STATS305A - Lecture 15

John Duchi
Scribed by Michael Howes

11/11/21

Contents

1 Overview

Today we will be talking about advanced predictive methods in regression. Namely

- Boosting with:
  - Decision stumps.
  - Decision trees.
- Locally weighted regression.

2 Boosting

Motivation: We don't know apriori what the "right" features, scalings, etc are.
Idea: iteratively add "best" features into a given model.
Setting: We have a collection of potential features

$$\Phi = \{\phi : \mathcal{X} \to \mathbb{R}\}.$$

2.1 Decision stumps

A common choice of Φ is the set of *decision stumps* $\phi : \mathbb{R}^d \to \{\pm 1\}$ where

$$\phi(x) = \begin{cases} 1 & \text{if } x_j \geq \tau, \\ -1 & \text{if } x_j < \tau. \end{cases}$$
$$= \text{sign}(x_j - \tau).$$

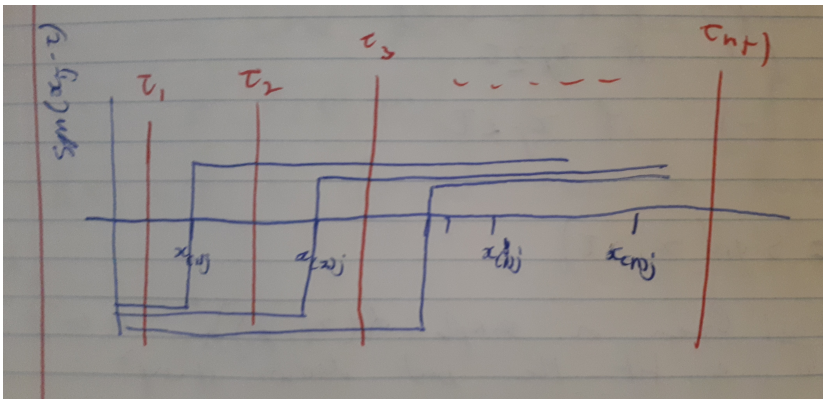
Our basic question is given a sample $(x_i, y_i)_{i=1}^n$ such that $x_i \in \mathbb{R}^d$ and $y_i \in \mathbb{R}$, how can we fit the the best decision stump? Our idea is to find the stump which is most correlted with the response y. Define

$$\widehat{c}(\tau, j) := \sum_{i=1}^{n} y_i \operatorname{sign}(x_{ij} - \tau),$$

which is the coreelation between a stump for feature j and the response $(y_i)_{i=1}^n$. We thus wish to choose a pair $(j, \tau) \in [d] \times \mathbb{R}$ that maximizes $|\widehat{c}(j, \tau)|$. How do we do this? The key observation is that we should sort our x_i for each coordinate j. For a fixed j write

$$x_{(1),j} \leq x_{(2),j} \leq \ldots \leq x_{(n),j}.$$

Note that $\widehat{c}(\tau, j)$ is constant on the intervals $\tau \in (-\infty, x_{(1),j})$, $\tau \in [x_{(i),j}, x_{(i+1),j})$ and $\tau \in [x_{(n),j}, \infty)$. This is because the signs $\operatorname{sign}(x_{ij} - \tau)$ are constant on said intervals (see picture). Thus we only need to test $n + 1$ values of τ for each j. These values correspond to the red lines.



We thus have the following algorithm for finding the best decision stump. For each coordinate sort x_{ij} (this takes time $O(dn \log(n))$). Evaluate $y^T \operatorname{sign}(x^{(j)} - \tau)$ for the $n + 1$ values of τ where $X = [x^{(1)}, \ldots, x^{(d)}]$. Since there are d coordinatex and $n + 1$ thresholds we need to caclulate $d(n + 1)$ inner products in $\mathbb{R}^n$ giving us a total run time of

