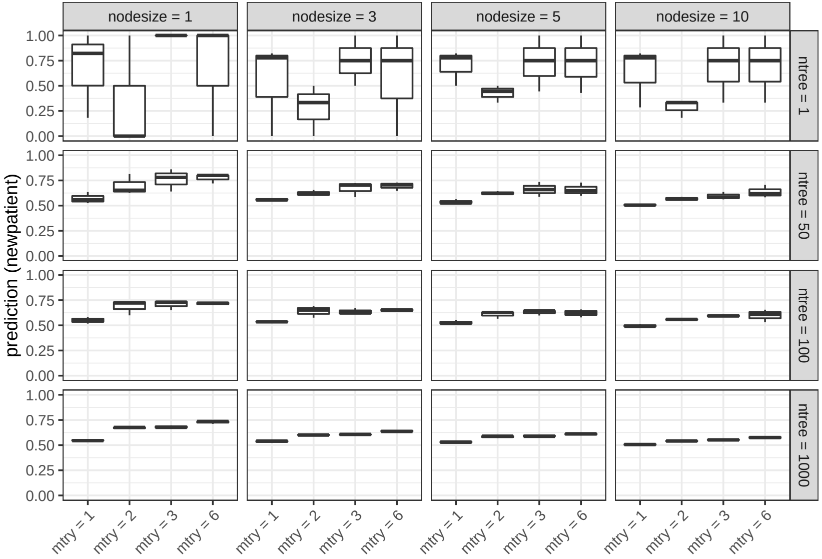
In this exercise, we will use the `rfsrc()` function from the `randomForestSRC` package to get forest prediction for `newpatient` for different values of hyperparameters

- The point is to see how sensitive the forest predictions are to the choice of hyperparameters

The exercise is described in **day3-practical.pdf**

- **Exercise 2**: Get predictions for `newpatient`

Exercise: Get predictions for `newpatient`



Tuning hyperparameters

Let's tune the random forest = pick hyperparameters that optimize predictive accuracy

Tuning hyperparameters

Let's tune the random forest = pick hyperparameters that optimize predictive accuracy

Look at the estimated error rate across the different choices of hyperparameters in the exercise:

```
                          [,1]
forest (ntree=50)    0.1229789
forest (ntree=100)   0.1225639
forest (ntree=1000)  0.1182131
forest (nodesize=3)  0.1097429
forest (nodesize=5)  0.1055143
forest (mtry=1)      0.1247416
forest (mtry=6)      0.1218394
```

Which one is the best one?

Tuning hyperparameters

Tuning a model is **tedious work** . . . there are lot of possible combinations of the parameters

Tuning hyperparameters

Propose (relevant) combinations of values for `mtry`, `nodesize` and `ntree`:

```
hyper.grid <- expand.grid(
  mtry = floor((ncol(Epo)-1)/c(4,3,2,1)),
  nodesize = c(1,3,5,10),
  ntree=c(5, 50, 100, 1000),
  oob.error = NA
)
```

Tuning hyperparameters

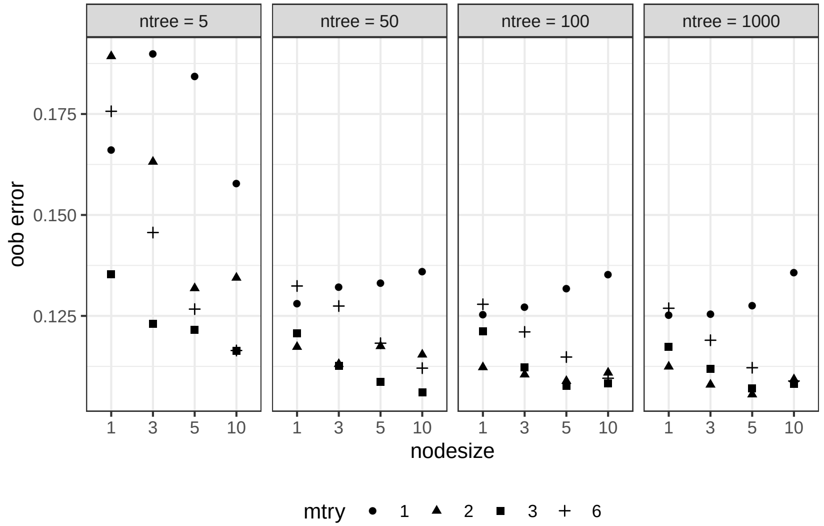
```
   mtry nodesize ntree
49    1        1  1000
50    2        1  1000
51    3        1  1000
52    6        1  1000
53    1        3  1000
54    2        3  1000
55    3        3  1000
56    6        3  1000
57    1        5  1000
58    2        5  1000
59    3        5  1000
60    6        5  1000
61    1       10  1000
62    2       10  1000
63    3       10  1000
64    6       10  1000
```

