Super-learner of a multinomial conditional distribution

Suppose we want to construct a super-learner of the conditional probability distribution $g_0(a \mid W) = P_0(A = a \mid W)$, where $a \in \mathcal{A}$. Let's denote the values of a with $\{0, 1, \ldots, K\}$. A valid loss function for the conditional density is

$$L(g)(O) = -\log g(A \mid W).$$

That is, $g_0 = \arg\min_g P_0 L(g)$, i.e., g_0 is the minimizer of the expectation of the log-likelihood loss.

Let $\hat{g}_k(P_n)$, $k = 1, \ldots, K$, be a collection of candidate estimators of g_0. The discrete super-learner is defined by

$$g_n = \hat{g}_{k_n}(P_n),$$

where

$$k_n = \arg\min_k E_{B_n} P^1_{n,B_n} L(\hat{g}(P^0_{n,B_n})) = E_{B_n} \frac{1}{np} \sum_{i:B_n(i)=1} L(\hat{g}(P^0_{n,B_n}))(O_i),$$

and $B_n \in \{0,1\}^n$ is a random sample split in training sample $\{i : B_n(i) = 0\}$ and validation sample $\{i : B_n(i) = 1\}$. Here p is the proportion of observations that are in the validation sample, P^1_{n,B_n}, P^0_{n,B_n} are the empirical probability distributions of the validation and training sample, respectively.

We can also define a parametric family of candidate estimators $\hat{g}_\alpha(P_n)$, indexed by a vector α, such as

$$\hat{g}_\alpha = \sum_{k=1}^{K} \alpha(k)\hat{g}_k$$

under the constraint that $\alpha(k) \geq 0$, $k = 1, \ldots, K$, and $\sum_k \alpha(k) = 1$. This choice of family is contained in the class of probability distributions. The super-learner for this family of candidate estimators is given by

$$g_n = \hat{g}_{\alpha_n}(P_n),$$

where

$$\alpha_n = \arg\min_\alpha E_{B_n} P^1_{n,B_n} L(\hat{g}_\alpha(P^0_{n,B_n})).$$

One might have to program this optimization over α, but existing routines are available for doing such constrained maximization problems. This step is often referred to as the meta-learner step.

Candidate estimators:

Candidate estimators based on multinomial logistic regression: Let's now discuss how one might construct a library of candidate estimators $\hat{g}_k$. To start with one can use existing parametric model based MLE and machine learning algorithms in R that fit a multinomial regression.
Candidate estimators based on machine learning for multinomial logistic regression: Secondly, one can use a machine learning algorithm such as polyclass in R that data adaptively fits a multinomial logistic regression, which itself has tuning parameters, again generating a class of candidate estimators. Note that one can also marry any of these choices with a screening algorithm, thereby creating more candidate estimators of interest. The screening can be particularly important when there are many variables.

Candidate estimators by fitting separate logistic regressions and using post-normalization:

Another easy way to construct candidate estimators is to code A in terms of Bernoullis $B_k = I(A = k)$, $k = 0, \ldots, K$. One could construct an estimator $\bar{g}_{0k}(W) \equiv P_0(B_k = 1 \mid W)$ using any of the logistic regression algorithms, for all $k = 0, \ldots, K$. Let's denote such an estimator with $\bar{g}_{nk}$. Then, this implies an estimator

$$g_n(a \mid W) = \frac{\bar{g}_{na}(W)}{\sum_{k=0}^{K} \bar{g}_{nk}(W)}.$$

In other words, we simply normalize these separate logistic regression estimators so that we obtain a valid conditional distribution. Again, this now generates an enormous amount of interesting algorithms, since we have available the whole machine learning literature for binary outcome regression.

Candidate estimators by estimating the conditional hazard with pooled logistic regression

Finally, we have used the following strategy in our research for construction of candidate estimators. Note that

$$g_0(a \mid W) = \lambda_0(a \mid W) S_0(a \mid W),$$

where

$$\lambda_0(a \mid W) = P_0(A = a \mid A \geq a, W),$$

and $S_0(a \mid W) = \prod_{s \leq a}(1 - \lambda_0(s \mid W))$ is the conditional survivor function $P_0(A > a \mid W)$. So we have now parameterized the conditional distribution of A, given W, by a conditional hazard $\lambda_0(a \mid W)$: $g_0 = g_{\lambda_0}$.