$$O(d(n + 1)n + dn \log(n)) = O(dn^2).$$

But with another idea we can reduce this run time. Suppose our thresholds from sorting $x^{(j)}$ are $\tau_1 < \tau_2 < \ldots < \tau_{n+1}$. Then

$$\begin{aligned}
\widehat{c}(\tau_1, j) &= y^T \mathbf{1} \\
\widehat{c}(\tau_2, j) &= y^T \mathbf{1} - 2y_{(1)} \\
&= \widehat{c}(\tau_1, j) + 2y_{(1)} \\
&\vdots \quad \vdots \\
\widehat{c}(\tau_{k+1}, j) &= \widehat{c}(\tau_k, j) - 2y_{(k)} \\
&\vdots \quad \vdots \\
\widehat{c}(\tau_{n+1}, j) &= \widehat{c}(\tau_k, n) - 2y_{(n)} \\
&= -y^T \mathbf{1},
\end{aligned}$$

where $(1), (2), \ldots, (n)$ are the indices from sorting $x^{(j)}$. That is,

$$x_{(1)}^{(j)} \leq x_{(2)}^{(j)} \leq \ldots \leq x_{(n)}^{(j)}.$$

Thus we can calculate the n inner products for coordinate j in linear time. Thus we can find the most correlated decision stump in time

$$O(dn\log(n)).$$

2.2 Boosting iteration

Let

$$X = \begin{bmatrix} x_1^T \\ \vdots \\ x_n^T \end{bmatrix},$$

be our matrix of features. Suppose that at iteration k we have a model $M_k(X)$ given by

$$M_k(X) = [\phi_1(X), \phi_2(X), \ldots, \phi_k(X)],$$

where

$$\phi_j(X) = \begin{bmatrix} \phi_j(x_1) \\ \vdots \\ \phi_j(x_n) \end{bmatrix} \in \mathbb{R}^n.$$

Also let

$$\widehat{\beta}^k = \underset{b}{\operatorname{argmin}} \left\| M_k(X) - y \right\|_2^2.$$

How do we add a feature to this model? Define,

$$R_k(\theta, \phi) = \frac{1}{2} \left\| M_k(X)\widehat{\beta}^k + \theta\phi(X) - y \right\|_2^2,$$

where $\theta \in \mathbb{R}$ and $\phi \in \Phi$ and

$$\phi(X) = \begin{bmatrix} \phi(x_1) \\ \vdots \\ \phi(x_n) \end{bmatrix} \in \mathbb{R}^n.$$

The quantity $R_k(\theta, \phi)$ is the residual sum of squares when we add a feature $\phi : \mathcal{X} \to \mathbb{R}$ and a parameter $\theta \in \mathbb{R}$ to our model. Note that $\widehat{\beta}^k$ is fixed. If we define

$$y^k = y - M_k(X)\widehat{\beta}^k,$$

then

$$R_k(\theta, \phi) = \frac{1}{2} \left\| y^k - \theta\phi(X) \right\|_2^2.$$

Thus we can think of minimizing $R_k(\theta, \phi)$ as a one-dimensional regression where we fit y^k onto $\phi(X)$. Our goal is to choose ϕ that minimizes or approximately minimizes $R_k(\theta, \phi)$. There are two methods for choosing this ϕ.

- Version 1: We choose ϕ to maximize $\left| \frac{\partial}{\partial\theta} R_k(\theta, \phi)|_{\theta=0} \right|$. That is, we do an approximation and choose the value of ϕ which gives us the best marginal benefit of adding ϕ.
- Version 2: We choose ϕ to minimize $\inf_\theta R_k(\theta, \phi)$. Such a ϕ would be an exact solution to our minimization problem.

Let's do some calculations for both these methods. Note that

$$\frac{\partial}{\partial\theta} R_k(\theta, \phi) = \theta \left\| \phi(X) \right\|_2^2 - \phi(X)^T y^k.$$

At $\theta = 0$ we thus have

$$\frac{\partial}{\partial\theta} R_k(\theta, \phi)\Big|_{\theta=0} = -\phi(X)^T y^k.$$

Thus we choose ϕ to maximize the aboslute value of the correlation between $\phi(X)$ and y^k. Note that for decision stumps we have seen how to do this efficiently.