Tuning hyperparameters

Compute the oob error for all combinations:

```
for (j in 1:nrow(hyper.grid)) {
  tmp.forest <-
    rfsrc(Y~age+sex+HbBase+Treat+Resection,
      Epo,
      mtry=hyper.grid[j, "mtry"],
      nodesize=hyper.grid[j, "nodesize"],
      ntree=hyper.grid[j, "ntree"], seed=1)
  hyper.grid[j, "oob.error"] <-
    loss.fun(Epo$Y, tmp.forest$predicted.oob)
}
```

Tuning hyperparameters



Tuning hyperparameters

Which combination gave the lowest estimated error rate?

```
hyper.grid[which.min(hyper.grid$oob.error), ]
```

```
   mtry nodesize ntree oob.error
58    2        5  1000 0.1055467
```

Tuning hyperparameters

Let's fit the corresponding, now tuned, forest:

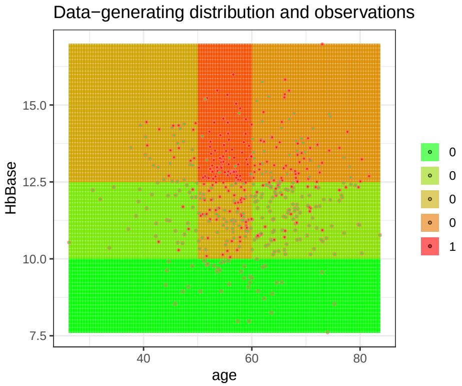
```
j <- which.min(hyper.grid$oob.error)
```

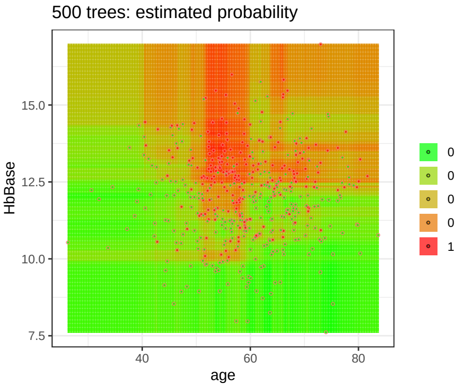
```
j
```

```
[1] 58
```

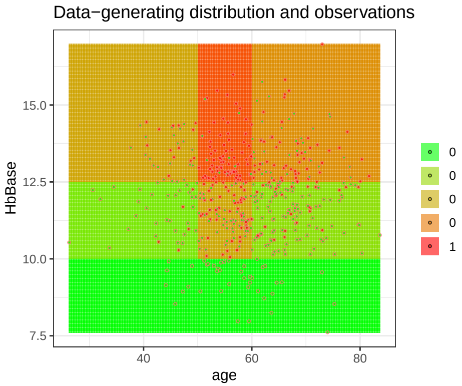
```
tuned.rf <-
    rfsrc(Y~age+sex+HbBase+Treat+Resection,
      Epo,
      mtry=hyper.grid[j, "mtry"],
      nodesize=hyper.grid[j, "nodesize"],
      ntree=hyper.grid[j, "ntree"], seed=1)
```

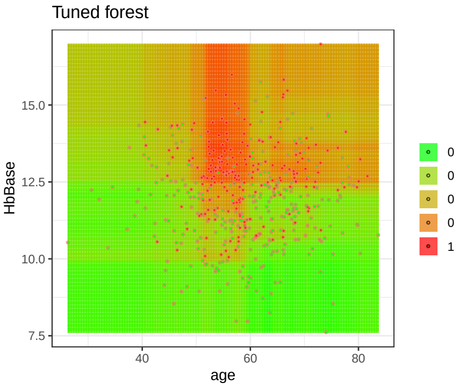
Tuning hyperparameters for the simulated data





