THE CURIOSITY CUP 2022
A Global SAS® Student Competition

Building Survival Decision Trees with SAS Software

Vasilev Iulii and Ivan Lazuhin,

Lomonosov Moscow State University – Team "Survivors"

ABSTRACT

This article is devoted to the research and development of survival decision trees with weighted log-rank split search criteria (Vasilev & Petrovskiy, 2022) on the SAS analytical platform, as well as to the construction of random forest ensembles based on such survival trees. The code is implemented as a set of SAS macros using the SAS Base language, SAS/STAT procedures, SAS SQL and the capabilities of the SAS/CONNECT package for parallel execution of the optimal partition search and parallel growing of trees in the ensemble (Vasilev I. , 2021). In contrast to the classical survival analysis approach based on Cox proportional hazards method implemented by PROC PHREG (Allison, 2010), the proposed technique allows one to work with categorical values and missing data directly, automatically find and visualize optimal patient strata with similar survival functions inside the group and different survival functions among different groups. Moreover, bagging ensembles based on the proposed survival trees significantly outperform classical regression methods. The proposed methods were developed and tested on real data of COVID patients in Moscow. Since this dataset is not public, the article presents the experimental results on the open dataset of Wuhan patients (Yan, 2020), where the proposed methods also show good results.

INTRODUCTION

Since the start of COVID-19 worldwide pandemic in March 2020, Systems Science and Engineering (CSSE) at Johns Hopkins University has recorded more than 350 million cases of COVID-19 infection and more than 5 million deaths as of January 2022. The daily increase in cases leads to an increase in the overheads of doctors and medical equipment and a decrease in the quality of the patients' treatment. Depending on the patient's condition, the exposed patient continues home or ambulance treatment. The problem is to evaluate the time and probability of the target (terminate) event for each patient, and in case of emergency, when the predicted survival time is short it is necessary to hospitalize the patient as soon as possible. Similar problem for already hospitalized patients is to estimate their expected time at hospital to optimize costs and prescribe the best treatment. The survival analysis helps to solve these types of problems. It includes a set of statistical methods for estimating the time-to-event and/or the probability of the event occurring. Traditional statistical methods need complete data that may not be available, or the time of the event may be unknown (the event didn't occur before the end of the study, or an observation was lost). In this case, events are called censored. The problems studied with the help of survival analysis are formulated in terms of survival S(t) and hazard h(t) functions (Kaplan, 1958). The survival function determines the probability of the event that has not yet occurred by time t. The hazard function is a conditional density, given that the event in question has not yet occurred prior to time t.

There are several ways to estimate the survival function. The popular nonparametric method is the Kaplan-Meier estimator. It estimates S(t) as the multiplication of all fractions of non-occurring events for each point in time before time t. In real-life problems, we have data with covariates, and we can assume that the survival function depends on them. The Cox

proportional hazards (Cox, 1972) model (Cox PH) is one of the most popular models for taking features into account. The idea behind the model is that the log-hazard of an individual is a linear function of their features, and a population-level baseline hazard is the same for all cases and it changes over time independently of covariates. Mathematically, the model assumes that all observations have the same form of the conditional hazard function h(t|x) consisting of baseline hazard function and a vector of weights. The weights of Cox PH are adjusted by maximum partial likelihood. The corresponding conditional survival function may be predicted for a particular observation with the baseline survival function and shifted relatively to the weight vector.

However, the method has several significant disadvantages:

1. The ratio of hazards for two different vectors (patients) is assumed to be a constant over time, and it is expected that significance of covariates does not change over time.
2. PH regression works with numerical features only, it means that categorical features should by recoded, and missing values should be imputed before building the model.
3. Assumption about dependency of time and probability of the event on the linear combination of covariates may have no ground for certain cases.
4. The model is poorly interpreted and visualized in the sense that, certainly, we can estimate the importance of covariates using PH regression coefficients and odds ratio, but we cannot use this information to segment cases (patients) into the groups with similar survival tendencies. Grouping patients according to their survival perspective was very important during early COVID waves.

