Gradient Boosting

David Rosenberg

New York University

March 11, 2015
AdaBoost produces a classification score function of the form

$$\sum_{m=1}^{M} \alpha_m G_m(x)$$

- each $G_m$ is a **weak classifier**
- The $G_m$’s are like basis functions, but they are learned from the data.
- Let’s move beyond classification models...
Adaptive Basis Function Model

- Hypothesis space $\mathcal{F}$
  - Can be classifiers or regression functions
  - These would be the “weak classifiers” or “base classifiers”

- An adaptive basis function expansion over $\mathcal{F}$ is
  \[
  f(x) = \sum_{m=1}^{M} \nu_m h_m(x),
  \]
  - Each $h_m \in \mathcal{F}$ is chosen in a learning process, and
  - $\nu_m$ are expansion coefficients.

- For example, $\mathcal{F}$ could be all decision trees of depth at most 4.
- We now discuss one approach to fitting such a model.
Forward Stagewise Additive Modeling

1. Initialize $f_0(x) = 0$.
2. For $m = 1$ to $M$:
   1. Compute:

   $$(\nu_m, h_m) = \arg\min_{\nu \in \mathbb{R}, h \in \mathcal{H}} \sum_{i=1}^{n} \ell \left\{ y_i, f_{m-1}(x_i) + \nu h(x_i) \right\}.$$  

   2. Set $f_m(x) = f_{m-1}(x) + \nu_m h(x)$.
3. Return: $f_M(x)$. 
Exponential Loss and AdaBoost

- Take loss function to be
  \[ \ell(y, f(x)) = \exp(-yf(x)). \]

- Let \( \mathcal{F} = \{ h(x) : \mathcal{X} \rightarrow \{-1, 1\} \} \) be a hypothesis space of weak classifiers.

- Then Forward Stagewise Additive Modeling (FSAM) reduces to AdaBoost.
  - (See HTF Section 10.4 for proof.)
Let’s examine the key step of FSAM a bit more closely:

\[(\nu_m, h_m) = \arg\min_{\nu \in \mathbb{R}, h \in \mathcal{F}} \sum_{i=1}^{n} \ell \left\{ y_i, f_{m-1}(x_i) + \nu h(x_i) \right\} \].

This looks like one step of a numerical optimization method:

- \( h(x_i) \) is like a step direction

This inspires a **new** approach to boosting.

- We can choose \( h_m \) to be something like a gradient in function space.
- Roughly speaking, it will be like the gradient projected onto \( \mathcal{F} \).
- Leads to a **functional gradient descent** method.

Note: This will be a new method. AdaBoost will **not** be a special case.
Functional Gradient Descent: Main Idea

- We want to minimize
  \[ \sum_{i=1}^{n} \ell(y_i, f(x_i)). \]
- Take functional gradient w.r.t. \( f \).
- Find function \( h \in \mathcal{F} \) closest to gradient.
- Take a step in this “projected gradient” direction \( h \).
Note that
\[ \sum_{i=1}^{n} \ell(y_i, f(x_i)) \]
only depends on \( f \) at the training points.

Define
\[ f = (f(x_1), \ldots, f(x_n))^T \]
and write the objective function as
\[ J(f) = \sum_{i=1}^{n} \ell(y_i, f_i). \]
Consider gradient descent on

\[ J(f) = \sum_{i=1}^{n} \ell(y_i, f_i). \]

The **negative gradient step direction** at \( f \) is

\[ -g = -\nabla_f J(f), \]

which we can easily calculate.
Unconstrained step direction is

\[-g = -\nabla_f J(f)\].

Suppose \( \mathcal{F} \) is our weak hypothesis space.

Find \( h \in \mathcal{F} \) that is closest to \(-g\) at the training points, in the \( \ell^2 \) sense:

\[
\min_{h \in \mathcal{F}} \sum_{i=1}^{n} (-g_i - h(x_i))^2.
\]

This is a least squares regression problem!

\( \mathcal{F} \) should have **real-valued** functions.

So the \( h \) that best approximates \(-g\) is our step direction.
Finally, we choose a stepsize.

Option 1 (Line search):

\[ \nu_m = \arg\min_{\nu > 0} \sum_{i=1}^{n} \ell\{ y_i, f_{m-1}(x_i) + \nu h_m(x_i) \} . \]

Option 2: (Shrinkage parameter)

- We consider \( \nu = 1 \) to be the full gradient step.
- Choose a fixed \( \nu \in (0, 1) \) – called a **shrinkage parameter**.
- A value of \( \nu = 0.1 \) is typical – optimize as a hyperparameter.
The Gradient Boosting Machine

1. Initialize $f_0(x) = 0$.

2. For $m = 1$ to $M$:
   
   1. Compute:
      
      $$g_m = \left( \frac{\partial}{\partial f(x_i)} \left( \sum_{i=1}^{n} \ell \{y_i, f(x_i)\} \right) \bigg|_{f(x_i) = f_{m-1}(x_i)} \right)_{i=1}^{n}$$
   
   2. Fit regression model to $-g_m$:
      
      $$h_m = \arg \min_{h \in F} \sum_{i=1}^{n} ((-g_m)_i - h(x_i))^2.$$
   
   3. Choose fixed step size $\nu_m = \nu \in (0, 1]$, or take
      
      $$\nu_m = \arg \min_{\nu > 0} \sum_{i=1}^{n} \ell \{y_i, f_{m-1}(x_i) + \nu h_m(x_i)\}.$$
   
   4. Take the step:
      
      $$f_m(x) = f_{m-1}(x) + \nu_m h_m(x)$$
The Gradient Boosting Machine: Recap

• Take any differentiable loss function.
• Choose a weak hypothesis space for regression.
• Choose number of steps (or a stopping criterion).
• Choose step size methodology.
• Then you’re good to go!
Gradient Tree Boosting

- Common form of gradient boosting machine takes

\[ F = \{ \text{regression trees of size } J \}, \]

where \( J \) is the number of terminal nodes.
- \( J = 2 \) gives decision stumps
- HTF recommends \( 4 \leq J \leq 8 \).

Software packages:
- Gradient tree boosting is implemented by the \texttt{gbm package} for R
- as \texttt{GradientBoostingClassifier} and \texttt{GradientBoostingRegressor} in \texttt{sklearn}

- For trees, there are other tweaks on the algorithm one can do
  - See HTF 10.9-10.12 and