We could now focus on constructing candidate estimators of $\lambda_0(a \mid W)$, which implies candidate estimators of g_0.
For every observation A_i, we can create $A_i + 1$ rows of data $(W, s, I(A_i = s))$, $s = 0, \ldots, A_i$, $i = 1, \ldots, n$. We now run a logistic regression estimator based on the pooled data set, ignoring ID, where we regress the binary outcome $I(A_i = s)$ on the covariates (W, s).
If one assumes a parametric model, then this is nothing else then using the maximum likelihood estimator, demonstrating that ignoring the ID is not inefficient. This defines now an estimator of $\lambda_0(s \mid W) = P_0(A = s \mid W, A \geq s)$ as a function of (s, W). Different choices of logistic regression based estimators will define different estimators.
The pooling across s is not very sensible if A is not an ordered variable. If A is categorical, we recommend to compute a separate logistic regression estimator of $\lambda_0(a \mid W)$ for each a (i.e., stratify by s in the above pooled data set).
In this manner, we can create an extensive library of candidate estimators.

Outline

Loss-based dissimilarity

Let $L(\psi)(O)$ be a loss function for $\psi_0 = \arg\min_\psi \int L(\psi)(o)dP_0(o)$. We can define a loss-based dissimilarity between a candidate ψ and true parameter value ψ_0:

$$d_0(\psi, \psi_0) = P_0 L(\psi) - P_0 L(\psi_0) = \int_o L(\psi)(o)dP_0(o) - \int_o L(\psi_0)(o)dP_0(o).$$

Cross-validation selector

Given a library of candidate estimator mappings $P_n \to \hat{\Psi}_k(P_n)$, $k = 1, \ldots, K_n$, we will define a cross-validation selector of k. Consider a V-fold cross-validation scheme that defines V sample splits in training and validation sample. For each sample split v, let $P_{n,v}$ be the empirical probability distribution of the training sample, and let $Val(v)$ be the set of indices that are in the validation sample. The cross-validation selector is defined by

$$k_n = \arg\min_k \frac{1}{V} \sum_{v=1}^{V} \frac{1}{np} \sum_{i \in Val(v)} L(\hat{\Psi}_k(P_{n,v}))(O_i).$$

Discrete super-learner

The discrete super-learner is defined as the estimator

$$\hat{\Psi}(P_n) = \hat{\Psi}_{k_n}(P_n).$$

Oracle inequality for quadratic loss-based dissimilarities

Suppose that $\sup_{\psi \in \text{-}, o} \mid L(\psi) - L(\psi_0) \mid (o) < M_1$ and

$$\sup_{\psi \in \text{-}} \frac{P_0 \{L(\psi) - L(\psi_0)\}^2}{P_0 L(\psi) - P_0 L(\psi_0)} < M_2.$$

Let $p = 1/V$, and $C(M_1, M_2, \delta) = O(M_1 + M_2/\delta)$. Then, for each $\delta > 0$, we have

$$E_0 \frac{1}{V} \sum_v d_0(\hat{\Psi}_{k_n}(P_{n,v}), \psi_0) \quad \leq \quad (1+\delta) E_0 \min_k \frac{1}{V} \sum_v d_0(\hat{\Psi}_k(P_{n,v}), \psi_0) + C(M_1, M_2, \delta) \frac{\log K_n}{np}$$

Asymptotic equivalence of cross-validation selector and oracle selector

Suppose that

$$\frac{\log K_n / n}{E_0 \min_k \frac{1}{V} \sum_v d_0(\hat{\Psi}_k(P_{n,v}), \psi_0)} \to 0.$$

Then,

$$\frac{E_0 \frac{1}{V} \sum_v d_0(\hat{\Psi}_{k_n}(P_{n,v}), \psi_0)}{E_0 \min_k \frac{1}{V} \sum_v d_0(\hat{\Psi}_k(P_{n,v}), \psi_0)} \to 1.$$

In words, if $K_n = n^m$ for some finite m, and the oracle selected estimator converges at a slower rate than $\log n/n$ (i.e., rate for a correctly specified parametric model), then the ratio of the dissimilarity of the cross-validated selected estimator and the truth and the dissimilarity of the oracle selected estimator and the truth converges to 1.
If, one of the candidate estimators happens to be based on a correctly specified parametric model, then the dissimilarity of the cross-validated selected estimator and the truth converges at rate $\log n/n$.

