ConCR-TMLE R Paper

David Chen, Thomas Gerds, Helene Rytgaard

Abstract

An abstract of less than 150 words.

Introduction

Data Structure

Consider a survival analysis on an interval $[0,\, t_{max}]$ with competing risks. Let T_j^a denote counterfactual time-to-event variables for event j and intervention a, for competing events $j \in \mathcal{J} = \{1, 2, \dots, J\}$ and an intervention $a \in \mathcal{A}$. Our counterfactual data structure can then be denoted by

$$(T_j^a,\, \mathbf{X}\, :\, a \in \mathcal{A},\, j \in \mathcal{J})$$

where $\mathbf{X} \in \mathbb{R}^d$ is a d-dimensional vector of baseline covariates. For a single time-point binary intervention, as in many randomized control trials, $\mathcal{A} = \{0, 1\}$ and the corresponding counterfactual data is

$$(T_j^1,\, T_j^0,\, \mathbf{X}\, :\, j \in \mathcal{J})$$

```
## sample data.table for counterfactual data
```

Let O denote the corresponding coarsened observed data where $O \sim P_0$. The observed data would include the time-to-censoring C, and observed intervention A. The time to first event (censoring or otherwise) we denote as $\widetilde{T} = \min(C,\, T_j\, :\, j \in \mathcal{J})$ with $\Delta = (\operatorname*{argmin}_j T_j) \times \mathbf{1}(\min_j T_j \leq C)$ marking which outcome is observed ($\Delta = 0$ being that censoring occurred). The observable right-censored survival data with competing events can then be represented as

$$O = (\widetilde{T},\, \Delta,\, A,\, \mathbf{X})$$

```
## sample data.table of the observed data correlating to the above
    counterfactual data example
```

This observed data also allows the "long-format" formulation, where with single time-point intervention variable A and baseline covariate vector $\mathbf{X}$,

$$O = (N_j(t),\, N_c(t),\, A,\, \mathbf{X}\, :\, j \in \mathcal{J}, t \leq \widetilde{T})$$

Here $N_j(t) = \mathbf{1}(\widetilde{T} \leq t, \Delta = j)$ and $N_c(t) = \mathbf{1}(\widetilde{T} \leq t, \Delta = 0)$ denote counting processes for event j and censoring respectively.

Under coarsening at random (CAR), the observed data likelihood can be factorized as

$$p(O) = p(\mathbf{X})\, \pi(A \mid \mathbf{X})\, \lambda_c(\widetilde{T} \mid A,\, \mathbf{X})^{\mathbf{1}(\Delta=0)} S_c(\widetilde{T}\text{-} \mid A,\, \mathbf{X}) \prod_{j=1}^{J} S(\widetilde{T}\text{-} \mid A,\, \mathbf{X}) \lambda_j(\widetilde{T} \mid A,\, \mathbf{X})^{\mathbf{1}(\Delta=j)}$$

where $\lambda_c(t \mid A,\, \mathbf{X})$ is the hazard of the censoring process and $\lambda_j(t \mid A,\, \mathbf{X})$ is the hazard of the j^{th} event process. Additionally

$$S_c(t \mid a,\, \mathbf{x}) = \exp\left(-\int_0^t \lambda_c(s \mid a,\, \mathbf{x})\, ds\right)$$

while in a pure competing risks setting

$$S(t \mid a,\, \mathbf{x}) = \exp\left(-\int_0^t \sum_{j=1}^{J} \lambda_j(s \mid a,\, \mathbf{x})\, ds\right)$$

and

$$\begin{aligned} F_j(t \mid a,\, \mathbf{x}) &= \int_0^t S(s\text{-} \mid a,\, \mathbf{x})\lambda_j(s \mid a,\, \mathbf{x})\, ds \\ &= \int_0^t \exp\Big(-\int_0^s \sum_{j=1}^{J} \lambda_j(u \mid a,\, \mathbf{x})\, du\Big)\lambda_j(s \mid a,\, \mathbf{x})\, ds. \end{aligned}$$