PROPOSED APPROACH

To avoid the above-described problems of classical PH regression approach we develop a survival decision tree, which hasn't been implemented on SAS analytical platform before. The code can be found in GitHub repository (Vasilev I. , 2021). Decision trees (Morgan, 1963) are based on the idea of recursive partitioning the feature space into groups (described by nodes/branches) dissimilar according to a selected split criterion. The root node contains all observations, and then, on the predefined criterion the algorithm chooses the best split (in the simplest case, the binary split). The process is recursively repeated on the child nodes until a stopping condition is satisfied. In addition, the tree pruning process cuts off the unnecessary leaves to improve the generalization ability.

WEIGHTED LOG-RANK CRITERIA

Ciampi et al. (Ciampi, 1986) suggested using the log-rank statistic to compare two samples (or survival functions) in the child nodes. A higher statistics value means more difference between the samples. But the log-rank criterion is calculated under the assumptions that the censoring indicator is uncorrelated with prediction, and survival probabilities are the same for events at the early and the late stages of the study. Researchers in (Lee, 2021), (Buyske, 2000) suggest using a weight function incorporated in the log-rank to improve its sensitivity. To increase the sensitivity of the log-rank criteria, we propose to use weighted log-rank criteria such as Wilcoxon (Breslow, 1970), Tarone-Ware (Tarone, 1977), Peto-Peto (Peto, 1972) tests. In our implementation, PROC LIFETEST is used as a tool for calculating weighted log-rank statistics and corresponding p-values while looking for the best splitting value for the variable (see macro %check_var in check_vars.sas file at (Vasilev I. , 2021)).

PROPOSED SURVIVAL DECISION TREE

We propose (Vasilev & Petrovskiy, 2022) the following approach for constructing a survival tree (see macro %build_tree in build_tree.sas file at (Vasilev I. , 2021)). Each node is partitioned recursively into two child nodes according to the best value of weighted log-rank

criterion. Consider our algorithm for finding the best split in a node for some feature (see macro %check_var in check_vars.sas file at (Vasilev I. , 2021)). If the feature is continuous, we bin it into given number of intervals using quantile-based discretization implemented by PROC RANK and get intermediate points as candidates for generating splitting hypotheses. In this case, each split point generates two splitting samples; fewer or equal values relate to the left branch and others to the right. If the feature is categorical, we map its values to the numeric scale using smoothed WOE approach based on event probability for this node. Each pair of sets generates left and right branches. If the feature contains missing values, we test extra hypothesis: missing values go to the left branch, to the right one, and they are placed in a separate branch. To avoid orphan nodes, we ignore hypotheses where one branch contains small number of cases (is controlled by minimum leaf size parameter). For all features, we choose the best split with a minimal Bonferroni adjusted p-value. The p-values are calculated by PROC LIFETEST where STARTA is defined by test hypothesis (left vs right branch) with selected weighted log-rank criteria (LOGRANK, PETO, TARONE or WILCOXON). Finally, we choose the feature with the minimal Bonferroni adjusted p-value in the best split. This process repeats itself recursively (see macro %split_node in build_tree.sas file at (Vasilev I. , 2021)) until one of the stopping conditions such as maximum depth or minimum p-value is satisfied. On each iteration we store the condition of the best split into special dataset which is used later for generating the scoring code of the tree. This code can be run in SAS Data Step. It assigns each record (patient) to the corresponding leaf of the tree according to the found rules and associates the record with the node's survival function based on Kaplan-Meier estimator built on the training set.

It is worth to note that we improve the performance of our code by using SAS/CONNECT abilities to run split search in parallel sessions for different subsets of features with RSUBMIT operator. To avoid overfitting, we implement simple tree pruning algorithm (see macro %prune_tree in build_tree.sas file at (Vasilev I. , 2021)). It uses holdout sample and Integrated AUC (Guo, 2017) as a selection criterion to sequentially remove the worst leaves on each step. Finally, the step with the best IAUC on holdout dataset is chosen as the final tree model.

