Super Learner for survival prediction from censored data: definition, simulations, and application

Yohann Foucher

https://github.com/foucher-y/talks

SESSTIM webconference, December 19, 2025

Plan

The problematic of the method choice for survival prediction

- The prediction of the probability that a subject experienced an event is often of interest.
- Several regressions can be used for right-censored data.:
  - Most of the studies use proportional hazard (PH)-based assumption.
  - Other models such as accelerated failure time (AFT) approaches are not frequent.
- Regressions are based on assumption such as log-linearity, PH, specific interactions, etc.
- Machine learning are increasingly used mainly because of their flexibility.
  - Random survival forests.
  - Survival neural networks.
  - Support-vector machines.
  - Etc.

A super learner (SL) allows us to combine regressions and algorithms.

- SL is an algorithm that uses cross-validation to estimate the performance of multiple machine learning models.
- It has been proven to be asymptotically as accurate as the best possible prediction algorithm that is tested.
- In 2011, Polley and van der Laan proposed a SL for right-censored data.
- Two R packages were available :
  - The first one was proposed by Golmakani et al. (2020). It allows us to obtain the linear predictor of a PH regression.
  - The second one was developed by Westling et al. (2021) with additional learners: several parametric PH models, a generalized additive Cox regression, and a random survival forest.
- **We aimed to extend these packages to additional learners and loss functions.**

Plan

Notations

- Consider a subject i in a sample of N independent subjects ($i = 1, ..., N$).
- We note the time-to-event by $T_i^\star$.
- Right-censoring leads the observation of $T_i = \min(T_i^\star, C_i)$, where C_i is the censoring time.
- Let $D_i = \mathbb{I}\{T_i^\star \leq C_i\}$ be the event indicator.
- The survival function at time t for a subject with the characteristics Z_i at baseline is defined by:

$$S(t \mid Z_i) = \mathbb{P}(T_i^\star > t \mid Z_i)$$

The SL is estimated by minimizing the cross-validated loss function.

- $S_m(.)$ is the survival function obtained by the m^{th} learner ($m = 1, ..., M$).
- w_m is the corresponding weight with respect to $\sum_1^M w_m = 1$ and $0 \leq w_m \leq 1$.
- The sample is randomly divided into V cross-validated sub-samples.
- For each of the folds, one can:
  - Estimate the M learners from the training subjects.
  - Tuning parameters should be tuned for each of the V folds.
  - Predict $\tilde{S}_m(.)$ of the leaving subjects.
- The weights $\hat{w}_m$ are then obtained by minimizing the loss function, i.e., distance between the observations and the predictions $\tilde{S}_{sl}(t \mid Z) = \sum_{m=1}^M w_m \tilde{S}_m(t \mid Z)$.
- The final SL is obtained by $\hat{S}_{sl}(t \mid Z) = \sum_{m=1}^M \hat{w}_m \hat{S}_m(t \mid Z)$, where $\hat{S}_m(t \mid Z)$ are estimated on the entire sample.

The implemented models without tuning parameters

- Parametric AFT models (Weibull, Gamma and generalized Gamma distributions).
  - Parameters are estimated by likelihood maximization by using the "flexsurv" package.
- Parametric PH models (Exponential or Gompertz distributions).
  - Parameters are estimated by likelihood maximization by using the "flexsurv" package.
- Semi-parametric PH models with a non-parametric baseline hazard function estimated by using the Breslow estimator
  - Parameters are estimated by likelihood maximization by using the "survival" package.
  - Option for covariates selection by forward AIC-based selection.
  - Non-parametric baseline hazard function estimated by using the Breslow estimator.

The implemented models/algorithms with tuning parameters

- Spline-based PH model proposed by Royston and Parmar.
  - Parameters are estimated by likelihood maximization by using the "flexsurv" package.
- Penalized PH models (Lasso, Ridge, or Elastic-Net).
  - Parameters are estimated by penalized likelihood maximization by using the "glmnet" package.
- Random survival forests.
  - Maximization of the Log-Rank statistic by using the "randomSRC" package.
- Survival neural networks.
  - Maximization of the entropy of a partial logistic regression approach "survivalPLANN" package.
  - https://github.com/chupverse/survivalPLANN

The implemented loss functions.

- The log-Likelihood.
- The area under the time-dependant ROC curve up to time t.
- The Pencina's concordance index at time t.
- The Uno's concordance index at time t.
- The Brier Score (BS) for right-censored data and a prediction at time t.
- The negative binomial log-likelihood (BLL) for a prognostic at time t.
- The integrated BS and BLL up to the maximum follow-up time.
- The restricted integrated BS and BLL up to a time t.

