A Discussion on GBDT: Gradient Boosting Decision Tree

Presented by Tom

March 6, 2012

Outline

Outline

Function Estimation

- Output(or response): a random variable y
- Input(or explanatory): a set of random variables $\mathbf{x} = \{x_1, \dots, x_n\}$
- Goal: using a training sample $\{y_i, \mathbf{x}_i\}_1^N$ of known $(y, \mathbf{x})$ values to obtain an estimate $\hat{F}(\mathbf{x})$ of the function $F^*(\mathbf{x})$ mapping $\mathbf{x}$ to y
- Minimizing the expected value of some specified loss function $L(y, F(\mathbf{x}))$:

$$F^* = \arg\min_F E_{y,\mathbf{x}} L(y, F(\mathbf{x})). \tag{1}$$

Numerical Optimization in Function Space

- Take a non-parametric approach
- Apply numerical optimization in function space
- Consider $\mathbf{F}(\mathbf{x})$ evaluated at each point $\mathbf{x}$ to a parameter and seek to minimize $\Phi(F) = E_{y,\mathbf{x}} L(y, F(\mathbf{x})) = E_{\mathbf{x}}[E_y(L(y, F(\mathbf{x})))|\mathbf{x}]$ at each individual $\mathbf{x}$, directly with respect to $F(\mathbf{x})$
- Numerical optimization paradigm:

$$F^*(\mathbf{x}) = \sum_{m=0}^{M} f_m(\mathbf{x}),$$

where $f_0(\mathbf{x})$ is an initial guess, and $\{f_m(\mathbf{x})\}_1^M$ are incremental functions (steps or boosts) defined by the optimization method

Numerical Optimization in Function Space

- Steepest-descent:

$$f_m(\mathbf{x}) = -\rho_m g_m(\mathbf{x})$$

with

$$g_m(\mathbf{x}) = [\frac{\partial \Phi(F(\mathbf{x}))}{\partial F(\mathbf{x})}]_{F(\mathbf{x})=F_{m-1}(\mathbf{x})}$$

and

$$F_{m-1}(\mathbf{x}) = \sum_{0}^{m-1} f_i(\mathbf{x})$$

- The multiplier ρ_m is given by the line search:

$$\rho_m = \arg\min_{\rho} E_{y,\mathbf{x}} L(y, F_{m-1}(\mathbf{x}) - \rho g_m(\mathbf{x}))$$

Finite Data

- Nonparametric approach breaks down when the joint distribution is estimated by a finite data sample
- Strength must be borrowed from nearby data points by imposing smoothness on the solution
- Assume a parameterized form and do parameterized optimization to minimize the corresponding data based estimate of expected loss:

$$\{\beta_m, \mathbf{a}_m\}_1^M = \arg \min_{\{\beta'_m, \mathbf{a}'_m\}_1^M} \sum_{i=1}^{N} L(y_i, \sum_{m=1}^{M} \beta'_m h(\mathbf{x}_i; \mathbf{a}'_m))$$

Finite Data

- In situation where this is infeasible one can try a greedy stagewise approach. For $m = 1, 2, \ldots, M$,

$$\{\beta_m, \mathbf{a}_m\} = \arg\min_{\{\beta, \mathbf{a}\}} \sum_{i=1}^{N} L(y_i, F_{m-1}(\mathbf{x}_i) + \beta h(\mathbf{x}_i; \mathbf{a}))$$

and then

$$F_m(\mathbf{x}) = F_{m-1}(\mathbf{x}) + \beta_m h(\mathbf{x}; \mathbf{a}_m)$$

Finite Data

- In signal processing this stagewise strategy is called matching pursuit
    - $L(y, F)$ is squared-error loss
    - $\{h(\mathbf{x}; \mathbf{a}_m)\}_1^M$ are called basis functions, usually taken from waveletlike dictionary
- In machine learning this stagewise strategy is called boosting
    - $y \in \{-1, 1\}$
    - $L(y, F)$ is either an exponential loss criterion e^{-yF} or negative binomial loglikelihood $\log(1 + e^{-2yF})$
    - $h(\mathbf{x}; \mathbf{a})$ is called a weak learner or base learner, and usually is a classification tree

