Individual Treatment Effect Prediction Using Model-Based Random Forests

Heidi Seibold, Achim Zeileis, Torsten Hothorn

https://eeecon.uibk.ac.at/~zeileis/

Motivation: Overall treatment effect

Base model:

```
R> basemodel <- model(response ~ treatment, data)
```

Motivation: Treatment-subgroup interaction

Base model:

```
R> basemodel <- model(response ~ treatment, data)
```

Subgroup interaction model:

```
R> sgrpmodel <- model(response ~ treatment * gender, data)
```

Motivation: Treatment-subgroup interaction

Base model:

```
R> basemodel <- model(response ~ treatment, data)
```

Subgroup interaction model:

```
R> sgrpmodel <- model(response ~ treatment * gender, data)
```

Equivalently:

```
R> sgmodel_m <- model(response ~ treatment, data,
+    subset = gender == "male")
R> sgmodel_f <- model(response ~ treatment, data,
+    subset = gender == "female")
```

Motivation: Treatment-subgroup interaction

Base model:

```
R> basemodel <- model(response ~ treatment, data)
```

Subgroup interaction model:

```
R> sgrpmodel <- model(response ~ treatment * gender, data)
```

Equivalently:

```
R> sgmodel_m <- model(response ~ treatment, data,
+    weights = as.numeric(gender == "male"))
R> sgmodel_f <- model(response ~ treatment, data,
+    weights = as.numeric(gender == "female"))
```

Motivation: Treatment-subgroup interaction

Base model:

```
R> basemodel <- model(response ~ treatment, data)
```

Subgroup interaction model:

```
R> sgrpmodel <- model(response ~ treatment * gender, data)
```

Equivalently:

```
R> sgmodel_m <- model(response ~ treatment, data,
+    weights = as.numeric(gender == "male"))
R> sgmodel_f <- model(response ~ treatment, data,
+    weights = as.numeric(gender == "female"))
```

Next steps:

- Find data-driven subgroups.
- Refine from *stratified* to *personalized* treatment effects.

From stratified to personalized treatment effects

Basic idea:

- Treatment-subgroup *interactions* can also be represented by *subgroups* or *weights*.
- Rather than hard 0/1 grouping, a soft weighting would enable observation-specific and thus personalized models.
- Use model-based forests and trees to find the weights in a data-driven way.

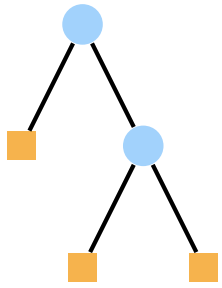
Stratified treatment effects

Goal: Find subgroups of observations that are (almost) homogenous with respect to the parameters of the base model.

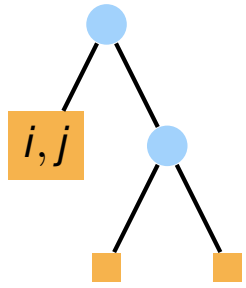
Model-based recursive partitioning:

1. Fit the *base model* to the data – e.g., intercept plus treatment effect.
2. Assess whether the *model scores* are associated with (or change along) any of the available covariates – e.g., using parameter instability tests (*strucchange*) or conditional inference (*coin*).
3. *Split* the sample along the covariate with the strongest association or instability. Choose breakpoint with highest improvement of the model fit – e.g., in terms of log-likelihood.
4. Repeat steps 1–3 recursively in the subgroups until some stopping criterion is met – e.g., for significance or sample size.

Stratified treatment effects

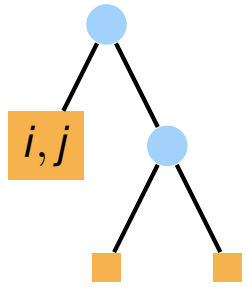


Stratified treatment effects



Weights: Only observations *j* in the same subgroup as observation *i* enter the corresponding subgroup model.

