Why and how to use random forest variable importance measures (and how you shouldn't)

Carolin Strobl (LMU München) and Achim Zeileis (WU Wien)

carolin.strobl@stat.uni-muenchen.de

useR! 2008, Dortmund

Introduction

Random forests

Introduction

Random forests

- have become increasingly popular in, e.g., genetics and the neurosciences

Introduction

Random forests

- have become increasingly popular in, e.g., genetics and the neurosciences [imagine a long list of references here]

Introduction

Random forests

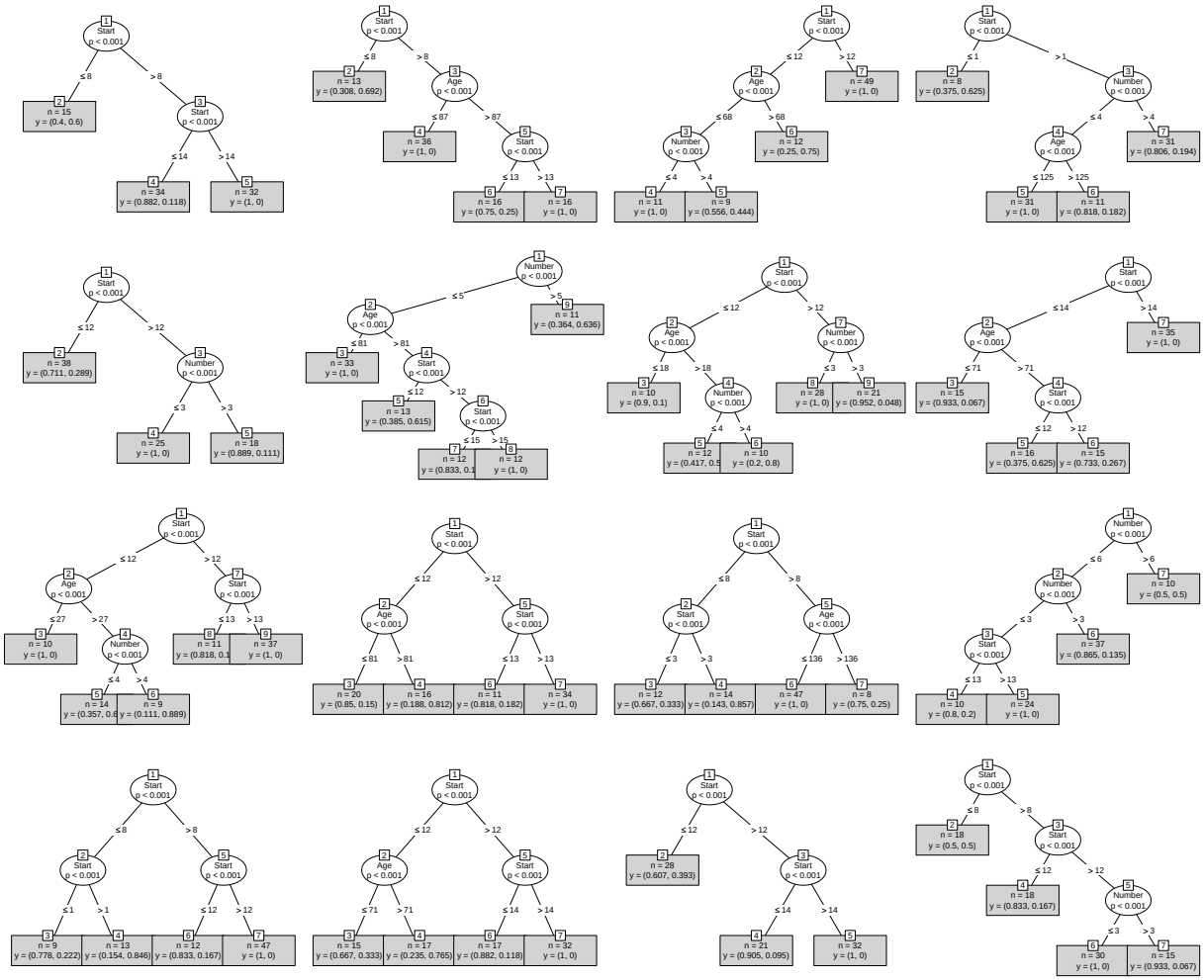
- have become increasingly popular in, e.g., genetics and the neurosciences [imagine a long list of references here]
- can deal with "small n large p"-problems, high-order interactions, correlated predictor variables

