15.095: Machine Learning under a Modern Optimization Lens

Lecture 10: Boosting

Overview

Broadly speaking, **boosting** is the idea of combining many weak machine learning models to create a single strong model.

We will focus on the problems of regression and classification.

Two examples of weak models:

1. One variable linear regression models
2. Decision trees with a single variable split ("stumps")

Boosting Setup

Data $(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_n, y_n)$, where $\mathbf{x}_i \in \mathbb{R}^p$ and y_i is in $\mathbb{R}$ or $\{0, 1\}$.

We are interested in decision rules of the form

$$\hat{y}^{(t)} = f(\mathbf{x}) := f_1(\mathbf{x}) + f_2(\mathbf{x}) + \cdots + f_t(\mathbf{x}),$$

where each $f_j \in \mathcal{F}$ is a "simple" function (e.g. $f(\mathbf{x}) = \mathbf{1}_{x_k < a}$).

Boosting Setup

$$f(\mathbf{x}) = f_1(\mathbf{x}) + f_2(\mathbf{x}) + \cdots + f_t(\mathbf{x}), \quad f_j \in \mathcal{F}$$

Instead of simultaneously optimizing over $f_1, \ldots, f_t$ to minimize

$$\min_{f_1,\ldots,f_t \in \mathcal{F}} \sum_{i=1}^{n} \ell\left(y_i, f_1(\mathbf{x}_i) + \cdots + f_t(\mathbf{x}_i)\right),$$

boosting typically proceeds in a *stage-wise* fashion:

$$f_1 = \operatorname*{argmin}_{f \in \mathcal{F}} \sum_{i=1}^{n} \ell\left(y_i, f(\mathbf{x}_i)\right),$$

$$f_t = \operatorname*{argmin}_{f \in \mathcal{F}} \sum_{i=1}^{n} \ell\left(y_i, f_1(\mathbf{x}_i) + \cdots + f_{t-1}(\mathbf{x}_i) + f(\mathbf{x}_i)\right), \quad t \geq 2.$$

Stagewise Linear Regression

Consider linear regression with

- ℓ as the square loss
- $\mathcal{F} = \{\boldsymbol{\beta} \in \mathbb{R}^p : \|\boldsymbol{\beta}\|_0 = 1\}$—linear models with a single nonzero coefficient

Initialization:

$$\begin{aligned} f_1 &\in \operatorname*{argmin}_{f \in \mathcal{F}} \sum_i \ell(y_i, f(\mathbf{x}_i)) = \sum_i (y_i - f(\mathbf{x}_i))^2 \\ &= \operatorname*{argmin}_{\substack{\beta \in \mathbb{R}, \\ j \in \{1,\dots,p\}}} \sum_i (y_i - \beta x_{ij})^2 . \end{aligned}$$

f_1 is best linear regression model with a single nonzero coefficient.

Stagewise Linear Regression—Update Step

Given $f_1, \ldots, f_{t-1} \in \mathcal{F}$, find

$$f_t \in \underset{f \in \mathcal{F}}{\operatorname{argmin}} \sum_i (\underbrace{y_i - f_1(\mathbf{x}_i) - \cdots - f_{t-1}(\mathbf{x}_i)}_{\text{residual component}} - f(\mathbf{x}_i))^2.$$

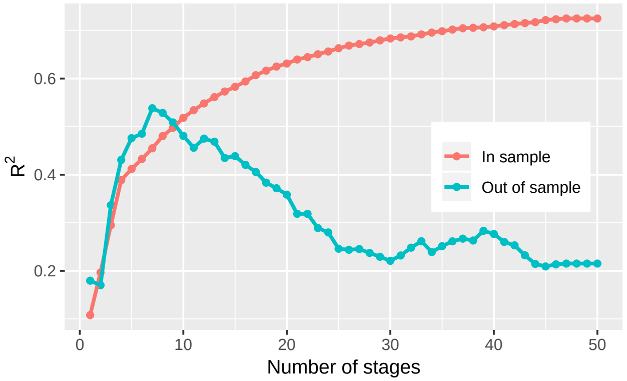
In other words, f_t is found by solving a regression problem on the residual component, i.e., the data

$$(\mathbf{x}_i, y_i - f_1(\mathbf{x}_i) - \cdots - f_{t-1}(\mathbf{x}_i)), \quad i = 1, \ldots, n.$$

Consistent with general intuition for boosting: add another weak model that explains that which is not already captured by the included models.

