Variable Selection and Model Choice in Survival Models with Time-Varying Effects

Boosting Survival Models

Benjamin Hofner [1]

Department of Medical Informatics, Biometry and Epidemiology (IMBE)
Friedrich-Alexander-Universität Erlangen-Nürnberg

joint work with Thomas Kneib and Torsten Hothorn

Department of Statistics
Ludwig-Maximilians-Universität München

useR! 2008

[1] benjamin.hofner@imbe.med.uni-erlangen.de

Introduction

Cox PH model:

$$\lambda_i(t) = \lambda(t, \mathbf{x}_i) = \lambda_0(t) \exp(\mathbf{x}_i' \beta)$$

with

- $\lambda_i(t)$ hazard rate of observation i $[i = 1, \ldots, n]$
- $\lambda_0(t)$ baseline hazard rate
- $\mathbf{x}_i$ vector of covariates for observation i $[i = 1, \ldots, n]$
- β vector of regression coefficients

Problem: restrictive model, not allowing for

- non-proportional hazards (e.g., time-varying effects)
- non-linear effects

Additive Hazard Regression

Generalisation: Additive Hazard Regression (Kneib & Fahrmeir, 2007)

$$\lambda_i(t) = \exp(\eta_i(t))$$

with

$$\eta_i(t) = \sum_{j=1}^{J} f_j(\mathbf{x_i}(t)),$$

generic representation of covariate effects $f_j(\mathbf{x}_i)$

a) linear effects: $f_j(\mathbf{x}_i(t)) = f_{\text{linear}}(\tilde{x}_i) = \tilde{x}_i\beta$

b) smooth effects: $f_j(\mathbf{x}_i(t)) = f_{\text{smooth}}(\tilde{x}_i)$

c) time-varying effects: $f_j(\mathbf{x}_i(t)) = f_{\text{smooth}}(t) \cdot \tilde{x}_i$

where $\tilde{x}_i \in \mathbf{x}_i(t)$.

Note:

c) includes log-baseline for $\tilde{x}_i \equiv 1$

P-Splines

flexible terms can be represented using P-splines (Eilers & Marx, 1996)

- model term (x can be either $\tilde{x}_i$ or t):

$$f_j(x) = \sum_{m=1}^{M} \beta_{jm} B_{jm}(x) \quad (j = 1, \ldots, J)$$

- penalty:

$$\text{pen}_j(\boldsymbol{\beta_j}) = \begin{cases} \kappa_j \, \boldsymbol{\beta_j}' \mathbf{K} \boldsymbol{\beta_j} & \text{cases b),c)} \\ 0 & \text{case a)} \end{cases}$$

with

- $\mathbf{K} = \mathbf{D}'\mathbf{D}$ (i.e., cross product of difference matrix $\mathbf{D}$)

$$\mathbf{D} \overset{e.g.}{=} \begin{pmatrix} 1 & -2 & 1 & \ldots & \\ 0 & 1 & -2 & 1 & \ldots \end{pmatrix}$$

- κ_j smoothing parameter
  (larger $\kappa_j \Rightarrow$ more penalization $\Rightarrow$ smoother fit)

Inference

Penalized Likelihood Criterion: (NB: this is the **full** log-likelihood)

$$\mathcal{L}_{\text{pen}}(\beta) = \sum_{i=1}^{n} \left[\delta_i \eta_i(t_i) - \int_0^{t_i} \exp(\eta_i(t))\, dt \right] - \sum_{j=0}^{J} \text{pen}_j(\beta_j)$$

- T_i true survival time
- C_i censoring time
- $t_i = \min(T_i, C_i)$ observed survival time (right censoring)
- $\delta_i = \mathbb{1}(T_i \leq C_i)$ indicator for non-censoring

Problem:

Estimation and in particular model choice

$\text{Cox}_{\text{flex}}\text{Boost}$

Aim:

Maximization of a (potentially) high-dimensional log-likelihood with different modeling alternatives

Thus, we use:

- Iterative algorithm
- Likelihood-based boosting algorithm
- Component-wise base-learners

Therefore:

- Use one base-learner $g_j(\cdot)$ for each covariate (or each model component) $\quad [\, j \in \{1, \ldots, J\} \,]$

Component-Wise Boosting

as a means of estimation and variable selection combined with model choice.