Tuning hyperparameters for the simulated data





Interpretable machine learning

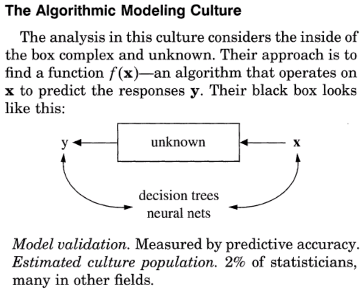
Variable importance

Logistic regression

Response: treatment successful yes/no

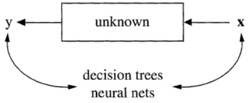
Factor	OddsRatio	StandardError	CI.95	pValue
(Intercept)	0.00	4.01	–	< 0.0001
Age	0.97	0.03	[0.91; 1.03]	0.2807
Sex:female	4.71	0.84	[0.91; 26.02]	0.0657
HbBase	3.25	0.27	[1.99; 5.91]	< 0.0001
Treatment:Epo	90.92	0.76	[23.9; 493.41]	< 0.0001
Resection:Incompl	1.75	0.81	[0.36; 9.03]	0.4924
Resection:Compl	4.14	0.69	[1.13; 17.36]	0.0395
Receptor:positive	5.81	0.66	[1.72; 23.39]	0.0076

Machine learning



The Algorithmic Modeling Culture

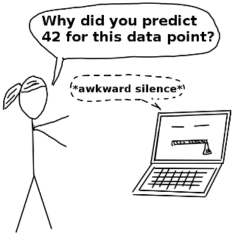
The analysis in this culture considers the inside of the box complex and unknown. Their approach is to find a function $f(\mathbf{x})$—an algorithm that operates on $\mathbf{x}$ to predict the responses $\mathbf{y}$. Their black box looks like this:



Model validation. Measured by predictive accuracy.
Estimated culture population. 2% of statisticians, many in other fields.

Interpretable machine learning

- Decision trees produce results that are easy to interpret
- Random forest results, on the other hand, are not per se so easy to interpret



- What predictor variables were important for the prediction?
- What effect did the predictor variables have on the prediction?

Variable importance

How important was a given variable for building the forest model?

We consider two different approaches

1. "VIMP"
2. Minimal depth

Variable importance - VIMP

VIMP (Variable IMPortance) is measured by the difference prediction error between:

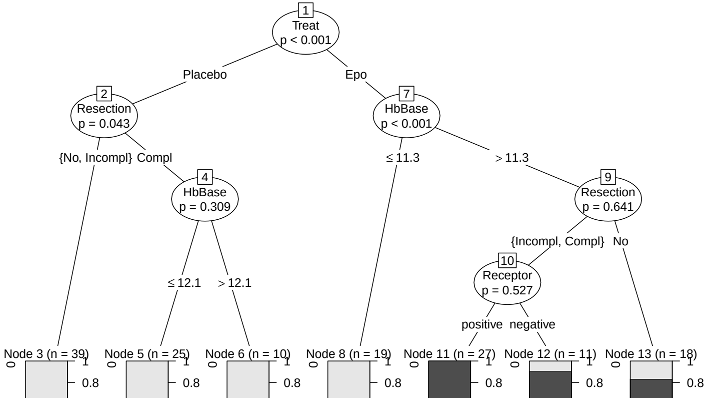
- running the forest with a "noised-up" version of X
- running the forest with X as was observed

If prediction performance decreases more for variable X_1 than for variable X_2, then importance(X_1) $>$ importance(X_2)

Variable importance - Minimal depth

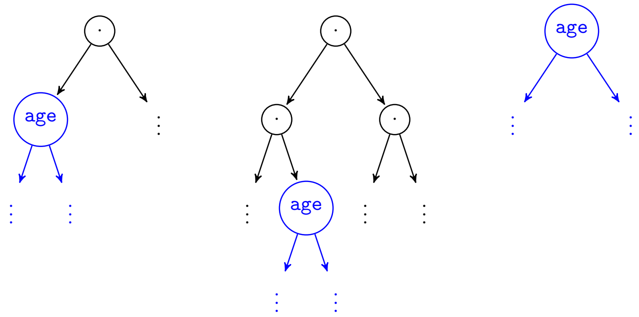
Recall:

- Trees are built by recursive partitioning
- They let the data decide which variables are important for splitting node



Variable importance - Minimal depth

The minimal depth is the average distance from the root node to the first split on a specific variable



The smaller the minimal depth, the more important is the variable

Variable importance - VIMP in R

VIMP for the Epo data:

```
tuned.rf <- rfsrc(Y~age+sex+HbBase+Treat+
                    Resection,
                  Epo,
                  mtry=hyper.grid[j, "mtry"],
                  nodesize=hyper.grid[j, "nodesize"],
                  ntree=hyper.grid[j, "ntree"],
                  seed=1,
                  importance=TRUE) # compute vimp
```

