Survival: Ovarian Cancer

cancer-informatics.org

2023-11-20

Introduction

Ovarian cancer (OS) can be subdivided into three main types: epithelial ovarian cancer, germ cell tumours, and stromal tumours with epithelial ovarian cancer being the most common type. In this article, we will focus on the early stage serous epithelial ovarian cancer, which is the most common type of ovarian cancer. Upon first diagnosis of early OS a staging laparotomy is performed to determine the extent of the disease. One well known study (ICON1) compared immediate adjuvant chemotherapy after surgery vs no adjuvant chemotherapy (see sources below). We provide a simulated dataset based on the ICON1 study, which can be used to perform survival analysis in R. **Please note that this dataset is simulated and for educational purposes only and should not be used for clinical decision-making.**

Used Packages

We will use the following packages in this article. Keep in mind that you might need to install the packages first before you can use them.

```
#Inspired by ICON1 https://academic.oup.com/jnci/article/95/2/125/2912345
#Mock dataset! Created with TwoArmSurvSim

#Load the Required Packages
library(survivalAnalysis)
library(ggplot2)
library(ggsurvfit)
library(ggsci)
library(svglite)
```

Load Data

First we need to obtain the data. Our mock dataset can be found under https://cancer-informatics.org/data sets/ovarian.csv You can either download the csv file first and reference the local file or point R directly to the url:

```
# Load the Ovarian Cancer Dataset:
data <- read.table("ovarian.csv", header=T, sep=",")
```

You can deduct the parameters `header = TRUE` and `sep = ","` from the csv file. Here are the first 5 lines of the csv file:

```
"Treatment",    "HighRisk", "Time",           "Status"
1,              1,          42.4503750028979, 1
0,              0,          18.3782004531846, 1
0,              1,          12.1239407387935, 1
0,              1,          10.1788407434407, 1
```

As you can see we provide the column names in the first line and use a comma as a separator.

Data preprocessing

First we examine the structure and content of the ovarian dataset to understand its variables and format:

```
# Explore the Dataset:
str(data)
```

```
## 'data.frame':    600 obs. of  4 variables:
##  $ Treatment: int  1 0 0 0 1 0 1 1 1 0 ...
##  $ HighRisk : int  1 0 1 1 1 1 1 1 0 1 ...
##  $ Time     : num  42.45 18.38 12.12 10.18 4.81 ...
##  $ Status   : int  1 1 1 1 1 1 0 1 0 1 ...
```

```
summary(data)
```

```
##    Treatment      HighRisk           Time            Status
##  Min.   :0.0   Min.   :0.0000   Min.   : 0.215   Min.   :0.0
##  1st Qu.:0.0   1st Qu.:0.0000   1st Qu.: 9.575   1st Qu.:0.0
##  Median :0.5   Median :0.0000   Median :14.766   Median :0.0
##  Mean   :0.5   Mean   :0.4833   Mean   :16.417   Mean   :0.4
##  3rd Qu.:1.0   3rd Qu.:1.0000   3rd Qu.:22.985   3rd Qu.:1.0
##  Max.   :1.0   Max.   :1.0000   Max.   :44.737   Max.   :1.0
```

The ovarian dataset contains information on 4 variables related to ovarian cancer, including survival time, survival status, treatment, and high-risk status.

It might be helpful to convert some columns into different data types. For instance, you might convert the treatment variable from numeric to factor with adequate labels:

```
#Convert to named factors
data$Treatment <- factor(data$Treatment, levels=c(0,1), labels=c("No CTX", "Adj. CTX"))
data$HighRisk <- factor(data$HighRisk, levels=c(0,1),
                        labels=c("Low Risk",
                                 "High Risk"))

#Adjust column names
colnames(data)[2] <- "RiskGroup"
```

