Package vignette for `TBFmultinomial`

Dynamic cause-specific variable selection for discrete time-to-event competing risks models

Rachel Heyard
University of Zurich

1 Introduction

The package `glmBfp` does objective Bayesian variable selection using a methodology based on test-based Bayes factors (TBF) for generalised linear models [Held et al., 2015] as well as for the Cox model [Held et al., 2016]. However, `glmBfp` cannot handle multinomial outcomes. Therefore, the package `TBFmultinomial` is an extension that allows mulitple outcomes as in the multinomial regression model. Most importantly the package has been developped for discrete time-to-event models with competing risks. The TBF methodology can easily be extended to these models, which are nothing else then multinomial regression models with a time-dependent intercept, see Heyard et al. [2017].

2 Data example

Our data example will be similar to the one presented in Heyard et al. [2017], but simplified. The goal of the analysis is to find a prediction model for the risk of acquiring a VAP. However if a patient is extubated or dies, a VAP cannot be diagnosed anymore. Extubation and death then compose the competing events/risks for VAP acquisition. The data is stored in the package as `VAP_data`.

```
library('TBFmultinomial')
```

```
## Loading required package:  VGAM
## Loading required package:  stats4
## Loading required package:  splines
## Loading required package:  nnet
## Loading required package:  parallel
## Loading required package:  stringr
## Loading required package:  plotrix
```

```
data("VAP_data")
dim(VAP_data)
```

```
## [1] 1640    7
```

```
head(VAP_data, 10)
```

```
##    ID day gender    type SAPSadmission SOFA    outcome
## 1   2   1      0 Medical            50    9 ventilated
## 2   2   2      0 Medical            50    8 ventilated
## 3   2   3      0 Medical            50    9 ventilated
## 4   2   4      0 Medical            50    9 ventilated
## 5   2   5      0 Medical            50    8        VAP
## 6   3   1      1 Medical            34   10 ventilated
## 9   3   4      1 Medical            34    6 ventilated
## 10  3   5      1 Medical            34    5 ventilated
## 12  3   7      1 Medical            34    2 ventilated
## 13  3   8      1 Medical            34    1 ventilated
```

```
table(VAP_data$outcome)
```

```
##
## ventilated       dead  extubated        VAP
##       1530         20         80         10
```

Each row in the data set stands for one day of ventilation of a patient. We have 1640 ventilation days for 90 distinct patients. We now want to find a prediction model for the variable `outcome` using the baseline variable `gender`, `type` (patient type, can be medical or surgical) and `SAPSadmission` (the simplified acute physiology score at admission) as well as the time-dependent variable `SOFA` (the daily sequential organ failure assessment score).

3 Dynamic Bayesian variable selection

We will now proceed step by step to dynamic Bayesian variable selection in order to define a prediction model for the time to acquire a VAP taking into account its competing risks.

3.1 Posterior model probability

The first step will be to fit the candidate models and compute their posterior probabilities using the function `PMP()`. Our methodology is based on the g-prior so that we need to decide on the g definition. We can either simply set g equal to the sample size with `method='g=n'`, or use an empirical Bayes (EB) approach like the local EB with `method='LEB'` or the global EB with `method='GEB'`. An other possibility is a fully Bayes approach with `method` $\in$ {`'ZS'`, `'ZSadapted'`, `'hyperG'`, `'hyperGN'`}. We refer to Held et al. [2015] for further detail on the g definition.

To use the `PMP()` function we first need to define the full model containing all the potential predictors with a time-dependent intercept defined as a natural spline with 4 degrees on the variable `day`:

```
full <- outcome ~ ns(day, df = 4) + gender + type + SAPSadmission + SOFA
```

The formula can be defined as a `formula-class` or as a character. Then we can apply the function on our data and use the default settings for the other parameters. By default a LEB approach is used for the estimation of g, a uniform (flat) prior is used on the candidate model space, the `nnet` package is used to fit the models with 150 iterations (max). We further need to tell the function that we are considering a discrete survival model by setting `discreteSurv` to `TRUE`.

```
PMP_LEB_flat <- PMP(fullModel = full, data = VAP_data, discreteSurv = TRUE)
```

