University of Zurich[UZH]

EBPI Epidemiology, Biostatistics and Prevention Institute

Score-based Transformation Learning

Torsten Hothorn

Dept. Biostats, NY State Univ. at Buffalo, May 6, 2021

Before we start...

The views presented in this talk are neither completely new nor completely my own.

References can be found in published papers.

I oversimplify and exaggerate quite a bit in some places.

Statistics 101

$Y_1, \dots, Y_N$ iid from model $Y_i \sim \mathcal{M}(\boldsymbol{\vartheta})$ with parameters $\boldsymbol{\vartheta} \in \Theta$

ML (as in "Maximum Likelihood")

$$\hat{\boldsymbol{\vartheta}} = \operatorname*{arg\,max}_{\boldsymbol{\vartheta} \in \Theta} \ell(\boldsymbol{\vartheta})$$

with log-likelihood

$$\ell(\boldsymbol{\vartheta}) = \sum_{i=1}^{N} \ell_i(\boldsymbol{\vartheta})$$

Machine Learning 101

$Y_1, \dots, Y_N$ iid from model $Y_i \sim \mathcal{M}(\boldsymbol{\vartheta})$ with parameters $\boldsymbol{\vartheta} \in \Theta$

ML (as in "Machine Learning")

$$\hat{\boldsymbol{\vartheta}} = \operatorname*{arg\,min}_{\boldsymbol{\vartheta} \in \Theta} R(\boldsymbol{\vartheta})$$

with empirical risk

$$R(\boldsymbol{\vartheta}) = \sum_{i=1}^{N} R_i(\boldsymbol{\vartheta})$$

“Interpretable” Machine Learning

“Predictive” modelling: Parameter(s) $\boldsymbol{\vartheta}(\mathbf{x})$ depend on explanatory variables $\mathbf{x}$, in the simplest case

$$\mathbb{E}(Y \mid \mathbf{X} = \mathbf{x}) = \vartheta(\mathbf{x})$$

“Interpretable”: $\vartheta(\mathbf{x})$ is human readable, for example $= \mathbf{x}^\top \boldsymbol{\beta}$

Today: Interpret and understand optimiser / algorithm

$$\hat{\boldsymbol{\vartheta}} = \operatorname*{arg\,max}_{\boldsymbol{\vartheta} \in \Theta} \ell(\boldsymbol{\vartheta})$$

in light of model

$$\mathcal{M}(\boldsymbol{\vartheta})$$

We have been there before...

1990+: neural networks and binary logistic regression

1995+: decision trees and mixture models

2000+: boosting and additive models

2010+: support vector machines and mixed models

2017+: random forests and locally adaptive maximum likelihood

Statistics 101

For "nice" models we have

$$\hat{\boldsymbol{\vartheta}} = \operatorname*{arg\,max}_{\boldsymbol{\vartheta} \in \Theta} \ell(\boldsymbol{\vartheta}) \iff \left. \frac{\partial \ell(\boldsymbol{\vartheta})}{\partial \boldsymbol{\vartheta}} \right|_{\boldsymbol{\vartheta} = \hat{\boldsymbol{\vartheta}}} = 0$$

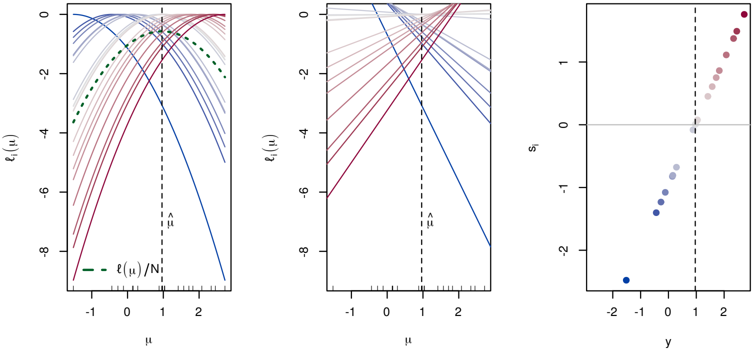
The contributions to these "estimating" or "score" equations are

$$0 = \left. \frac{\partial \ell(\boldsymbol{\vartheta})}{\partial \boldsymbol{\vartheta}} \right|_{\boldsymbol{\vartheta} = \hat{\boldsymbol{\vartheta}}} = \sum_{i=1}^{N} \left. \frac{\partial \ell_i(\boldsymbol{\vartheta})}{\partial \boldsymbol{\vartheta}} \right|_{\boldsymbol{\vartheta} = \hat{\boldsymbol{\vartheta}}} =: \sum_{i=1}^{N} \mathbf{S}_i$$

What can we learn from the scores (or score contributions) $\mathbf{S}_i$ (apart from $N^{-1} \sum_i \mathbf{S}_i \mathbf{S}_i^\top \overset{\mathbb{P}}{\to} I(\vartheta)$)?