Cox$_{\text{flex}}$Boost

Aim:

Maximization of a (potentially) high-dimensional log-likelihood with different modeling alternatives

Thus, we use:

- Iterative algorithm
- Likelihood-based boosting algorithm
- Component-wise base-learners

Therefore:

- Use one base-learner $g_j(\cdot)$ for each covariate (or each model component) $[\, j \in \{1, \ldots, J\} \,]$

Component-Wise Boosting

as a means of estimation and variable selection combined with model choice.

Cox_{flex}Boost

Aim:

Maximization of a (potentially) high-dimensional log-likelihood with different modeling alternatives

Thus, we use:

- Iterative algorithm
- Likelihood-based boosting algorithm
- Component-wise base-learners

Therefore:

- Use one base-learner $g_j(\cdot)$ for each covariate (or each model component) $[\ j \in \{1, \ldots, J\}\ ]$

Component-Wise Boosting

as a means of estimation and variable selection combined with model choice.

Cox$_{\text{flex}}$Boost Algorithm

(i) **Initialization:** Iteration index $m := 0$.

- Function estimates (for all $j \in \{1, \dots, J\}$):

$$\hat{f}_j^{[0]}(\cdot) \equiv 0$$

- Offset (MLE for constant log hazard):

$$\hat{\eta}^{[0]}(\cdot) \equiv \log\left(\frac{\sum_{i=1}^n \delta_i}{\sum_{i=1}^n t_i}\right)$$

(ii) **Estimation:** $m := m + 1$.
Fit all (linear/P-spline) base-learners separately

$$\hat{g}_j = g_j(\cdot\ ; \hat{\boldsymbol{\beta}}_j),\ \forall j \in \{1, \ldots, J\},$$

by penalized MLE, i.e.,

$$\hat{\boldsymbol{\beta}}_j = \arg\max_{\boldsymbol{\beta}} \mathcal{L}_{j,\text{pen}}^{[m]}(\boldsymbol{\beta})$$

with the penalized log-likelihood (analogously as above)

$$\begin{aligned}\mathcal{L}_{j,\text{pen}}^{[m]}(\boldsymbol{\beta}) \quad = \quad & \sum_{i=1}^{n} \Big[\delta_i \cdot (\hat{\eta}_i^{[m-1]} + g_j(x_i(t_i); \boldsymbol{\beta})) \\ & - \int_0^{t_i} \exp\left\{ \hat{\eta}_i^{[m-1]}(\tilde{\mathbf{t}}) + g_j(x_i(\tilde{\mathbf{t}}); \boldsymbol{\beta}) \right\} d\tilde{\mathbf{t}} \Big] - \text{pen}_j(\boldsymbol{\beta}),\end{aligned}$$

with the additive predictor η_i split

- into the estimate from previous iteration $\hat{\eta}_i^{[m-1]}$
- and the current base-learner $g_j(\cdot; \boldsymbol{\beta})$

(iii) **Selection:** Choose base-learner $\hat{g}_{j^*}$ with

$$j^* = \arg \max_{j \in \{1,\ldots,J\}} \mathcal{L}^{[m]}_{j,\text{unpen}}(\hat{\beta}_j)$$

(iv) **Update:**

- Function estimates (for all $j \in \{1, \ldots, J\}$):

$$\hat{f}_j^{[m]} = \begin{cases} \hat{f}_j^{[m-1]} + \nu \cdot \hat{g}_j & j = j^* \\ \hat{f}_j^{[m-1]} & j \neq j^* \end{cases}$$

- Additive predictor (= fit):

$$\hat{\eta}^{[m]} = \hat{\eta}^{[m-1]} + \nu \cdot \hat{g}_{j^*}$$

with step-length $\nu \in (0, 1]$ (here: $\nu = 0.1$)

(v) **Stopping rule:** Continue iterating steps (ii) to (iv) until $m = m_{\text{stop}}$