ENSEMBLE OF TREES

Though the proposed survival decision trees demonstrate acceptable accuracy and provide powerful tool for visualization and interpretable knowledge discovery, they usually don't outperform well-tuned PH regression. To check whether the performance of our models can be improved via bootstrap aggregation approach, we developed the method (Vasilev & Petrovskiy, 2022) and the code (see macro %build_forest and macro %score_forest in main.sas file at (Vasilev I. , 2021)) for constructing random forest of the proposed models. We add sampling without replacement implemented by PROC SURVEYSELECT before building each tree and randomly filter out 75% of variables in each attempt in split search step. Different trees can be built in separate sessions using SAS/CONNECT, but in this case, split search works in sequential mode. We plan to implement the algorithm for optimal ensemble size selection based on OOB in the future. For now, the size of the ensemble is defined by prespecified parameter (equal to 10 by default). The bagging model risk or time prediction is the mean of all models' predictions in the ensemble.

EXPERIMENTS

DATA PREPARATION

The proposed approach was developed and applied to the real-life Moscow hospital dataset, but the dataset is not public. To confirm the performance of our approach, we apply the developed methods to public dataset of COVID-19-infected patients in Wuhan, China (Yan, 2020). The dataset contains a sequence of clinical analysis over time for patients, the age, gender, time of admission, time of discharge, and the outcome for each patient from 10

January to 18 February 2020. The target survival time is calculated as difference between discharge and admission time (in days), death indicator (outcome) is set as a censoring flag in our task. Data originating from pregnant and breastfeeding women, patients younger than 18 years, and records with more than 80% of missing values are excluded from the subsequent analysis. The average age of the patients was 58.83 years, and 59.7% of all patients are male. Of the 375 cases included in the subsequent analysis, 201 recovered from COVID-19 and were discharged from the hospital, while the remaining 174 have deceased.

Since each patient may have several records with clinical analysis corresponding to different days, we convert these sequences of values into patient's features calculating minimum, maximum and average value of each type of analysis and add variables with av_, mx_ and mn_ prefixes to the feature space. Besides, for each initial clinical feature we calculate its trend as a Pearson correlation coefficient over time using PROC CORR. (see macro %prep_patients in main.sas file at (Vasilev I. , 2021)).

To honestly estimate our models and compare them with baseline PH regression model we randomly split the data into train, test, and validation samples in a proportion of 50/25/25 with stratification by event probability. For bagging ensembles, we use only train dataset for training and test dataset for performance estimation. For each weighted log-rank criterion, the decision trees and bagging ensembles are fit on the train sample. Also, the decision trees are pruned on the validation sample. Baseline Cox proportional hazard regression model is trained on the train sample using PROC PHREG with forward stepwise selecting method (with maxstep=100 option). After that, the best step on validation dataset is selected using IAUC criteria (see macro %best_phreg in main.sas file at (Vasilev I. , 2021)).

METRICS AND EVALUATION

We use the Integrated AUC (Guo, 2017) to evaluate the performance of the models. This metric estimates time-dependent concordance measures from AUC and weights derived from the survival time distribution. The resulting measure is a weighted sum of the estimated AUC at each unique failure time where weights are proportional.

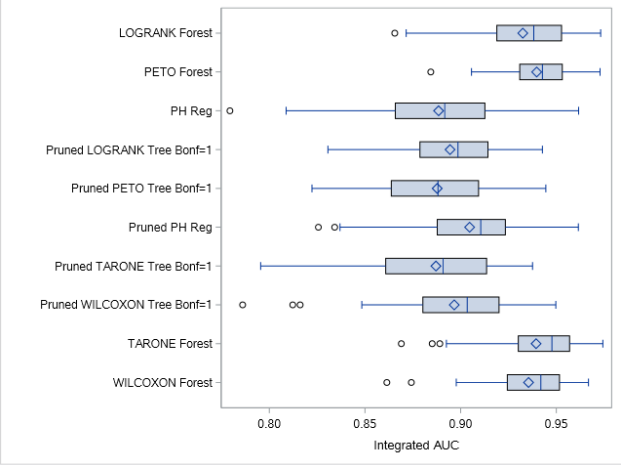


Figure 1. COVID-19-infected patients in Wuhan, China dataset Integrated AUC results.

The final evaluation is done via series of 50 independent experimental runs. Each run uses its own randomly generated partitioning on train/validation/test datasets (with stratification by event rate). PETO, WILCOXON, LOGRANK and TARONE pruned trees and bagging ensembles are trained as well as complete and pruned (on validation dataset) stepwise forward PH regression models. Integrated AUCs are calculated for every model on the test part of the dataset in each run, and final results are presented in the form of boxplot graph (see Figure 1). Default metaparameters are max depth 15, pval threshold 0.15, minimum leaf size 15, ensemble size 10, number of bins 20.