Then, using the generic function `as.data.frame()`, we can nicely represent an object of class `PMP`; the models are ordered by their posterior probability. So the first element in the data frame is the model with the highest PMP: the maximum a posteriori (MAP) model is the one with only `SOFA` as predictor.

```
class(PMP_LEB_flat)

## [1] "PMP"  "list"

as.data.frame(PMP_LEB_flat)

##       posterior  logPrior gender  type SAPSadmission  SOFA
## 5  5.328383e-01 -2.772589  FALSE FALSE         FALSE  TRUE
## 10 3.572925e-01 -2.772589  FALSE  TRUE         FALSE  TRUE
## 15 3.607272e-02 -2.772589  FALSE  TRUE          TRUE  TRUE
## 11 2.774031e-02 -2.772589  FALSE FALSE          TRUE  TRUE
## 13 2.427082e-02 -2.772589   TRUE  TRUE         FALSE  TRUE
## 8  1.671044e-02 -2.772589   TRUE FALSE         FALSE  TRUE
## 16 3.392558e-03 -2.772589   TRUE  TRUE          TRUE  TRUE
## 14 1.682432e-03 -2.772589   TRUE FALSE          TRUE  TRUE
## 9  3.331786e-15 -2.772589  FALSE  TRUE          TRUE FALSE
## 4  3.038039e-15 -2.772589  FALSE FALSE          TRUE FALSE
## 12 2.025561e-15 -2.772589   TRUE  TRUE          TRUE FALSE
## 7  1.566718e-15 -2.772589   TRUE FALSE          TRUE FALSE
## 3  2.615036e-16 -2.772589  FALSE  TRUE         FALSE FALSE
## 6  2.593422e-16 -2.772589   TRUE  TRUE         FALSE FALSE
## 2  1.677275e-16 -2.772589   TRUE FALSE         FALSE FALSE
## 1  1.590463e-16 -2.772589  FALSE FALSE         FALSE FALSE
```

Instead of defining a full model as an input for the function, we can also fix the candidate models before and store them in a character vector with the first element being the reference model and the last the full model. Then we set

the parameter `candidateModels` to this vector and leave `fullModel` undefined. In this way, we can fix some variables to be included by default.

3.2 Posterior inclusion probability

Using the `PMP`-object we can also compute the posterior inclusion probabilities (PIPs) with the `postInclusionProb()` function.

```
postInclusionProb(PMP_LEB_flat)
```

```
##        gender          type SAPSadmission          SOFA
##    0.04605625    0.42102855    0.06888802    1.00000000
```

3.3 Cause-specific variable selection

The PIPs refer to the importance of a variable as a predictor for all outcomes together. We may want to quantify the relevance of a variable for the prediction of each outcome individually. Therefore we proceed to cause-specific variable selection `CSVS` as described in Heyard et al. [2017]. The function `CSVS()` can be applied on one particular model either fitted using `multinom()` of the package `nnet` or using `vglm()` from `VGAM`. Note that we need a fixed g, so we cannot use the fully Bayes methods for CSVS:

```
# we first fit the model:
model_full_nnet <- multinom(formula = full, data = VAP_data,
                            maxit = 150, trace = FALSE)
# and then apply the function
test_CSVS_nnet <- CSVS(g = tail(PMP_LEB_flat$G, 1), model = model_full_nnet,
                       discreteSurv = TRUE, package = 'nnet')
```

The function `plot_CSVS` then plots the results:

```
res <- plot_CSVS(CSVSobject = test_CSVS_nnet,
                 namesVar = NULL, shrunken = TRUE,
                 standardized = TRUE, numberIntercepts = 5)
```

```
## $before
##      gender1 typeSurgical SAPSadmission       SOFA
## 1  0.7360920   -0.7644011     1.2416076  2.8581988
## 2 -0.9435592   -2.4763176    -0.7786595 -6.5665457
## 3 -0.6957668   -0.9161588    -0.7380157  0.5375009
##
## $after
##   gender1 typeSurgical SAPSadmission      SOFA
## 1       0     0.000000             0  3.759587
## 2       0    -2.288231             0 -6.686574
## 3       0     0.000000             0  0.000000
```

The color scale in Figure 1 is defined with white to red corresponding to 0.538 to 6.567 for the upper plot and to 0 to 6.687 for the lower plot. Furthermore, the outcomes are defined as 1:dead, 2:extubated and 3:VAP.

3.4 Dynamic variable selection using landmarking

In a very last step, we can proceed to dynamic variable selection via landmarking using the function `PIPs_by_landmarking()`. The landmarking technique has been introduced by van Houwelingen [2007], used in connection with PIPs by Held et al. [2016] and been extended to the context of discrete time-to-event competing risks model by Heyard et al. [2017]. To do so, we need to set the same parameters as for `PMP()`. Further, we need to specify the landmark length in days (here `landmarkLength=4`), the last landmark (here `lastlandmark=20`) and the name of the variable indication the time (here `timeVariableName = 'day'`).