Introduction

Random forests

- have become increasingly popular in, e.g., genetics and the neurosciences [imagine a long list of references here]
- can deal with "small n large p"-problems, high-order interactions, correlated predictor variables
- are used not only for prediction, but also to assess variable importance

(Small) random forest



Construction of a random forest

Construction of a random forest

- draw `ntree` bootstrap samples from original sample

Construction of a random forest

- draw `ntree` bootstrap samples from original sample
- fit a classification tree to each bootstrap sample

  $\Rightarrow$ `ntree` trees

Construction of a random forest

- draw `ntree` bootstrap samples from original sample
- fit a classification tree to each bootstrap sample

  $\Rightarrow$ `ntree` trees
- creates diverse set of trees because
  - trees are instable w.r.t. changes in learning data

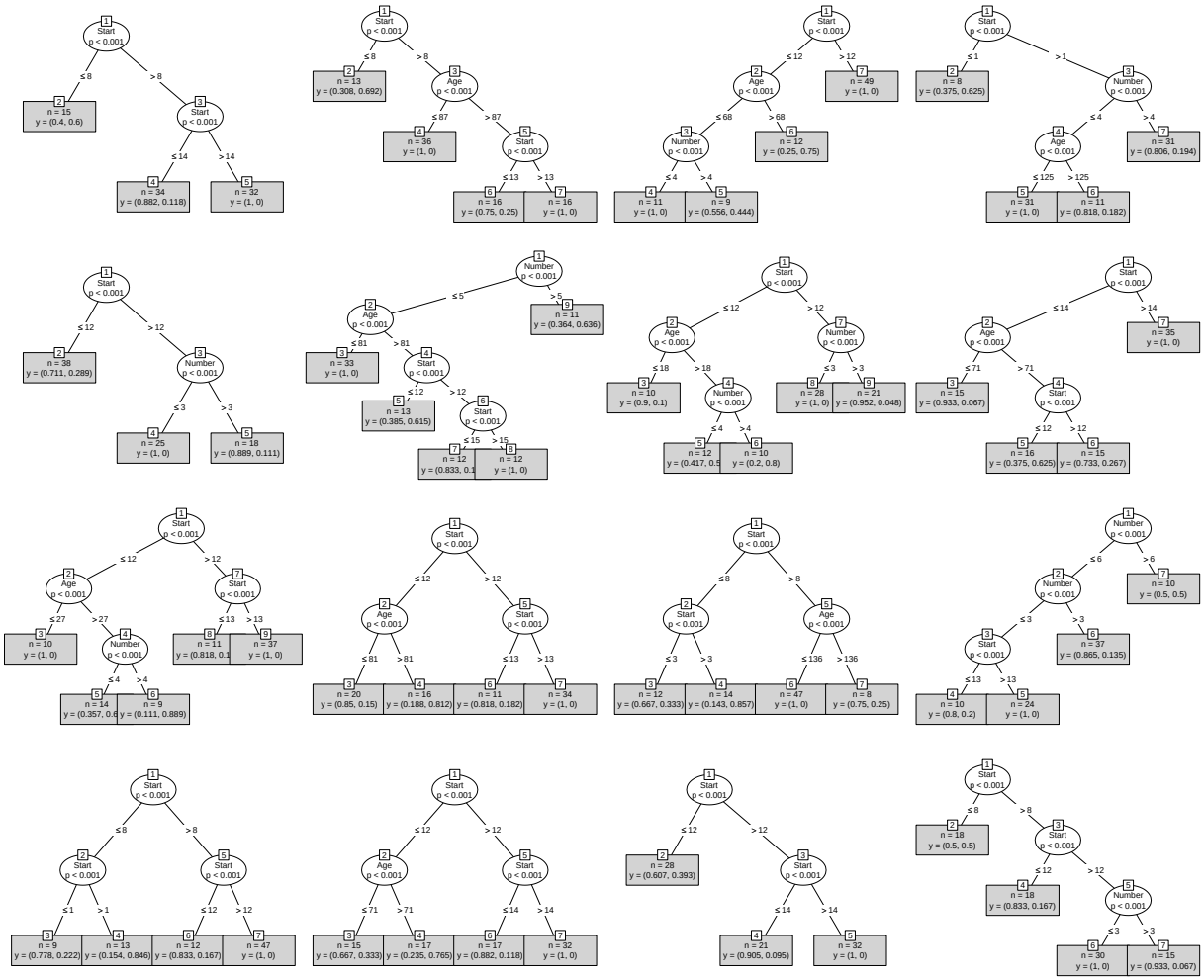
    $\Rightarrow$ `ntree` different looking trees (bagging)
  - randomly preselect `mtry` splitting variables in each split