Oracle inequality for non-quadratic loss-based dissimilarities

Suppose that $\sup_{\psi \in \text{-}, o} \mid L(\psi) - L(\psi_0) \mid (o) < M_1$. Let $p = 1/V$ and $C(M_1) = O(M_1)$. Then, for each $\delta > 0$, we have

$$E_0 \frac{1}{V} \sum_v d_0(\hat{\Psi}_{k_n}(P_{n,v}), \psi_0) \quad \leq \quad (1+\delta) E_0 \min_k \frac{1}{V} \sum_v d_0(\hat{\Psi}_k(P_{n,v}), \psi_0)$$
$$C(M_1 \frac{(\log K_n)^{0.5}}{(np)^{0.5}}.$$

Causal effect of treatment on right-censored survival time

Let $O = (W, A, \Delta = I(T \leq C), \tilde{T} = \min(T, C))$, T is a survival time, C is a right-censoring time, A is a binary treatment, W are baseline covariates. Suppose that C is independent of T, given A, W. Consider the conditional survivor function:

$$S(t_0 \mid A, W) = P(T > t_0 \mid A, W).$$

Let

$$\lambda(t \mid A, W) = P(T = t \mid A, W, T \geq t)$$

be the conditional hazard of survival at time t. We have

$$S(t_0 \mid A, W) = \prod_{s \leq t_0} (1 - \lambda(s \mid A, W)).$$

If T is continuous, then this writes as

$$S(t_0 \mid A, W) = \prod_{s \leq t_0} (1 - d\Lambda(s \mid A, W)).$$

If C is independent of T, given A, W, then we can identify the conditional hazard as follows:

$$\lambda(t \mid A, W) = P(\tilde{T} = t, \Delta = 1 \mid \tilde{T} \geq t, A, W).$$

Let's denote the right-hand side with $\tilde{\lambda}(t \mid A, W)$. Define

$$\Psi_a(P) = E_P S(t_0 \mid A = a, W).$$

Under a causal model in which $T = T_A$, T_0, T_1 treatment specific counterfactual survival times, and the assumption that A is independent of T_0, T_1, given W, we have

$$\Psi_a(P) = P(T_a > t_0),$$

the counterfactual survival function at t_0 under intervention $A = a$.

Prediction of survival based on right-censored data

Suppose that we want to estimate $\psi_0(t_0, A, W) = S_0(t_0 \mid A, W)$ at a given point t_0, based on the right-censored data structure $O = (W, A, \Delta, \tilde{T})$. If there would not be right-censoring, then a valid loss function would be

$$L^F(\psi)(W, A, T) = (I(T > t_0) - \psi(t_0, A, W))^2.$$

Let $\bar{G}(t \mid A, W) = P(C \geq t \mid A, W)$. We can define the Inverse probability of censoring weighted loss function:

$$L_{G_0}(\psi)(O) = \frac{L^F(\psi)(W, A, \tilde{T})\Delta}{\bar{G}(\tilde{T} \mid A, W)}.$$

An improved IPCW-loss is given by:

$$L_{G_0}(\psi)(O) = \frac{L^F(\psi)(W, A, \tilde{T})\{I(\tilde{T} \leq t_0, \Delta = 1) + I(\tilde{T} > t_0)\}}{\bar{G}_0(\min(\tilde{T}, t_0) \mid A, W)}.$$

The cross-validation selector is now defined by:

$$k_n = \arg\min_k \frac{1}{V} \sum_v \sum_{i \in VAL(v)} L_{G_{n,v}}(\hat{\Psi}_k(P_{n,v}))(O_i),$$

where $G_{n,v}$ is an estimator of G_0 based on the training sample $P_{n,v}$.