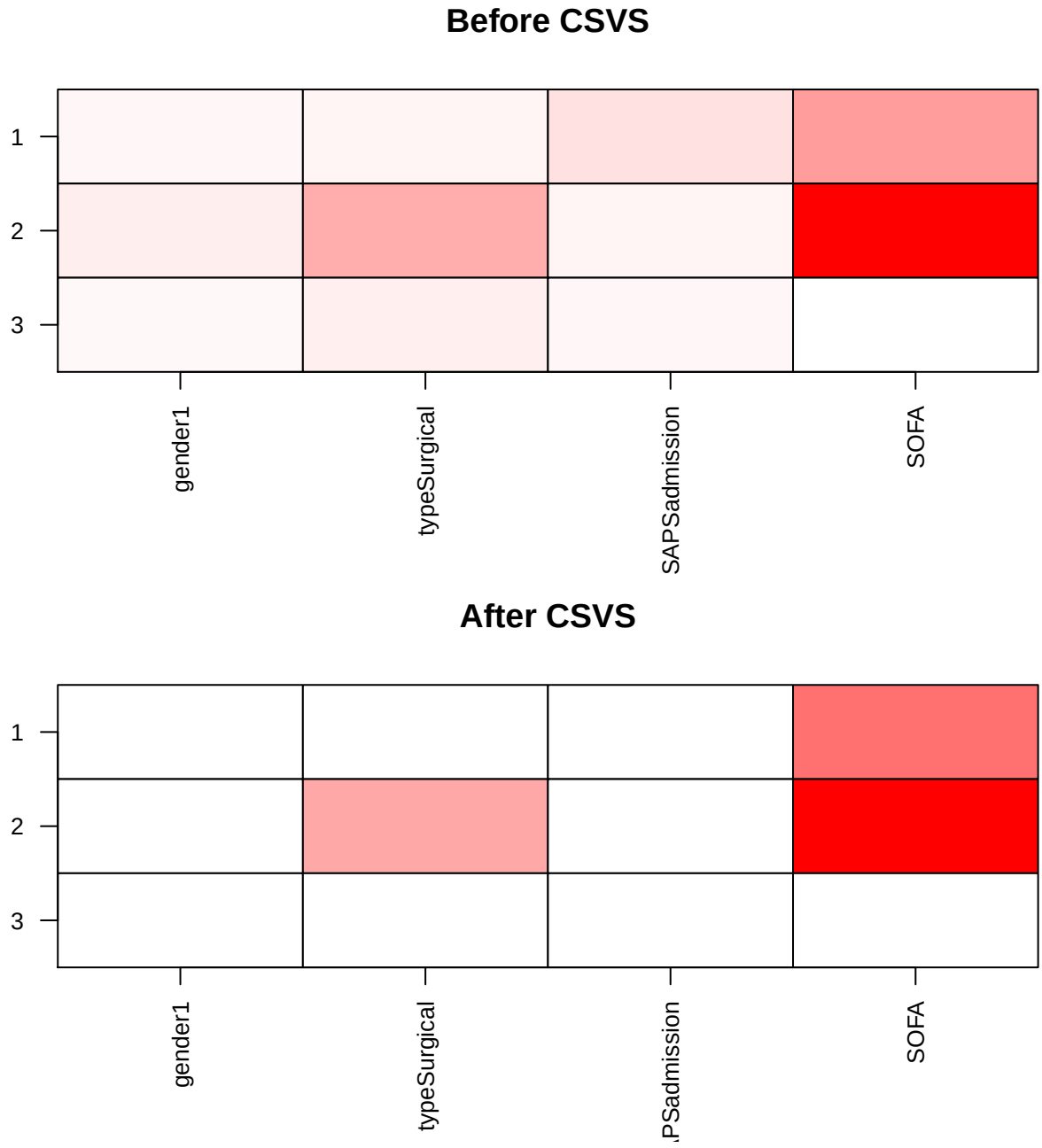

Figure 1: Absolute values of the shrunken standardized coefficients before and after CSVS.

```
pips_landmark <-
  PIPs_by_landmarking(fullModel = full, data = VAP_data, discreteSurv = TRUE,
                      numberCores = 1, landmarkLength = 4, lastlandmark = 20,
                      timeVariableName = 'day')
```

```
pips_matrix <- matrix(unlist(pips_landmark), nrow = length(pips_landmark),
                      byrow = TRUE)
colnames(pips_matrix) <- names(pips_landmark[[1]])
par(mfrow = c(2,2), las = 1)
for(i in 1:ncol(pips_matrix)){
  plot(seq(0, 20, by = 4), pips_matrix[ , i], type = 'b',
       xlab = 'Landmark (in days)', pch = 19,
       ylab = 'Probability',
       main = colnames(pips_matrix)[i],
       ylim = c(0, 1))
  abline(h = .5, col = 'blue', lty = 2)
}
```

See 2 for the evolution of the PIPs over time.

4 (Simple) multinomial regression

We can as well apply the TBF methodology on multinomial regression models by setting the parameter `discreteSurv` to `FALSE`.