    $\Rightarrow$ `ntree` more different looking trees (random forest)

Random forests in R

- `randomForest` (pkg: `randomForest`)
  - reference implementation based on CART trees (Breiman, 2001; Liaw and Wiener, 2008)
  - – for variables of different types: biased in favor of continuous variables and variables with many categories (Strobl, Boulesteix, Zeileis, and Hothorn, 2007)
- `cforest` (pkg: `party`)
  - based on unbiased conditional inference trees (Hothorn, Hornik, and Zeileis, 2006)
  - \+ for variables of different types: unbiased when subsampling, instead of bootstrap sampling, is used (Strobl, Boulesteix, Zeileis, and Hothorn, 2007)

(Small) random forest



Measuring variable importance

- Gini importance

  mean Gini gain produced by X_j over all trees

  - `obj <- randomForest(..., importance=TRUE)`
    `obj$importance` column: `MeanDecreaseGini`
    `importance(obj, type=2)`

for variables of different types: biased in favor of continuous variables and variables with many categories

Measuring variable importance

- permutation importance

  mean decrease in classification accuracy after permuting X_j over all trees

  - `obj <- randomForest(..., importance=TRUE)`
    `obj$importance` column: `MeanDecreaseAccuracy`
    `importance(obj, type=1)`
  - `obj <- cforest(...)`
    `varimp(obj)`

for variables of different types: unbiased only when subsampling is used as in `cforest(..., controls = cforest_unbiased())`

The permutation importance

within each tree t

$$VI^{(t)}(\mathbf{x}_j) = \frac{\sum_{i \in \overline{\mathfrak{B}}^{(t)}} I\left(y_i = \hat{y}_i^{(t)}\right)}{\left|\overline{\mathfrak{B}}^{(t)}\right|} - \frac{\sum_{i \in \overline{\mathfrak{B}}^{(t)}} I\left(y_i = \hat{y}_{i,\pi_j}^{(t)}\right)}{\left|\overline{\mathfrak{B}}^{(t)}\right|}$$

$\hat{y}_i^{(t)} = f^{(t)}(\mathbf{x}_i)$ = predicted class before permuting

$\hat{y}_{i,\pi_j}^{(t)} = f^{(t)}(\mathbf{x}_{i,\pi_j})$ = predicted class after permuting X_j

$\mathbf{x}_{i,\pi_j} = \left(x_{i,1}, \ldots, x_{i,j-1}, \; x_{\pi_j(i),j}, x_{i,j+1}, \ldots, x_{i,p}\right)$

Note: $VI^{(t)}(\mathbf{x}_j) = 0$ by definition, if X_j is not in tree t

The permutation importance

over all trees:

1. raw importance

$$VI(\mathbf{x}_j) = \frac{\sum_{t=1}^{ntree} VI^{(t)}(\mathbf{x}_j)}{ntree}$$

- `obj <- randomForest(..., importance=TRUE)`
  `importance(obj, type=1, scale=FALSE)`

The permutation importance

over all trees:

2. scaled importance (z-score)

$$\frac{VI(\mathbf{x}_j)}{\frac{\hat{\sigma}}{\sqrt{ntree}}} = z_j$$

- `obj <- randomForest(..., importance=TRUE)`
  `importance(obj, type=1, scale=TRUE)` (default)

Tests for variable importance

for variable selection purposes

Tests for variable importance

for variable selection purposes

- Breiman and Cutler (2008): simple significance test based on normality of z-score

  `randomForest`, `scale=TRUE` $+$ α-quantile of N(0,1)

Tests for variable importance

for variable selection purposes

- Breiman and Cutler (2008): simple significance test based on normality of z-score

  `randomForest`, `scale=TRUE` + α-quantile of N(0,1)
- Diaz-Uriarte and Alvarez de Andrés (2006): backward elimination (throw out least important variables until out-of-bag prediction accuracy drops)

  `varSelRF` (pkg: `varSelRF`), dep. on `randomForest`

Tests for variable importance

for variable selection purposes

- Breiman and Cutler (2008): simple significance test based on normality of z-score
  `randomForest`, `scale=TRUE` + α-quantile of N(0,1)
- Diaz-Uriarte and Alvarez de Andrés (2006): backward elimination (throw out least important variables until out-of-bag prediction accuracy drops)
  `varSelRF` (pkg: `varSelRF`), dep. on `randomForest`
- Diaz-Uriarte (2007) and Rodenburg et al. (2008): plots and significance test (randomly permute response values to mimic the overall null hypothesis that none of the predictor variables is relevant = baseline)

