Model-based boosting in \texttt{R}

Build your own family

Andreas Mayr

andreas.mayr@imbe.med.uni-erlangen.de

Institut für Medizininformatik, Biometrie und Epidemiologie (IMBE)
Friedrich-Alexander-Universität Erlangen-Nürnberg

Statistical Computing 2011
Build your own family

Family(ngradient, loss = NULL, risk = NULL,
offset = function(y, w)
  optimize(risk, interval = range(y), y=y, w=w)$minimum, ...)
Example: quantile regression
Quantile regression

What’s our loss?

- Quantile regression, in comparison to standard regression, fits quantiles rather than the expected mean of the conditional distribution function
  \[ y_i = \hat{f}_{\tau i} + \varepsilon_{\tau i} \]

- The appropriate loss function for the \( \tau \)-quantile is the check-function \( \rho_\tau(\cdot) \):
  \[
  \rho_\tau(y_i - \hat{f}_{\tau i}) = \begin{cases} 
    (y_i - \hat{f}_{\tau i}) \cdot \tau & (y_i - \hat{f}_{\tau i}) \geq 0 \\
    (y_i - \hat{f}_{\tau i}) \cdot (\tau - 1) & (y_i - \hat{f}_{\tau i}) < 0.
  \end{cases}
  \]

\[
\text{loss} = \text{function}(y, f) \ \text{tau} \times (y - f) \times ((y - f) \geq 0) + \\
(\text{tau} - 1) \times (y - f) \times ((y - f) < 0)
\]
Quantile regression

What’s our loss?

Loss function for standard regression (left) and the check function for quantile regression (right) for different values of $\tau$. 
Quantile regression
What’s the gradient?

- The check-function is not differentiable at the point 0. But in practice, as the response is continuous, we can ignore this by defining:

\[
\frac{\partial \rho_\tau(y_i, f_{\tau i})}{\partial f} = \begin{cases} 
\tau & (y_i - f_{\tau i}) \geq 0 \\
\tau - 1 & (y_i - f_{\tau i}) < 0
\end{cases}
\]

\[
\text{ngradient} = \text{function}(y, f, w = 1) \\
\text{tau} * ((y - f) \geq 0) + (\text{tau} - 1) * ((y - f) < 0)
\]
Quantile regression

Construct our own family

\[
R> \text{OurQuantReg} <- \text{function}(\text{tau} = 0.5)\{ \\
+ \hspace{1em} \text{Family(} \\
+ \hspace{2em} \text{ngradient} = \text{function}(y,f,w = 1) \\
+ \hspace{3em} \text{tau} * ((y - f) >=0) + (\text{tau} - 1) * ((y - f) < 0), \\
+ \hspace{2em} \text{loss} = \text{function}(y, f) \text{tau} * (y - f) * ((y - f) >= 0) + \\
+ \hspace{3em} (\text{tau} - 1) * (y - f) * ((y - f) < 0), \\
+ \hspace{2em} \text{offset} = \text{function}(y, w = 1) \text{median}(y), \\
+ \hspace{2em} \text{name} = "Our new family for quantile regression" \\
+ \hspace{1em} \}) \\
R> \text{OurQuantReg()} \\
\]

Our new family for quantile regression

Loss function: \(\text{tau} * (y - f) * ((y - f) >= 0) + (\text{tau} - 1) * (y - f) * ((y - f) < 0)\)
Let’s try how it works
Our new family for quantile regression

Let's try how it works

```r
R> set.seed(0804)
R> x <- runif(100)
R> y <- 2 + 3*x + x*rnorm(100)
R> plot(x,y)
```
Our new family for quantile regression

Let's try how it works

```r
R> ctrl <- boost_control(mstop=500)
R> gb1 <- glmboost(y~x, family=OurQuantReg(tau=0.1), control=ctrl)
R> gb2 <- glmboost(y~x, family=OurQuantReg(tau=0.9), control=ctrl)
R> lines(x, fitted(gb1), col=2)
R> lines(x, fitted(gb2), col=2)
```
Our new family for quantile regression

Let’s try how it works

Compare to \texttt{rq()} of Koenker’s \texttt{quantreg} package, which can be seen as the gold standard for low-dimensional quantile regression, based on linear programming:

\begin{verbatim}
R> library(quantreg)
R> rq1 <- rq(y~x, tau=0.1)
R> rq2 <- rq(y~x, tau=0.9)
R> rbind(coef(rq1), coef(rq2))

(Intercept) x
[1,] 1.998975 1.740673
[2,] 2.009817 4.111221

R> rbind(coef(gb1, off2int=TRUE), coef(gb2, off2int=TRUE))

(Intercept) x
[1,] 1.998600 1.740263
[2,] 2.010295 4.111727
\end{verbatim}
Our new family for quantile regression

Let's try how it works

Compare to the true coefficients:

\[ R> \text{abline}(a=2, b= 3 + \text{qnorm}(0.1), \text{col}=4) \]
\[ R> \text{abline}(a=2, b= 3 + \text{qnorm}(0.9), \text{col}=4) \]
For more on quantile regression

Family \texttt{QuantReg()} is implemented in \texttt{mboost} (Fenske et al., 2011).