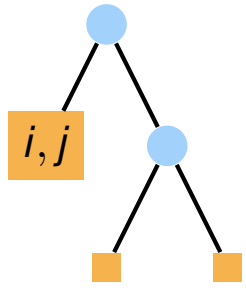
Stratified treatment effects



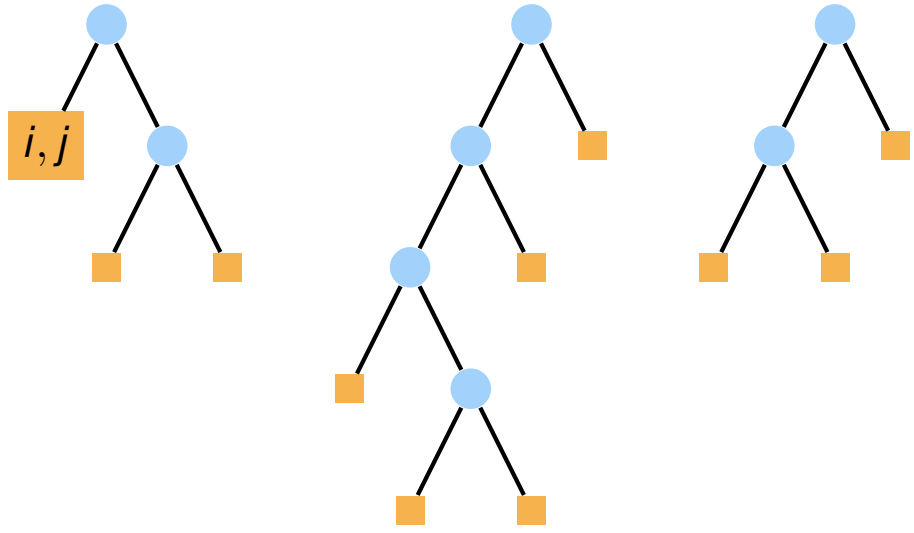
Weights: Only observations *j* in the same subgroup as observation *i* enter the corresponding subgroup model.

```
R> sgmodel_1 <- model(response ~ treatment, data,
+    weights = as.numeric(subgroup == 1))
```

Personalized treatment effects

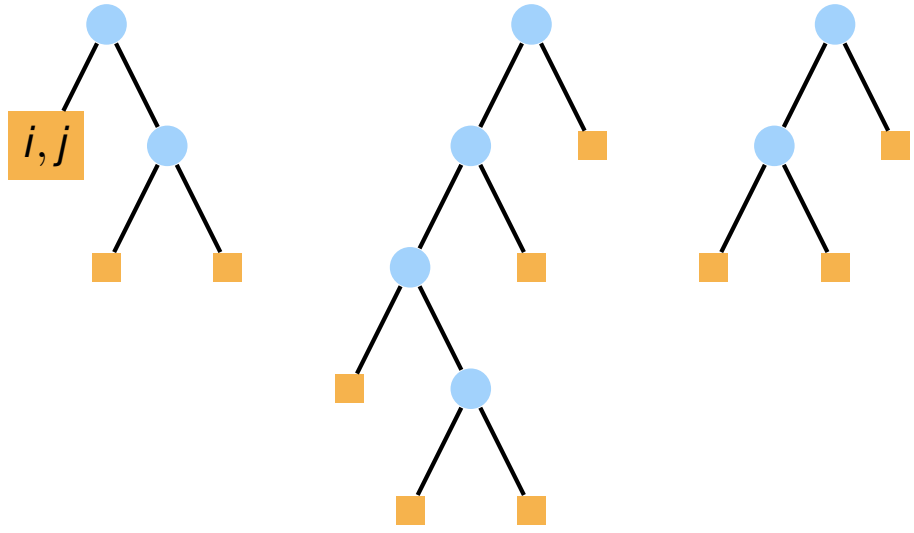


Personalized treatment effects



Weights: Obtain a finer measure of similarity between all observations j and observation i via a forest/ensemble of trees.