CONCLUSION

SAS analytical platform helps us to develop survival decision trees growing and pruning algorithms, as well as bagging ensembles based on them to solve the survival analysis problem. The implementation uses SAS Macro, SAS Base, SAS SQL languages, SAS/STAT procs and SAS/CONNECT technologies. The proposed algorithms allow to build interpretable survival models using incomplete data and categorical features; and they don't lean on the assumption of proportional hazards and linear combination of covariates. At the same time, the proposed ensemble methods demonstrate very promising performance on COVID-19 patients data outperforming classical PH regression approach.

REFERENCES

Allison, P. D. (2010). *Survival analysis using SAS: a practical guide.* Sas Institute.

Breslow, N. (1970). A generalized Kruskal-Wallis test for comparing K samples subject to unequal patterns of censorship. *Biometrika*, 579-594.

Buyske, S. R. (2000). A class of weighted log-rank tests for survival data when the event is rare. *Journal of the American Statistical Association 95.449*, 249-258.

Ciampi, A. e. (1986). Stratification by stepwise regression, correspondence analysis and recursive partition: a comparison of three methods of analysis for survival data with covariates. *Computational statistics & data analysis 4.3* , 185-204.

Cox, D. R. (1972). Regression models and life-tables. *Journal of the Royal Statistical Society: Series B (Methodological) 34.2* , 187-202.

Guo, C. a. (2017). A Simple Generalisation of the Area Under the ROC Curve for Multiple Class Classification Problems. *Paper SAS462-2017, Cary, NC: SAS Institute*.

Kaplan, E. L. (1958). Nonparametric estimation from incomplete observations. *Journal of the American statistical association 53.282*, 457-481.

Lee, S.-H. (2021). Weighted Log-Rank Statistics for Accelerated Failure Time Model. *Stats 4.2*, 348-358.

Morgan, J. N. (1963). Problems in the analysis of survey data, and a proposal. *Journal of the American statistical association 58.302*, 415-434.

Peto, R. a. (1972). Asymptotically efficient rank invariant test procedures. *Journal of the Royal Statistical Society: Series A (General)*, 185-198.

Tarone, R. E. (1977). On distribution-free tests for equality of survival distributions. *Biometrika*, 156-160.

Vasilev, I. (2021, December 1). *SAS_Curiosity_Cup_2022.* Retrieved from GitHub: https://github.com/iuliivasilev/SAS_Curiosity_Cup_2022

Vasilev, I., & Petrovskiy, M. a. (2022). Survival Analysis Algorithms based on Decision Trees with Weighted Log-rank Criteria. *In Proceedings of the 11th International Conference on Pattern Recognition Applications and Methods*, 132-140.

Yan, L. e. (2020). An interpretable mortality prediction model for COVID-19 patients. *Nature machine intelligence*, 283-288.

APPENDIX: VISUALIZATION EXAMPLES

SURVIVAL TREE EXAMPLE

Below there is a visual example of the trained and pruned WILCOXON-based survival tree on Wuhan COVID-19 patients dataset. The main settings are: maximum depth is 15, number on bins for continuous features is 20, p-value threshold 0.15, Bonferroni adjustment is on):