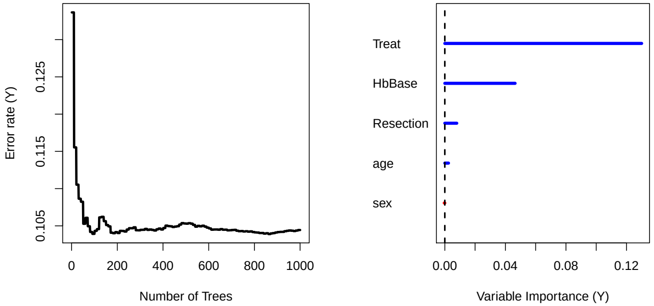
```
tuned.rf$importance
```

```
       age        sex     HbBase      Treat  Resection
0.07420322 0.00589100 0.23239730 0.36622336 0.02681411
```

What variables are most important?

Variable importance - VIMP in R

```
plot(tuned.rf)
```



Variable importance - VIMP in R

Minimal depth for the Epo data:

```
  age        sex     HbBase      Treat Resection
1.550      3.688      1.300      0.992     2.072
```

The forest provides a threshold (cut-off) value:

```
[1] 2.343143
```

What variables are important?

Variable importance

What you should not do...

```
rfsrc(Y~age+sex+HbBase+Treat+Resection+Receptor,
      ... ,
      mtry = 1, # <-- nooo
      ... )
```

Variable importance

What you should not do...

```
rfsrc(Y~age+sex+HbBase+Treat+Resection+Receptor,
      ... ,
      mtry = 1, # <-- nooo
      ... )
```

mtry=1:

```
          mtry1
age       1.688
sex       2.017
HbBase    1.697
Treat     1.867
Resection 1.903
```

Threshold=

```
[1] 2.051348
```

Variable importance

What you should not do...

```
rfsrc(Y~age+sex+HbBase+Treat+Resection+Receptor,
     ... ,
     mtry = 1, # <-- nooo
     ... )
```

mtry=1:

```
          mtry1
age       1.688
sex       2.017
HbBase    1.697
Treat     1.867
Resection 1.903
```

Threshold=

```
[1] 2.051348
```

mtry=2:

```
          mtry2
age       1.550
sex       3.688
HbBase    1.300
Treat     0.992
Resection 2.072
```

Threshold=

```
[1] 2.343143
```

Variable importance measures from random forests

Why is this?

`mtry` controls split-variable randomization:

- for each node only a small number of randomly selected predictors are used to find the best split of that node (= `mtry`)
- this is done as part of the randomization of trees
- (it ensures some of the theoretical properties of the forests)

In fact, if we are interested in variable importance (rather than predictive accuracy) we should choose a high value for this.

Variable importance on simulated data

- Two uncorrelated variables `x1` and `x2` with the same effect
- One variable `c1` correlated with `x1` but with no effect
- Two correlated variables `z1` and `z2` with the same effect
- Ten noise variables `w1`,. . . , `w10`

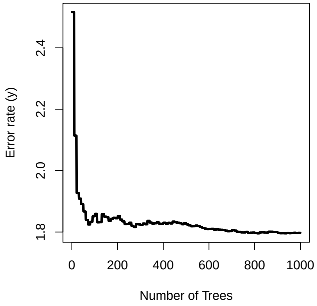
```r
x1 <- runif(n)
x2 <- runif(n)
z1 <- rnorm(n, mean=0, sd=0.3)
z2 <- rnorm(n, mean=z1+0.1, sd=0.3)
c1 <- rnorm(n, mean=x1+0.1, sd=0.3)
w <- matrix(runif(n*10), ncol=10)
y <- rnorm(n, mean=0.1+2.5*x1+2.5*x2+2.5*z1+2.5*z2)
```

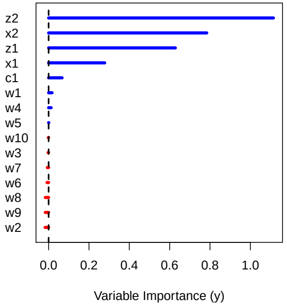
Variable importance on simulated data

```
rf.sim <- rfsrc(y~x1+x2+c1+z1+z2+
                  w1+w2+w3+w4+w5+w6+w7+w8+w9+w10,
                sim.data,
                seed=3, ntree=1000,
                importance=TRUE)
```