Outline

Physical Meaning

- Suppose for a particular loss $L(y, F)$ and base learner $h(\mathbf{x}, \mathbf{a})$, the solution to $(\beta_m, \mathbf{a}_m)$ is difficult to obtain
- Given any approximator $F_{m-1}(\mathbf{x})$, the function $\beta_m h(\mathbf{x}; \mathbf{a}_m)$ can be viewed as the best greedy step toward the data-based estimate of $F^*(\mathbf{x})$, under the constraint that the step direction $h(\mathbf{x}; \mathbf{a}_m)$ be a member of the parameterized class of functions
- The data-based analogue of the unconstrained negative gradient:

$$-g_m(\mathbf{x}_i) = -[\frac{\partial L(y_i, F(\mathbf{x}_i))}{\partial F(\mathbf{x}_i)}]_{F(\mathbf{x})=F_{m-1}(\mathbf{x})}$$

gives the best steepest-descent step direction in the N-dimensional data space at $F_{m-1}(\mathbf{x})$

Physical Meaning

- This gradient is defined only at the data points $\{\mathbf{x}_i\}_1^N$ and cannot be generalized to other x-values
- One possibility for generalization is to choose that member of the parameterized class $h(\mathbf{x}; \mathbf{a}_m)$ that produces $\mathbf{h}_m = \{h(\mathbf{x}_i; \mathbf{a}_m)\}_1^N$ most parallel to $-\mathbf{g}_m \in R^N$
- It can be obtained from the solution:

$$\mathbf{a}_m = \arg\min_{\mathbf{a},\beta} \sum_{i=1}^{N} [-g_m(\mathbf{x}_i) - \beta h(\mathbf{x}_i; \mathbf{a})]^2$$

Physical Meaning

- This constrained negative gradient is used in place of the unconstrained one in the steepest-descent strategy. Specifically, the line search is performed:

$$\rho_m = \arg\min_{\rho} \sum_{i=1}^{N} L(y_i, F_{m-1}(\mathbf{x}_i) + \rho h(\mathbf{x}_i; \mathbf{a}_m))$$

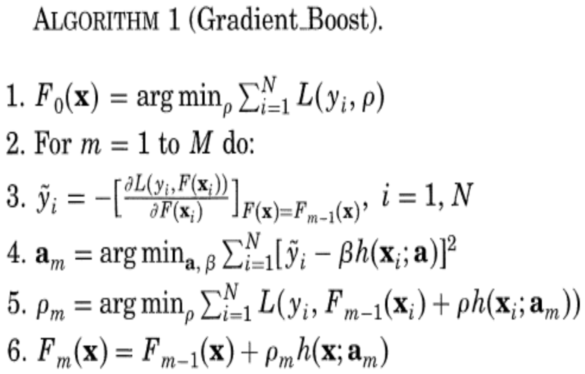
  and the approximate updated

$$F_m(\mathbf{x}) = F_{m-1}(\mathbf{x}) + \rho_m h(\mathbf{x}; \mathbf{a}_m)$$

Physical Meaning

- Advantage: replace the difficult function minimization problem $(\beta_m, \mathbf{a}_m)$ by least-squares function minimization, followed by only a single parameter optimization based on the original criterion
- For any $h(\mathbf{x}; \mathbf{a})$ for which a feasible least-squares algorithm exists for solving above formula, one can use this approach to minimize any differentiable loss $L(y, F)$ in conjunction with forward stage-wise additive modeling.

Generic Algorithm using Steepest-Descent



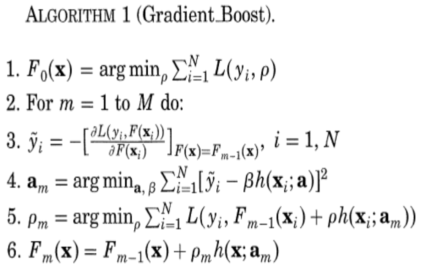
ALGORITHM 1 (Gradient_Boost).

```
1. $F_0(\mathbf{x}) = \arg\min_\rho \sum_{i=1}^N L(y_i, \rho)$
2. For $m = 1$ to $M$ do:
3. $\tilde{y}_i = -\left[\frac{\partial L(y_i, F(\mathbf{x}_i))}{\partial F(\mathbf{x}_i)}\right]_{F(\mathbf{x})=F_{m-1}(\mathbf{x})}, \ i = 1, N$
4. $\mathbf{a}_m = \arg\min_{\mathbf{a}, \beta} \sum_{i=1}^N [\tilde{y}_i - \beta h(\mathbf{x}_i; \mathbf{a})]^2$
5. $\rho_m = \arg\min_\rho \sum_{i=1}^N L(y_i, F_{m-1}(\mathbf{x}_i) + \rho h(\mathbf{x}_i; \mathbf{a}_m))$
6. $F_m(\mathbf{x}) = F_{m-1}(\mathbf{x}) + \rho_m h(\mathbf{x}; \mathbf{a}_m)$
```

