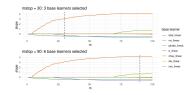
Introduction to Machine Learning

Boosting Gradient Boosting: CWB Basics 1





Learning goals

- Concept of CWB
- Which base learners do we use
- Built-in feature selection

COMPONENTWISE GRADIENT BOOSTING

GB (with trees), has strong predictive performance but is difficult to interpret unless the base learners are stumps.

The aim of CWB is to find a model that exhibits:

- strong predictive performance,
- interpretable components,
- automatic selection of components,
- is sparser than a model fitted with maximum-likelihood estimation.

This is achieved by using "nice" base learners which yield familiar statistical models in the end.

Because of this, CWB is also often referred to as **model-based boosting**.



BASE LEARNERS

In GB only one kind of base learner $\ensuremath{\mathcal{B}}$ is used, e.g., regression trees.

For CWB we generalize this to multiple base learner sets $\{\mathcal{B}_1,...\mathcal{B}_J\}$ with associated parameter spaces $\{\Theta_1,...\Theta_J\}$, where $j\in\{1,2,...,J\}$ indexes the type of base learner.

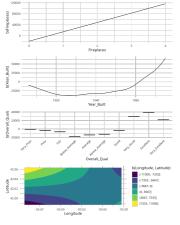




In each iteration, base learners are fitted to the **pseudo residuals** $\tilde{r}^{[m]}$.

BASE LEARNERS / 2

We restrict the base learners to additive model components, i.e.,



linear effect

non-linear (spline) effect

dummy encoded linear model of a cat, feature

tensor product spline for interaction modelling (e.g. spatial effects)

More advanced base learners could also be Markov random fields, random effects, or trees.



BASE LEARNERS / 3

0.0 2.5 5.0 7.5 10.0

х

 $b_j(x|\theta^{[1]})$

Two BLs of the same type can simply be added by adding up their parameter vectors:

$$b_{j}(\mathbf{x}, \boldsymbol{\theta}^{[1]}) + b_{j}(\mathbf{x}, \boldsymbol{\theta}^{[2]}) = b_{j}(\mathbf{x}, \boldsymbol{\theta}^{[1]} + \boldsymbol{\theta}^{[2]}).$$

$$+ \frac{\widehat{\mathbb{E}}_{\mathbf{x}}^{\frac{3}{2}}}{\widehat{\mathbb{E}}_{\mathbf{x}}^{\frac{1}{2}}} = \mathbf{E}_{\mathbf{y}}^{\mathbf{x}} + \mathbf{E}_{\mathbf{x}}^{\mathbf{x}} = \mathbf{E}_{\mathbf{x}}^{\mathbf{x}} + \mathbf{E}_{\mathbf{x}}^{\mathbf{x}} = \mathbf{E}_{\mathbf{x}}^{\mathbf{x}}$$

0.0 2.5 5.0 7.5 10.0

Thus, if
$$\{b_j(\mathbf{x}, \boldsymbol{\theta}^{[1]}), b_j(\mathbf{x}, \boldsymbol{\theta}^{[2]})\} \in \mathcal{B}_j$$
, then $b_j(\mathbf{x}, \boldsymbol{\theta}^{[1]} + \boldsymbol{\theta}^{[2]}) \in \mathcal{B}_j$.



0.0 2.5 5.0 7.5 10.0

Different from GB, multiple base learners $b_i \in \mathcal{B}_i$, $j = 1, \ldots, J$, are fitted and only best-fitting one is selected and updated.

Algorithm Componentwise Gradient Boosting.

1: Initialize
$$f^{[0]}(\mathbf{x}) = \arg\min_{\theta_0 \in \mathbb{R}} \sum_{i=1}^n L(y^{(i)}, \theta_0)$$

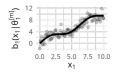
2: **for** $m = 1 \to M$ **do**
3: For all i : $\tilde{r}^{[m](i)} = -\left[\frac{\partial L(y, f)}{\partial f}\right]_{f=f^{[m-1]}(\mathbf{x}^{(i)}), y=y^{(i)}}$
4: **for** $j = 1 \to J$ **do**
5: Fit regression base learner $b_j \in \mathcal{B}_j$ to the vector of pseudo-residuals $\tilde{r}^{[m]}$:
6: $\hat{\theta}_j^{[m]} = \arg\min_{\theta \in \Theta_j} \sum_{i=1}^n (\tilde{r}^{[m](i)} - b_j(\mathbf{x}^{(i)}, \theta))^2$
7: **end for**
8: $j^{[m]} = \arg\min_j \sum_{i=1}^n (\tilde{r}^{[m](i)} - \hat{b}_j(\mathbf{x}^{(i)}, \hat{\theta}_j^{[m]}))^2$
9: Update $f^{[m]}(\mathbf{x}) = f^{[m-1]}(\mathbf{x}) + \alpha \hat{b}_j(\mathbf{x}, \hat{\theta}_j^{[m]})$
10: **end for**
11: Output $\hat{f}(\mathbf{x}) = f^{[M]}(\mathbf{x})$

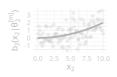


Algorithm Componentwise Gradient Boosting (inner loop).