Variable importance on simulated data

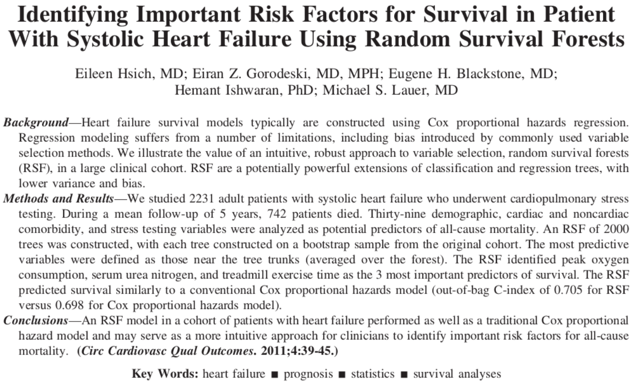
- Two uncorrelated variables `x1` and `x2` with the same effect
- One variable `c1` correlated with `x1` but with no effect
- Two correlated variables `z1` and `z2` with the same effect
- Ten noise variables `w1`, . . . , `w10`





Exercise: Identifying risk factors with variable importance

In this exercise we will look at the analysis of Hsich et al. (2011):



Identifying Important Risk Factors for Survival in Patient With Systolic Heart Failure Using Random Survival Forests

Eileen Hsich, MD; Eiran Z. Gorodeski, MD, MPH; Eugene H. Blackstone, MD; Hemant Ishwaran, PhD; Michael S. Lauer, MD

Background—Heart failure survival models typically are constructed using Cox proportional hazards regression. Regression modeling suffers from a number of limitations, including bias introduced by commonly used variable selection methods. We illustrate the value of an intuitive, robust approach to variable selection, random survival forests (RSF), in a large clinical cohort. RSF are a potentially powerful extensions of classification and regression trees, with lower variance and bias.

Methods and Results—We studied 2231 adult patients with systolic heart failure who underwent cardiopulmonary stress testing. During a mean follow-up of 5 years, 742 patients died. Thirty-nine demographic, cardiac and noncardiac comorbidity, and stress testing variables were analyzed as potential predictors of all-cause mortality. An RSF of 2000 trees was constructed, with each tree constructed on a bootstrap sample from the original cohort. The most predictive variables were defined as those near the tree trunks (averaged over the forest). The RSF identified peak oxygen consumption, serum urea nitrogen, and treadmill exercise time as the 3 most important predictors of survival. The RSF predicted survival similarly to a conventional Cox proportional hazards model (out-of-bag C-index of 0.705 for RSF versus 0.698 for Cox proportional hazards model).

Conclusions—An RSF model in a cohort of patients with heart failure performed as well as a traditional Cox proportional hazard model and may serve as a more intuitive approach for clinicians to identify important risk factors for all-cause mortality. **(*Circ Cardiovasc Qual Outcomes*. 2011;4:39-45.)**

Key Words: heart failure ■ prognosis ■ statistics ■ survival analyses

The exercise is described in **day3-practical.pdf**

- **Exercise 3**: Identifying risk factors

Effects of predictor variables on the final prediction

Plots can be useful to assess the effect of predictor variables on the final prediction

Effects of predictor variables on the final prediction

Plots can be useful to assess the effect of predictor variables on the final prediction. There are different ways to do so:

Partial Dependence Plots (PDPs)

- Average forest predictions as a function of predictor variables
- Obtained by marginalizing the forest prediction over the other features/covariates
- Can show if the relationship is linear, monotonic or more complex

Individual Conditional Expectation (ICE) plots

- Looking at the individual predictions as a function of predictor variables

Partial Dependence Plots (PDPs)

Say, we want to know how

$$\hat{P}(Y = 1 \mid \text{age}, \text{Gender}, \text{HbBase}, \text{Treatment}, \text{Resection})$$

varies when HbBase varies

We can estimate this by:

$$\hat{P}^{\text{HbBase}}(b) = \frac{1}{n}\sum_{i=1}^{n} \hat{P}(Y_i = 1 \mid \text{age}_i, \text{Gender}_i, \text{HbBase} = b, \text{Treatment}_i, \text{Resection}_i)$$

