Penalized Cox PH Models

Steve & BB

2020 Mar 17 (Tue)

We describe gradient descent (proximal and accelerated) algorithms for penalised (elastic net) Cox proportional hazard model. We first discuss gradient descent methods and then implement a penalised Cox-PH model with constant covariates. We will then extend these ideas to implement penalised time-dependent Cox PH models. `coxnet` package has already implemented this model using coordinate descent algorithm but does not accommodate time varying covariates.

Introduction

Let T denote the time until an event (e.g., diagnosis date) takes place. However, in survival framework, we do not always observe T and instead, observe $\{t_i, \delta_i, x_i\}_{i=1}^n$, where t_i is the observed (follow-up) failure time, δ_i is an indicator variable which is 1 if the event has occurred during follow-up time, otherwise, $\delta_i = 0$ if the event is censored. x_i is a vector of predictors. If we assume uniquely increasing failure times, $t_1 < t_2 < \cdots t_n$, Cox PH models can be used. The hazard for individual i at time t is defined as

$$h_i(t) = h_0(t) \exp{(x_i^T \beta)}$$

where $h_0(t)$ is the shared baseline hazard and $e^{x_i^T \beta}$ is the relative hazard risk for individual i. The conditional probability that individual i fails at time t_i given that one of the subjects in the risk set R_i failed at time t_i is defined as

$$\frac{\exp{(x_i^T \beta)}}{\sum_{j \in R_i} \exp{(x_j^T \beta)}}.$$

Since the conditional probabilities conditionally independent across different event times, the full likelihood function can be computed by multiplying the individual likelihoods over all failure times

$$L(\beta) = \prod_i \frac{\exp{(x_i^T \beta)}}{\sum_{j \in R_i} \exp{(x_j^T \beta)}}.$$

The likelihood defined above assumes that the observed failure times are unique. We will extend our discussion in other sections to extend it to tied failure times. The denominator in the likelihood can be rewritten as

$$\sum_{j=1}^n Y_j(t_i) \exp{(x_j^T \beta)}$$

where $Y_i(t)$ is an indicator that subject i is at risk ($= 1$) otherwise ($= 0$), at time t. **(I think this definition would be key in implementing time-varying covariates?)**.

Using definition by Kvamme, Borgan, and Scheel (2019), (and let $w_j = \exp(x_j^T \beta)$), we can simplify the Cox PH likelihood as

$$L(\beta) = \prod_i \left(\frac{w_i}{\sum_{j \in R_i} w_j} \right)^{\delta_i}. \tag{1}$$

Denote the relative hazard (probability of failure?) for individual i by w_i, the absolute failure probability for individual i at time t_j is

$$\pi_{ij} = Y_i(t_j) \frac{w_i}{W_j}$$

where $W_i = \sum_{j \in R_i} w_j$ is the total hazard for all individual at risk when individual i fails.

Using 1 and letting $\eta_i = x_i^T \beta$, the partial log-likelihood is defined as

$$\begin{aligned} l(\beta) &= \sum_i^n \delta_i \log w_i - \sum_i^n \delta_i \log W_i \\ &= \sum_i^n \delta_i \eta_i - \sum_i^n \delta_i \log W_i. \end{aligned} \tag{2}$$

The closed form expression for $\hat{\beta}$ can be obtained by setting the derivative of 2 0. In this case, we compute the derivative and then illustrate how to use gradient descent based methods to approximate $\hat{\beta}$. Taking partial derivative of 2 with respect to the kth linear predictor

$$\begin{aligned} \frac{\partial l(\beta)}{\partial \eta_k} &= \delta_k - \sum_i^n \pi_{ki} \delta_i \\ \implies \frac{\partial l(\beta)}{\partial \eta} &= \delta - \mathbf{P}\delta \end{aligned} \tag{3}$$

where $P = \pi_{ij}$ is a $n \times n$ lower triangular matrix. Applying chain rule ($\eta = \mathbf{X}^T \beta$)

$$\frac{\partial l(\beta)}{\partial \beta} = \mathbf{X}^T(\delta - \mathbf{P}\delta) \tag{4}$$

Gradient descent methods

We first review how gradient descent works. Consider the objective function

$$\min_{\beta} \; f(\beta)$$

where f is differentiable, convex and the domain of f is in $\mathbb{R}^n$. To perform gradient descent, choose $\beta^{(0)} \in \mathbb{R}^n$ and then repeat

$$\beta^{(k)} = \beta^{(k-1)} - \gamma_k \nabla f(\beta^{(k-1)}), \;\; k = 1, 2, 3, \cdots$$

Step sizes γ_k can be:

- **Fixed step size**: take $\gamma_k = \gamma \ \forall k = 1, 2, 3, \cdots$. Can potentially lead to divergence issues if the value is too large or too slow if too small.
- **Backtracking line search**: adaptively chooses step size. First fix parameters $0 < \alpha_1 < 1$ and $0 < \alpha_2 \leq 1/2$. Starting with $\gamma = 1$, at each k, and while

  $$f(\beta - \gamma \nabla f(\beta)) > f(\beta) - \alpha_1 \gamma ||\nabla f(\beta)||_2^2$$

  shrink $\gamma = \alpha_1 \gamma$. Else perform the gradient descent update

  $$\beta^+ = \beta - \gamma \nabla f(\beta).$$

For a more comprehensive discussion on these methods, please Tibshirani (2010).

Proximal gradient descent

Consider a composite function

$$f(\beta) = g(\beta) + h(\beta) \tag{5}$$

- g is differentiable and convex, and domain of f is in $\mathbb{R}^n$. h is convex but necessarily differentiable.

If f is not differentiable but $f = g + h$, g is differentiable, Tibshirani (2010) illustrated the *proximal gradient descent* approximation based on proximal mapping defined as

$$\begin{aligned} \text{prox}_{h,\gamma}(\beta) &= \underset{z}{\text{argmin}} \ \frac{1}{2\gamma} ||\beta - z||_2^2 + h(z) \\ &= S_{h\gamma}(\beta) \end{aligned}$$

where $S(x, \lambda) = \text{sgn}(x)(|x| - \lambda)_+$ (Simon et al. 2011).

In the gradient descent update, choose $\beta^{(0)}$ and then repeat

$$\beta^{(k)} = \text{prox}_{h,\gamma}\left(\beta^{(k-1)} - \gamma_k \nabla g(\beta^{(k-1)})\right), \ k = 1, 2, 3, \cdots$$

Accelerated proximal gradient descent

As before, consider the composite function in 5 with the same properties as defined above. Accelerated proximal gradient method chooses initial value $\beta^{(0)} = \beta^{(-1)} \in \mathbb{R}^n$, and then repeats

$$v = \beta^{(k-1)} + \frac{k-2}{k+1}\left(\beta^{(k-1)} - \beta^{(k-2)}\right) \tag{6}$$

$$\beta^{(k)} = \text{prox}_{\gamma}\left(v - \gamma_k \nabla g(v)\right), \ k = 1, 2, 3, \cdots \tag{7}$$

$k = 1$ is simply proximal gradient update while v in 6 carries some "momentum" from the previous iterations.

Penalised proximal gradient descent for a Cox PH model

We easily extend 2 (and subsequently 4) to add the elastic net regularisation term so that the penalised log-likelihood (and subsequently the gradient function) becomes

$$\text{pen } l(\beta)_{\alpha\lambda} = -l(\beta) + \lambda P(\beta)_\alpha$$

where

$$\lambda P(\beta)_\alpha = \lambda \left(\alpha \sum_{i=1}^{p} |\beta_i| + 0.5(1-\alpha) \sum_{i=1}^{p} \beta_i^2 \right) \tag{8}$$

is the penalisation term. Following the result of $\partial \lambda P(\beta)_\alpha / \partial \beta$ by Simon et al. (2011), the proximal gradient descent update becomes

$$\begin{aligned} \beta^{(k)} &= S_{\gamma\alpha\lambda} \left(\beta^{(k-1)} - \gamma \nabla \text{pen } l(\beta)_{\alpha\lambda} \right) \\ &= S_{\gamma\alpha\lambda} \left(\beta^{(k-1)} + \frac{\gamma}{N} \mathbf{X}^T (\delta - \mathbf{P}\delta) - \gamma\lambda(1-\alpha)\beta^{(k-1)} \right) \end{aligned} \tag{9}$$

combines l_1 (lasso) and l_2 (ridge) penalties. The lasso chooses only the nonzero coefficients. Althogh this tends to work in most applications, if two predictors are correlated, it will pick one and completely ignore the other. On the hand, ridge penalises coefficients towards zero, but not to exactly zero. Elastic net penaly, 8 combines the strength of both lasso and rigde.

References

Kvamme, Håvard, Ørnulf Borgan, and Ida Scheel. 2019. "Time-to-Event Prediction with Neural Networks and Cox Regression." *Journal of Machine Learning Research* 20 (129): 1–30.

Simon, Noah, Jerome Friedman, Trevor Hastie, and Rob Tibshirani. 2011. "Regularization Paths for Cox's Proportional Hazards Model via Coordinate Descent." *Journal of Statistical Software* 39 (5): 1.

Tibshirani, Ryan. 2010. "Proximal Gradient Descent and Acceleration." *Lecture Notes.*