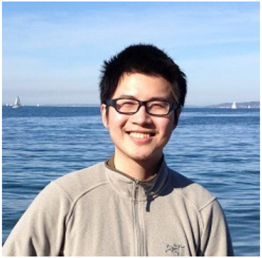
Hacking the Xgboost

Raul Sanchez-Vazquez

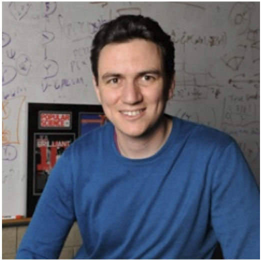
Kunumi

22 June 2018

Introduction



(a) Tianqi Chen



(b) Carlos Guestrin

Figure 1: Authors

Introduction

Tianqi Chen:

- (2013 – Present) Ph.D. Student (University of Washington)
  - Social Media and Recommender Systems
- (2012 – 2012) Intern (Huawei Noah's Ark Research Lab)
- (2009 – 2012) Teaching Assistant (Shanghai Jiao Tong University)
  - Compiler
  - Operating System
  - Database Management System
  - Data Mining and Machine Learning in Practice
- (2011 – 2011) Visiting Scholar (DERI)
  - Information extraction from technology reviews
- (2010 – 2013) Master in Computer Science (Shanghai Jiao Tong University)
- (2006 – 2010) Bsc Computer Science (Shanghai Jiao Tong University)

Introduction

Carlos Guestrin:

- (2016 – Present) Senior Director of AI and Machine Learning (Apple)
- (2012 – Present) Professor of Machine Learning (University of Washington)
- (2013 – 2016) Founder & CEO (NameDato, Inc.)
- (2009 – 2012) Co-Founder (Flashgroup)
- (2004 – 2012) Associate Professor (Carnegie Mellon University)
- (2003 – 2004) Senior Researcher (Intel Corporation)
- (1998 – 2003) Ph.D Computer Science (Stanford University)
- (1993 – 1998) Mechatronics, Robotics, and Automation Engineering (Universidade de São Paulo)

Gradient Boosting

Dataset:

$$\mathcal{D} = \{(x_i, y_i)\}$$

Given n examples and m features $|\mathcal{D}| = n$ and $x_i \in \mathbb{R}^m$

$$\hat{y}_i = \sum_{k=1}^{K} f_k(x_i), f_k \in \mathcal{F} \tag{1}$$

Where K is the number of week learners and f_k is a single week learner drawn from $\mathcal{F}$, which is the space of all possible learners (i.e. decision trees, neural nets, k-neares-neighbors, linear regression, SVM,).

Decision Tree

In the specific case of XGBoost, $\mathcal{F}$ is the space of regression trees (known as CART):

$$\mathcal{F} = \{f(x) = w_q(x)\}(q : \mathbb{R}^m \to T, w \in \mathbb{R}^{\mathbb{T}})$$

- q represents the structure of each tree that maps to leaf indexes
- T is the number of leaves in the tree
- f_k corresponds to an independent tree structure q and weights w
- w_i corresponds to the score in the i-th leaf

Example of decision tree

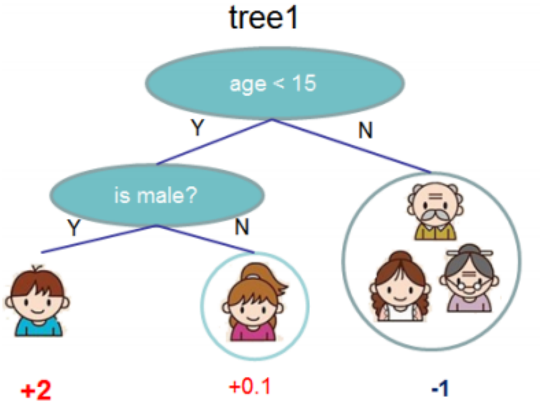


Figure 2: Tree example

- $T = 3$
- $w = \{w_1, w_2, w_3\} = \{2, .1, -1\}$
- q is the tree structure

Loss Function

Regularized learning objective:

$$\mathcal{L}(\phi) = \sum_i l(\hat{y}_i, y_i) + \sum_k \Omega(f_k) \tag{2}$$

Where:

- $\Omega(f) = \gamma T + \frac{1}{2}\lambda \|w\|^2$
- The function l must be differentiable

Gradient Tree Boosting

- The tree ensemble cannot be optimized using traditional optimization methods.
- For such reasons, the ensemble model is trained in an **additive manner**