Tests for variable importance

problems of these approaches:

Tests for variable importance

problems of these approaches:

- (at least) Breiman and Cutler (2008): strange statistical properties (Strobl and Zeileis, 2008)

Tests for variable importance

problems of these approaches:

- (at least) Breiman and Cutler (2008): strange statistical properties (Strobl and Zeileis, 2008)
- all: preference of correlated predictor variables (see also Nicodemus and Shugart, 2007; Archer and Kimes, 2008)

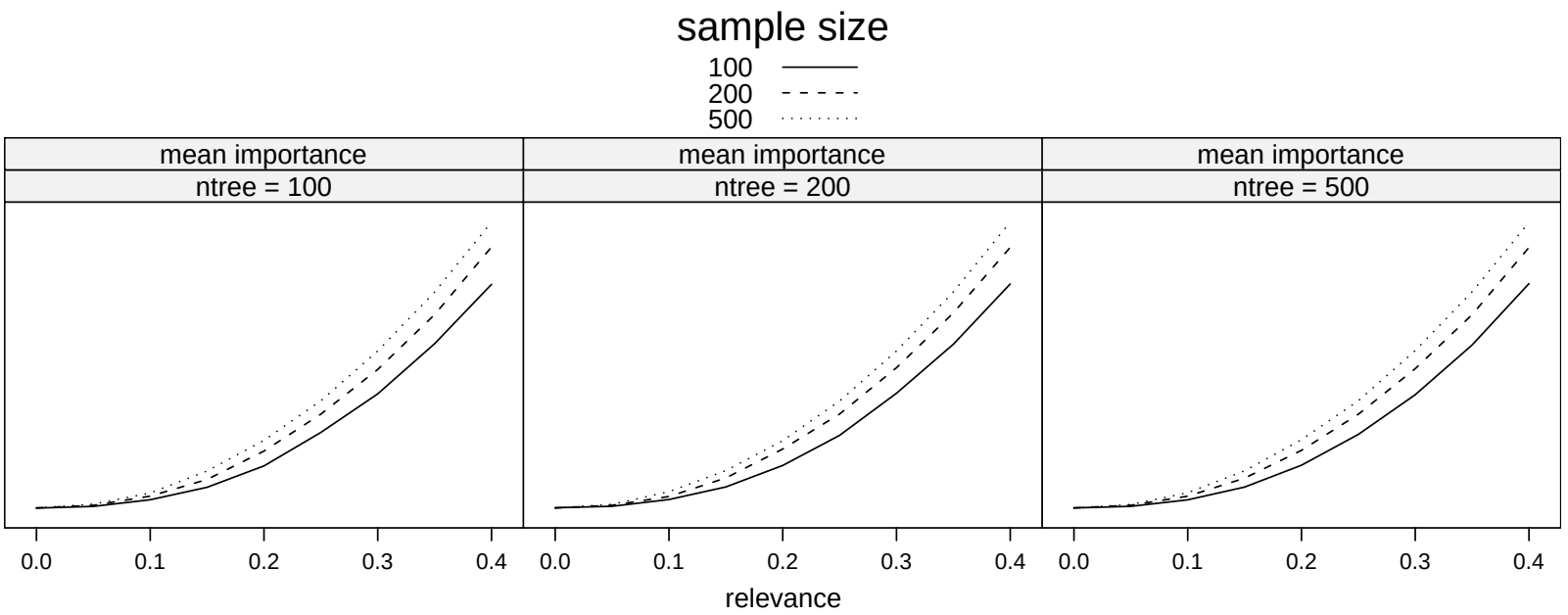
Breiman and Cutler's test

under the null hypothesis of zero importance:

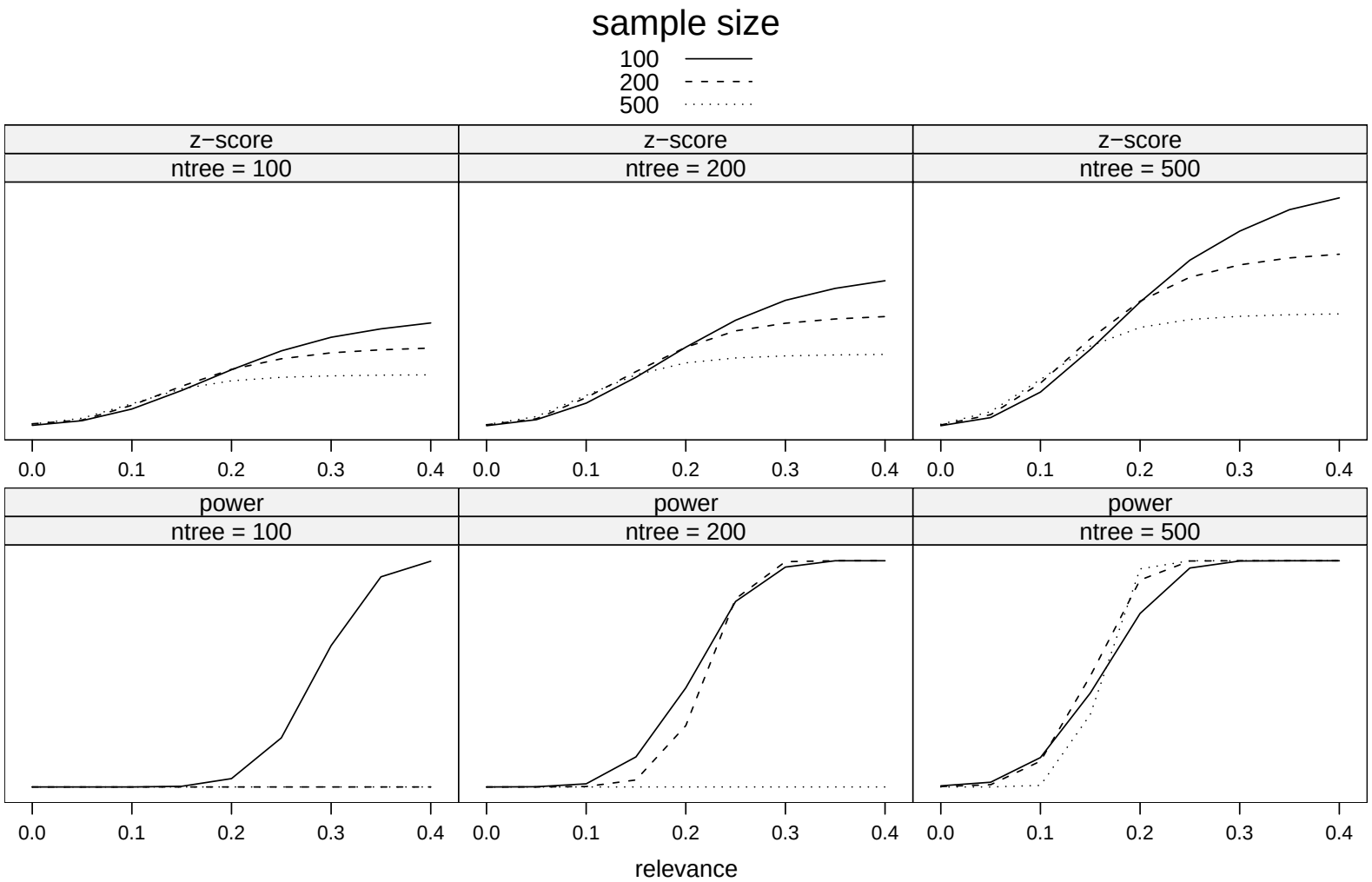
$$z_j \overset{as.}{\sim} N(0,1)$$

if z_j exceeds the α-quantile of N(0,1) $\Rightarrow$ reject the null hypothesis of zero importance for variable X_j

Raw importance



z-score and power



Findings

z-score and power

- increase in `ntree`
- decrease in sample size

$\Rightarrow$ rather use raw, unscaled permutation importance!

```
importance(obj, type=1, scale=FALSE)
```

```
varimp(obj)
```

What null hypothesis were we testing in the first place?

obs	Y	X_j	Z
1	y_1	$x_{\pi_j(1),j}$	z_1
$\vdots$	$\vdots$	$\vdots$	$\vdots$
i	y_i	$x_{\pi_j(i),j}$	z_i
$\vdots$	$\vdots$	$\vdots$	$\vdots$
n	y_n	$x_{\pi_j(n),j}$	z_n

$$H_0 : X_j \perp Y, Z \text{ or } X_j \perp Y \wedge X_j \perp Z$$

$$P(Y, X_j, Z) \overset{H_0}{=} P(Y, Z) \cdot P(X_j)$$

What null hypothesis were we testing in the first place?

the current null hypothesis reflects independence of X_j from both Y and the remaining predictor variables Z

What null hypothesis were we testing in the first place?

the current null hypothesis reflects independence of X_j from both Y and the remaining predictor variables Z

$\Rightarrow$ a high variable importance can result from violation of

either one!