Stagewise Linear Regression—Example

Numerical example:



There has been recent work showing that this type of boosting is actually subgradient descent applied to the dual of the Lasso problem.

An Abstract View

$$\min_{f \in \mathcal{F}} \sum_i \ell(y_i, \hat{y}_i + f(\mathbf{x}_i)) \; + \; R(f),$$

where $\hat{y}_i = f_1(\mathbf{x}_i) + \cdots + f_{t-1}(\mathbf{x}_i)$ and R is some *regularizer*.

Many choices of ℓ:

- e.g. classification loss

Many choices of $\mathcal{F}$:

- e.g. trees with a fixed depth or number of leaves

Many choices of R:

- e.g. $R(\text{tree}) = \gamma \cdot (\#\text{ leaves}) + \lambda \sum_j (\text{weight on leaf } j)^2/2$

In all cases, there is the general question of how to (approximately) solve the optimization problem.

Second Order Approximation

$$\min_{f \in \mathcal{F}} \sum_i \ell(y_i, \hat{y}_i + f(\mathbf{x}_i)) \; + \; R(f)$$

Taylor approximation:

$$\ell(y_i, \hat{y}_i + f(\mathbf{x}_i)) \approx \ell(y_i, \hat{y}_i) + f(\mathbf{x}_i) \frac{\partial \ell(y_i, \hat{y}_i)}{\partial \hat{y}_i} + f^2(\mathbf{x}_i) \, \frac{\partial^2 \ell(y_i, \hat{y}_i)}{\partial \hat{y}_i^2} \Big/ 2.$$

This leads to the new optimization criterion

$$\min_{f \in \mathcal{F}} \sum_i \left(g_i f(\mathbf{x}_i) + h_i f^2(\mathbf{x}_i)/2 \right) \; + \; R(f),$$

benefits of writing in this way?

where $g_i = \frac{\partial \ell(y_i, \hat{y}_i)}{\partial \hat{y}_i}$ (**g**radient) and $h_i = \frac{\partial^2 \ell(y_i, \hat{y}_i)}{\partial \hat{y}_i^2}$ (**h**essian).

Sanity Check

$$\min_{f \in \mathcal{F}} \sum_i \ell(y_i, \hat{y}_i + f(\mathbf{x}_i)) \ + \ R(f)$$

If we are doing regression with the usual ℓ_2 loss, then regardless of $\mathcal{F}$, the problem

$$\min_{f \in \mathcal{F}} \sum_i \left(g_i f(\mathbf{x}_i) + h_i f^2(\mathbf{x}_i)\right) \ + \ R(f)$$

always coincides with the original approach:

$$\begin{aligned}
\ell(y_i, \hat{y}_i + f(\mathbf{x}_i)) &= (y_i - \hat{y}_i - f(\mathbf{x}_i))^2 \\
&= (y_i - \hat{y}_i)^2 \underbrace{-2(y_i - \hat{y}_i)}_{g_i} f(\mathbf{x}_i) + \underbrace{2}_{h_i} \cdot f^2(\mathbf{x}_i)/2 \\
&= \ell(y_i, \hat{y}_i) + g_i f(\mathbf{x}_i) + h_i f^2(\mathbf{x}_i)/2.
\end{aligned}$$

Boosted Trees

How to solve

$$\min_{f \in \mathcal{F}} \sum_i \left(g_i f(\mathbf{x}_i) + h_i f^2(\mathbf{x}_i)/2\right) \; + \; R(f)$$

for more complicated choices of $\mathcal{F}$?

e.g. $\mathcal{F} = \{\text{trees with at most } T \text{ leaves}\}$

The optimization problem is typically solved using heuristics.