Example

$Y_i \sim \mathrm{N}(\mu, 1)$ with $N = 20$. Estimate mean $\boldsymbol{\vartheta} = \mu$.
$\ell_i(\mu) = -1/2(Y_i - \mu)^2$, $S_{i,\mu} = Y_i - \hat{\mu}$



Residuals

In normal linear models

$$(Y_i \mid \mathbf{X} = \mathbf{x}_i) = \mu + \mathbf{x}_i^\top \boldsymbol{\beta} + \sigma Z, \quad Z \sim \mathrm{N}(0, 1)$$

with L_2 risk (or normal log-likelihood, σ is "nuisance")

$$\ell_i((\mu, \boldsymbol{\beta})) = -1/2(Y_i - (\mu + \mathbf{x}_i^\top \boldsymbol{\beta}))^2$$

we obtain score contributions

$$\mathbf{S}_i = (Y_i - (\mu + \mathbf{x}_i^\top \boldsymbol{\beta}))(1, \mathbf{x}_i)^\top$$

and least-squares residuals

$$S_{i,1} = S_{i,\mu} = Y_i - (\mu + \mathbf{x}_i^\top \boldsymbol{\beta}) = \text{observed - predicted}$$

Residuals

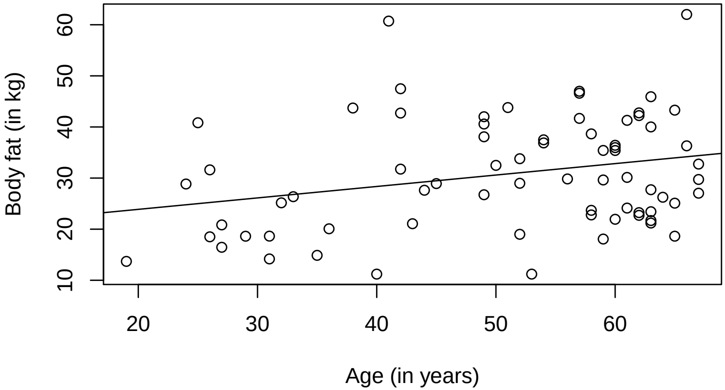
Least-squares residuals are very helpful for model diagnostics.

Idea:

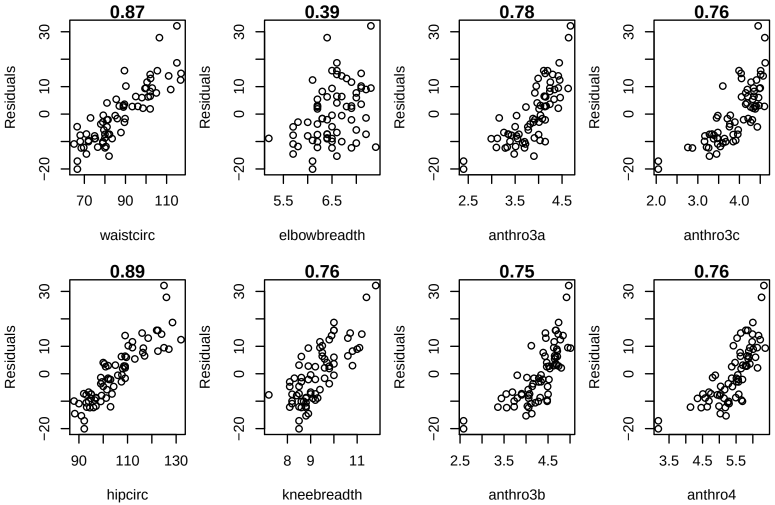
1. Start with a simple model
2. Check if residuals can be explained by covariates
3. Add most important covariate to model
4. Iterate