Personalized treatment effects

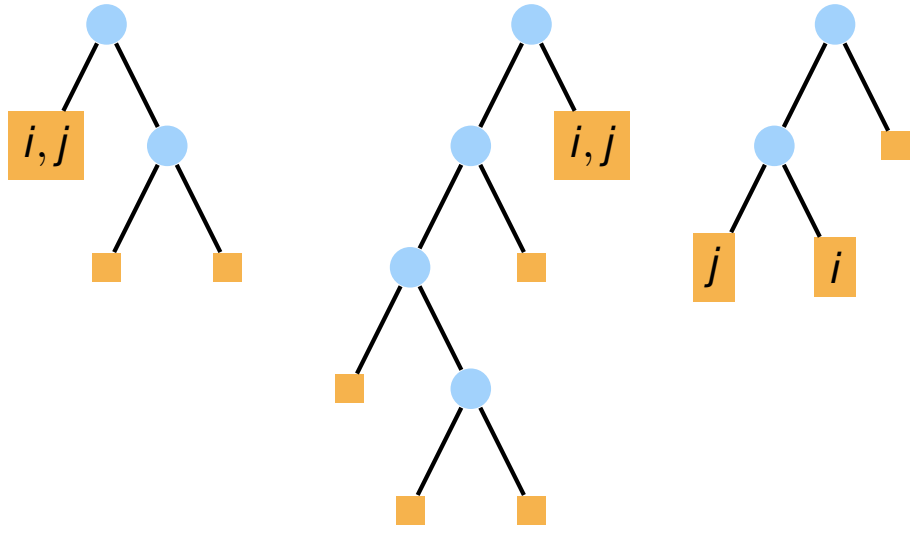


Weights: Obtain a finer measure of similarity between all observations j and observation i via a forest/ensemble of trees.

Randomization:

- Subsample of the training data (per tree).
- Subsample of covariates (per node).

Personalized treatment effects



Weights: Obtain a finer measure of similarity between all observations j and observation i via a forest/ensemble of trees.

Aggregate: The weight of observation j for modeling the treatment effect for observation i is the sum (or mean) of assignments to the same node $\Rightarrow w_{ij} = 2$ (or equivalently $2/3$).

Personalized treatment effects

Personalized model:

```
R> pmodel_i <- model(response ~ treatment, data, weights = w_i)
```

- Observation j enters $w_{ij} = 2$ times in pmodel_i.
- Observations j are the entire learning data.
- Observations i may be in-sample observations from the learning data or new out-of-sample observations.

PRO-ACT database

Pooled Resource Open-Access ALS Clinical Trials Database:

- Amyotrophic lateral sclerosis.
- Riluzole versus no treatment.
- 23 phase-2 clinical trials.
- Two primary endpoints:
  - Survival time
    (3306 patients, 18 covariates).
  - ALS functional rating scale
    (2534 patients, 57 covariates).

`https://nctu.partners.org/ProACT/`

PRO-ACT database

Pooled Resource Open-Access ALS Clinical Trials Database:

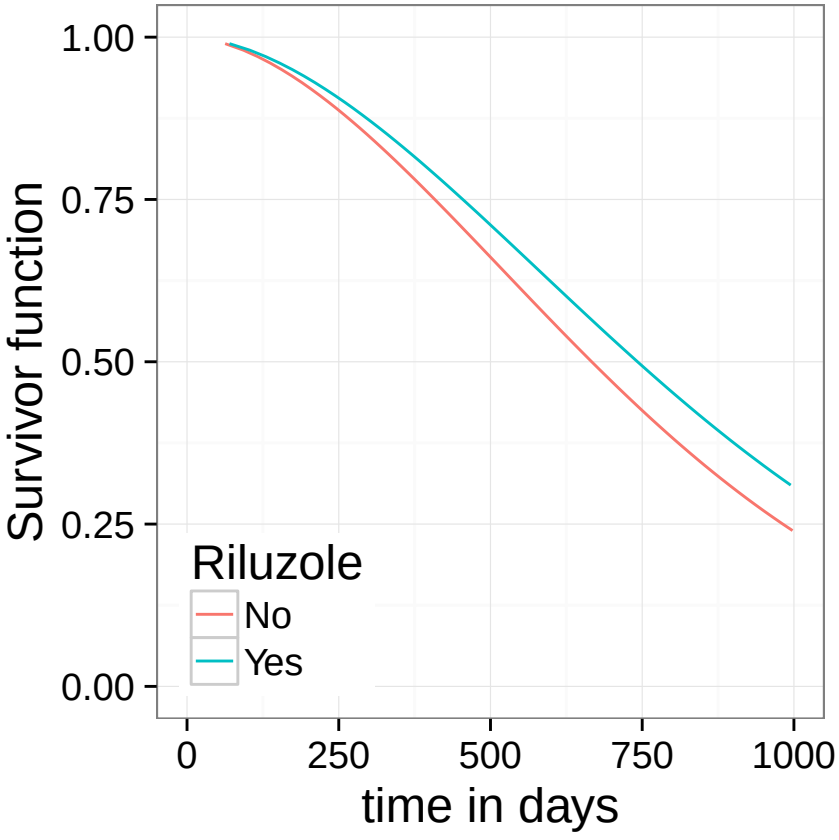
- Amyotrophic lateral sclerosis.
- Riluzole versus no treatment.
- 23 phase-2 clinical trials.
- Two primary endpoints:
  - Survival time
    (3306 patients, 18 covariates).
  - ALS functional rating scale
    (2534 patients, 57 covariates).