Let $\hat{y}_i^t$ be the prediction of the i-th instance at iteration t. We will need to add f_t to minimize the following objective:

$$\mathcal{L}^{(t)} = \sum_{i=1}^{n} l(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)) + \Omega(f_t)$$

Newton's Method

$$f_T(x) = f_T(x_n + \Delta x) \approx f(x_n) + f'(x_n)\Delta x + \frac{1}{2} f''(x_n)\Delta x^2 \,.$$

Figure 3: Newton's method in optimization

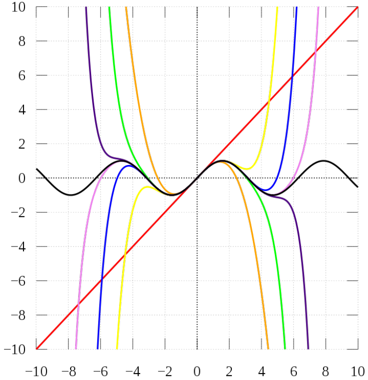


Figure 4: Taylor Expansion

Second Order approximation

Regularized objective:

$$\mathcal{L}^{(t)} = \sum_{i=1}^{n} l(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)) + \Omega(f_t)$$

Applying second-order approximation:

$$\mathcal{L}^{(t)} \approx \sum_{i=1}^{n} \left[l(y_i, \hat{y}^{(t-1)}) + g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i) \right] + \Omega(f_t)$$

Where

- $g_i = \partial_{\hat{y}(t-1)} l(y_i, \hat{y}^{t-1})$ first gradient on the loss function
- $h_i = \partial^2_{\hat{y}(t-1)} l(y_i, \hat{y}^{t-1})$ second gradient on the loss function

Simplified Loss

Simplifying the second-order approximation equation:

$$\tilde{\mathcal{L}}^{(t)} = \sum_{i=1}^{n} \left[g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i) \right] + \Omega(f_t) \tag{3}$$

Define $I_j = \{i | q(x_i) = j\}$ as the instance set of leaf j. We can rewrite Eq 3 by expanding Ω:

$$\tilde{\mathcal{L}}^{(t)} = \sum_{i=1}^{n} \left[g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i) \right] + \gamma T + \frac{1}{2} \lambda \sum_{j=1}^{T} w_j^2$$

Simplified Loss

$I_j = \{i | q(x_i) = j\}$ instance set of leaf j.

$$\tilde{\mathcal{L}}^{(t)} = \sum_{i=1}^{n} \left[g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i) \right] + \gamma T + \frac{1}{2} \lambda \sum_{j=1}^{T} w_j^2$$

$$\tilde{\mathcal{L}}^{(t)} = \sum_{j=1}^{T} \left[(\sum_{i \in I_j} g_i) w_j + \frac{1}{2} (\sum_{i \in I_j} h_i) w_j^2 \right] + \gamma T + \frac{1}{2} \lambda \sum_{j=1}^{T} w_j^2$$

$$\tilde{\mathcal{L}}^{(t)} = \sum_{j=1}^{T} \left[(\sum_{i \in I_j} g_i) w_j + \frac{1}{2} (\sum_{i \in I_j} h_i) w_j^2 + \frac{1}{2} \lambda w_j^2 \right] + \gamma T$$

$$\tilde{\mathcal{L}}^{(t)} = \sum_{j=1}^{T} \left[(\sum_{i \in I_j} g_i) w_j + \frac{1}{2} w_j^2 (\sum_{i \in I_j} h_i + \lambda) \right] + \gamma T$$

Optimal Weight

$$\tilde{\mathcal{L}}^{(t)} = \sum_{j=1}^{T} \left[(\sum_{i \in I_j} g_i) w_j + \frac{1}{2} w_j^2 (\sum_{i \in I_j} h_i + \lambda) \right] + \gamma T \tag{4}$$

To compute the optimal leaf weight w_j^* from Eq 4, we derivate the loss with respect to w considering a single leave and set it to 0:

$$\frac{\tilde{\mathcal{L}}^{(w)}}{dw} = (\sum_{i \in I_j} g_i) + w_j (\sum_{i \in I_j} h_i + \lambda)$$

$$0 = (\sum_{i \in I_j} g_i) + w_j (\sum_{i \in I_j} h_i + \lambda)$$

$$-w_j (\sum_{i \in I_j} h_i + \lambda) = (\sum_{i \in I_j} g_i)$$

$$w_j^* = -\frac{\sum_{i \in I_j} g_i}{\sum_{i \in I_j} h_i + \lambda} \tag{5}$$

Optimal Weight and Tree Structure

Substituing Eq 5 on Eq 4

Eq 5:

$$\tilde{\mathcal{L}}^{(t)} = \sum_{j=1}^{T} \left[(\sum_{i \in I_j} g_i) w_j + \frac{1}{2} w_j^2 (\sum_{i \in I_j} h_i + \lambda) \right] + \gamma T$$