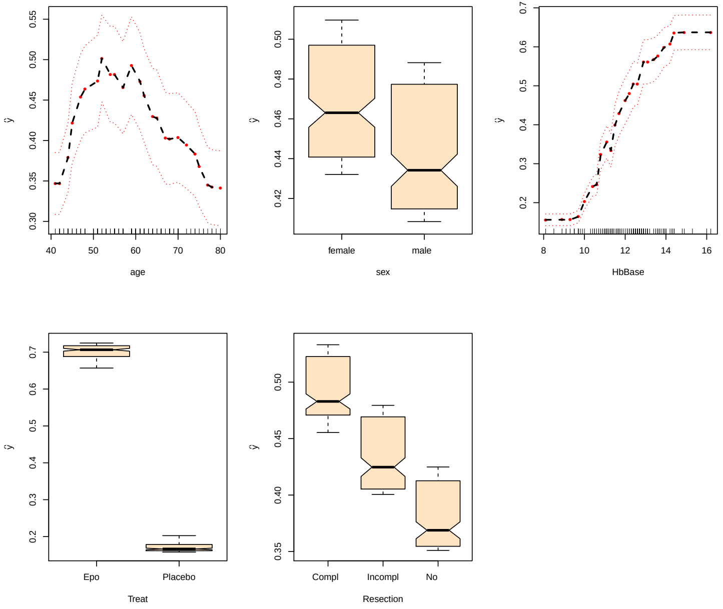
- We marginalize the forest prediction over the other features/covariates

Partial Dependence Plots (PDPs)

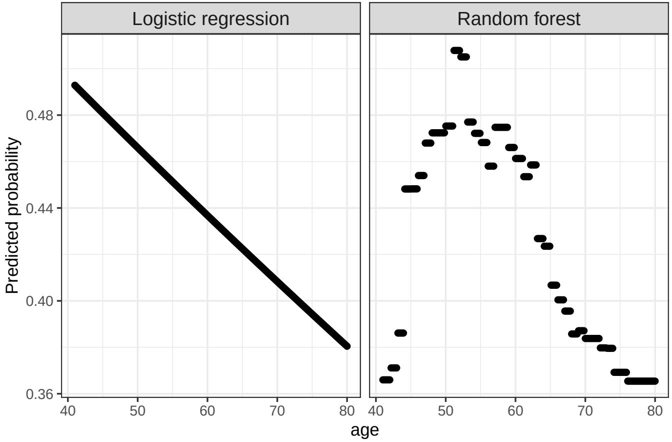
In R, we can plot these estimates for all variables by simply writing:

```
plot.variable(tuned.rf, partial=TRUE, plots.per.page=3)
```

Partial Dependence Plots (PDPs)



Partial Dependence Plots (PDPs)



It is clear that the random forest captures a highly nonlinear effect of age on the predicted probability!

Individual Conditional Expectation (ICE) plots

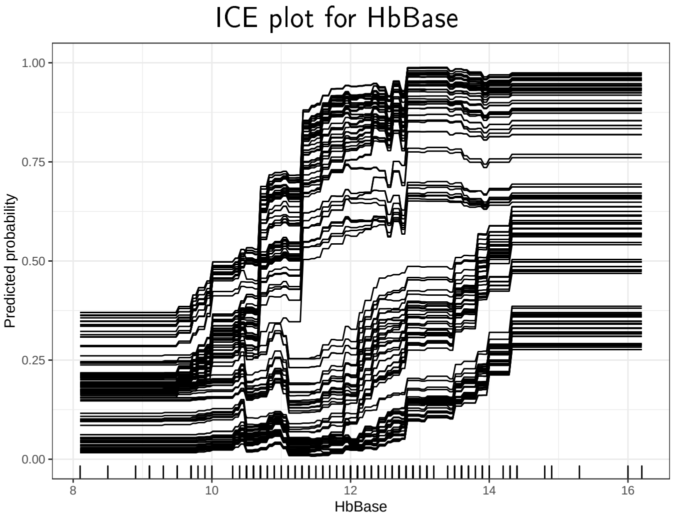
The ICE plot shows the variation of

$$\hat{P}_i^{\text{HbBase}}(b) = \hat{P}(Y = 1 \mid \text{age}_i, \text{Gender}_i, \text{HbBase} = b, \text{Treatment}_i, \text{Resection}_i)$$

for each individual i one by one

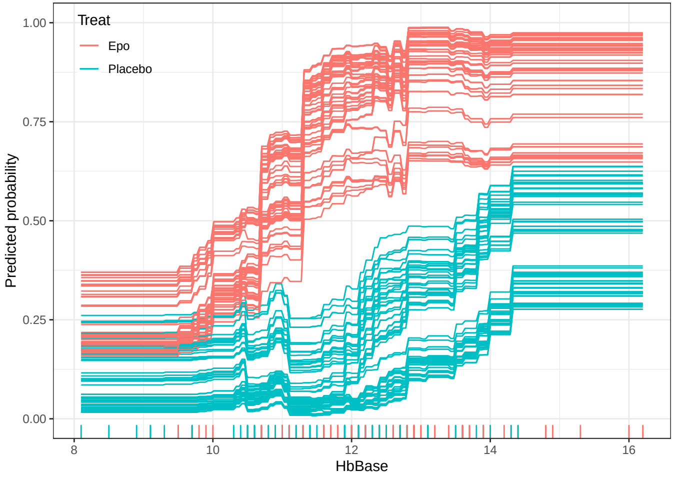
- This can very useful if there are interactions
- Do the curves follow the same course (e.g., changepoints, linearity, etc) for all individuals?

