

# Gradient Boosting Practice: Poisson Response

*David S. Rosenberg*

Suppose we're trying to predict a distribution of count from some input covariates. The simplest distribution in this situation is the Poisson distribution:

$$p(k; \lambda) = \frac{e^{-\lambda} \lambda^k}{k!}$$

on  $k = 0, 1, 2, 3, \dots$   $\lambda \in (0, \infty)$ .

## 1 Linear Conditional Probability Model

- Input:  $x \in \mathbf{R}^d$ .
- Output:  $y \in \{0, 1, 2, \dots\}$
- Data:

$$\mathcal{D} = ((x_1, y_1), \dots, (x_n, y_n)) \in (\mathbf{R}^d \times \{0, 1, 2, \dots\})^n$$

assume is sampled i.i.d. from some distribution  $P_{\mathcal{X} \times \mathcal{Y}}$ .

- Action:  $\lambda \in (0, \infty)$ , where  $\lambda$  is the parameter of a Poisson distribution.

We've got to map input  $x$  to action  $\lambda$  in our action space, which is  $(0, \infty)$ .

$$x \mapsto \underbrace{w^T x}_{\text{score } s \in \mathbf{R}} \mapsto \underbrace{\lambda}_{\in (0, \infty)}$$

To map the score  $s$  into our action space, we could use the transfer function  $\psi(s) = \exp(s)$ . Then

$$\lambda = \exp(w^T x).$$

So if we predict  $\lambda$ , what's the probability of an observed count  $k$  for input vector  $x$ ?

$$\begin{aligned} p(y = k \mid x; w) &= \frac{e^{-\lambda(x)} \lambda(x)^k}{k!} \\ &= \frac{e^{-\exp(w^T x)} [\exp(w^T x)]^k}{k!}. \end{aligned}$$

The conditional likelihood for particular example  $(x_i, y_i)$ , (where  $y_i$  is a count) is

$$p(y = y_i \mid x_i; w) = \frac{e^{-\exp(w^T x_i)} [\exp(w^T x_i)]^{y_i}}{y_i!}.$$

Easier to work with the log:

$$\log p(y = y_i \mid x_i; w) = -\exp(w^T x_i) + y_i w^T x_i - \log(y_i!)$$

What do we need to find to fit this model?  $w$ . our strategy is to use maximum log-likelihood:

$$\begin{aligned} \log L_{\mathcal{D}}(w) &= \log p(\mathcal{D}; w) \\ &= \sum_{i=1}^n \log p(y = y_i \mid x_i; w) \\ &= \sum_{i=1}^n [-\exp(w^T x_i) + y_i w^T x_i - \log(y_i!)] \end{aligned}$$

So find  $w$  maximizing this log-likelihood and we're done. Can use standard gradient based methods.

## 2 Nonlinear approach

In a nonlinear approach, we'll replace the linear score function  $s = w^T x$  with a nonlinear function  $s = f(x)$ :

$$x \mapsto \underbrace{f(x)}_{\text{score } s \in \mathbf{R}} \mapsto \underbrace{\lambda}_{\in (0, \infty)}.$$

Again, we can use the transfer function  $\psi(s) = \exp(s)$ . So

$$\lambda = \exp(f(x)).$$

For score function  $f$ , the probability of  $y_i \mid x_i$  is:

$$p(y = y_i \mid x_i; f) = \frac{e^{-\exp(f(x_i))} [\exp(f(x_i))]^{y_i}}{y_i!}.$$

Easier to work with the log:

$$\log p(y = y_i \mid x_i; f) = -\exp(f(x_i)) + y_i f(x_i) - \log(y_i!)$$

Somehow we want to find a function  $f$  that gives high log-likelihood to our observed data:

$$\log L_{\mathcal{D}}(w) = \sum_{i=1}^n [-\exp(f(x_i)) + y_i f(x_i) - \log(y_i!)]$$

### 3 Gradient Boosting Approach

Let's differentiate  $\log p(y = y_i \mid x_i; f)$  w.r.t.  $f(x_i)$ :

$$\frac{\partial}{\partial f(x_i)} \log p(y = y_i \mid x_i; f) = -\exp(f(x_i)) + y_i$$

Now differentiating the full log-likelihood is

$$\begin{aligned} \frac{\partial}{\partial f(x_i)} [\log L_{\mathcal{D}}(f)] &= \frac{\partial}{\partial f(x_i)} [-\exp(f(x_i)) + y_i f(x_i) - \log(y_i!)] \\ &= -\exp(f(x_i)) + y_i \end{aligned}$$

So optimal unconstrained step direction for changing the vector of evaluations  $\mathbf{f} = (f(x_1), \dots, f(x_n))$  is

$$-\mathbf{g} = (-y_1 + \exp(f(x_1)), \dots, -y_n + \exp(f(x_n)))$$

Fix some base hypothesis space  $\mathcal{H}$  of functions  $h : \mathbf{R}^d \rightarrow \mathbf{R}$ . Then, our actual step direction will be the  $h \in \mathcal{H}$  that best fits  $-\mathbf{g}$  in the least squares sense:

$$\begin{aligned} &\arg \min_{h \in \mathcal{H}} \sum_{i=1}^n (-\mathbf{g}_i - h(x_i))^2 \\ &= \arg \min_{h \in \mathcal{H}} \sum_{i=1}^n ([-y_i + \exp(f(x_i))] - h(x_i))^2 \end{aligned}$$

So to recap:

1. Up to this point, our score function is  $f$ .
2. We want to improve  $f$ .
3. The optimal step direction for  $f(x_i)$  is  $-y_i + \exp(f(x_i))$ . We can evaluate this. It's a real number.
4. So we have a bunch of  $(x_i, -\mathbf{g}_i)$  pairs that we will use regression over  $\mathcal{H}$  to fit.

Then we add something like  $0.1h$  to  $f$  and repeat.