References

L. Held, D. Sabanés Bové, and I. Gravestock. Approximate Bayesian model selection with the deviance statistic. *Statistical Science*, 30(2):242–257, 05 2015. doi: 10.1214/14-STS510.

L. Held, I. Gravestock, and D. Sabanés Bové. Objective Bayesian model selection for Cox regression. *Statistics in Medicine*, page 5376–5390, 2016. ISSN 1097-0258. doi: 10.1002/sim.7089. sim.7089.

R. Heyard, J.-F. Timsit, W. I. Essaied, and L. Held. Dynamic clinical prediction models for discrete time-to-event data with competing risks; a case study on the outcomerea database. Technical report, University of Zurich, 2017.

H. C. van Houwelingen. Dynamic prediction by landmarking in event history analysis. *Scandinavian Journal of Statistics*, 34:70–85, 2007.

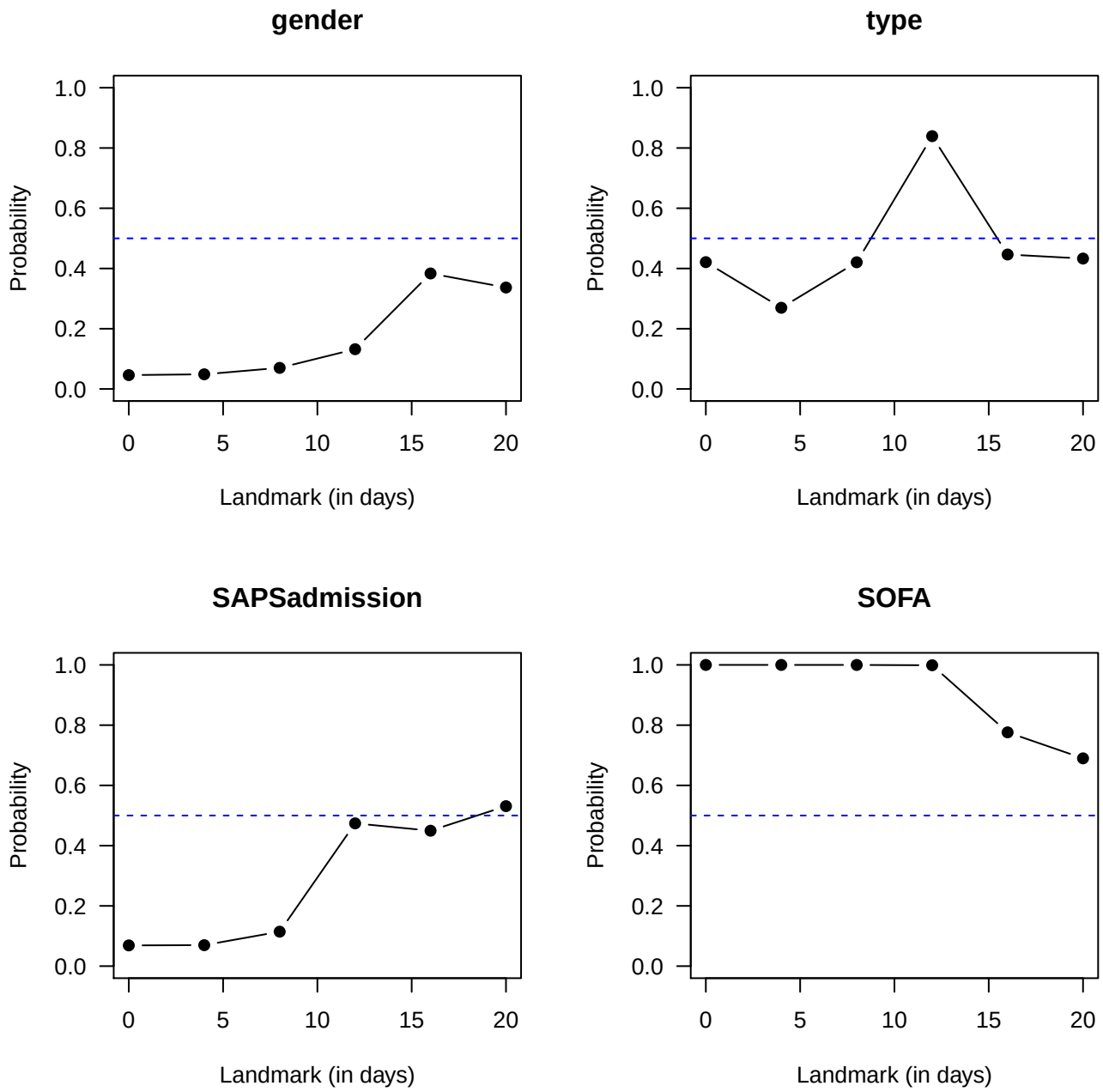

Figure 2: The posterior inclusion probabilities for each landmark.