`https://nctu.partners.org/ProACT/`

Survival time: Weibull model

Base model:

$$\mathrm{P}(Y \leq y | X = x) \quad = \quad F\left(\frac{\log(y) - \alpha_1 - \beta x}{\alpha_2}\right)$$



	estimate	2.5 %	97.5 %
α_1	6.71	6.64	6.77
β	0.11	0.03	0.18
$\log(\alpha_2)$	−0.58	−0.64	−0.53

Survival time: Weibull model

Base model:

```
R> library("survival")
R> basemodel <- survreg(Surv(survival.time, cens) ~ Riluzole,
+    data = ALSsurvdata, dist = "weibull")
```

Score extractor:

```
R> wbscore <- function(data, weights) {
+
+    mod <- survreg(Surv(survival.time, cens) ~ Riluzole,
+      data = data, weights = weights, subset = weights > 0,
+      dist = "weibull", init = c(6.7, 0))
+
+    ef <- as.matrix(sandwich::estfun(mod))
+
+    ret <- matrix(0, nrow = nrow(data), ncol = ncol(ef))
+    ret[weights > 0,] <- ef
+    ret
+  }
```

Survival time: Weibull forest

Weibull forest:

```
R> alsforest <-  cforest(
+    survival.time + cens + Riluzole ~ age + gender + etc,
+    data = ALSsurvdata, ytrafo = wbscore,
+    ntree = 100, perturb = list(replace = FALSE))
```

Weights:

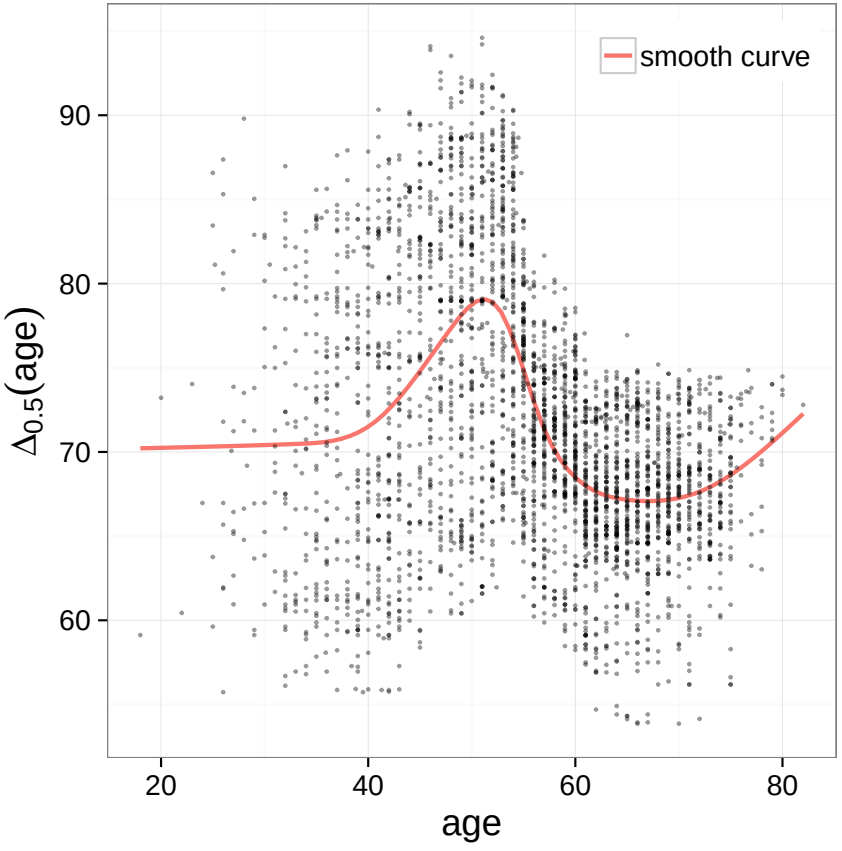
```
R> w <- predict(alsforest, type = "weights", OOB = TRUE)
```