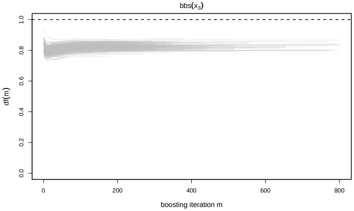
Some Aspects of Cox$_{\text{flex}}$Boost

Estimation	full penalized MLE $\cdot \nu$ (step-length)
Selection	based on unpenalized log-likelihood $\mathcal{L}_{j,\text{unpen}}^{[m]}$
Base-Learners	specified by (initial) degrees of freedom, i.e., $\text{df}_j = \widetilde{df}_j$

- Likelihood-based boosting (in general):
  See, e.g., Tutz and Binder (2006)
- Above aspects in Cox$_{\text{flex}}$Boost:
  See, e.g., model based boosting (Bühlmann & Hothorn, 2007)

Degrees of Freedom

- Specifying df more intuitive than specifying smoothing parameter κ
- Comparable to other modeling components, e.g., linear effects
- Problem: Not constant over the (boosting) iterations
  - But simulation studies showed: No big deviation from the initial $\text{df}_j = \widetilde{df}_j$



Estimated degrees of freedom traced over the boosting steps for the flexible base-learners of x_3 (in 200 replicates) and initially specified degrees of freedom (dashed line).

Model Choice

Recall from generic representation:

$f_j(\tilde{x}_i)$ can be a

a) linear effect: $f_j(\mathbf{x}_i(t)) = f_{\mathrm{linear}}(\tilde{x}_i) = \tilde{x}_i\beta$

b) smooth effect: $f_j(\mathbf{x}_i(t)) = f_{\mathrm{smooth}}(\tilde{x}_i)$

c) time-varying effect: $f_j(\mathbf{x}_i(t)) = f_{\mathrm{smooth}}(t) \cdot \tilde{x}_i$

- $\Rightarrow$ We see: $\tilde{x}_i$ can enter the model in 3 different ways
- But how?
- Add all possibilities as base-learners to the model. Boosting can chose between the possibilities
- But the df must be comparable! Otherwise: more flexible base-learners are preferred

Model Choice

Recall from generic representation:

$f_j(\tilde{x}_i)$ can be a

a) linear effect: $f_j(\mathbf{x}_i(t)) = f_{\text{linear}}(\tilde{x}_i) = \tilde{x}_i\beta$

b) smooth effect: $f_j(\mathbf{x}_i(t)) = f_{\text{smooth}}(\tilde{x}_i)$

c) time-varying effect: $f_j(\mathbf{x}_i(t)) = f_{\text{smooth}}(t) \cdot \tilde{x}_i$

- $\Rightarrow$ We see: $\tilde{x}_i$ can enter the model in 3 different ways
- But how?
- Add all possibilities as base-learners to the model.
  Boosting can chose between the possibilities
- But the df must be comparable!
  Otherwise: more flexible base-learners are preferred

- For higher order differences ($d \geq 2$): df > 1 ($\kappa \rightarrow \infty$)
- Polynomial of order $d - 1$ remains unpenalized
- **Solution:**

Decomposition (based on Kneib, Hothorn, & Tutz, 2008)

$$g(x) = \underbrace{\beta_0 + \beta_1 x + \ldots + \beta_{d-1} x^{d-1}}_{\text{unpenalized, parametric part}} \quad + \quad \underbrace{g_{centered}(x)}_{\text{deviation from polynomial}}$$

- Add unpenalized part as separate, parametric base-learners
- Assign df $= 1$ to the centered effect (and add as P-spline base-learner)
- Analogously for time-varying effects

Technical realization (see Fahrmeir, Kneib, & Lang, 2004):

decomposing the vector of regression coefficients β into $(\widetilde{\beta}_{unpen}, \widetilde{\beta}_{pen})$ utilizing a spectral decomposition of the penalty matrix

Early Stopping

1. Run the algorithm m_{stop}-times (previously defined).
2. Determine new $\widehat{m}_{\text{stop,opt}} \leq m_{\text{stop}}$:
    - ... based on out-of-bag sample (with simulations easy to use)
    - ... based on information criterion, e.g., AIC

$\Rightarrow$ Prevents algorithm to stop in a local maximum (of the log-likelihood)

