demo

Tristal Li

2023-03-29

Import packages and dataset

```
# Uncomment the following line to install the packages
#install.packages(c("tidyverse", "survival", "ggsurvfit", "condSURV", "ggfortify"))
library(tidyverse)
```

```
## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
## ✓ dplyr     1.1.0     ✓ readr     2.1.4
## ✓ forcats   1.0.0     ✓ stringr   1.5.0
## ✓ ggplot2   3.4.1     ✓ tibble    3.1.8
## ✓ lubridate 1.9.2     ✓ tidyr     1.3.0
## ✓ purrr     1.0.1
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
## ℹ Use the  ]8;;http://conflicted.r-lib.org/ conflicted package ]8;;  to force all conflicts to become er
rors
```

```
library(survival)
library(ggsurvfit)
library(condSURV)
library(ggfortify)
colon_orig = read.csv("colon.csv")
```

Data summary

Chemotherapy for Stage B/C colon cancer

```
head(colon_orig)
```

```
##   id study      rx sex age obstruct perfor adhere nodes status differ extent
## 1  1     1 Lev+5FU   1  43        0      0      0     5      1      2      3
## 2  1     1 Lev+5FU   1  43        0      0      0     5      1      2      3
## 3  2     1 Lev+5FU   1  63        0      0      0     1      0      2      3
## 4  2     1 Lev+5FU   1  63        0      0      0     1      0      2      3
## 5  3     1     Obs   0  71        0      0      1     7      1      2      2
## 6  3     1     Obs   0  71        0      0      1     7      1      2      2
##   surg node4 time etype
## 1    0     1 1521     2
## 2    0     1  968     1
## 3    0     0 3087     2
## 4    0     0 3087     1
## 5    0     1  963     2
## 6    0     1  542     1
```

Columns:

id:	id
study:	1 for all patients
rx:	Treatment - Obs(ervation), Lev(amisole), Lev(amisole)+5-FU
sex:	1=male
age:	in years
obstruct:	obstruction of colon by tumour
perfor:	perforation of colon
adhere:	adherence to nearby organs
nodes:	number of lymph nodes with detectable cancer
time:	days until event or censoring
status:	censoring status
differ:	differentiation of tumour (1=well, 2=moderate, 3=poor)
extent:	Extent of local spread (1=submucosa, 2=muscle, 3=serosa, 4=contiguous structures)
surg:	time from surgery to registration (0=short, 1=long)
node4:	more than 4 positive lymph nodes
etype:	event type: 1=recurrence,2=death

```
# Only include death event data
colon = filter(colon_orig, etype==2)[-16]
```

Important data for survival analysis:

1. `status` : *censor* data, censoring occurs when the event of interest is not observed for some subjects before the study is terminated.
2. `time` : *time to event* or censoring

Kaplan Meier Analysis

First, use Surv() to build the standard survival object.

```
km = with(colon, Surv(time, status))
head(km,80)
```

```
##  [1] 1521  3087+  963   293   659  1767   420  3192+ 3173+ 3308+ 2908+ 3309+
## [13] 2085  2910  2754+ 3214+  406   522   887  3329+ 2789   739   709  2969+
## [25] 2889+ 1772   384   968   218   133  3238+ 3019+ 1745  2527  1387  3024+
## [37]  570  2815+ 2901+  553   905  3030+  685  2740+ 2899+ 2598+ 2840+ 2802+
## [49] 2781+  833  1290  1620  2765+ 2708+ 2737+ 1178  2765+ 2883+ 2679+ 2925+
## [61]  472  2772+  474  2739+  365  2653+ 2726+  774  3078+ 2133  2127   400
## [73] 3185+ 1237  1219  3017+  806  2985+ 2969+ 1995
```

Note: “+” after the time in the print out of `km` indicates censoring.

Next, produce the Kaplan-Meier estimates of the probability of survival over time.

```
km_fit = survfit(Surv(time, status) ~ 1, data=colon)
# Print the estimate of survival probability for some selected time
summary(km_fit, times = c(8,24,62,100*(1:10)))
```