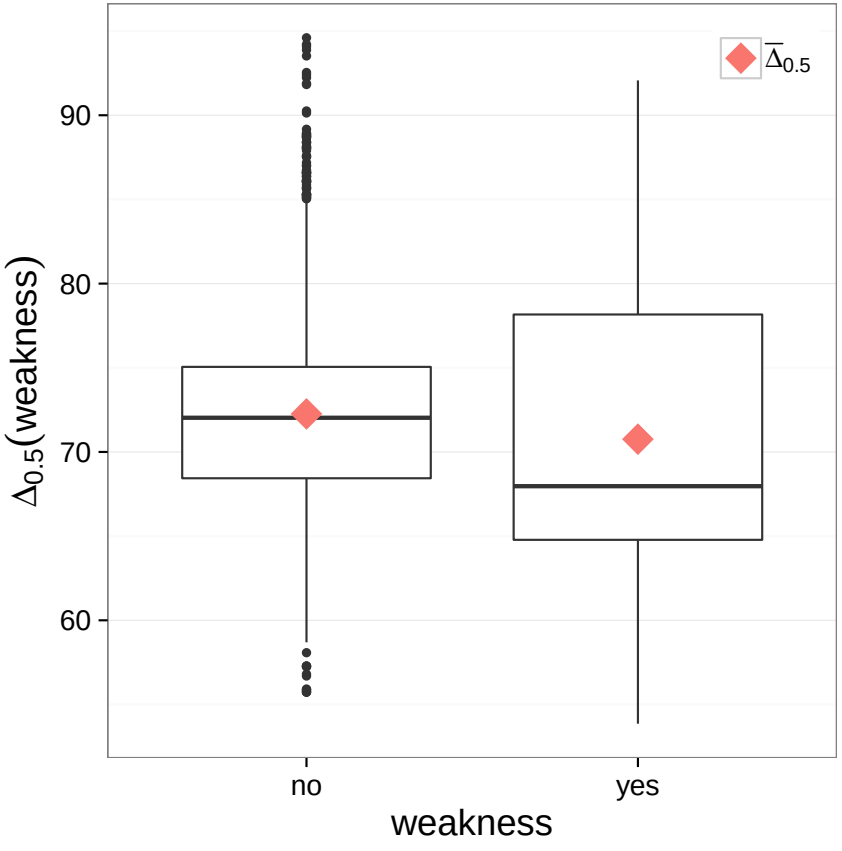
Personalized model for patient *i*:

```
R> pmodel_i <- survreg(Surv(survival.time, cens) ~ Riluzole,
+    data = ALSsurvdata, dist = "weibull", weights = w[, i])
```

Survival time: Dependence plots

Visualization: Dependence of median survival time difference on most important patient characteristics.