We'll now turn to the second version. Define

$$R_k^*(\phi) = \inf_\theta R_k(\theta, \phi).$$

We wish to choose ϕ that minimizes $R_k^*(\phi)$. Recall that

$$R_k(\theta, \phi) = \frac{1}{2} \left\| y^k - \theta\phi(X) \right\|_2^2.$$

Thus the infimum in R_k^* is obtained and it is obtained at

$$\theta_\phi = \frac{\phi(X)^T y_k}{\|\phi(X)\|_2^2}.$$

Thus

$$\begin{aligned} R_k^*(\phi) &= R_k(\theta_\phi, \phi) \\ &= \frac{1}{2} \left\| y^k - \frac{\phi(X)^T y^k}{\|\phi(X)\|_2^2} \phi(X) \right\|_2^2 \\ &= \frac{1}{2} \left\| y^k \right\|_2^2 - \frac{1}{2} \left\| \frac{\phi(X)^T y^k}{\|\phi(X)\|_2^2} \phi(X) \right\|_2^2 \\ &= \frac{1}{2} \left\| y^k \right\|_2^2 - \frac{1}{2} \frac{(\phi(X)^T y^k)^2}{\|\phi(X)\|_2^2}. \end{aligned}$$

Thus $R_k^*(\phi)$ is also maximized when the absolute value of the correlation between $\phi(X)$ and y^k is maximized. Just like with the previous version/method. With both methods we choose ϕ_{k+1} to be the element of Φ that is most correlated with y^k.

After choosing ϕ_{k+1} set

$$M_{k+1}(X) = [\phi_1(X), \ldots, \phi_{k+1}(X)],$$

and

$$\widehat{\beta}_{k+1} = \operatorname*{argmin}_{b \in \mathbb{R}^{k+1}} \|M_{k+1}(X)b - y\|_2^2.$$

This is an example of *fully-corrective boosting*. In standing boosting we do the following:

$$\begin{aligned} \widehat{\theta}_{k+1} &= \operatorname*{argmin}_{\theta \in \mathbb{R}} R_k(\theta, \phi_{k+1}) \\ &= \operatorname*{argmin}_{\theta \in \mathbb{R}} \frac{1}{2} \left\| y^k - \theta\phi_{k+1}(X) \right\|_2^2 \\ &= \frac{\phi_{k+1}(X)^T y^k}{\|\phi_{k+1}(X)\|_2^2}, \end{aligned}$$

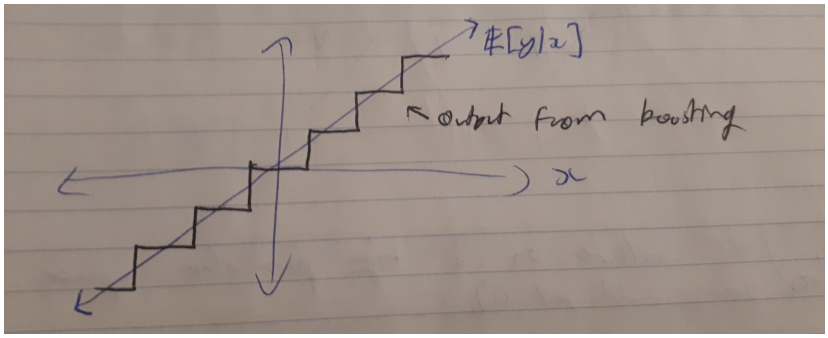
and then set

$$\widehat{\beta}_{k+1} = \begin{bmatrix} \widehat{\beta}_k \\ \widehat{\theta}_{k+1} \end{bmatrix}.$$