$\Rightarrow$ Early stopping prevents overfitting

Variable Selection and Model Choice

... is achieved by

- selection of base-learner (in step (iii) of Cox$_{\text{flex}}$Boost), i.e., component-wise boosting
  and
- early stopping

Simulation-Results (in Short)

- Good variable selection strategy
- Good model choice strategy if only linear and smooth effects are used
- Selection bias in favor of time-varying base-learners (if present) $\Rightarrow$ standardizing time could be a solution
- Estimates are better if model choice is performed

Computational Aspects

Cox$_{\text{flex}}$Boost is implemented using R

- Crucial computation: Integral in $\mathcal{L}_{j,\text{pen}}^{[m]}(\boldsymbol{\beta})$:

$$\int_0^{t_i} \exp\left\{\hat{\eta}_i^{[m-1]}(\tilde{\mathbf{t}}) + g_j(x_i(\tilde{\mathbf{t}}); \boldsymbol{\beta})\right\} d\tilde{\mathbf{t}}$$

  - time consuming
  - very often evaluated (maximization of $\mathcal{L}_{j,\text{pen}}^{[m]}(\boldsymbol{\beta})$)

- R-function `integrate()` slow in this context
  $\Rightarrow$ (specialized) vectorized trapezoid integration implemented
  $\Rightarrow \approx 100$ times quicker
- Efficient storage of matrices can reduce computational burden
  $\Rightarrow$ recycling of results

Computational Aspects

Cox$_{\text{flex}}$Boost is implemented using R

- Crucial computation: Integral in $\mathcal{L}_{j,\text{pen}}^{[m]}(\beta)$:

$$\int_0^{t_i} \exp\left\{\hat{\eta}_i^{[m-1]}(\tilde{\mathbf{t}}) + g_j(x_i(\tilde{\mathbf{t}});\beta)\right\} d\tilde{\mathbf{t}}$$

  - time consuming
  - very often evaluated (maximization of $\mathcal{L}_{j,\text{pen}}^{[m]}(\beta)$)
- R-function `integrate()` slow in this context
  $\Rightarrow$ (specialized) vectorized trapezoid integration implemented
  $\Rightarrow \approx$ 100 times quicker
- Efficient storage of matrices can reduce computational burden
  $\Rightarrow$ recycling of results

Computational Aspects

Cox$_{\text{flex}}$Boost is implemented using R

- Crucial computation: Integral in $\mathcal{L}^{[m]}_{j,\text{pen}}(\beta)$:

$$\int_0^{t_i} \exp\left\{\hat{\eta}_i^{[m-1]}(\tilde{\mathbf{t}}) + g_j(x_i(\tilde{\mathbf{t}});\beta)\right\} d\tilde{\mathbf{t}}$$

  - time consuming
  - very often evaluated (maximization of $\mathcal{L}^{[m]}_{j,\text{pen}}(\beta)$)
- `R`-function `integrate()` slow in this context
  $\Rightarrow$ (specialized) vectorized trapezoid integration implemented
  $\Rightarrow \approx 100$ times quicker
- Efficient storage of matrices can reduce computational burden
  $\Rightarrow$ recycling of results

Summary & Outlook

Cox_{flex}Boost ...

- ... allows for variable selection and model choice.
- ... allows for flexible modeling
  - flexible, non-linear effects
  - time-varying effects (i.e., non-proportional hazards)
- ... provides functions to manipulate and show results (`summary()`, `plot()`, `subset()`, ...)

To be continued ...

- Formula for AIC (for Boosting in Survival Models)
- Include mandatory covariates (update in each step)
- Measure for variable importance: e.g., $\int |\hat{f}_j^{[m_{stop}]}(\cdot)|$

Summary & Outlook

Cox_{flex}Boost ...

- ...allows for variable selection and model choice.
- ...allows for flexible modeling
  - flexible, non-linear effects
  - time-varying effects (i.e., non-proportional hazards)
- ...provides functions to manipulate and show results (`summary()`, `plot()`, `subset()`, ...)

To be continued ...