Example: Body fat for 71 females explained by anthropometric measurements

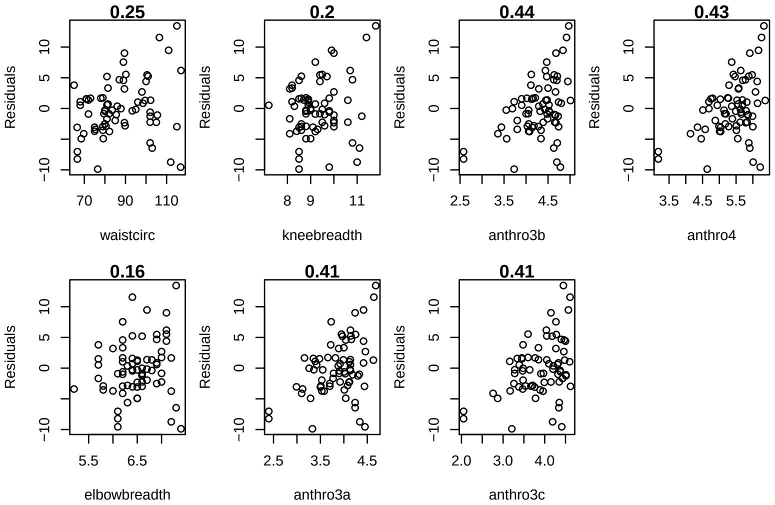
Body Fat $\sim$ Age



Body Fat ∼ Age


0.87
0.39
0.78
0.76
0.89
0.76
0.75
0.76
Residuals
waistcirc
elbowbreadth
anthro3a
anthro3c
hipcirc
kneebreadth
anthro3b
anthro4

Body Fat $\sim$ Age + Hip Circumference



Oups!

This was L_2 boosting with $m_{\text{stop}} = 3$ iterations, step size $\nu = 1$ and simple linear models as base-learners.

With $m_{\text{stop}} = 2$ and univariate smoothers, this procedure was described as “twicing” by John Tukey in his book “Exploratory Data Analysis” (1977) as I learned from Peter Bühlmann.

How can we use this idea to estimate parameters in (way more) complex models, let’s say in transformation models? (I’m lazy, so I want to cover as many models as possible with as little work as possible.)

More General Residuals

There is no intercept in linear transformation models

$$\mathbb{P}(Y \leq y \mid \mathbf{X} = \mathbf{x}) = F_Z(h(y) + \mathbf{x}^\top \boldsymbol{\beta})$$

for example in a Cox model with $F_Z = \text{cloglog}^{-1}$ (where h is the log-cumulative baseline hazard function and $\boldsymbol{\beta}$ log-hazard ratios).

Trick: Introduce $\alpha \equiv 0$ in the model

$$\mathbb{P}(Y \leq y \mid \mathbf{X} = \mathbf{x}) = F_Z(h(y) + \mathbf{x}^\top \boldsymbol{\beta} + \alpha)$$

and use score $S_{i,\alpha}$ (which is equivalent to least-squares residuals $S_{i,\mu}$ in a linear model) as a residual.

Example: Score Test for Comparing Two Groups

Model ($\alpha \equiv 0$):

$$\begin{aligned} \mathbb{P}(Y \leq y \mid \text{placebo}) &= \text{expit}(h(y) + \alpha) \\ \mathbb{P}(Y \leq y \mid \text{treatment}) &= \text{expit}(h(y) + \beta + \alpha) \end{aligned}$$

$H_0 : \beta = 0$ vs. log-odds ratio alternatives

Observe $(y, \mathbf{x})_i, i = 1, \ldots, N$ (independent etc)

Under H_0 (!!!), estimate cumulative distribution function

$$F_Y(y) = \mathbb{P}(Y \leq y)$$

from the whole sample

Example: Score Test for Comparing Two Groups

Maybe very simple by ECDF

$$\hat{F}_{Y,N}(y_i) = (N+1)^{-1} \sum_{j=1}^{N} \mathbb{1}(y_j \leq y_i) = (N+1)^{-1} R_i$$

where R_i is the rank of the ith response value in the whole sample

Then: $\hat{h}(y_i) = \text{logit}((N+1)^{-1} R_i)$

Example: Score Test for Comparing Two Groups

Plug-in $\hat{h}(y_i)$ and compute score wrt $\alpha \equiv 0$

$$S_{i,\alpha} = \left. \frac{\partial \ell_i(\hat{h}(y_i), \alpha)}{\partial \alpha} \right|_{\alpha=0} = 1 - 2R_i/(N+1)$$

Use "correlation" between score and treatment as test statistic:

$$\sum_{i=1}^{N} S_{i,\alpha} \mathbb{1}(\mathbf{x}_i = \text{treatment}) \cong \sum_{i=1}^{N} R_i \mathbb{1}(\mathbf{x}_i = \text{treatment}) = W$$