Eq 4:

$$w_j^* = -\frac{\sum_{i \in I_j} g_i}{\sum_{i \in I_j} h_i + \lambda}$$

We obtain a measure of the quality of a tree structure q:

$$\sum_{j=1}^{T} \Big[\sum_{i \in I_j} g_i \left(-\frac{\sum_{i \in I_j} g_i}{\sum_{i \in I_j} h_i + \lambda} \right) + \frac{1}{2} \left(-\frac{\sum_{i \in I_j} g_i}{\sum_{i \in I_j} h_i + \lambda} \right)^2 (\sum_{i \in I_j} h_i + \lambda) \Big] + \gamma T$$

Optimal Weight and Tree Structure

$$\sum_{j=1}^{T}\Big[\sum_{i\in I_j} g_i\left(-\frac{\sum_{i\in I_j} g_i}{\sum_{i\in I_j} h_i+\lambda}\right)+\frac{1}{2}\left(-\frac{\sum_{i\in I_j} g_i}{\sum_{i\in I_j} h_i+\lambda}\right)^2\Big(\sum_{i\in I_j} h_i+\lambda\Big)\Big]+\gamma T$$

$$\sum_{j=1}^{T}\Big[-\frac{(\sum_{i\in I_j} g_i)^2}{(\sum_{i\in I_j} h_i+\lambda)}+\frac{1}{2}\left(\frac{(\sum_{i\in I_j} g_i)^2(\sum_{i\in I_j} h_i+\lambda)}{(\sum_{i\in I_j} h_i+\lambda)^2}\right)\Big]+\gamma T$$

$$\sum_{j=1}^{T}\Big[-\frac{(\sum_{i\in I_j} g_i)^2}{(\sum_{i\in I_j} h_i+\lambda)}+\frac{1}{2}\frac{(\sum_{i\in I_j} g_i)^2}{(\sum_{i\in I_j} h_i+\lambda)}\Big]+\gamma T$$

$$\tilde{\mathcal{L}}^{(t)}(q)=-\frac{1}{2}\sum_{j=1}^{T}\Big[\frac{(\sum_{i\in I_j} g_i)^2}{\sum_{i\in I_j} h_i+\lambda}\Big]+\gamma T \qquad (6)$$

Optimal Weight and Tree Structure

Eq 6 can be used as a scoring function to measure the quality of a tree structure q:

$$\tilde{\mathcal{L}}^{(t)}(q) = -\frac{1}{2}\sum_{j=1}^{T}\left[\frac{(\sum_{i\in I_j} g_i)^2}{\sum_{i\in I_j} h_i + \lambda}\right] + \gamma T$$

Normally it is impossible to enumerate all possible tree structures q. A greedy algorithm starts from a single leaf and iteratively adds branches to the tree.
Assume that I_L and I_R are the instance sets of left and right nodes after the split:

$$\mathcal{L} = \frac{1}{2}\left[\frac{(\sum_{i\in I_L} g_i)^2}{\sum_{i\in I_L} h_i + \lambda} + \frac{(\sum_{i\in I_R} g_i)^2}{\sum_{i\in I_R} h_i + \lambda} - \frac{(\sum_{i\in I} g_i)^2}{\sum_{i\in I} h_i + \lambda}\right] - \gamma \qquad (7)$$

Algorithm 1: Exact Greedy Algorithm for Split Finding

Input: I, instance set of current node
Input: d, feature dimension
$gain \leftarrow 0$
$G \leftarrow \sum_{i \in I} g_i$, $H \leftarrow \sum_{i \in I} h_i$
for $k = 1$ ***to*** m **do**
  $G_L \leftarrow 0, \; H_L \leftarrow 0$
  for j *in sorted*(I, *by* $\mathbf{x}_{jk}$) **do**
    $G_L \leftarrow G_L + g_j, \; H_L \leftarrow H_L + h_j$
    $G_R \leftarrow G - G_L, \; H_R \leftarrow H - H_L$
    $score \leftarrow \max(score, \frac{G_L^2}{H_L+\lambda} + \frac{G_R^2}{H_R+\lambda} - \frac{G^2}{H+\lambda})$
  end
end
Output: Split with max score

Figure 5: Split finding