Individual Conditional Expectation (ICE) plots



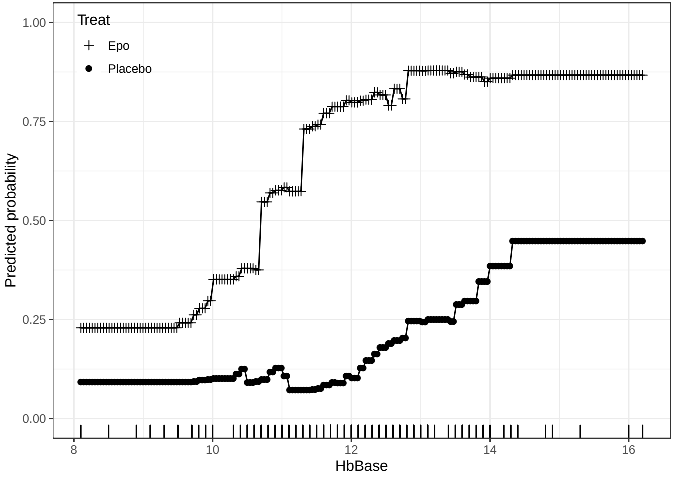
Individual Conditional Expectation (ICE) plots

ICE plot for HbBase, colored by Treat

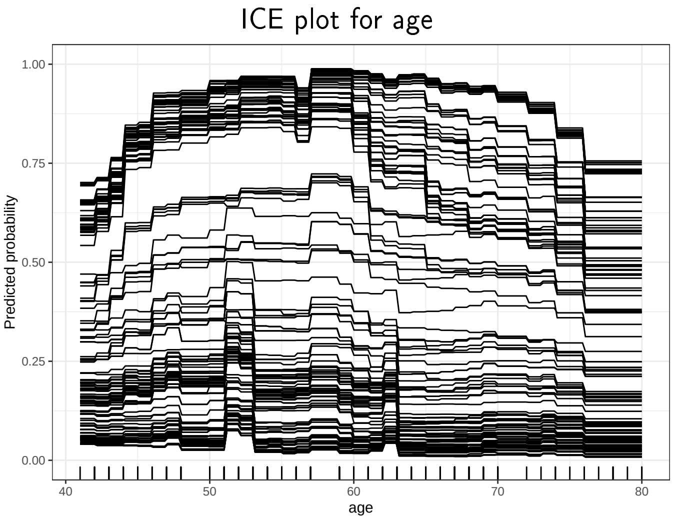


Individual Conditional Expectation (ICE) plots

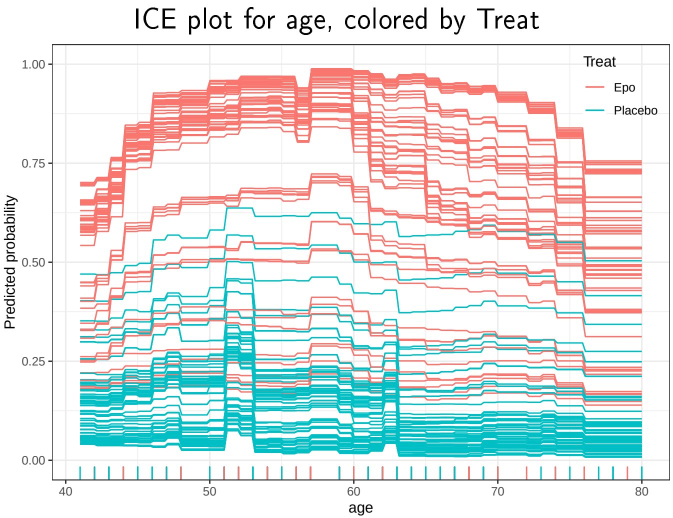
PDP plot for HbBase, computed in groups defined by Treat



