Exploring Methodologies for ROC Curve Covariate Study with R

Francisco Machado e Costa[1(✉)] and Ana Cristina Braga[2]

[1] School of Engineering, University of Minho, Campus de Gualtar, Braga, Portugal
[2] ALGORITMI Centre, University of Minho, Campus de Azurém, Guimarães, Portugal
acb@dps.uminho.pt

Abstract. The ROC curve is a statistical tool used broadly to help professionals from several fields of study gauge the ability of a binary classifier. Recent theoretical advancements have allowed the ROC curve to better examine existing confounding variables in its analysis allowing greater calibration for markers and classifiers.

A few packages developed for the R language have already incorporated these newfound concepts and are currently available to aid users in the covariate study.

This article combines different ROC curve, adjusted ROC curve and covariate specific ROC curve methodologies across packages to study the effect of sex on the CRIB score system with a resampling strategy using parallel computing.

Results show a confounding effect on roughly 15% of cases with similar results across packages confirming a consensus among methods and providing a robust methodology for future use.

Keywords: ROC curve · AROC · Resampling

1 Introduction

The problem of assigning an object to one of two classes is limitless in application and the core concept behind decision statistics. Systems of classification are flawed however, and errors in classification can occur more often than not requiring the existence of performance classifiers such as the ROC curve.

The ROC curve, defined as a plot of False Positive Fraction (FPF), or 1-specificity and True Positive Fraction (TPF), or sensitivity, pairs obtained by varying threshold c as (x,y) axis respectively.

Defining Y_D and $Y_{\bar{D}}$ as continuous variables for diseased and non diseased groups respectively with cumulative distribution functions F_D and $F_{\bar{D}}$, we assume all test outcomes greater than c belong to the diseased group, with

This work has been supported by FCT - Fundação para a Ciência e Tecnologia within the R&D Units Project Scope: UIDB/00319/2020. The authors express their gratitude to the Portuguese National Registry for supplying the dataset used in this study.

© Springer Nature Switzerland AG 2021
O. Gervasi et al. (Eds.): ICCSA 2021, LNCS 12952, pp. 563–576, 2021.
https://doi.org/10.1007/978-3-030-86973-1_40

$c \in I\!R$. Subsequently, each given c will determine TPF, $TPF(c) = Pr(Y_D \geq c) = 1 - F_D(c)$ and similarly FPF, $FPF(c) = Pr(Y_{\bar{D}} \geq c) = 1 - F_{\bar{D}}(c)$.

Given the standard equation, $ROC(\cdot) = \{(FPF(c), TPF(c)), c \in \mathbb{R}\}$ [1]. By converting FPF at threshold c to t, such as, $t = FPF(c) = 1 - F_{\bar{D}}(c)$, the ROC curve is defined as $\{(t, ROC(t)) : t \in [0, 1]\}$ [2], where

$$ROC(t) = Pr\{Y_D > F_{\bar{D}}^{-1}(1-t)\} = 1 - F_D\{F_{\bar{D}}^{-1}(1-t)\}, 0 \leq t \leq 1 \quad (1)$$

To avoid bias analysis or incomplete conclusions about test accuracy covariate information must be incorporated in the ROC methodology [3]. To this goal, several mathematical approaches have been proposed in recent years to better solve this problem [1,3,4].

By denoting $\mathbf{X}_D$ and $\mathbf{X}_{\bar{D}}$ as diseased and non diseased vector of covariates of interest and a covariate value $\mathbf{x}$, the covariate-specific ROC curve is built as

$$ROC(t|\mathbf{x}) = Pr\{Y_D > F_{\bar{D}}^{-1}(1-t|\mathbf{X}_{\bar{D}} = \mathbf{x})|\mathbf{X}_D = \mathbf{x}\} \quad (2)$$

$$= 1 - F_D\{F_{\bar{D}}^{-1}(1-t|\mathbf{X}_{\bar{D}} = \mathbf{x})|\mathbf{X}_D = \mathbf{x}\} \quad (3)$$

The covariate specific ROC curve builds different curves and displays a different test accuracy for each value of $\mathbf{x}$, crucial in identifying optimal and suboptimal populations and variable values for tests. This method, while helpful for categorical or binary variables lacked the necessary global adjustment and measurement methods for continuous variables.