Target Parameter

For a target parameter of the treatment regime a^*, cause $k \in J$ cumulative risk at time τ

$$\Psi_{a^*,k,\tau}(P_0) = \mathbb{E}\left[F_k(\tau \mid A = a^*,\, \mathbf{X})\right]$$

the corresponding efficient influence function $D^*_{a^*,k,\tau}(P)(O)$ is

$$\begin{aligned} &\sum_{j=1}^{J} \int_0^\tau \frac{\mathbf{1}(A = a^*)\,\mathbf{1}(s \leq \tau)}{\pi(A \mid \mathbf{X})\, S_c(s\text{-} \mid A,\, \mathbf{X})} \left(\mathbf{1}(\delta = k) - \frac{F_k(\tau \mid A,\, \mathbf{X}) - F_k(s \mid A,\, \mathbf{X})}{S(s \mid A,\, \mathbf{X})}\right) \\ &\qquad \Big(N_j(ds) - \mathbf{1}(\widetilde{T} \geq s)\, \lambda_j(s \mid A,\, \mathbf{X})\Big)\; ds \\ &\qquad\qquad + F_k(t \mid A = a^*,\, \mathbf{X}) - \Psi_{a^*,k,\tau}(P_0) \end{aligned}$$

with a clever covariate $h_{a^*,k,j,\tau,s}$

$$h_{a^*,\,k,\,j,\,\tau,\,s} = \frac{\mathbf{1}(A = a^*)\,\mathbf{1}(s \leq \tau)}{\pi(A \mid \mathbf{X}) S_c(s\text{-} \mid A,\, \mathbf{X})} \left(\mathbf{1}(\delta = k) - \frac{F_k(\tau \mid A,\, \mathbf{X}) - F_k(s \mid A,\, \mathbf{X})}{S(s \mid A,\, \mathbf{X})}\right)$$

The components of the data distribution that must be estimated are $g(A \mid \mathbf{X})$ and $S_c(t \mid A,\, \mathbf{X})$, $\lambda_j(t \mid A,\, \mathbf{X})$, $F_j(t \mid A,\, \mathbf{X})$, and $S(t \mid A,\, \mathbf{X})$

Estimation

Cross-Validation Specification

Let $D_n = \{O_i\}_{i=1}^n$ be an observed sample of n i.i.d observations of $O \sim P_0$. For V-fold cross validation, let $B_n = \{1, ..., V\}^n$ be a random vector that assigns the n observations into V validation folds. For each $v \in \{1, ..., V\}$ we then define training set $D_v^{\mathcal{T}} = \{O_i : B_n(i) = v\}$ with the corresponding validation set $D_v^{\mathcal{V}} = \{O_i : B_n(i) \neq v\}$.

Stratified Cross-Validation

```
# StrataIDs <- as.numeric(factor(paste0(Data[["Trt"]], ":", Data[["
    Event"]])))
# CVFolds <- origami::make_folds(n = Data, fold_fun = origami::folds_
    vfold, strata_ids = StrataIDs)
```

Propensity Score Estimation

For the conditional distribution of A given $\mathbf{X}$, $\pi(\cdot \mid \mathbf{X})$, and $\hat{\pi} : D_n \to \hat{\pi}(D_n)$, let L_π be a loss function such that the risk $\mathbb{E}_0 \left[L_\pi(\hat{\pi}, O)\right]$ is minimized when $\hat{\pi} = \pi_0$. For instance, with a binary A, we may specify the negative log loss $L_\pi(\hat{\pi}, O) =$ -$\log\left(\hat{\pi}(1 \mid \mathbf{X})^A \, \hat{\pi}(0 \mid \mathbf{X})\right)^{1-A}\big)$. Let $\mathcal{M}_\pi$ be the set of candidate propensity score models. The discrete superlearner selector chooses the candidate propensity score model with the minimal cross validated risk

$$\hat{\pi}^{SL} = \underset{\hat{\pi} \in \mathcal{M}_\pi}{\text{argmin}} \sum_{v=1}^{V} P_{D_v^{\mathcal{V}}} \; L_\pi(\hat{\pi}(D_v^{\mathcal{T}}), D_v^{\mathcal{V}})$$

```
getInitialEstimates <- function(...) {
  getPropScores <- function(...) {
    getSl3PropScores <- function {
      TrtTask <- sl3::make_sl3_Task(
       data = Data[, -c("Time", "Event", "ID")],
       covariates = colnames(CovDataTable),
       outcome = "Trt")

   TrtSL <- sl3::Lrnr_sl$new(learners = Models[["A"]], folds =
    CVFolds)
   TrtFit <- TrtSL$train(TrtTask)
    }
  }
}
```