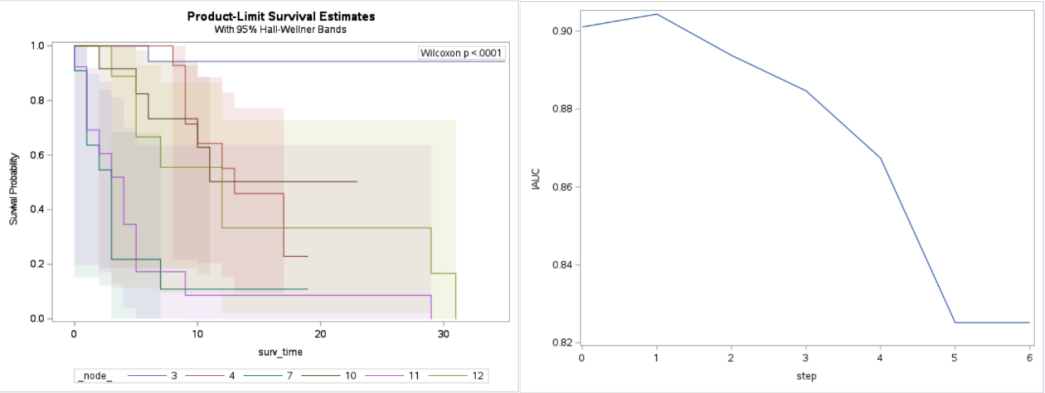


Figure 2. ODS Output for the trained and pruned WILCOXON-based survival tree. The survival functions for each tree leaf is on the left. The IAUC per step history of the pruning process is on the right.

if (((1=1)) and (((mn_procalc ne .) and (mn_procalc<0.115)))) and (not (((av_Lactdehydr ne .) and (av_Lactdehydr<360.20835)))) then do; _node_= 4;t= 9.8125;xb= -0.33860186;end;
if ((((1=1)) and (((mn_procalc ne .) and (mn_procalc<0.115)))) and (((av_Lactdehydr ne .) and (av_Lactdehydr<360.20835)))) and (((mx_hematocrit ne .) and (mx_hematocrit<35.9))) then do; _node_= 5;t= 12.625;xb=-3.229294253;end;
if ((((1=1)) and (((mn_procalc ne .) and (mn_procalc<0.115)))) and (((av_Lactdehydr ne .) and (av_Lactdehydr<360.20835)))) and (not (((mx_hematocrit ne .) and (mx_hematocrit<35.9)))) then do; _node_= 6;t=16.140350877;xb= -17.7830726;end;
if (((1=1)) and (not (((mn_procalc ne .) and (mn_procalc<0.115))))) and (((mx_schloride eq .) or (mx_schloride<98.2))) then do; _node_= 7;t=2.7666666667;xb=1.0931600309;end;
if ((((1=1)) and (not (((mn_procalc ne .) and (mn_procalc<0.115))))) and (not (((mx_schloride eq .) or (mx_schloride<98.2))))) and (not (((mx_prothromb ne .) and (mx_prothromb<83)))) then do; _node_= 10;t=10.222222222;xb=-1.426407463;end;
if (((((1=1)) and (not (((mn_procalc ne .) and (mn_procalc<0.115))))) and (not (((mx_schloride eq .) or (mx_schloride<98.2))))) and (((mx_prothromb ne .) and (mx_prothromb<83)))) and (((trnd_indbilirubin eq .) or (trnd_indbilirubin<-0.996625))) then do; _node_= 11;t=5.9523809524;xb=0.5655400831;end;
if (((((1=1)) and (not (((mn_procalc ne .) and (mn_procalc<0.115))))) and (not (((mx_schloride eq .) or (mx_schloride<98.2))))) and (((mx_prothromb ne .) and (mx_prothromb<83)))) and (not (((trnd_indbilirubin eq .) or (trnd_indbilirubin<-0.996625)))) then do; _node_= 12;t=8.5238095238;xb= 0;end;

Table 1. Score code generated for each leaf of the tree.

According to the Figure 1 we can say that the initial tree has seven leaves after training and the best IAUC (higher than 0.9) is obtained after the first step of pruning. Each node has its own survival function and the highest survival odds are for the samples from node 3, and the worst are for node 11. According to the Table 1. Score code generated for each leaf of the tree., node 3 strata correspond to patients with small average procalcitonin (less than 0.115) and lactate dehydrogenase (less than 360.2) levels. On the other hand, the highest risk is for patients from node 11 with high (or unmeasured) average level of procalcitonin, high (or unmeasured) maximum level of serum chloride (higher than 98.2), small maximum level prothrombin activity (less than 83) and strong negative trend of indirect bilirubin level over time (less than -0.99).