To mend this issue, along with the covariate **specific** ROC, the covariate **adjusted** ROC curve, AROC [3], was defined as,

$$AROC(t) = \int ROC(t|\mathbf{x}) dH_D(\mathbf{x}) \quad (4)$$

where,

$$H_D(\mathbf{x}) = Pr(\mathbf{X}_D \leq \mathbf{x}),$$

is the cumulative distribution function of $\mathbf{X}_D$. Therefore the AROC is a weighted average of covariate specific ROC curves according to the distribution of the covariates in the diseased group [2]. Janes and Pepe [3] also demonstrated that the AROC curve can also be expressed as

$$AROC(t) = Pr\{Y_D > F_{\bar{D}}^{-1}(1-t|\mathbf{X}_D)\} = Pr\{1 - F_{\bar{D}}(Y_D|\mathbf{X}_D)) \leq t\}, \quad (5)$$

proving that the AROC summarizes the covariate-specific performance of the test.

To establish a statistically significant confounding effect of a given covariate on the ROC curve we have explored the possibility of using the fit model's parameters present in both adjusted and covariate specific methodologies however another well established alternative is available in some scenarios, ROC curve comparison.

When comparing ROC curves, Area Under the Curve (AUC) is frequently used to indicate higher performance provided no curve crossing occur in areas of statistical interest. The Z-test, is often used for this comparison [5]. As long as the covariate is binary in nature, it is possible to separate the data by this covariate to detect differences of performance of the classification system which infers a confounding effect of the covariate in question.

This article will explore these different ROC curve methodologies to detect covariate effect on the CRIB score system utilizing R and recent R packages with a parallel computing and resampling strategy.

2 Methodology

2.1 Dataset

The dataset used is part of the Portuguese National Registry on low weight newborns between 2013 and 2018 and was made available for research purposes. The original data included possible confounding observations such as repeated ids and twins that were removed to ensure no unaccounted variable. After ensuring all id's were unique, these were promptly removed along with any possible identifiable features to abide by European Union's anonymity and data protection standards. The resulting dataset composed of 3823 unique entries registering gestational age in weeks, the mothers' age, in years, the biological sex of the infant (1-Male; 2-Female), CRIB score (0–21), survival (0-Survival; 1-Death) and other possibly relevant covariates were used for the remainder of the study. An abridged dataset with relevant information was published and is available for reference [6].

2.2 Resampling and Parallel Computing

Instead of using the full dataset on a single test to describe the confounding effect of the 'sex' covariate, a resampling strategy was employed to better describe the score's system accuracy and this covariate's effect.

To better represent the original data, sampling followed a 9:1 ratio of survival and death distributed equally among both sexes to ensure proportionality.

Random samples of $n = 500$, $n = 1000$, and $n = 1500$ with replacement were taken and processed through the three tests, this was done iteratively with $i = 100$ and $i = 500$, each time creating a new sample and building three distinct ROC curves.

Given the number of required iterations and high density of data, a parallel computing strategy was implemented. The package `foreach` together with the `doParallel` and `parallel` packages allow several CPU cores to calculate each process in tandem which, in practice, divides the amount of time spent on the full iteration process by the number of cores provided although diminishing returns apply due to different bottlenecks in the process. Note that the packages and code used were meant to apply to a Windows10 machine and different packages could be needed on a Linux or OSX based system due to different core processing each operating system uses.

2.3 ROC Packages

To apply the adjusted ROC and covariate specific ROC curve methodology the `ROCnReg` package was selected. This is a recently released package that imports some methods from a previously existing one of the same authors [7] adding to the original AROC methods, covariate specific/conditional functions. The methods used, *cROC.sp()* and *AROC.sp()*, for covariate specific and covariate adjusted ROC respectively, are both based on a semiparametric regression model to estimate $F_{\bar{D}}(\cdot|\mathbf{x})$ [4]. The covariate specific method however, does so for both healthy and diseased populations, providing two separate significance test values.

For the ROC curve and AUC comparison, the Braga methodology [5] and `Comp2ROC` package were selected. The method uses a collection of sampling lines similar to multi-objective distinct optimization algorithms, allowing ROC curve comparison in several regions of space evaluating statistical performance and generating confidence intervals with nonparametric bootstrap resampling method. This allows curve crossing to occur while still detecting significant differences in AUC values and ROC performance across the whole curve [5,8].