Plan

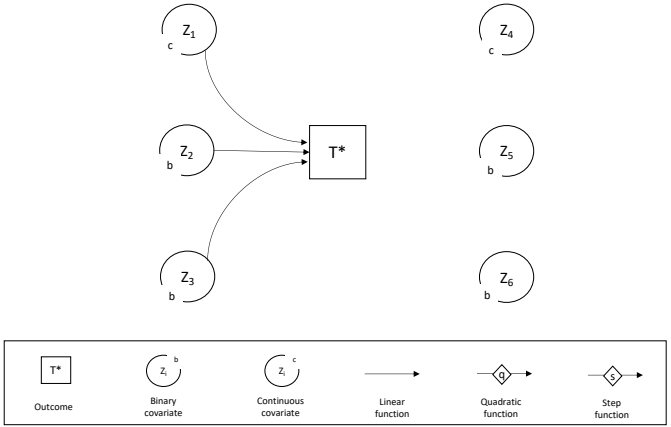
The objectives of the simulations

- We principally aim to describe the performances of our proposed package.
- We also compared our SL with the package proposed by Westling (2021).
- In the SL proposed by Westling we included the 5 possible learners: the PH model with the Breslow estimator, the Exponential PH model, the random survival forest, the Lasso PH model and the PH model with covariates' selection ($p < 0.05$).
- In our proposed SL, we included the 7 learners: the PH model with the Breslow estimator, the same model with forward selection based on AIC minimization, the Elastic-Net PH model, the random survival forest, the Exponential PH model, the Gamma distribution-based AFT model, and the survival neural network.
- We used the default grid of the tuning parameters.
- The SL weights were estimated by minimizing the integrated BS (IBS).

The data generation

- We generated 1000 data sets for each scenario.
- The times-to-event were obtained from Weibull distributions and the PH assumption.
- The censoring times were generated from uniform distributions to obtain a 40% censoring rate.
- We studied two sample sizes for learning (200 and 500).
- The validation samples were composed of 500 subjects.
- We proposed two contrasting simple and complex scenarios.

The design of the simple scenario



Simple scenario: the SL performed as well as semi-parametric approaches

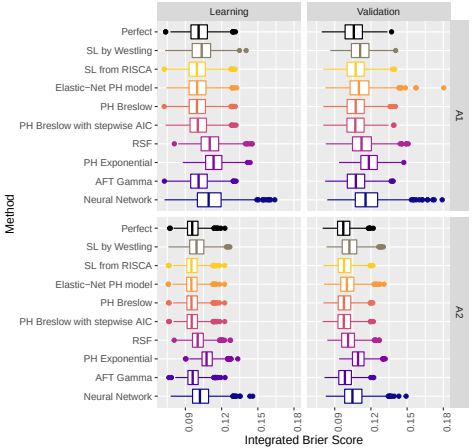
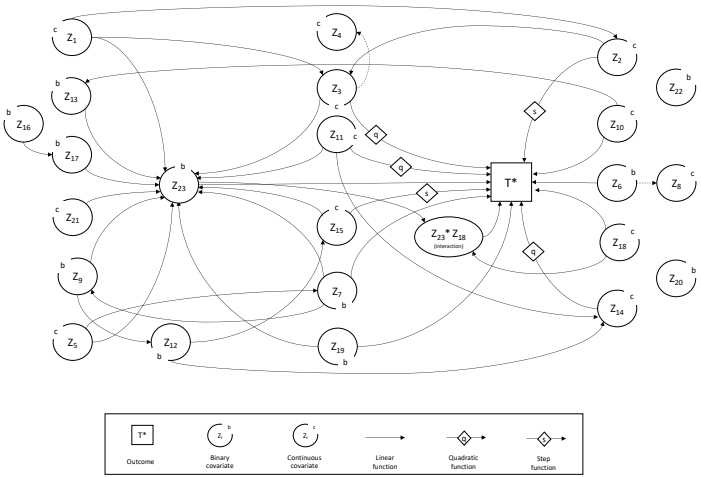


Figure 1: Simulation results in the simple context. A1-$N = 200$ for learning in the two top plots. A2-$N = 500$ for learning in the two bottom plots.

The design of the complex scenario



Complex scenario: the SL performed the best

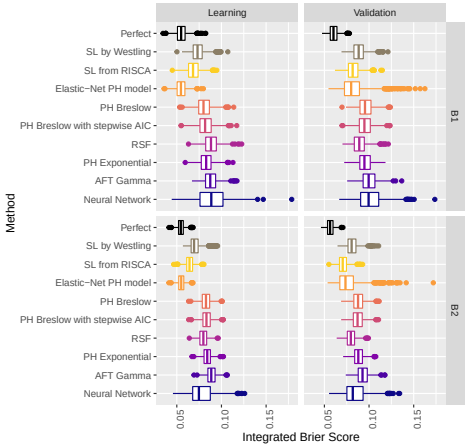


Figure 2: Simulation results in the complex context. A1-$N = 200$ for learning in the two top plots. A2-$N = 500$ for learning in the two bottom plots.