Outline

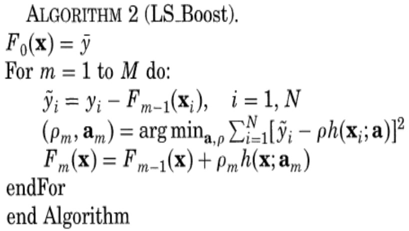
Least-Squares (LS) Regression

- $L(y, F) = (y - F)^2/2$
- Gradient boosting on squared-error loss produces the usual stagewise approach of iteratively fitting the current residuals



ALGORITHM 1 (Gradient_Boost).

```
1. $F_0(\mathbf{x}) = \arg\min_\rho \sum_{i=1}^N L(y_i, \rho)$
2. For $m = 1$ to $M$ do:
3. $\tilde{y}_i = -\left[\frac{\partial L(y_i, F(\mathbf{x}_i))}{\partial F(\mathbf{x}_i)}\right]_{F(\mathbf{x})=F_{m-1}(\mathbf{x})}, \ i = 1, N$
4. $\mathbf{a}_m = \arg\min_{\mathbf{a}, \beta} \sum_{i=1}^N [\tilde{y}_i - \beta h(\mathbf{x}_i; \mathbf{a})]^2$
5. $\rho_m = \arg\min_\rho \sum_{i=1}^N L(y_i, F_{m-1}(\mathbf{x}_i) + \rho h(\mathbf{x}_i; \mathbf{a}_m))$
6. $F_m(\mathbf{x}) = F_{m-1}(\mathbf{x}) + \rho_m h(\mathbf{x}; \mathbf{a}_m)$
```



ALGORITHM 2 (LS_Boost).

```
$F_0(\mathbf{x}) = \bar{y}$
For $m = 1$ to $M$ do:
    $\tilde{y}_i = y_i - F_{m-1}(\mathbf{x}_i), \quad i = 1, N$
    $(\rho_m, \mathbf{a}_m) = \arg\min_{\mathbf{a}, \rho} \sum_{i=1}^N [\tilde{y}_i - \rho h(\mathbf{x}_i; \mathbf{a})]^2$
    $F_m(\mathbf{x}) = F_{m-1}(\mathbf{x}) + \rho_m h(\mathbf{x}; \mathbf{a}_m)$
endFor
end Algorithm
```

Least Absolute Deviation (LAD) Regression

- Loss function: $L(y, F) = |y - F|$

$$\tilde{y}_i = sign(y_i - F_{m-1}(\mathbf{x}_i))$$

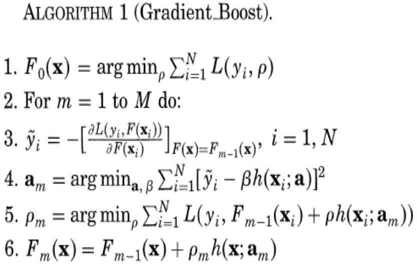
- Consider the special case where each base learner is an J-terminal node regression tree, each regression tree has the additive form:

$$h((x); \{b_j, R_j\}_1^J) = \sum_{j=1}^{J} b_j 1(\mathbf{x} \in R_j)$$

- $\{R_j\}_1^J$ are disjoint regions that collectively cover the space of all joint values of the predictor variables (x)

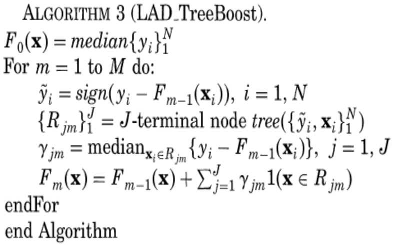
Least Absolute Deviation (LAD) Regression

- Through some transformations, we could obtain the algorithm as follows:



ALGORITHM 1 (Gradient_Boost).

```
1. $F_0(\mathbf{x}) = \arg\min_\rho \sum_{i=1}^N L(y_i, \rho)$
2. For $m = 1$ to $M$ do:
3. $\tilde{y}_i = -\left[\frac{\partial L(y_i, F(\mathbf{x}_i))}{\partial F(\mathbf{x}_i)}\right]_{F(\mathbf{x})=F_{m-1}(\mathbf{x})}, \; i = 1, N$
4. $\mathbf{a}_m = \arg\min_{\mathbf{a}, \beta} \sum_{i=1}^N [\tilde{y}_i - \beta h(\mathbf{x}_i; \mathbf{a})]^2$
5. $\rho_m = \arg\min_\rho \sum_{i=1}^N L(y_i, F_{m-1}(\mathbf{x}_i) + \rho h(\mathbf{x}_i; \mathbf{a}_m))$
6. $F_m(\mathbf{x}) = F_{m-1}(\mathbf{x}) + \rho_m h(\mathbf{x}; \mathbf{a}_m)$
```