2.4 Code

Working with R and Rstudio the data and packages were imported. For replication purposes, the session info detailing the operating system and package versions will be available in Sect. 5, additionally, custom functions were built in a different file, `comp2conv.R` to simplify code reading but will be detailed also.

The data is imported, and the sample size is defined for this demonstration as $n = 1000$. However, and as mentioned before, values of $n = 500$, $n = 1000$, and $n = 1500$ were run through the analysis.

```
library(parallel)
library(doParallel)
library(foreach)
library(ROCnReg)
library(dplyr)
library(Comp2ROC)
set.seed('1234')
source('comp2conv.R')


neonatal <- read.csv('NeonatalPortugal2018_abridged.csv',
                             sep = ';')
neonatal$sex <- as.factor(neonatal$sex)


sample_size <- 1000
surv_sample <- sample_size * 0.9
dea_sample <- sample_size * 0.1
```

After defining the initial parameters, CPU cores are allocated, in this instance, due to machine limitations, 4 cores are allocated for the analysis. A timing function is used to detail time expenditure on the analysis however this has no bearing on the ROC analysis.

A `foreach` cycle follows, which is similar to the standard `for` however allows for parallel computing thanks to the *%dopar%* parameter. In this instance a 100 cycle is represented by $i = 1 : 100$.

A new sample is created in each cycle with *mysample()* function and be processed by each ROC curve approach in *doROCp* which builds the ROC curves and returns p-values from the fit models and Z-test, where values p-value $<$ 0.05 point to a rejection of the null hypothesis, i.e. indicates the covariate is influencing the analysis. This function returns a list of said p-values which are recorded into the *res* dataframe object due to the *'cbind'* parameters.

After completing an i number of cycles, the parallelization is turned off and time spent on the analysis is recorded and displayed.

```
1
2  cluster <- makeCluster(4) #number of cores
3  registerDoParallel(cluster)
4
5  start_time <- Sys.time()
6
7  res <-
8    foreach (i = 1:100, .combine = 'cbind') %dopar% {
9      mysample <- makesample(neonatal, surv_sample, dea_
           sample)
10
11     mypvalues <- doROCp(mysample)
12
13   }
14
15 parallel::stopCluster(cluster)
16
17 end_time <- Sys.time()
18
19 print(end_time - start_time)
20 }
```

Finally, the p-value of each iteration and sample were analysed to reveal the rate at which the methods rejected the null hypotheses, giving the covariate 'sex' relevant weight in the outcome of the low weight infant.

```
1
2  sum(res[1,] < 0.05) / ncol(res) # AROC
3  sum(res[2,] < 0.05) / ncol(res) # cROC Healthy
4  sum(res[3,] < 0.05) / ncol(res) # cROC Diseased
5  sum(res[4,] < 0.05) / ncol(res) # Comp2ROC
```

Table 1. Results of the iteration process denoting H_0 hypothesis rejection.

		AROC	cROC		Comp2ROC
			Healthy	Diseased	
n = 500	r_1	0.10	0.10	0.14	0.08
i = 100	r_2	0.15	0.15	0.09	0.05
	r_3	0.06	0.06	0.08	0.06
n = 500	r_1	0.10	0.10	0.10	0.06
i = 500	r_2	0.08	0.08	0.08	0.06
	r_3	0.10	0.10	0.07	0.05
n = 1000	r_1	0.15	0.15	0.15	0.09
i = 100	r_2	0.17	0.17	0.12	0.07
	r_3	0.18	0.18	0.11	0.08
n = 1000	r_1	0.13	0.13	0.15	0.07
i = 500	r_2	0.15	0.15	0.13	0.06
	r_3	0.15	0.15	0.16	0.08
n = 1500	r_1	0.21	0.21	0.15	0.04
i = 100	r_2	0.17	0.17	0.20	0.03
	r_3	0.15	0.15	0.14	0.06
n = 1500	r_1	0.18	0.18	0.19	0.05
i = 500	r_2	0.20	0.20	0.16	0.07
	r_3	0.16	0.16	0.19	0.08

Scripts for *makesample()* and *doROCp()* are detailed in Sect. 5. This analysis was performed three times per sample size, iteration pair - $(n; i)$.

3 Results

Each $(n; i)$ pair rejection rate was recorded with triplicates (r_1, r_2, and r_3) in Table 1, and for illustration purposes one iteration of ROC curves built with each method was also recorded.