Next, we can perform any necessary data cleaning and transformation steps. Please note that our example already utilizes a fairly clean dataset, so we do not need to perform any additional data cleaning steps. For instance, you might need to check for missing values in the selected variables and handle them appropriately. In this example, we'll remove any rows with missing values.

```
# Check for missing values.
# The result is zero as there are no missing values in the selected dataset.
sum(is.na(data))
```

```
## [1] 0
```

```
# Remove rows with missing values;
# In our example this step is redundant as we already know that
# there are no missing values in the dataset.
data_filtered <- na.omit(data)
```

Multivariate Analysis

In our example we have two independent variables (treatment and high-risk status) and one dependent variable (survival time). To calculate the impact of the independent variables on the dependent variable we can use a Cox proportional hazards model.

```r
#run multivariate analysis
result <- analyse_multivariate(data_filtered,
                               c("Time", "Status"),
                               covariates = c("Treatment", "RiskGroup"))
result$summary
```

```
## Call:
## coxph(formula = Surv(Time, Status) ~ Treatment + RiskGroup, data = data)
##
##   n= 600, number of events= 240
##
##                         coef exp(coef) se(coef)      z Pr(>|z|)
## TreatmentAdj. CTX  -0.6837    0.5048   0.1338 -5.110 3.22e-07 ***
## RiskGroupHigh Risk  0.5102    1.6657   0.1313  3.886 0.000102 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##                     exp(coef) exp(-coef) lower .95 upper .95
## TreatmentAdj. CTX      0.5048     1.9812    0.3883    0.6561
## RiskGroupHigh Risk     1.6657     0.6003    1.2877    2.1546
##
## Concordance= 0.61  (se = 0.019 )
## Likelihood ratio test= 41.4  on 2 df,   p=1e-09
## Wald test            = 40.24  on 2 df,   p=2e-09
## Score (logrank) test = 41.46  on 2 df,   p=1e-09
```

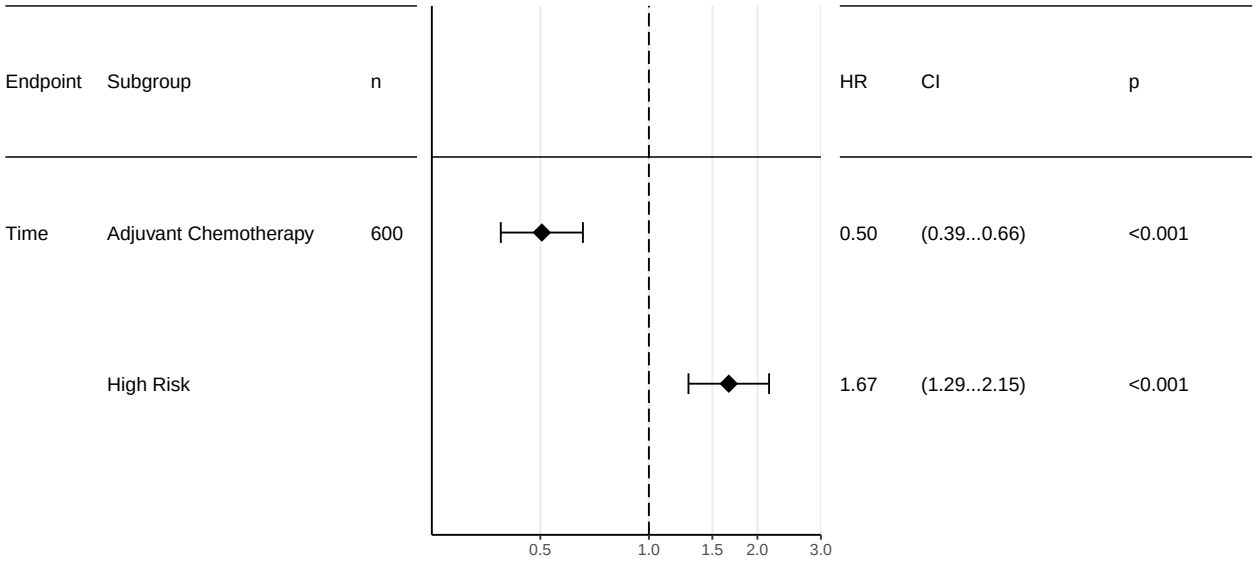
Our results show that both treatment and high-risk status have a significant impact on survival time. The hazard ratio for treatment is 0.5048, which means that patients receiving adjuvant chemotherapy have a 50% lower risk of death compared to patients receiving no adjuvant chemotherapy. The hazard ratio for high-risk status is 1.6657, which means that patients with high-risk status have a 66% higher risk of death compared to patients with low-risk status.