Boosted Trees

$$\min_{f \in \mathcal{F}} \sum_i \left(g_i f(\mathbf{x}_i) + h_i f^2(\mathbf{x}_i)/2 \right) \; + \; R(f)$$

where $\mathcal{F} = \{$(axis-parallel-split) trees with at most T leaves$\}$ and

$$R(\text{tree}) = \gamma \cdot (\#\text{ leaves}) + \lambda \sum_j (\text{weight on leaf } j)^2/2$$

Begin by parametrizing $\mathcal{F}$: any $f \in \mathcal{F}$ can be expressed as

$$f(\mathbf{x}_i) = \sum_{j \in J} w_j \mathbf{1}_{i \in L_j},$$

for some partition of $\{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ into disjoint leaves with $L_j = \{i : \mathbf{x}_i \text{ falls in leaf } j\}$. (And $|J| \leq T$.)

Boosted Trees

With this parametrization, the objective function becomes

$$\begin{aligned}\sum_i \left[g_i f(\mathbf{x}_i) + \frac{h_i}{2} f^2(\mathbf{x}_i)\right] + R(f) &= \sum_{i,j} g_i w_j \mathbf{1}_{i\in L_j} + \sum_i \frac{h_i}{2}\left[\sum_j w_j \mathbf{1}_{i\in L_j}\right]^2 + R(f)\\ &= \sum_{i,j}\left(g_i w_j + h_i w_j^2/2\right)\mathbf{1}_{i\in L_j} + \gamma|J| + \lambda\sum_j w_j^2/2\\ &= \sum_j \sum_{i\in L_j}\left(g_i w_j + (h_i+\lambda) w_j^2/2\right) + \gamma|J|.\end{aligned}$$

Therefore, the optimization problem can be written as

$$\min_{\substack{\mathbf{w}\in\mathbb{R}^J,\\ \{L_j\}_{j\in J}}} \sum_j \sum_{i\in L_j}\left(g_i w_j + (h_i+\lambda) w_j^2/2\right) + \gamma|J|,$$

taken over all partitions of $\{\mathbf{x}_1,\ldots,\mathbf{x}_n\}$ with $|J| \le T$.

Boosted Trees

Can eliminate weight variable (given leaves):

$$\begin{aligned}\min_{\mathbf{w}\in\mathbb{R}^J}\sum_j\sum_{i\in L_j}(g_i w_j + (h_i+\lambda)w_j^2/2 + \gamma|J| \\ &= \sum_j \min_{w\in\mathbb{R}}\left(\sum_{i\in L_j} g_i\right) w + \left(\sum_{i\in L_j} h_i + \lambda\right) w^2/2 + \gamma|J| \\ &= -\sum_j \frac{G_j^2}{2(H_j+\lambda)} + \gamma|J|,\end{aligned}$$

where $G_j = \sum_{i\in L_j} g_i$ and $H_j = \sum_{i\in L_j} h_i$. Therefore, optimization problem can be written as

$$\min_{\{L_j\}_{j\in J}} -\sum_j \frac{G_j^2}{2(H_j+\lambda)} + \gamma|J|$$

subject to $G_j = \sum_{i\in L_j} g_i$ and $H_j = \sum_{i\in L_j} h_i$.

Boosted Trees Heuristic

How does this help us? Recall that trees are typically found in a top-down, greedy fashion.

Given some partition $\{L_j\}_{j \in J}$, when is it worthwhile (in terms of objective) to split L_1, say, into two disjoint sets?

$$\begin{aligned}\Delta(\mathrm{obj}, L_1 \mapsto L_{1'} \cup L_{1''}) &= -\sum_{j \neq 1} \frac{G_j^2}{2(H_j + \lambda)} - \frac{G_{1'}^2}{2(H_{1'} + \lambda)} - \frac{G_{1''}^2}{2(H_{1''} + \lambda)} \\ &\quad + \gamma(|J| + 1) + \sum_j \frac{G_j^2}{2(H_j + \lambda)} - \gamma|J| \\ &= \gamma + \frac{(G_{1'} + G_{1''})^2}{2(H_{1'} + H_{1''} + \lambda)} - \frac{G_{1'}^2}{2(H_{1'} + \lambda)} - \frac{G_{1''}^2}{2(H_{1''} + \lambda)}.\end{aligned}$$

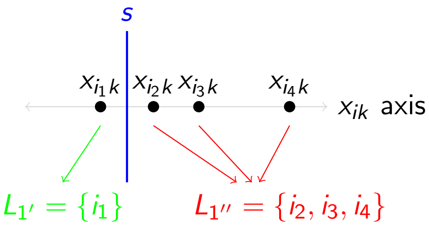
Need to decide if $\Delta(\mathrm{obj}) < 0$ (i.e., have better objective with extra split)—need to check all splits of L_1 into $L_{1'}$ and $L_{1''}$?