- 4: for $j = 1 \rightarrow J$ do
- 5: Fit regression base learner $b_j \in \mathcal{B}_j$ to the vector of pseudo-residuals $\tilde{r}^{[m]}$:
- 6: $\hat{\boldsymbol{\theta}}_{j}^{[m]} = \arg\min_{\boldsymbol{\theta} \in \boldsymbol{\Theta}_{j}} \sum_{i=1}^{n} (\tilde{\boldsymbol{\tau}}^{[m](i)} b_{j}(\mathbf{x}^{(i)}, \boldsymbol{\theta}))^{2}$
- 7: end for
- 8: $j^{[m]} = \arg\min_{j} \sum_{i=1}^{n} (\tilde{r}^{[m](i)} \hat{b}_{j}(\mathbf{x}^{(i)}, \hat{\theta}_{j}^{[m]}))^{2}$

Iteration
$$m, j = 1, \sum_{i=1}^{n} (\tilde{r}^{[m](i)} - \hat{b}_1(x_1^{(i)}, \hat{\theta}_1^{[m]}))^2 = 24.4$$
:





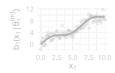


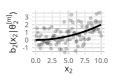


Algorithm Componentwise Gradient Boosting (inner loop).

- 4: for $j = 1 \rightarrow J$ do
- 5: Fit regression base learner $b_j \in \mathcal{B}_j$ to the vector of pseudo-residuals $\tilde{r}^{[m]}$:
- 6: $\hat{\boldsymbol{\theta}}_{j}^{[m]} = \arg\min_{\boldsymbol{\theta} \in \boldsymbol{\Theta}_{j}} \sum_{i=1}^{n} (\tilde{\boldsymbol{\tau}}^{[m](i)} b_{j}(\mathbf{x}^{(i)}, \boldsymbol{\theta}))^{2}$
- 7: end for
- 8: $j^{[m]} = \arg\min_{j} \sum_{i=1}^{n} (\tilde{r}^{[m](i)} \hat{b}_{j}(\mathbf{x}^{(i)}, \hat{\boldsymbol{\theta}}_{j}^{[m]}))^{2}$

Iteration
$$m, j = 2, \sum_{i=1}^{n} (\tilde{r}^{[m](i)} - \hat{b}_2(x_2^{(i)}, \hat{\theta}_2^{[m]}))^2 = 43.2$$
:





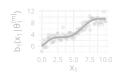


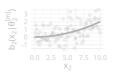


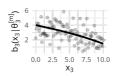
Algorithm Componentwise Gradient Boosting (inner loop).

- 4: for $j = 1 \rightarrow J$ do
- 5: Fit regression base learner $b_j \in \mathcal{B}_j$ to the vector of pseudo-residuals $\tilde{r}^{[m]}$:
- 6: $\hat{\boldsymbol{\theta}}_{j}^{[m]} = \arg\min_{\boldsymbol{\theta} \in \boldsymbol{\Theta}_{j}} \sum_{i=1}^{n} (\tilde{r}^{[m](i)} b_{j}(\mathbf{x}^{(i)}, \boldsymbol{\theta}))^{2}$
- 7: end for
- 8: $j^{[m]} = \arg\min_{j} \sum_{i=1}^{n} (\tilde{r}^{[m](i)} \hat{b}_{j}(\mathbf{x}^{(i)}, \hat{\boldsymbol{\theta}}_{j}^{[m]}))^{2}$

Iteration
$$m, j = 3, \sum_{i=1}^{n} (\tilde{r}^{[m](i)} - \hat{b}_3(x_3^{(i)}, \hat{\theta}_3^{[m]}))^2 = 35.2$$
:





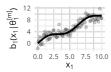


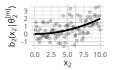


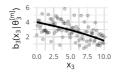
Algorithm Componentwise Gradient Boosting (inner loop).

- 4: for $j = 1 \rightarrow J$ do
- 5: Fit regression base learner $b_j \in \mathcal{B}_j$ to the vector of pseudo-residuals $\tilde{r}^{[m]}$:
- 6: $\hat{\boldsymbol{\theta}}_{j}^{[m]} = \arg\min_{\boldsymbol{\theta} \in \boldsymbol{\Theta}_{j}} \sum_{i=1}^{n} (\tilde{r}^{[m](i)} b_{j}(\mathbf{x}^{(i)}, \boldsymbol{\theta}))^{2}$
- 7: end for
- 8: $j^{[m]} = \operatorname{arg\,min}_j \sum_{i=1}^n (\tilde{r}^{[m](i)} \hat{b}_j(\mathbf{x}^{(i)}, \hat{\boldsymbol{\theta}}_j^{[m]}))^2$

Iteration $m: \Rightarrow i^{[m]} = 1$









FEATURE SELECTION IN CWB

In CWB, we often define BLs on a single feature

$$b_j(x_j,\theta)$$
 for $j=1,2,\ldots,p$.

Allows natural form of feature selection:

- When we select the best BL in one iter of CWB, we thereby also only select one (associated) feature
- Note that a feature (or rather a BL associated with it) can be selected in multiple iters, so ≤ M features are selected