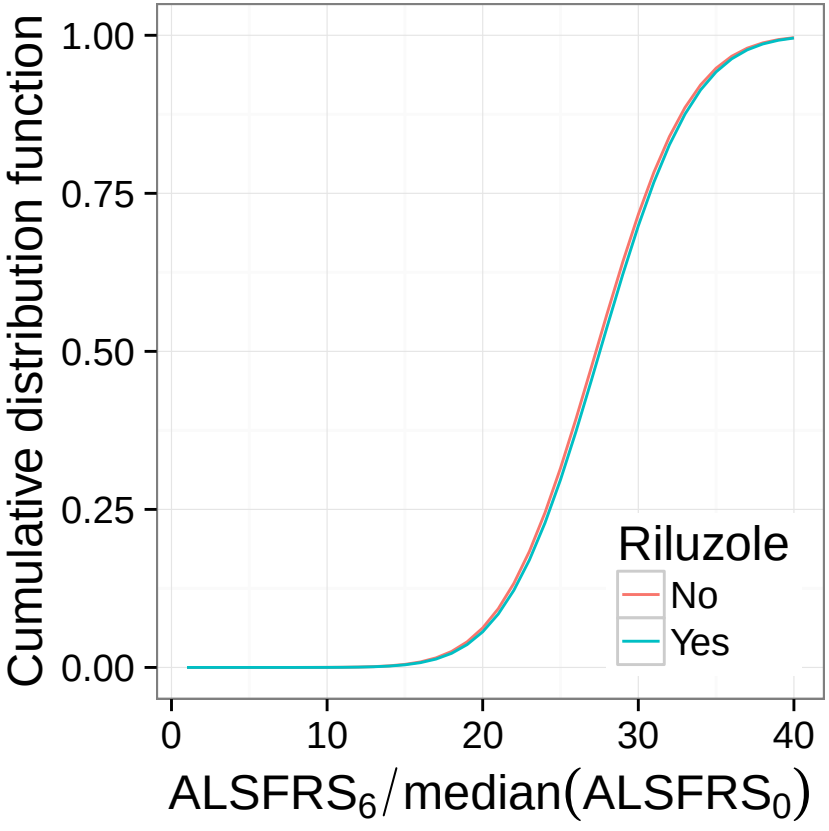




ALSFRS: Gaussian GLM with log link

Base model:

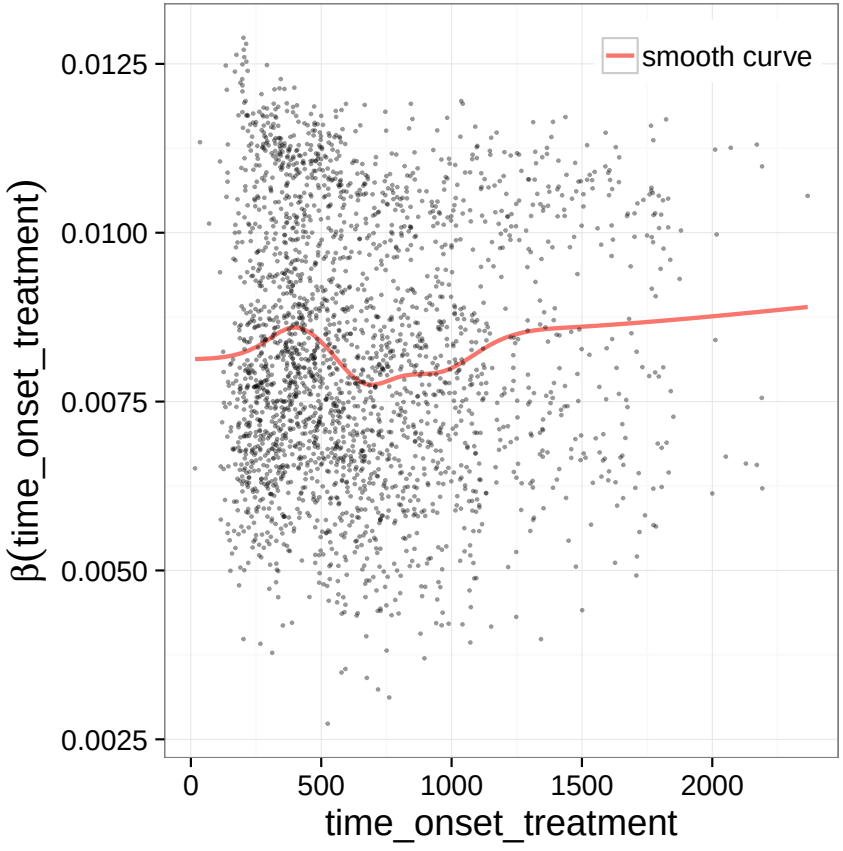
$$\mathrm{E}\left(\frac{\mathrm{ALSFRS}_6}{\mathrm{ALSFRS}_0}\middle| X = x\right) = \frac{\mathrm{E}(\mathrm{ALSFRS}_6|X=x)}{\mathrm{ALSFRS}_0} = \exp\{\alpha + \beta x\}$$

