LogitBoost with Trees Applied to the WCCI 2006 Performance Prediction Challenge Datasets

Roman Lutz

Seminar für Statistik, ETH Zürich, Switzerland

18. July 2006

Overview

1. Motivation
2. LogitBoost with trees
   - The logistic framework
   - The LogitBoost algorithm
   - The learner
   - Shrinkage
   - Remarks and extensions
3. Results
4. Summary

Why LogitBoost with trees?

Why boosting?

- Successful in various real-world applications.
- Topic of my PhD Thesis.

Why LogitBoost?

- "Statistical version of boosting".

Why trees?

- Easy to control the degree of interaction.
- No transformation of variables necessary.

Why LogitBoost with trees?

Why boosting?

- Successful in various real-world applications.
- Topic of my PhD Thesis.

Why LogitBoost?

- “Statistical version of boosting”.

Why trees?

- Easy to control the degree of interaction.
- No transformation of variables necessary.

Why LogitBoost with trees?

Why boosting?

- Successful in various real-world applications.
- Topic of my PhD Thesis.

Why LogitBoost?

- “Statistical version of boosting”.

Why trees?

- Easy to control the degree of interaction.
- No transformation of variables necessary.

Why LogitBoost with trees?

Why boosting?

- Successful in various real-world applications.
- Topic of my PhD Thesis.

Why LogitBoost?

- “Statistical version of boosting”.

Why trees?

- Easy to control the degree of interaction.
- No transformation of variables necessary.

Why LogitBoost with trees?

Why boosting?

- Successful in various real-world applications.
- Topic of my PhD Thesis.

Why LogitBoost?

- "Statistical version of boosting".

Why trees?

- Easy to control the degree of interaction.
- No transformation of variables necessary.

Why LogitBoost with trees?

Why boosting?

- Successful in various real-world applications.
- Topic of my PhD Thesis.

Why LogitBoost?

- “Statistical version of boosting”.

Why trees?

- Easy to control the degree of interaction.
- No transformation of variables necessary.

Why LogitBoost with trees?

Why boosting?

- Successful in various real-world applications.
- Topic of my PhD Thesis.

Why LogitBoost?

- “Statistical version of boosting”.

Why trees?

- Easy to control the degree of interaction.
- No transformation of variables necessary.

Why LogitBoost with trees?

Why boosting?

- Successful in various real-world applications.
- Topic of my PhD Thesis.

Why LogitBoost?

- “Statistical version of boosting”.

Why trees?

- Easy to control the degree of interaction.
- No transformation of variables necessary.

Terminology

Training data: $(x_1, y_1), \ldots, (x_n, y_n), \quad x_i \in \mathbb{R}^p, \quad y_i \in \{0, 1\}$.

Logistic framework:

- Conditional probabilities: $p(x) = P[Y = 1 | X = x]$.
- "Predictor": $F : \mathbb{R}^p \to \mathbb{R}$.
- Link: $p(x) = \frac{\exp(F(x))}{1+\exp(F(x))}$ and $F(x) = \log\left(\frac{p(x)}{1-p(x)}\right)$.

Classification rule:

- Classify a new observation as $+1$ if $p(x_{new}) > \text{cut-off}$.
- The cut-off is the proportion of class $+1$ in the data (because the balanced error rate (BER) is used).

The LogitBoost algorithm in words

LogitBoost (Friedman, Hastie, Tibshirani (2000)) uses Newton steps for fitting a logistic model by maximum binomial likelihood.