Suggestion: Conditional permutation scheme

obs	Y	X_j	Z
1	y_1	$x_{\pi_{j\|Z=a}(1),j}$	$z_1 = a$
3	y_3	$x_{\pi_{j\|Z=a}(3),j}$	$z_3 = a$
27	y_{27}	$x_{\pi_{j\|Z=a}(27),j}$	$z_{27} = a$
6	y_6	$x_{\pi_{j\|Z=b}(6),j}$	$z_6 = b$
14	y_{14}	$x_{\pi_{j\|Z=b}(14),j}$	$z_{14} = b$
33	y_{33}	$x_{\pi_{j\|Z=b}(33),j}$	$z_{33} = b$
$\vdots$	$\vdots$	$\vdots$	$\vdots$

$$H_0 : X_j \perp Y|Z$$

$$P(Y, X_j|Z) \overset{H_0}{=} P(Y|Z) \cdot P(X_j|Z)$$

$$\text{or} \quad P(Y|X_j, Z) \overset{H_0}{=} P(Y|Z)$$

Technically

- use any partition of the feature space for conditioning

Technically

- use any partition of the feature space for conditioning
- here: use binary partition already learned by tree
  (use cutpoints as bisectors of feature space)

Technically

- use any partition of the feature space for conditioning
- here: use binary partition already learned by tree
  (use cutpoints as bisectors of feature space)
- condition on correlated variables or select some

Technically

- use any partition of the feature space for conditioning
- here: use binary partition already learned by tree (use cutpoints as bisectors of feature space)
- condition on correlated variables or select some

Strobl et al. (2008)