Boosted Trees Heuristic

How to compute $\Delta(\text{obj})$ over *all* possible splits of L_1? (And L_2, and so on.)

Recall that we are looking at axis-parallel split trees, so a split of L_1 would look like $\{x_k < s\} \cup \{x_k \geq s\}$.

Sort $\mathbf{x}_i$ $(i \in L_1)$ by x_{ik}:



Compute $\Delta(\text{obj}, L_1 \mapsto L_{1'} \cup L_{1''})$. Pick best of all possible splits (lowest Δ), and repeat across all p possible features. If final $\Delta^* < 0$, split L_1.

Boosted Tree Heuristic

To recap: tree heuristic approximately solves

$$\min_{f \in \mathcal{F}} \sum_i \left(g_i f(\mathbf{x}_i) + h_i f^2(\mathbf{x}_i)/2 \right) \; + \; R(f)$$

using top-down approach. Starting with $L_1 = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$,

1. For all p features, sort $\mathbf{x}_i$ along kth axis.
2. Find best split for each feature, and pick the best split amongst all the p possible choices.
3. If this split lowers objective ($\Delta < 0$), then add it and return to (1) (if $\#$ leaves $\leq T$). Otherwise, stop.

This is a single stage in the boosting process.

Boosted Trees Algorithm

Putting it all together: initialize $f_1 = 0$.

For $t \geq 2$, find approximate solution f_t to

$$\min_{f \in \mathcal{F}} \sum_i \ell\left(y_i, f_1(\mathbf{x}_i) + \cdots + f_{t-1}(\mathbf{x}_i) + f(\mathbf{x}_i)\right) + R(f)$$

using the (Taylor expansion) gradient approximation and subsequent tree heuristic.

This algorithm is implemented in the popular package `XGBoost`, available in many different languages.

Summary

Boosting involves the building successively better models by adding new weak models.

In linear regression case, this corresponded to adding new models that explained (predicted) the leftover residual component.

In the more general setup, this corresponded to adding new models that improve a specific gradient-based objective.

Ensemble Methods

Boosting in an example of one type of *ensemble method*: a machine learning technique that combines multiple models to create a single model.

Another example is Random Forest (RF), which we have encountered before.

- Regression: RF averages the predictions across many trees.
- Classification: RF takes the most frequent prediction across many trees.

Despite the nominal similarity in aggregation, there are fundamental differences between boosting and RF. To understand the difference, let's focus on the regression setting.

Back to Bias-Variance Tradeoff

The old adage: Error = Bias + Variance

$$\mathbb{E}\left[(y-\hat{f}(\mathbf{x}))^2\right] = \Big(\mathbb{E}[\hat{f}(\mathbf{x})-f(\mathbf{x})]\Big)^2 + \mathbb{E}\left[\Big(\hat{f}(\mathbf{x})-\mathbb{E}[\hat{f}(\mathbf{x})]\Big)^2\right] + \mathbb{E}\left[(y-\mathbb{E}[y])^2\right]$$
$$= \Big(\mathrm{bias}(\hat{f}(\mathbf{x}))\Big)^2 + \mathrm{var}(\hat{f}(\mathbf{x})) + \mathrm{var}(y).$$

Boosting uses weak learners (high bias, low variance).
$\hookrightarrow$ Aggregating helps to reduce overall bias (and variance).

In contrast, RF uses strong learners (low bias, high variance).
$\hookrightarrow$ Aggregating helps to reduce overall variance (*assuming individual trees are not strongly correlated*).

Conclusions

Boosting is a powerful methodology for combining many weak models to create a single powerful method.