ALGORITHM 3 (LAD_TreeBoost).

```
$F_0(\mathbf{x}) = median\{y_i\}_1^N$
For $m = 1$ to $M$ do:
    $\tilde{y}_i = sign(y_i - F_{m-1}(\mathbf{x}_i)), \; i = 1, N$
    $\{R_{jm}\}_1^J = J$-terminal node $tree(\{\tilde{y}_i, \mathbf{x}_i\}_1^N)$
    $\gamma_{jm} = \text{median}_{\mathbf{x}_i \in R_{jm}} \{y_i - F_{m-1}(\mathbf{x}_i)\}, \; j = 1, J$
    $F_m(\mathbf{x}) = F_{m-1}(\mathbf{x}) + \sum_{j=1}^J \gamma_{jm} 1(\mathbf{x} \in R_{jm})$
endFor
end Algorithm
```

Least Absolute Deviation (LAD) Regression

- This algorithm is highly robust
  - The trees use only order information on the individual input variables x_j
  - The pseudoresponses $\tilde{y}_i$ have only two values, $\tilde{y}_i = \{-1, 1\}$
  - Terminal node updates are based on medians

Other Regression Techniques

- M-regression
- Two-class logistic regression and classification
- Multiclass logistic regression and classification
- Please refer to (Jerome H. Friedman 2001)

Regularization

- Fitting the training data too closely can be counterproductive
- Reducing the expected loss on the training data beyond some point causes the population-based loss to stop decreasing and often to start increasing
- Regularization methods attempt to prevent overfitting by constraining the fitting procedure

Regularization

- For additive expansions a natural regularization parameter is the number of components M
- Controlling the value of M regulates the degree to which expected loss on the training data can be minimized
- It has often been found that regularization through shrinkage provides superior results

$$F_m(\mathbf{x}) = F_{m-1}(\mathbf{x}) + \nu \times \rho_m h(\mathbf{x}; \mathbf{a}_m)$$

Regularization

- Decreasing the value of ν increases the best value for M
- We could tune parameters according to applications

Outline

Advantages

- All TreeBoost procedures are invariant under all strictly monotone transformations of the individual input variables. For example, using $x_j, \log x_j, e^{x_j}, x_j^a$
- Eliminate the sensitivity to long-tailed distributions and outliers
- Trees tend to be robust against the addition of irrelevant input variables

Comparison with Single Tree Models

- Disadvantage of single tree models:
  - Inaccurate for smaller trees
  - Instable for larger trees, involve high-order interactions
- Mitigated by boosting:
  - Produce piecewise constant approximations, but the granularity is much finer
  - Enhance stability by using small trees and averaging over many of them

Scalability

- After sorting the input variables, the computation of the regression TreeBoost procedues (LS,LAD and M TreeBoost) scales linearly with the number of observations N, the number of input variables n and the number of iterations M. Scales roughly as the logarithm of the size of the constituent trees J. The classification algorithm L_K *TreeBoost* scales linearly with the number of classes K
- More data become available after modeling is complete, boosting can be continued on the new data starting from the previous solution
- Boosting on successive subsets of data can also be used when there is insufficient random access main memory to store the entire data set

Outline

Resources

- R Package:
  - http://cran.r-project.org/web/packages/gbm/index.html
  - http://cran.r-project.org/web/packages/mboost/index.html
  - http://cran.r-project.org/web/packages/gbev/index.html
- Java: weka.sourceforge.net/doc/weka/classifiers/meta/AdditiveRegression.htm
- C++:
  - https://sites.google.com/site/rtranking/
  - https://mloss.org/software/view/332/

References

- Jerome H. Friedman, Greedy Function Approximation: A Gradient Boosting Machine, 2001
- L. Breiman, J. H. Friedman, R. Olshen and C. Stone, Classification and Regression Trees, 1983
- Y. Freund and R. Schapire, Experiments with a new boosting algorithm, 1996
- J. H. Friedman, T. Hastie and R. Tibshirani, Additive logistic regression: a statistical view of boosting, 2000
- S. Mallat and Z. Zhang, Matching pursuits with time frequency dictionaries, 1993
- R. Schapire and Y. Singer, Improved boosting algorithms using confidence-rated predictions, 1998
- http://en.wikipedia.org/wiki/Gradient_boosting

QA

Thanks for your attention!