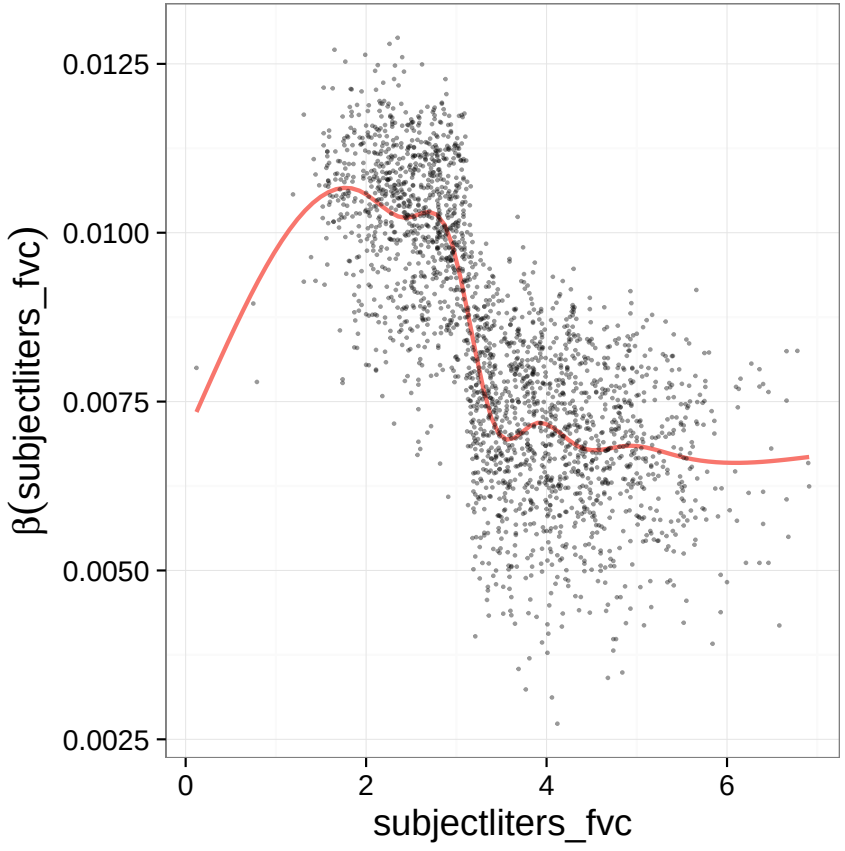


	estimate	2.5 %	97.5 %
α	−0.16	−0.17	−0.15
β	0.01	−0.01	0.02

ALSFRS: Dependence plots

Visualization: Dependence of treatment effect β_i (log-scale) on most important patient characteristics.





Check for overfitting

Assessment: Difference in log-likelihood against base model.

$$\begin{aligned}\Delta(\ell) &= \sum_{i=1}^{n} \ell((\text{response}, \text{treatment})_i, \text{pmodel}_i) - \\ &\quad \sum_{i=1}^{n} \ell((\text{response}, \text{treatment})_i, \text{basemodel})\end{aligned}$$

Comparison: Observed vs. maximum obtained in 50 parametric bootstrap samples drawn under the base-model null hypothesis.

$\Delta(\ell)$	Survival	ALSFRS
Observed	71.5	851.0
Maximum bootstrapped	1.0	122.7

References

Seibold H, Zeileis A, Hothorn T (2017). "Individual Treatment Effect Prediction for ALS Patients." *Statistical Methods in Medical Research*, Forthcoming. Preprint version at `https://arxiv.org/abs/1604.08720`

Seibold H, Zeileis A, Hothorn T (2016). "Model-Based Recursive Partitioning for Subgroup Analyses." *International Journal of Biostatistics*, **12**(1), 45–63. `doi:10.1515/ijb-2015-0032`

Hothorn T, Zeileis A (2015). "partykit: A Modular Toolkit for Recursive Partytioning in R." *Journal of Machine Learning Research*, **16**, 3905–3909. `http://www.jmlr.org/papers/v16/hothorn15a.html`

R package:
`https://CRAN.R-project.org/package=partykit`

Replication materials for personalized models:
`https://github.com/HeidiSeibold/personalised_medicine`