Individual Conditional Expectation (ICE) plots



Individual Conditional Expectation (ICE) plots



Interpretable machine learning

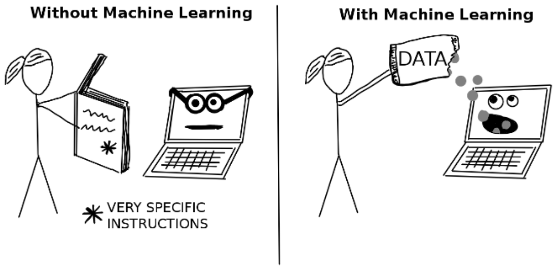
Variable importance measures can tell us what variables seem important for prediction

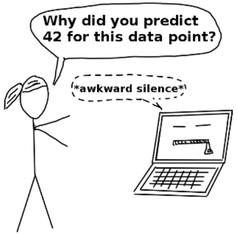
- Beware of correlated predictors

PDPs and ICE plots can show us how predicted probabilities vary as a function of predictor values

- PDPs show the average variation
  - Beware of hidden interactions
- ICE show the individual variations

Logistic regression versus random forests





Logistic regression versus random forests

When utilizing a logistic regression approach:

- We must specify the model[12]

$$\hat{P}(Y = 1 \mid \text{age}, \text{Gender}, \text{HbBase}, \text{Treatment}, \text{Resection}, \text{Epo})$$
$$= \text{expit}(\beta_0 + \beta_1 \text{age} + \beta_2 \text{age} : \text{female} + \cdots) \qquad (*)$$

Based on the model we may:

- Predict $\hat{P}(Y = 1)$ for a new patient
- Interpret odds ratios, conditional on holding the other features fixed (p-values, confidence intervals, etc)

But all inference relies on $(*)$ being correct and prespecified.

[12] Interactions, quadratic terms (e.g., age^2), . . .

Logistic regression versus random forests

When utilizing a random forest approach:

- The forest automatically detects nonlinear effects and complex interactions
- "Model selection" is comprised by hyperparameter tuning

Based on the model we may:

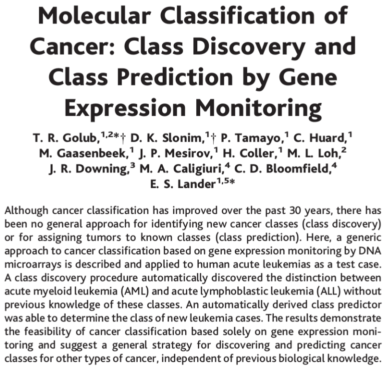
- Predict $\hat{P}(Y = 1)$ for a new patient *often with high accuracy*
- Obtaining interpretable measures from the random forest[13] is applied after model training, e.g.:
    - Variable importance, PDPs, ICEs, ...

But, inference (confidence intervals, p-values) is not so obvious.

And, beware that everything depends on the random seed.

[13] And other machine learning methods

Exercise: Predicting tumor class (Golub et al., 1999)



Molecular Classification of Cancer: Class Discovery and Class Prediction by Gene Expression Monitoring

T. R. Golub,[1,2*†] D. K. Slonim,[1†] P. Tamayo,[1] C. Huard,[1] M. Gaasenbeek,[1] J. P. Mesirov,[1] H. Coller,[1] M. L. Loh,[2] J. R. Downing,[3] M. A. Caligiuri,[4] C. D. Bloomfield,[4] E. S. Lander[1,5*]

Although cancer classification has improved over the past 30 years, there has been no general approach for identifying new cancer classes (class discovery) or for assigning tumors to known classes (class prediction). Here, a generic approach to cancer classification based on gene expression monitoring by DNA microarrays is described and applied to human acute leukemias as a test case. A class discovery procedure automatically discovered the distinction between acute myeloid leukemia (AML) and acute lymphoblastic leukemia (ALL) without previous knowledge of these classes. An automatically derived class predictor was able to determine the class of new leukemia cases. The results demonstrate the feasibility of cancer classification based solely on gene expression monitoring and suggest a general strategy for discovering and predicting cancer classes for other types of cancer, independent of previous biological knowledge.

- Accurate cancer classification can be used to target specific therapies to distinct tumor types
- We could use a random forest model to provide a data-based classification algorithm based on gene expression monitoring

Exercise: Predicting tumor class (Golub et al., 1999)

In this practical we will work with a dataset containing information on 38 tumor mRNA samples from 38 individuals and the gene expression values from 3051 genes

We will go through the steps on the lectures slides to explore these data

- The goal of the analysis is to predict the tumor class

The exercise is described in **day3-practical.pdf**

- **Exercise 4**: Predicting tumor class

That was it . . .

Comments and suggestions for this material are very much welcome at mark.bech.knudsen@sund.ku.dk ☺