Example: Score Test for Comparing Two Groups

Plug-in $\hat{h}(y_i)$ and compute score wrt $\alpha \equiv 0$

$$S_{i,\alpha} = \left. \frac{\partial \ell_i(\hat{h}(y_i), \alpha)}{\partial \alpha} \right|_{\alpha=0} = 1 - 2R_i/(N+1)$$

Use "correlation" between score and treatment as test statistic:

$$\sum_{i=1}^{N} S_{i,\alpha} \mathbb{1}(\mathbf{x}_i = \text{treatment}) \cong \sum_{i=1}^{N} R_i \mathbb{1}(\mathbf{x}_i = \text{treatment}) = W$$

Oups: Wilcoxon-Mann-Whitney-Rank-Sum Test

Log-rank Test

Estimate h under the null $\beta = 0$ in model

$$\begin{aligned} \mathbb{P}(Y \leq y \mid \text{placebo}) &= \text{cloglog}^{-1}(h(y) + \alpha) \\ \mathbb{P}(Y \leq y \mid \text{treatment}) &= \text{cloglog}^{-1}(h(y) + \beta + \alpha) \end{aligned}$$

Use $h(y_i) = \log(-\log(1 - R_i/(N+1)))$ with ranks $R_1, \dots, R_N$

The derivative of the corresponding log-likelihood with respect to $\alpha \equiv 0$ is then

$$S_{i,\alpha} = 1 + \log(1 - R_i/(N+1))$$

Residuals in Machine Learning

In the remainder of this talk, I demonstrate that boosting, trees and forests can be understood as algorithms implementing the same simple idea:

1. Start with a simple model
2. Check if residuals can be explained by covariates
3. Add most important covariate to model
4. Iterate

This understanding helps us to apply these procedures to interesting models (outside the classical "regression and classification" framework).

L_2 boosting (original)

$$f = \arg\max_f \sum_{i=1}^{N} \ell_i(f(\mathbf{x}_i))$$

via functional gradient descent with negative gradient

$$u_i^{[m]} = \left.\frac{\partial \ell_i(f)}{\partial f}\right|_{f=\hat{f}^{[m]}(\mathbf{x}_i)}$$

and updates

$$f^{[m+1]}(\mathbf{x}_i) = f^{[m+1]}(\mathbf{x}_i) + \nu g^{[m]}(\mathbf{x}_i)$$

based on least-squares $g^{[m]} : u_i^{[m]} \sim \mathbf{x}_i$.

L_2 boosting (simplified but identical)

$$f = \arg\max_f \sum_{i=1}^{N} \ell_i(f(\mathbf{x}_i) + \alpha), \quad \alpha \equiv 0$$

via gradient descent with negative gradient

$$u_i^{[m]} = \left. \frac{\partial \ell_i(\hat{f}^{[m]}(\mathbf{x}_i) + \alpha)}{\partial \alpha} \right|_{\alpha=0}$$

and updates

$$\hat{f}^{[m+1]}(\mathbf{x}_i) = \hat{f}^{[m+1]}(\mathbf{x}_i) + \nu \hat{g}^{[m]}(\mathbf{x}_i)$$

based on least-squares $\hat{g}^{[m]} : u_i^{[m]} \sim \mathbf{x}_i$.

L_2 boosting (generalised)

$$(f, \boldsymbol{\vartheta}) = \operatorname*{arg\,max}_{f, \boldsymbol{\vartheta}} \sum_{i=1}^{N} \ell_i(\alpha + f(\mathbf{x}_i), \boldsymbol{\vartheta}), \quad \alpha \equiv 0$$

via gradient descent with negative gradient

$$\begin{aligned}
\hat{\boldsymbol{\vartheta}}^{[m]} &= \operatorname*{arg\,max}_{\boldsymbol{\vartheta}} \sum_{i=1}^{N} \ell_i(\alpha + \hat{f}^{[m-1]}(\mathbf{x}_i), \boldsymbol{\vartheta}) \\
u_i^{[m]} &= \left. \frac{\partial \ell_i(\alpha + \hat{f}^{[m]}(\mathbf{x}_i), \hat{\boldsymbol{\vartheta}}^{[m]})}{\partial \alpha} \right|_{\alpha=0}
\end{aligned}$$

and updates

$$\hat{f}^{[m+1]}(\mathbf{x}_i) = \hat{f}^{[m+1]}(\mathbf{x}_i) + \nu \hat{g}^{[m]}(\mathbf{x}_i)$$

based on least-squares $\hat{g}^{[m]} : u_i^{[m]} \sim \mathbf{x}_i$.