Complex scenario: running times

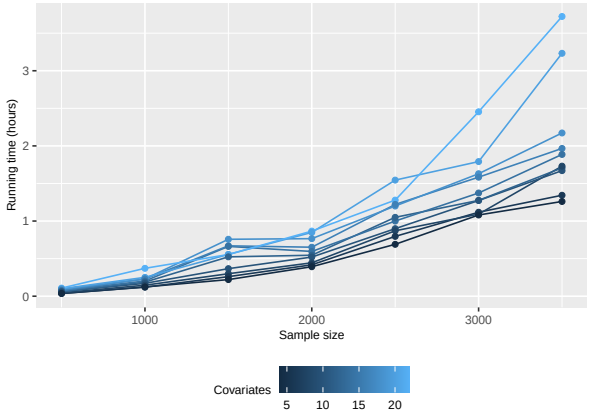


Figure 3: Running times according to the sample size and the number of covariates. Results obtained for a MacBook Pro 2.6 GHz Intel Core i7 6 cores.

Plan

Objectives & data

- To construct an algorithm for predicting the time to disease progression after the initiation of a first-line treatment (baseline $T^\star = 0$).
- To compare its prognostic capacities up to 2 years with those of the existing Rio score .
- A data set 1,300 simulated patients from the OFSEP cohort.

```
library("survivalSL")
data(dataOFSEP); head(dataOFSEP)

#> time event age duration period gender relapse edss t1 t2 rio
#>  2.1195051 1 34 1113 0 1 1  low   0 0 1
#>  0.8160995 1 37 1296 0 1 1  high 1+ 0 1
#>  1.9709546 1 33  995 1 1 0  low   0 0 0
#>  2.5881311 1 35  858 1 1 0  low   0 0 0
#>  1.4726224 1 31  759 1 0 2+ miss 1+ 0 2
#>  1.6970962 1 40 1642 1 1 0  high  0 0 0
```

Data management...

```
# codage of the predictors
dataOFSEP$relapse.1 <- 1*(dataOFSEP$relapse=="1")
dataOFSEP$relapse.2 <- 1*(dataOFSEP$relapse=="2+")
dataOFSEP$edss.1 <- 1*(dataOFSEP$edss=="low")
dataOFSEP$edss.2 <- 1*(dataOFSEP$edss=="high")
dataOFSEP$t1.1 <- 1*(dataOFSEP$t1=="0")
dataOFSEP$t1.2 <- 1*(dataOFSEP$t1=="1+")

#two-thirds of the sample to train and other third validation
set.seed(117)
dataOFSEP$train <- 1*rbinom(n=dim(dataOFSEP)[1], size=1, prob=2/3)
dataTRAIN <- dataOFSEP[dataOFSEP$train==1,]
dataVALID <- dataOFSEP[dataOFSEP$train==0,]
```

The SL training

- Learners:
  - Elastic-Net PH model with non-parametric Breslow hazard function,.
  - Royston-Parmar Spline-based model.
  - AFT model with generalized Gamma distribution.
  - Random survival forest.
- We used the default grids to estimate the tuning parameters.
- The weights were estimated to minimize the C-index at 2 years with a 30-fold CV.

```
.f <- Surv(time, event) ~ age + duration + period + gender +
relapse.1 + relapse.2 + edss.1 + edss.2 + t1.1 + t1.2

sl1 <- survivalSL(formula=.f, metric="uno_ci", data=dataTRAIN,
methods=c("LIB_COXen", "LIB_PHspline", "LIB_AFTggamma", "LIB_RSF"),
cv=30, optim.local.min=TRUE, show_progress=FALSE, seed=117)
```

The contribution of the learners

```
print(sl1, digits=4)

#> leaners weights
#> 1 LIB_COXen 0.1775
#> 2 LIB_PHspline 0.2519
#> 3 LIB_AFTggamma 0.3039
#> 4 LIB_RSF 0.2667

#> Minimum of the 30-fold CV of the metric uno_ci:0.6851.
```