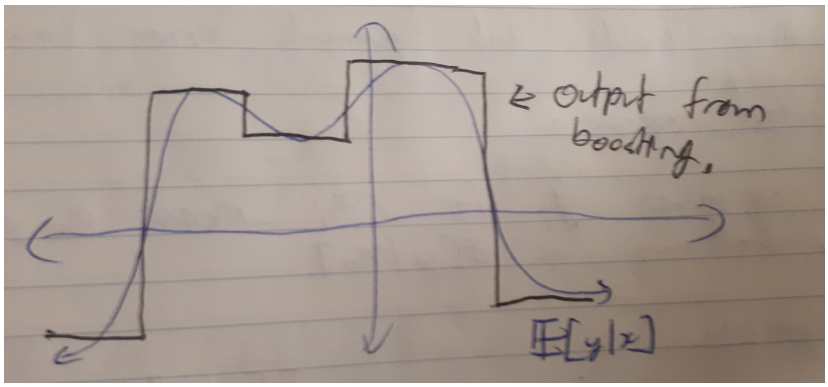
So we do not update/correct the first k coefficients.

2.3 Higher level commentary

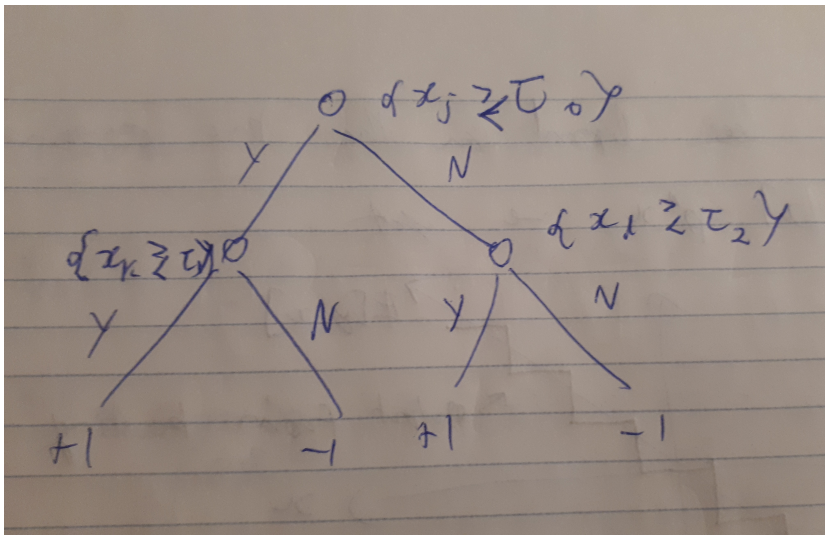
- The collection Φ is often called the weak learner. The choice of Φ matters a lot.
- If we used decision stumps, then we for free fix the scaling of the features and we can get a lot of nice non-linearity.
- The performance of boosting does depend on the "true" relationship between y and x. If the true relationship is linear so $\mathbb{E}[y|x] = \beta^T x$, boosting with decision stumps gives something like:



which is not very good. On the other hand if $\mathbb{E}[y|x]$ is non-linear, then boosting with decision trees works well. We would get results like this:



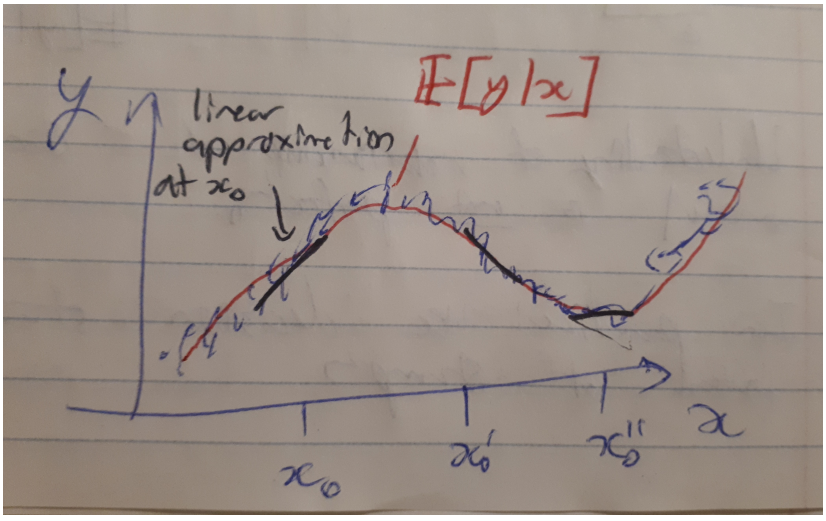
- There isn't much understanding of the relationship between the strength of Φ and the final model.
- In partice we use cas use decision trees (or forests). For decision trees our weak learners are something like:



- One perspective is that boosting is "just" coordinate descent on a very large parameter space.
- What we called <u>version 1</u> is called *gradient boosting* and is implemented in XG-Boost. What we called <u>version 2</u> doesn't really have a name. Version 2 does not generalize to other loss functions as well as version 1. It is just because we have linear models and square error that the calculations were possible.

3 Locally weighted least squares

Idea: use responses y_i for x_i near a given x_0 to make predictions of $\mathbb{E}[y|x_0]$. We want to predict y at x_0 and we will use the idea that locally $\mathbb{E}[y|x]$ might be linear even if globally $\mathbb{E}[y|x]$ is non-linear. This would be the case if our looks something like this:



Our prediction at $x_0 \in \mathbb{R}^d$ will be

$$\widehat{f}(x_0) = a_0 + b_0^T x_0,$$

where (a_0, b_0) minimize

$$\sum_{i=1}^{n} W\left(\frac{\|x_i - x_0\|_2^2}{\omega}\right)(y_i - a - b^T x_i)^2,$$

where $W : \mathbb{R} \to \mathbb{R}_+$ satisfies $W(0) = 1$, $W(t) \leq 1$ for $t \neq 0$. For example maybe $W(t) = \exp(-\frac{1}{2}t^2)$ or $W(t) = (1 - |t|)_+^2$. In general $W(t)$ is some sort of function with a bump at 0 that goes to zero away from 0. The parameter ω is called a bandwidth parameter.

3.1 Fitting and predicting

Suppose we want to make a prediction at x_0. Define $D(x_0) = \text{diag}\left(W\left(\frac{\|x_0 - x_i\|_2^2}{\omega}\right)\right) \in \mathbb{R}^{n \times n}$. Also define $Z = [1, X]$ where

$$X = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix},$$

Let $\theta = \begin{bmatrix} a \\ b \end{bmatrix} \in \mathbb{R}^{1+d}$, then

$$\begin{aligned} \widehat{\theta} &= \underset{\theta}{\text{argmin}}(y - Z\theta)^T D(x_0)(y - Z\theta) \\ &= (Z^T D(x_0) Z)^{-1} Z^T D(x_0) y. \end{aligned}$$

Our prediction at x_0 is thus

$$\widehat{f}(x_0) = \begin{bmatrix} 1 \\ x_0 \end{bmatrix}^T \widehat{\theta} = \begin{bmatrix} 1 \\ x_0 \end{bmatrix}^T (Z^T D(x_0) Z)^{-1} Z^T D(x_0) y.$$

Note, that for predicting at x_i we have

$$\widehat{y}_i = w(x_i)^T y,$$

where

$$w(x_i) = D(x_i) Z (Z^T D(x_i) Z)^{-1} \begin{bmatrix} 1 \\ x_i \end{bmatrix}.$$

Thus $\widehat{y} = Hy$ where

$$H = \begin{bmatrix} w(x_1)^T \\ \vdots \\ w(x_n)^T \end{bmatrix}.$$

The matrix H is symmetric but does not satisfy $H^2 = H$ so H is not a projection matrix. But, since $\widehat{y} = Hy$ is linear in y all of our theory on $\mathbb{E}[\|\widehat{y} - y\|_2^2]$ still goes through.