Example

$Y_i \in \{y_1, \dots, y_K\}$
proportional odds logistic regression

$$\mathbb{P}(Y_i \leq y_k \mid \mathbf{X} = \mathbf{x}_i) = \text{expit}(\vartheta_k + f(\mathbf{x}_i) + \alpha), \quad \alpha \equiv 0$$

where $\exp(f(\mathbf{x}_i))$ is odds ratio comparing the odds given $\mathbf{x}_i$ to the odds of $\mathbf{x}$ with $f(\mathbf{x}) = 0$.

The non-decreasing intercept thresholds $\boldsymbol{\vartheta} = (\vartheta_1, \dots, \vartheta_{K-1})^\top$ are "nuisance" parameters in the log-likelihood

$$\begin{aligned}\ell_i(\alpha + f(\mathbf{x}_i), \boldsymbol{\vartheta}) = \log(&\text{expit}(\vartheta_k + f(\mathbf{x}_i) + \alpha) - \\ &\text{expit}(\vartheta_{k-1} + f(\mathbf{x}_i) + \alpha))\end{aligned}$$

for $Y_i = y_k$ with $\vartheta_0 = -\infty$ and $\vartheta_K = \infty$.

Shift-transformation Boosting

$$\mathbb{P}(Y_i \le y \mid \mathbf{X} = \mathbf{x}_i) = F_Z(\mathbf{a}(y)^\top \boldsymbol{\vartheta} + f(\mathbf{x}_i) + \alpha), \quad \alpha \equiv 0$$

with log-likelihood contributions $\ell_i(\alpha + f(\mathbf{x}_i), \boldsymbol{\vartheta})$ and $h(y) = \mathbf{a}(y)^\top \boldsymbol{\vartheta}$.

This includes Cox or Weibull models, reverse time proportional hazards (for Lehmann alternatives), continuous outcome logistic (proportional odds) regression, Box-Cox type models etc. under all forms of random censoring and truncation for continuous and discrete (incl. count) data.

`stmboost()` in R add-on package **tbm** (from CRAN)

Boosting Partially Linear Models

$$\mathbb{P}(Y_i \leq y \mid \mathbf{X} = \mathbf{x}_i) = F_Z(\mathbf{a}(y)^\top \boldsymbol{\vartheta} + \mathbf{x}^\top \boldsymbol{\beta} + f(\mathbf{x}_i) + \alpha), \quad \alpha \equiv 0$$

where $\boldsymbol{\beta}$ is relative low-dimensional and shall be estimated without penalisation. Treat $\boldsymbol{\vartheta}$ and $\boldsymbol{\beta}$ as "nuisance" parameters.

Random Forests

Now understood as Adaptive Local Likelihood Estimators

$$\hat{\boldsymbol{\vartheta}}(\mathbf{x}) := \underset{\boldsymbol{\vartheta} \in \Theta}{\arg\max} \sum_{i=1}^{N} w_i(\mathbf{x}) \ell_i(\boldsymbol{\vartheta})$$

Conditioning works via weight functions $w_i(\mathbf{x})$ only. These weights come from trees.

Trees & Forests

Start with $\hat{\vartheta}$, compute $S_{i,\alpha}$ and try to find best cutpoint model of the form

$$\mathbb{E}(S_\alpha \mid \mathbf{X} = \mathbf{x}) = \mu + \beta I(x_p \le \xi)$$

This is a stump fitted to residuals by least squares.

Split into two groups wrt. $x_p \le \xi$ and proceed recursively.

Build many trees on subsamples and compute the weights $w_i(\mathbf{x})$ as the number of times $\mathbf{x}_i$ and $\mathbf{x}$ are element of the same terminal node (essentially).

More than Residuals

We did not look at all scores but only at the score wrt. an intercept term so far.

Now use all scores

$$\mathbf{S}_i = \left. \frac{\partial \ell_i(\boldsymbol{\vartheta})}{\partial \boldsymbol{\vartheta}} \right|_{\boldsymbol{\vartheta} = \hat{\boldsymbol{\vartheta}}}$$

simultaneously.

Regressing these scores was suggested for the assessment of parameter instability explained by covariates in the 1970ies.

mob()

This principle is employed in the MOB (model-based recursive partitioning)

This now also works with generalised residuals in transformation models.