Analysing Table 1 across ROC methods we first note the same values presented across all $(n; i)$ pairs for AROC and cROC pertaining to the healthy population. This is a result of the same fit procedure being used in both methods and the adjusted ROC being a weighted average of cROC curves, which, given the 9:1 ratio of healthy and diseased sample populations heavily weighs in the calculation. Looking at the Healthy and Diseased populations of the cROC method however, also shows very slight differences between ratios of null hypothesis rejection.

The adjusted covariates method, performed with `ROCnReg` , can be seen in Fig. 1 displaying the AROC curve, AAUC and confidence intervals. Rejection rates vary between 10–20% with the increase of sample size that can be seen

in Table 1. The covariate-specific method, of the same package shows similar rejection rates of 10–20% and 8–19% for surviving and deceased population respectively, the samples are separated by covariate and plotted separately as can be seen in Fig. 2 however, unlike the curve comparison method.

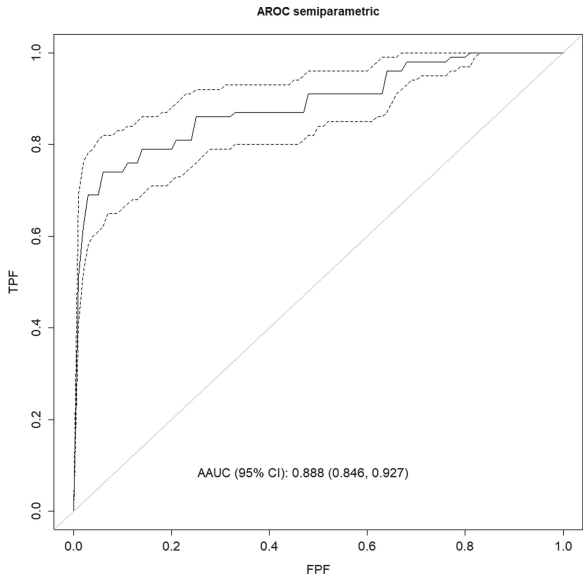

Fig. 1. AROC curve

Another global interpretation of Table 1 shows values ratios of null hypothesis rejection for the Comp2ROC method to be significantly smaller than other methods. Comp2ROC is registering H_0 rejection of Z-test rather than the fit parameters of AROC and cROC, meaning that, instead of measuring the covariate's weight in the adjustment, it records the differences of AUC and performance between covariate specific curves. These differences in global values could mean that, while the theoretical relevant weight of the covariate might be detected in the adjustment methods, the practical performance of the curve remains unchanged in a few instances. This curve comparison method generates ROC curves for both sexes as well as partial AUC performance by sampling lines and performance of sampling data and can be seen in Fig. 3. Z- statistic p-value recorded shows a rejection rate between 5–9% with an exception on r_2 of the $(n = 1500; i = 100)$ pair indicating statistical influence of the 'sex' covariate in these cases.

Results show slight differences with the increase of the sample size (n) however an iteration (i) number above 100 appears to be redundant given no large differences can be found both within or across methods.

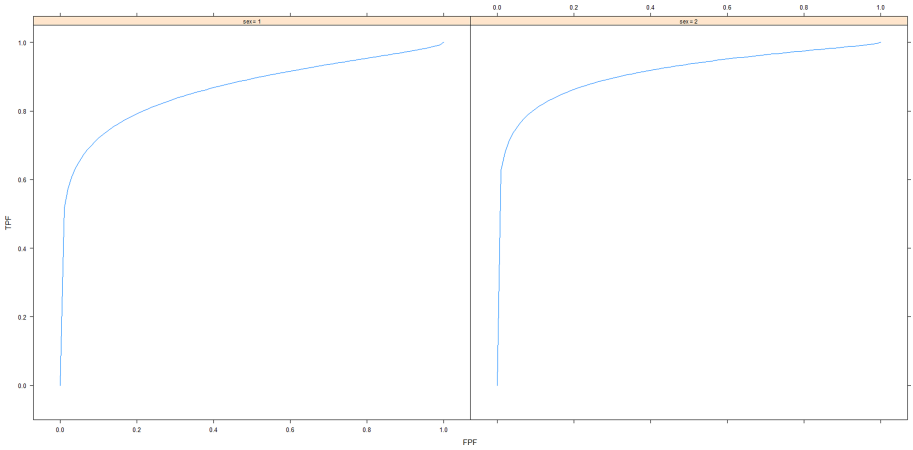

Fig. 2. Covariate specific ROC curves for male (left) and female (right)