Your own tuning grid

```
.tune <- vector("list",4)
.tune[[2]] <- list(k=1:6) # number of knots

sl2 <- survivalSL(formula=.f, metric="uno_ci", data=dataTRAIN,
methods=c("LIB_COXen", "LIB_PHspline", "LIB_AFTggamma", "LIB_RSF"),
cv=30, optim.local.min=TRUE, param.tune=.tune, show_progress=FALSE,
seed=117)

print(sl2, digits=4)
#> leaners weights
#> 1 LIB_COXen 0.2486
#> 2 LIB_PHspline 0.3340
#> 3 LIB_AFTggamma 0.1773
#> 4 LIB_RSF 0.2401

#> Minimum of the 30-fold CV of the metric uno_ci:0.684.
```

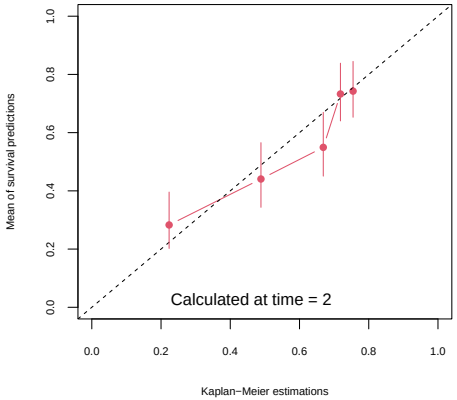
Predictive metrics from the training and validation samples

```
rbind(train = summary(sl2, method="sl", pro.time=2, digits=2)$metrics,
test = summary(sl2, newdata=dataVALID, method="sl", pro.time=2, digits
    =2)$metrics)

#> p_ci uno_ci auc bs ibs ribs bll ibll ribll ll
#> train 0.7463 0.7444 0.8056 0.1838 0.0813 0.0840 0.5487 NaN NaN
    -465.7561
#> test 0.6737 0.6713 0.7036 0.2170 0.0924 0.0969 0.6232 NaN NaN
    -595.7444
```

Calibration plot at 2 years from the validation sample

```
plot(sl2, newdata=dataVALID, cex.lab=0.70,cex.axis=0.70, n.groups=5,
    pro.time=2, col=2)
```



Discriminative capacities up to 2 years from the validation sample

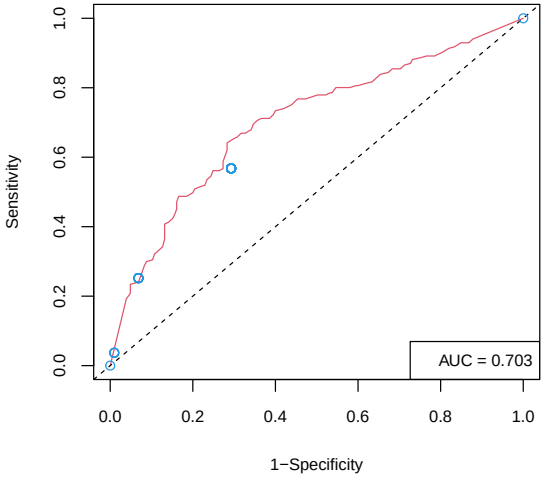
```
library("RISCA")
.pred <- predict(sl2, newdata=dataVALID)
dataVALID$sl <- 1 - .pred$predictions$sl[,sum(.pred$times<2)]

roc.sl <- roc.time(times="time", failures="event", variable="sl",
    confounders=~1, data=dataVALID, pro.time=2,precision=seq(0.1, 0.9,
     by=0.01))

roc.rio <- roc.time(times="time", failures="event", variable="rio",
    confounders=~1, data=dataVALID, pro.time=2)

plot(roc.sl, col=2, type="l", xlab="1-Specificity", ylab="Sensitivity"
    ,cex.lab=0.8, cex.axis=0.8)
points(x=1-roc.rio$table$sp, y=roc.rio$table$se, col=4)
legend("bottomright", legend=
paste("AUC =", round(roc.sl$auc, 3)), cex=0.8 )
```

Discriminative capacities up to 2 years from the validation sample



Plan

Conclusions

- Compared to the available R-based SL in this context of survival analysis, our proposition allows a larger set of candidate learners and loss functions.
- The simulation study showed that our proposed functions performed well.
- The R package is available at:

  https://cran.r-project.org/web/packages/survivalSL/index.html
- Bug reports on the Github:

  https://github.com/chupverse/survivalSL/issues
- Full open paper for details:

  https://journal.r-project.org/articles/RJ-2024-037

Acknowledgements

- Camille Sabathé (Post-doctorate position at Inserm U1246, Nantes University).
- Amina Belkebir Mekki (PhD student, CIC Inserm 1402, Poitiers University).
- Thomas Ollard (PhD student, CIC Inserm 1402, Poitiers University).