- Formula for AIC (for Boosting in Survival Models)
- Include mandatory covariates (update in each step)
- Measure for variable importance: e.g., $\int |\hat{f}_j^{[m_{stop}]}(\cdot)|$

Literature

Bühlmann, P., & Hothorn, T. (2007). Boosting algorithms: Regularization, prediction and model fitting. *Statistical Science*, *22*(4), 477-505.

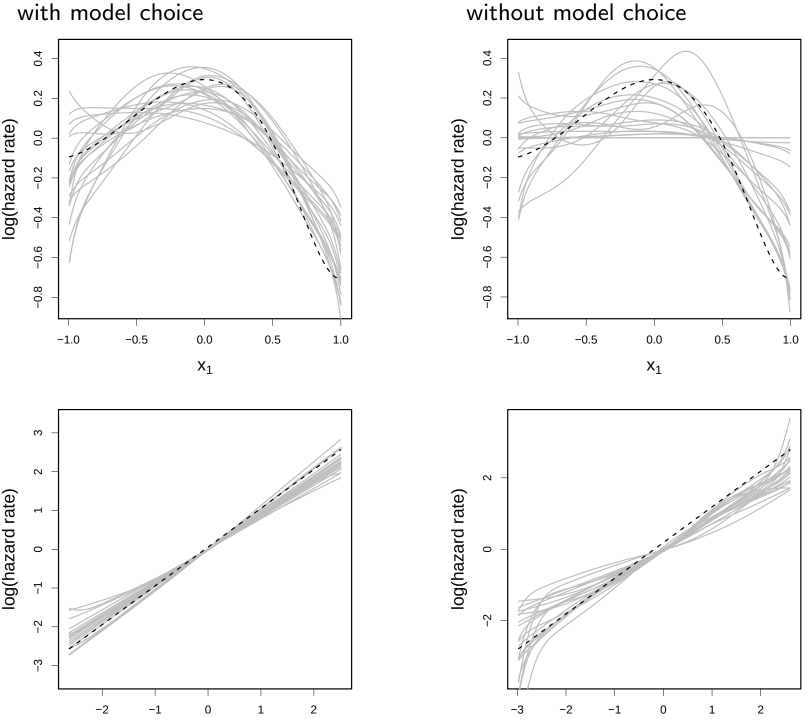
Eilers, P. H. C., & Marx, B. D. (1996). Flexible smoothing with B-splines and penalties. *Statistical Science*, *11*(2), 89–121.

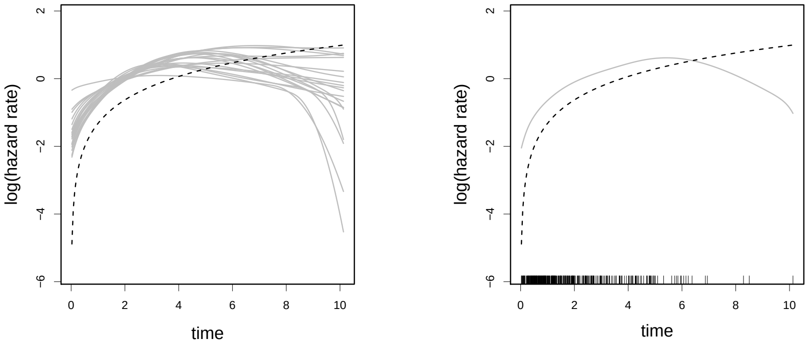
Fahrmeir, L., Kneib, T., & Lang, S. (2004). Penalized structured additive regression: A Bayesian perspective. *Statistica Sinica*, *14*, 731–761.

Kneib, T., & Fahrmeir, L. (2007). A mixed model approach for geoadditive hazard regression. *Scand. J. Statist.*, *34*, 207–228.

Kneib, T., Hothorn, T., & Tutz, G. (2008). Variable selection and model choice in geoadditive regression. *Biometrics (accepted)*.

Tutz, G., & Binder, H. (2006). Generalized additive modelling with implicit variable selection by likelihood-based boosting. *Biometrics*, *62*, 961–971.


with model choice
without model choice
log(hazard rate)
x_1


log(hazard rate)
2
0
−2
−4
−6
0
2
4
6
8
10
time
log(hazard rate)
2
0
−2
−4
−6
0
2
4
6
8
10
time