available in `cforest` from version `0.9-994`: `varimp(obj, conditional = TRUE)`

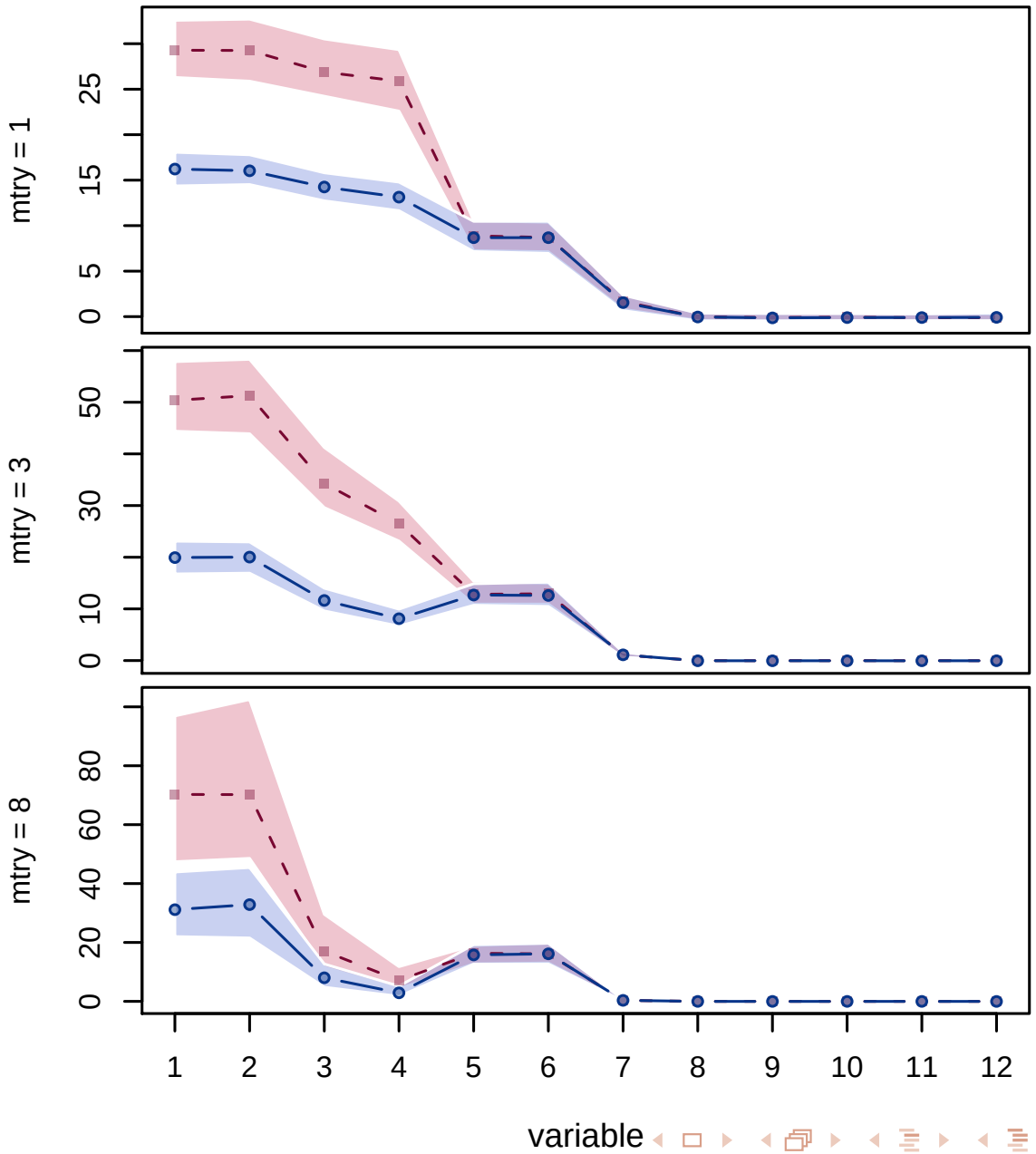
Simulation study

- dgp: $y_i = \beta_1 \cdot x_{i,1} + \cdots + \beta_{12} \cdot x_{i,12} + \varepsilon_i$, $\varepsilon_i \overset{i.i.d.}{\sim} N(0, 0.5)$
- $X_1, \ldots, X_{12} \sim N(0, \mathbf{\Sigma})$

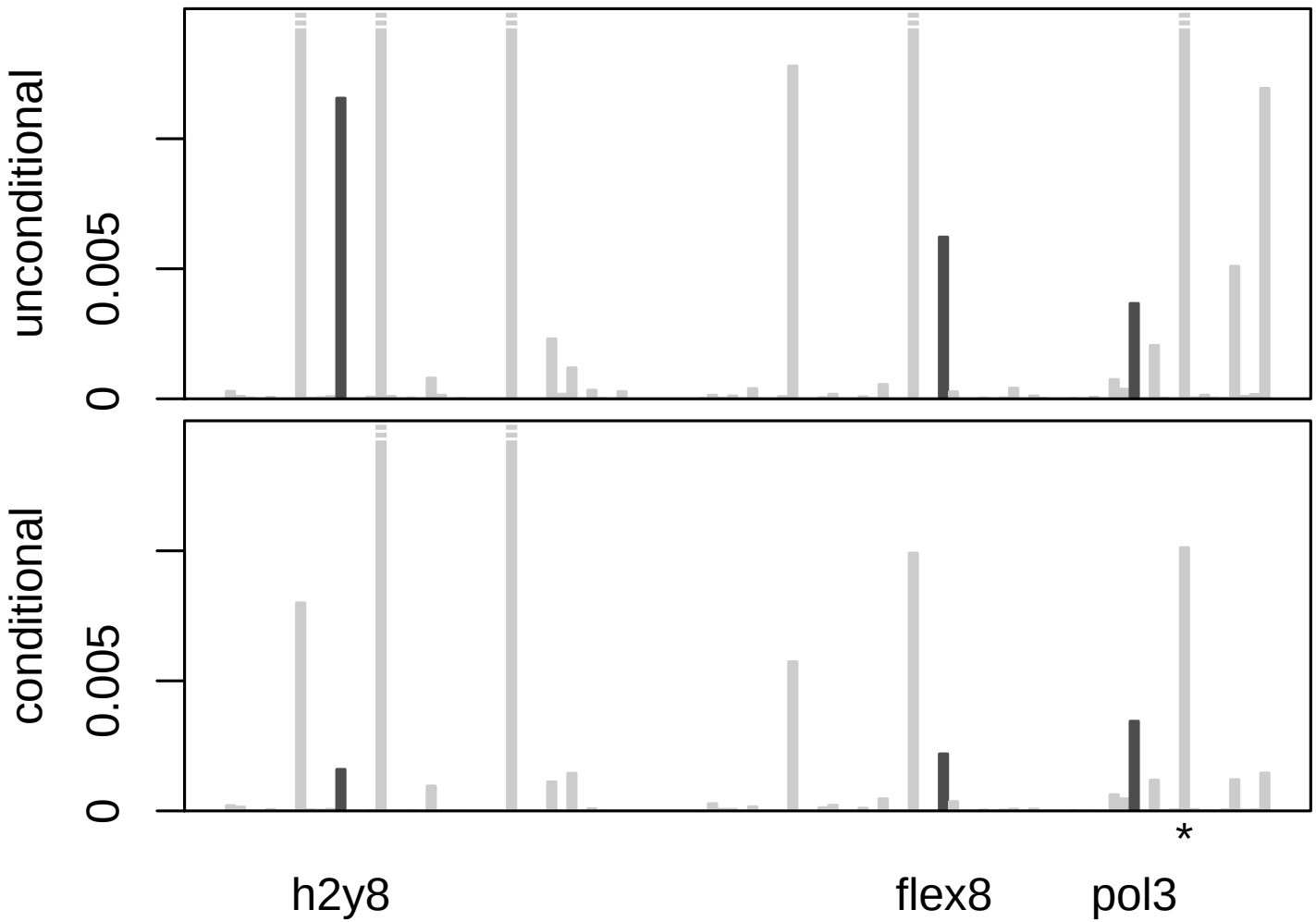
$$\mathbf{\Sigma} = \begin{pmatrix} 1 & 0.9 & 0.9 & 0.9 & 0 & \cdots & 0 \\ 0.9 & 1 & 0.9 & 0.9 & 0 & \cdots & 0 \\ 0.9 & 0.9 & 1 & 0.9 & 0 & \cdots & 0 \\ 0.9 & 0.9 & 0.9 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