AUC OVER TIME COMPARISON EXAMPLE

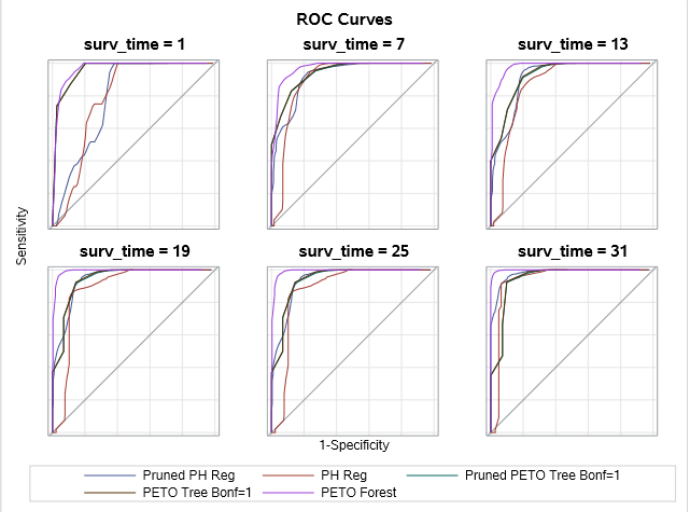
Technically we calculate Integrated AUC using PROC PHREG with ROC statement for baseline and challenging models, receptions parameter to visualize AUC(t). Results of the proposed tree-based models are substituted with PRED= option and model /NOFIT option in PROC PHREG for comparison using ROC statement (see macro %main in main.sas file at (Vasilev I. , 2021)).

```
ods output iauc=iauc;
proc phreg data=ttt_final plots=roc rocoptions(IAUC at=1 to 35 by 6);
   model surv_time*outcome(0)=phreg_last phreg_best / nofit;
```

```
    roc 'Pruned PH Reg' pred=phreg_best;
    roc 'PH Reg' pred=phreg_last;
    &rocs.
run;
```

where rocs macro variable is constructed in the loop adding results for survival trees and their ensembles for each type of test defined by macro variable tst:

```
%let rocs = roc %str(%')Pruned &tst. Tree%str(%') pred=t_&tst._pruned
%str(;) roc %str(%')&tst. Tree%str(%') pred=t_&tst._rules %str(;) roc
%str(%')&tst. Forest%str(%') pred=t_&tst._forest %str(;) &rocs;
```



Integrated Time-Dependent Area Under the Curve	
Source	**Estimate**
Pruned PH Reg	0.9199
PH Reg	0.8897
Pruned PETO Tree Bonf=1	0.9388
PETO Tree Bonf=1	0.9408
PETO Forest	0.9788

Figure 3. ODS Output of comparing AUCs at time moments (1 to 35 step 6) for different PETO-based trees and PH regressions models.

APPENDIX: LIST OF ALL CLINICAL VARIABLES IN THE DATASET

Hypersensitive cardiac troponin, hemoglobin, serum chloride, prothrombin time, procalcitonin, eosinophils, Interleukin 2, Alkaline phosphatase, albumin, basophil, Interleukin 10, Total bilirubin, Platelet count, monocytes, antithrombin, Interleukin 8, indirect bilirubin, Red blood cell distribution width, neutrophils, total protein, Quantification of Treponema pallidum antibodies, Prothrombin activity, HBsAg, mean corpuscular volume, hematocrit, White blood cell count, Tumor necrosis factor, mean corpuscular hemoglobin concentration, fibrinogen, Interleukin 1, Urea, lymphocyte count, PH value, Red blood cell count, Eosinophil count, Corrected calcium, Serum potassium, glucose, neutrophils count, Direct bilirubin, Mean platelet volume, ferritin, RBC distribution width SD, lymphocyte, HCV antibody quantification, D-D dimer, Total cholesterol, aspartate aminotransferase, Uric acid, HCO3, calcium, Amino-terminal brain natriuretic peptide precursor, Lactate dehydrogenase, platelet large cell ratio, Interleukin 6, Fibrin degradation products, monocytes count, PLT distribution width, globulin, glutamyl transpeptidase, ISR, basophil count, 2019-nCoV nucleic acid detection, mean corpuscular hemoglobin, Activation of partial thromboplastin time, Hypersensitive c-reactive protein, HIV antibody quantification, serum sodium, thrombocytosis, ESR, glutamic-pyruvic transaminase, eGFR, creatinine.