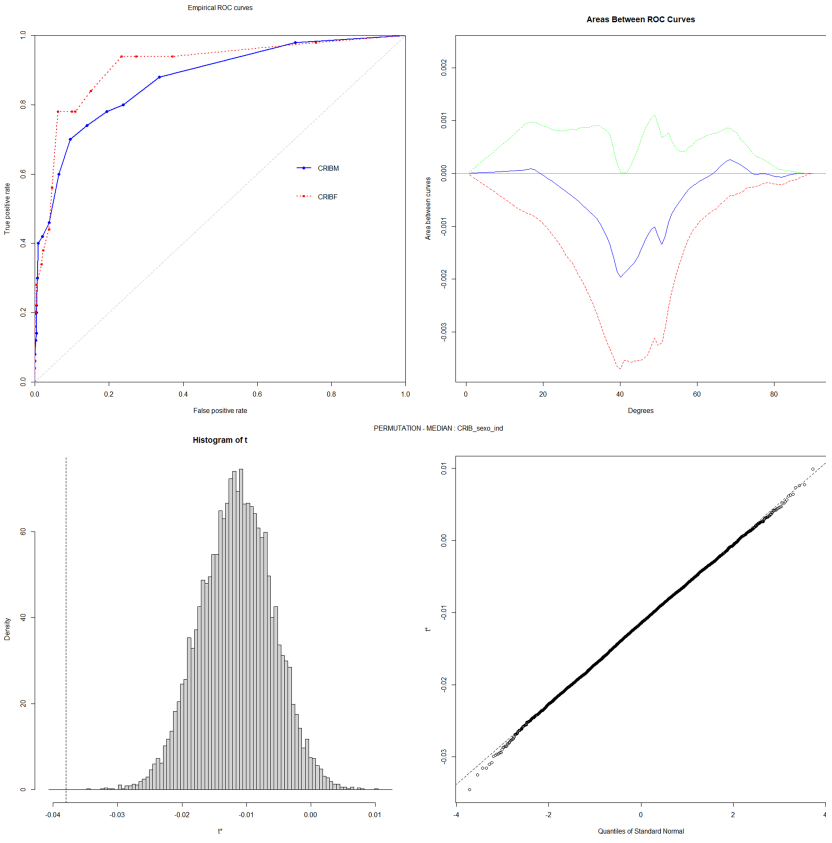

Fig. 3. Comp2ROC results

4 Discussion and Conclusion

While the curve comparison method and Z-test are known and robust procedures to analyse ROC curve and ROC curve data, the adjusted covariate and covariate specific methods are still recent methods that are slowly being implemented across software. A previous R package by the same authors was explored previously [9] however, the new `ROCnReg` shows improvements and additions to these methods keeping the same overall input-output structure as the previous one, which will allow users to transition seamlessly to the more updated version.

This computational work took the initial (null) hypothesis that the CRIB score was not influenced by the sex covariate, and explored said hypothesis by applying the newest methods for ROC curve and covariate analysis in tandem with their tried and tested counterparts and a robust resampling procedure. The results seen in Table 1, detail the rejection rates of the null hypothesis, exhibiting an influence of the covariate on the adjustment of the curve on 10–20% of cases for AROC and cROC methods, as well as a practical distinction between male and female differentiated ROC curves in 5–9% of cases in the Z-test of the `Comp2ROC` methods.

These results show a clear consensus between methods and packages of the convolution factor of the sex covariate in CRIB scores.

Applying parallel computing methods allowed for a more expedient analysis, building a more complete table of results with replicates and different combinations of $(n; i)$ pairs that show a much clearer understanding of the effects of the covariates on the CRIB score as well as how the different methods behave.

The methodology presented in this paper should provide a framework to help guide improvements to similar score systems and markers in their covariate studies that can lead to the development of better calibrated systems for healthcare and all other systems where ROC curves can be implemented.