X_j	$\mathbf{X_1}$	$\mathbf{X_2}$	$\mathbf{X_3}$	$\mathbf{X_4}$	X_5	X_6	X_7	X_8	$\cdots$	X_{12}
β_j	**5**	**5**	**2**	**0**	-5	-5	-2	0	$\cdots$	0

Results



Peptide-binding data



Summary

Summary

if your predictor variables are of different types:

use `cforest` (pkg: `party`) with default option `controls = cforest_unbiased()`

with permutation importance `varimp(obj)`

Summary

if your predictor variables are of different types:

use `cforest` (pkg: `party`) with default option `controls = cforest_unbiased()`

with permutation importance `varimp(obj)`

otherwise: feel free to use `cforest` (pkg: `party`)

with permutation importance `varimp(obj)`

or `randomForest` (pkg: `randomForest`)

with permutation importance `importance(obj, type=1)`

or Gini importance `importance(obj, type=2)`

but don't fall for the z-score! (i.e. set `scale=FALSE`)

Summary

if your predictor variables are of different types:

use `cforest` (pkg: `party`) with default option `controls = cforest_unbiased()`

with permutation importance `varimp(obj)`

otherwise: feel free to use `cforest` (pkg: `party`)

with permutation importance `varimp(obj)`

or `randomForest` (pkg: `randomForest`)

with permutation importance `importance(obj, type=1)`

or Gini importance `importance(obj, type=2)`

but don't fall for the z-score! (i.e. set `scale=FALSE`)

if your predictor variables are highly correlated: use the conditional importance in `cforest` (pkg: `party`)

Archer, K. J. and R. V. Kimes (2008). Empirical characterization of random forest variable importance measures. *Computational Statistics & Data Analysis 52*(4), 2249–2260.

Breiman, L. (2001). Random forests. *Machine Learning 45*(1), 5–32.

Breiman, L. and A. Cutler (2008). Random forests – Classification manual. Website accessed in 1/2008; http://www.math.usu.edu/~adele/forests.

Breiman, L., A. Cutler, A. Liaw, and M. Wiener (2006). *Breiman and Cutler's Random Forests for Classification and Regression*. R package version 4.5-16.

Diaz-Uriarte, R. (2007). `GeneSrF` and `varselrf`: A web-based tool and R package for gene selection and classification using random forest. *BMC Bioinformatics 8:328*.

Hothorn, T., K. Hornik, and A. Zeileis (2006). Unbiased recursive partitioning: A conditional inference framework. *Journal of Computational and Graphical Statistics 15*(3), 651–674.

Strobl, C., A.-L. Boulesteix, A. Zeileis, and T. Hothorn (2007). Bias in random forest variable importance measures: Illustrations, sources and a solution. *BMC Bioinformatics 8:25*.

Strobl, C. and A. Zeileis (2008). Danger: High power! – exploring the statistical properties of a test for random forest variable importance. In *Proceedings of the 18th International Conference on Computational Statistics, Porto, Portugal*.

Strobl, C., A.-L. Boulesteix, T. Kneib, T. Augustin, and A. Zeileis (2008). Conditional variable importance for random forests. *BMC Bioinformatics 9:307*.