Forest Plot

In the next step we can visualize the results using a forest plot.

```r
# this is required to adjust the labels in the forest plot
# try creating the plot without this helper variable to see why we use it
covariates_labels <-   c(
  "RiskGroup:High Risk" = "High Risk",
  "Treatment:Adj. CTX" = "Adjuvant Chemotherapy"
)

#create forest plot from multivariate analysis
forest_plot(result,
            factor_labeller = covariates_labels,
            endpoint_labeller = c(futime="OS"),
            orderer = ~order(HR),
            labels_displayed = c("endpoint", "factor", "n"),
            HR_x_breaks = c(0.25, 0.5, 1, 1.5, 2, 3),
            HR_x_limits = c(0.25,3)
)
```

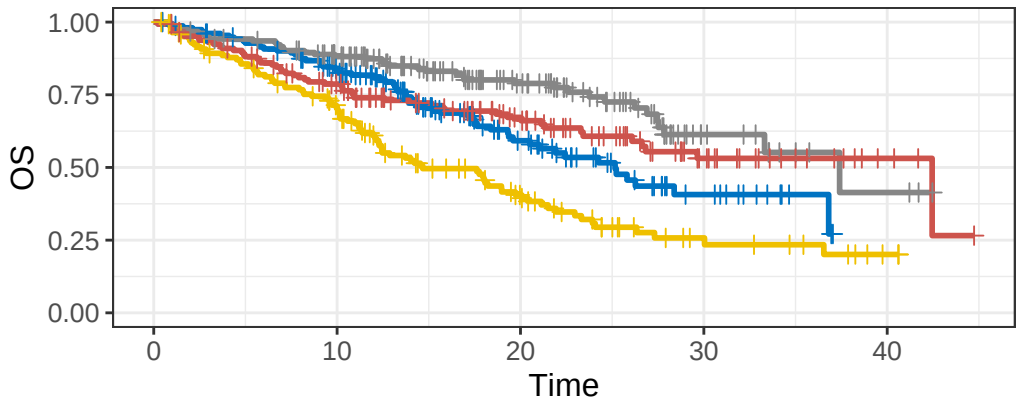


We first create a helper variable `covariates_labels` to adjust the labels in the forest plot. Next, we create the forest plot using the `forest_plot()` function from the survivalAnalysis package. It accepts our previously generated multivariate analysis result as input. Finally, we save the plot as a pdf and svg file.

Survival Analysis

Survival data is typically shown using Kaplan-Meier survival curves.

```
# create a kaplan meier plot
surv_obj <- survfit2(Surv(Time, Status) ~ Treatment+RiskGroup, data = data_filtered)
ggsurvfit(surv_obj, linewidth = 1) + add_pvalue()+
  scale_color_jco() + add_censor_mark() +
  labs(
    y = "OS",
    x = "Time"
  ) + ylim(0,1)+add_risktable()
```



p<0.001

At Risk					
No CTX, Low Risk	155	116	46	13	0
No CTX, High Risk	145	90	37	11	2
Adj. CTX, Low Risk	155	123	64	13	3
Adj. CTX, High Risk	145	103	59	21	4
Events					
No CTX, Low Risk	0	25	50	60	61
No CTX, High Risk	0	40	74	85	87
Adj. CTX, Low Risk	0	18	28	37	39
Adj. CTX, High Risk	0	30	43	52	52

In our example we use the `survfit2()` function to create the survival object by utilizing a formula (`dependent variable (outcome) ~ independent variable(s)`). Our formula defines `Treatment` and `RiskGroup` as independent variables.