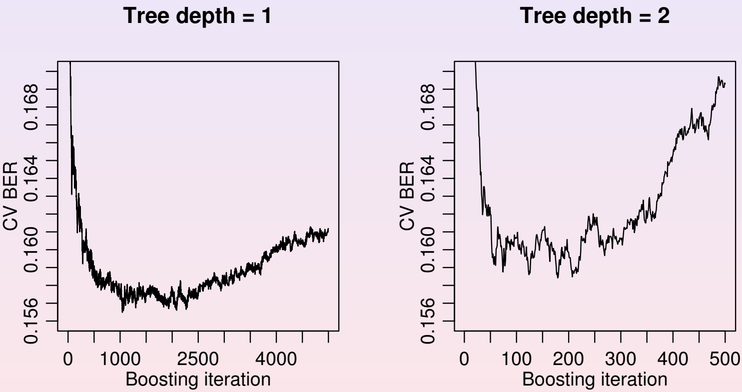
The LogitBoost algorithm in code

1. Start with $F^{(0)}(x_i) = 0$ and $p(x_i) = \frac{1}{2}$, $i = 1, \ldots, n$.
2. Repeat for $m = 1, \ldots, M$:
    1. Compute the weights and working response
    $$w_i = p(x_i)(1 - p(x_i)), \quad z_i = \frac{y_i - p(x_i)}{p(x_i)(1 - p(x_i))}.$$
    2. Fit the function $f^{(m)}(x)$, using the tree-based learner, by a weighted least-squares regression of z_i to x_i using weights w_i.
    3. Update $F^{(m)}(x_i) = F^{(m-1)}(x_i) + \nu f^{(m)}(x_i), \quad 0 < \nu \leq 1$ and $p(x_i) = \frac{\exp(F^{(m)}(x_i))}{1+\exp(F^{(m)}(x_i))}$.

The learner

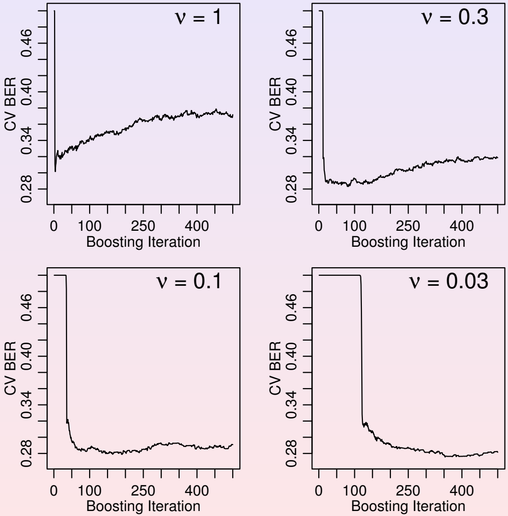
The learner (fitting method) is a regression tree of prefixed depth: 1, 2, 3, 4, or 5 (in each iteration, a tree of the same depth is fitted).

The tree depth and the number of iterations M are chosen by 10-fold cross-validation (CV) to minimize the balanced error rate (BER).


Tree depth = 1
CV BER
0.156
0.160
0.164
0.168
0
1000
2500
4000
Boosting iteration
Tree depth = 2
CV BER
0.156
0.160
0.164
0.168
0
100
200
300
400
500
Boosting iteration

Shrinkage

- The ν is the so-called shrinkage factor.
- The natural value is 1, but smaller values are often a better choice.
- Start with $\nu = 1$. If the CV BER curve is too rough, reduce ν by a factor of approximately 3 and rerun LogitBoost.


ν = 1
ν = 0.3
ν = 0.1
ν = 0.03
CV BER
0.28
0.34
0.40
0.46
0
100
250
400
Boosting Iteration

Remarks and extensions

PCA for Nova. First 400 principal components are taken for LogitBoost.

Further modifications (challenge submissions 2 - 4):

- Variable pre-selection by Wilcoxon/Fisher exact test. Variables with a p-value above 0.1 are dropped.
- Predicted probabilities of LogitBoost with and without variable pre-selection averaged.
- Intercept adaptation: Add the same constant to all $F(x_i)$ so that the average of the resulting probabilities $p(x_i)$ is exactly the proportion of class $+1$ in the data.

BER guess

The BER guess is the CV BER at the stopping iteration.

- Additional computation: 0.
- Is too optimistic, because the number of iterations is explicitly chosen to minimize the CV BER.
- But the estimation of generalisation BER by CV is biased upward.
- $\rightarrow$ The two effects could cancel each other out.

Results

Plain LogitBoost (not the winning submission):

Dataset	Tree depth	ν	No. of iterations	CV BER = BER guess	BER on test set
Ada	1	0.3	1043	0.1565	0.1712
Gina	5	0.3	741	0.0415	0.0385
Hiva	2	0.03	353	0.2756	0.2888
Nova	2	0.1	294	0.0506	0.0491
Sylva	1	0.3	273	0.0058	0.0064
Average				0.1060	0.1108

Summary

- LogitBoost with trees is very competitive.
- Use only large trees if really necessary.
- Use shrinkage (e.g. $\nu = 0.1$).