5 Supplementary

The following code snippets detail the session info for replication purposes as well as functions used in the main article.

```
> sessionInfo()
R version 4.0.4 (2021-02-15)
Platform: x86_64-w64-mingw32/x64 (64-bit)
Running under: Windows 10 x64 (build 19041)

Matrix products: default

locale:
[1] LC_COLLATE=Portuguese_Portugal.1252  LC_CTYPE=
    Portuguese_Portugal.1252
[3] LC_MONETARY=Portuguese_Portugal.1252 LC_NUMERIC=C
[5] LC_TIME=Portuguese_Portugal.1252

```

```
attached base packages:
[1] parallel  stats     graphics  grDevices utils
    datasets  methods   base

other attached packages:
[1] Comp2ROC_1.1.4     boot_1.3-26        ROCR_1.0-7
    gplots_3.1.1       dplyr_1.0.5
[6] ROCnReg_1.0-5      doParallel_1.0.16  iterators_1.0.13
    foreach_1.5.1

loaded via a namespace (and not attached):
 [1] gtools_3.8.2             spatstat.linnet_1.65-3
 tidyselect_1.1.0             purrr_0.3.4
 [5] splines_4.0.4            lattice_0.20-41
 vctrs_0.3.6                  generics_0.1.0
 [9] spatstat.utils_2.1-0     mgcv_1.8-33
 utf8_1.1.4                   rlang_0.4.10
[13] spatstat.data_2.0-0      pillar_1.5.1
spatstat_2.0-1                glue_1.4.2
[17] matrixStats_0.58.0       lifecycle_1.0.0
spatstat.core_1.65-5          MatrixModels_0.5-0
[21] moments_0.14             caTools_1.18.1
codetools_0.2-18              SparseM_1.81
[25] quantreg_5.85            fansi_0.4.2
Rcpp_1.0.6                    KernSmooth_2.23-18
[29] conquer_1.0.2            tensor_1.5
abind_1.4-5                   deldir_0.2-10
[33] nor1mix_1.3-0            spatstat.sparse_1.2-1
polyclip_1.10-0               grid_4.0.4
[37] quadprog_1.5-8           tools_4.0.4
bitops_1.0-6                  magrittr_2.0.1
[41] goftest_1.2-2            tibble_3.1.0
np_0.60-10                    crayon_1.4.1
[45] pbivnorm_0.6.0           pkgconfig_2.0.3
ellipsis_0.3.1                MASS_7.3-53
[49] Matrix_1.3-2             cubature_2.0.4.1
R6_2.5.0                      rpart_4.1-15
[53] spatstat.geom_1.65-5     nlme_3.1-152
compiler_4.0.4
```

The *makesample()* function, used in each iteration of the process, receives the full dataset and survival and death population sizes as inputs, with these parameters and using `dplyr` sample population are selected and bound together with *rbind()* returning a dataframe of sample population ready to be processed.

```
makesample <- function(dataframe, surv_size, dea_size) {
  surv1 <-
    dplyr::sample_n(dataframe[dataframe$survival == 0 &
```

```
                                    dataframe$sex == 1,],
                                    surv_size / 2,
                                    replace = T)
  surv2 <-
    dplyr::sample_n(dataframe[dataframe$survival == 0 &
                                    dataframe$sex == 2,],
                                    surv_size / 2,
                                    replace = T)
  dea1 <-
    dplyr::sample_n(dataframe[dataframe$survival == 1 &
                                    dataframe$sex == 1,],
                                    dea_size / 2,
                                    replace = T)
  dea2 <-
    dplyr::sample_n(dataframe[dataframe$survival == 1 &
                                    dataframe$sex == 2,],
                                    dea_size / 2,
                                    replace = T)

  nn <- rbind(surv1, surv2, dea1, dea2)
}
```

The function *doROCp()* is the main process for the article, it receives the dataframe and runs all three ROC analysis methods. Given the focus of the article on a specific covariate and dataframe, all parameters for each ROC procedure are already entered, however, one can easily generalize the function to include these parameters in the input.