`trafotree()` in R add-on package **trtf** (from CRAN)

Forests

Using generalised split criteria utilising all scores in such a tree makes the tree sensitive to changes in *all* parameters $\boldsymbol{\vartheta}$ (not just a mean or log-hazard ratio or ...).

We obtain a transformation model with predictive distribution

$$\hat{\mathbb{P}}(Y \le y \mid \mathbf{X} = \mathbf{x}) = F_Z(\mathbf{a}(y)^\top \hat{\boldsymbol{\vartheta}}(\mathbf{x}))$$

and also a likelihood and thus proper scoring rule for evaluating the forest. Alterative to quantile regression forests.

`traforest()` in R add-on package **trtf** (from CRAN)

mob() & model4you & grf

Other important application is the estimation of additive models using trees and forests.

For example, estimation of heterogenuous treatment effects

$$(Y \mid \mathbf{X} = \mathbf{x}) = \mu(\mathbf{x}) + \beta(\mathbf{x}) I(\text{treated}) + \sigma Z$$

model4you uses mob to estimate $\mu(\mathbf{x})$ and $\beta(\mathbf{x})$ simultaneously (for linear or other models).

grf employs a similar principle sequentially (for linear models).

Boosting Conditional Transformation Models

$$\boldsymbol{\vartheta} = \arg\max_{\boldsymbol{\vartheta}} \sum_{i=1}^{N} \ell_i(\boldsymbol{\vartheta}(\mathbf{x}_i))$$

via functional gradient descent with negative gradient

$$\mathbf{u}_i^{[m]} = \left. \frac{\partial \ell_i(\boldsymbol{\vartheta})}{\partial \boldsymbol{\vartheta}} \right|_{\boldsymbol{\vartheta} = \hat{\boldsymbol{\vartheta}}^{[m]}(\mathbf{x}_i)}$$

and updates

$$\hat{\boldsymbol{\vartheta}}^{[m+1]}(\mathbf{x}_i) = \hat{\boldsymbol{\vartheta}}^{[m+1]}(\mathbf{x}_i) + \nu \hat{g}^{[m]}(\mathbf{x}_i)$$

based on multivariate least-squares $\hat{g}^{[m]} : \mathbf{u}_i^{[m]} \sim \mathbf{x}_i$.

`ctmboost()` in R add-on package **tbm**

Boosting Body Fat

Linear model

$$\mathbb{P}(Y_i \leq y \mid \mathbf{X} = \mathbf{x}_i) \quad = \quad \Phi(\vartheta_1 + \vartheta_2 y + f(\mathbf{x}_i))$$

Box-Cox type model

$$\mathbb{P}(Y_i \leq y \mid \mathbf{X} = \mathbf{x}_i) \quad = \quad \Phi(\mathbf{a}(y)^\top \boldsymbol{\vartheta} + f(\mathbf{x}_i))$$

Conditional transformation model

$$\mathbb{P}(Y_i \leq y \mid \mathbf{X} = \mathbf{x}_i) \quad = \quad \Phi(\mathbf{a}(y)^\top \boldsymbol{\vartheta}(\mathbf{x}_i))$$

Summary

- Boosting, trees, and forests can be understood as algorithms leveraging the information contained in residuals or scores for increasing model complexity.
- *You* start with an *appropriate* model featuring *interpretable* parameters.
- *Your* model defines the log-likelihood and scores.
- Simple least-squares fitting is used to explain score variability by covariates and *automagically* estimates a more complex model.
- This principle is universally applicable (on paper and in silico).
- Transformation models are a convenient starting point.

http://ctm.R-forge.R-project.org

Resources

- "Transformation Boosting Machines", STCO, **tbm**, http://dx.doi.org/10.1007/s11222-019-09870-4
- "(Survival) Transformation Forests", **trtf**, https://arxiv.org/abs/1701.02110, SMMR http://dx.doi.org/10.1177/0962280219862586
- "Most Likely Transformations", SJoS, **mlt**, **tram**, http://dx.doi.org/10.1111/sjos.12291
- "Conditional Transformation Models", JRSS-B, http://dx.doi.org/10.1111/rssb.12017
- "Model-based Recursive Partitioning", JCGS, **partykit** http://dx.doi.org/10.1198/106186008X319331,
- "Model-based Recursive Partitioning for Subgroup Analyses", IJB, **model4you** http://dx.doi.org/10.1515/ijb-2015-0032
- "Model-based Forests", SMMR, **model4you**, http://dx.doi.org/10.1177/0962280217693034, AOAS http://dx.doi.org/10.1214/19-AOAS1247