As mentioned the function will compute each ROC curve and save the different p-values according to how each function records the object. Afterwards, the function returns a list of all 4 p-values ordered by function to always know which p-value relates to each method.

```
doROCp <- function(mysample) {
  # - AROC
  spCribSex <- ROCnReg::AROC.sp(
    formula.h = 'crib ~ sex',
    group = 'survival',
    tag.h = 0,
    data = mysample
  )
  spvalue <-
    signif(as.numeric(summary(spCribSex$fit)$coefficients
        [2, 4]), 4)

  #-----
  # - cROC
  cspCS <- ROCnReg::cROC.sp(
    formula.h = 'crib ~ sex',
```

```
    formula.d = 'crib ~ sex',
    group = 'survival',
    tag.h = 0,
    data = mysample
  )
  pvalue_h <-
    signif(as.numeric(summary(cspCS$fit$h)$coefficients[2,
        4]), 4)
  pvalue_d <-
    signif(as.numeric(summary(cspCS$fit$d)$coefficients[2,
        4]), 4)
  #----
  # - Comp2ROC
  nnM <- mysample[mysample$sex == 1 , c('crib', 'survival'
      )]
  nnF <- mysample[mysample$sex == 2 , c('crib', 'survival'
      )]


  datafromfunc <-
    comp_converter(nnM, nnF, 'survival', 'survival', FALSE
        )
  results <-
    Comp2ROC::roc.curves.boot(datafromfunc, 1000, 0.05,
        name = 'CRIB_sexo_ind', "CRIBM", "CRIBF", FALSE)
  #----

  res <-
    c(spvalue, pvalue_h, pvalue_d, results$pvalue1) #[AROC
        ,cROC_h,cROC_d,Comp2ROC]
  res
}
```

The previous function uses the *comp_converter()* function to perform the ROC curve comparison of the `Comp2ROC` package. This function is generalized and converts the initial dataframe to the structure needed for this specific package.

```
comp_converter <- function(df1, df2, rescol1, rescol2,
    related) {
  if (length(df1) == 2 && length(df2 == 2)) {
    var1 <- names(df1)[names(df1) != rescol1]
    var2 <- names(df2)[names(df2) != rescol2]

    list1 <- df1[order(df1[[rescol1]]), ]
    list2 <- df2[order(df2[[rescol2]]), ]


    listall <- list(list1[[var1]] ,
                    list1[[rescol1]] ,
```

```
                    list2[[var2]] ,
                    list2[[rescol2]])

    var1n <- paste(var1, '1', sep = '_')
    var2n <- paste(var2, '2', sep = '_')
    rescol1n <- paste(rescol1, '1', sep = '_')
    rescol2n <- paste(rescol2, '2', sep = '_')

    names(listall) <- c(var1n, rescol1n, var2n, rescol2n)


    comp2rocdata <- read.manually.introduced(
      listall,
      modality1 = var1n,
      testdirection1 = TRUE,
      modality2 = var2n,
      testdirection2 = TRUE,
      status1 = rescol1n,
      related = related,
      status2 = rescol2n
    )


  } else {
    stop('Dataframes must have 2 columns each')
  }


}
```

References

1. Pepe, M.S.: The Statistical Evaluation of Medical Tests for Classification and Prediction. Oxford University Press, Oxford (2003)
2. Rodriguez-Alvarez, M.X., et al.: Bootstrap-based procedures for inference in nonparametric receiver-operating characteristic curve regression analysis. Stat. Methods Med. Res. **27**(3), 740–764 (2018)
3. Janes, H., Pepe, M.S.: Adjusting for covariate effects on classification accuracy using the covariate-adjusted receiver operating characteristic curve. Biometrika **96**(2), 371–382 (2009)
4. Rodriguez-Alvarez, M.X., Inacio, V.: ROCnReg: ROC Curve Inference with and without Covariates (2021). https://CRAN.R-project.org/package=ROCnReg, R package version 1.0-5
5. Braga, A.C., et al.: An alternative method for global and partial comparison of two diagnostic systems based on ROC curves. J. Stat. Comput. Simul. **83**(2), 307–325 (2013)

6. Machado e Costa, F., Braga, A.C.: NeonatalPortugal2018. Mendeley Data (2019). https://doi.org/10.17632/br8tnh3h47.1
7. Rodriguez-Alvarez, M.X., Inacio, V.: AROC: covariate-adjusted receiver operating characteristic curve inference (2018). https://CRAN.R-project.org/package=AROC, R package version 1.0
8. Braga, A.C., et al.: Comp2ROC: compare two ROC curves that intersect (2016). https://CRAN.R-project.org/package=Comp2ROC, R package version 1.1.4
9. Machado e Costa, F., Braga, A.C.: Adjusting ROC curve for covariates with AROC R package. In: Gervasi, O., et al. (eds.) ICCSA 2020. LNCS, vol. 12251, pp. 185–198. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58808-3_15