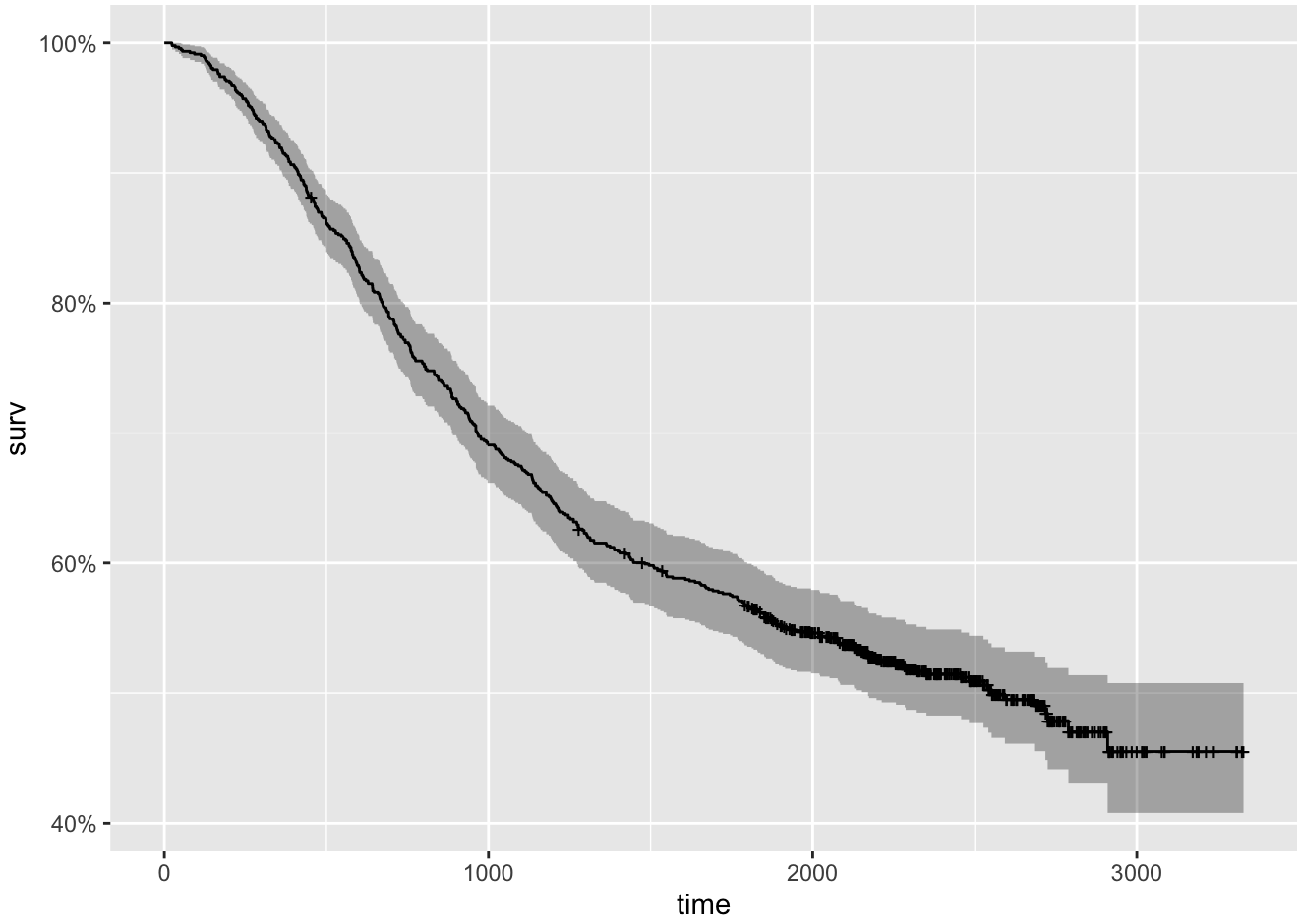
```
## Call: survfit(formula = Surv(time, status) ~ 1, data = colon)
##
##  time n.risk n.event survival std.err lower 95% CI upper 95% CI
##     8    929       0    1.000 0.00000        1.000        1.000
##    24    928       2    0.998 0.00152        0.995        1.000
##    62    923       4    0.994 0.00263        0.988        0.999
##   100    921       2    0.991 0.00303        0.985        0.997
##   200    902      19    0.971 0.00551        0.960        0.982
##   300    873      29    0.940 0.00781        0.925        0.955
##   400    842      32    0.905 0.00961        0.887        0.924
##   500    799      41    0.861 0.01135        0.839        0.884
##   600    768      31    0.828 0.01239        0.804        0.852
##   700    731      37    0.788 0.01342        0.762        0.815
##   800    699      32    0.753 0.01415        0.726        0.782
##   900    674      25    0.726 0.01463        0.698        0.756
##  1000    641      33    0.691 0.01517        0.662        0.721
```

Interpretation of this summary table:

1. `time` : time point t
2. `n.risk` : the number of subjects at risk immediately before the time point t. Being “at risk” at time t means that the subject has not had an event before time t, and is not censored before or at time t.
3. `n.event` : the number of subjects who have events at time t.
4. `survival` : the proportion of survival (probability)
5. `std.err` : the standard error of the estimated survival.
6. `lower 95% CI` and `upper 95% CI` : lower and upper 95% confidence bounds for the proportion of survival.

Now, we can have a simplest Kaplan Meier curve using `autoplot()` function in `{ggfortify}` package

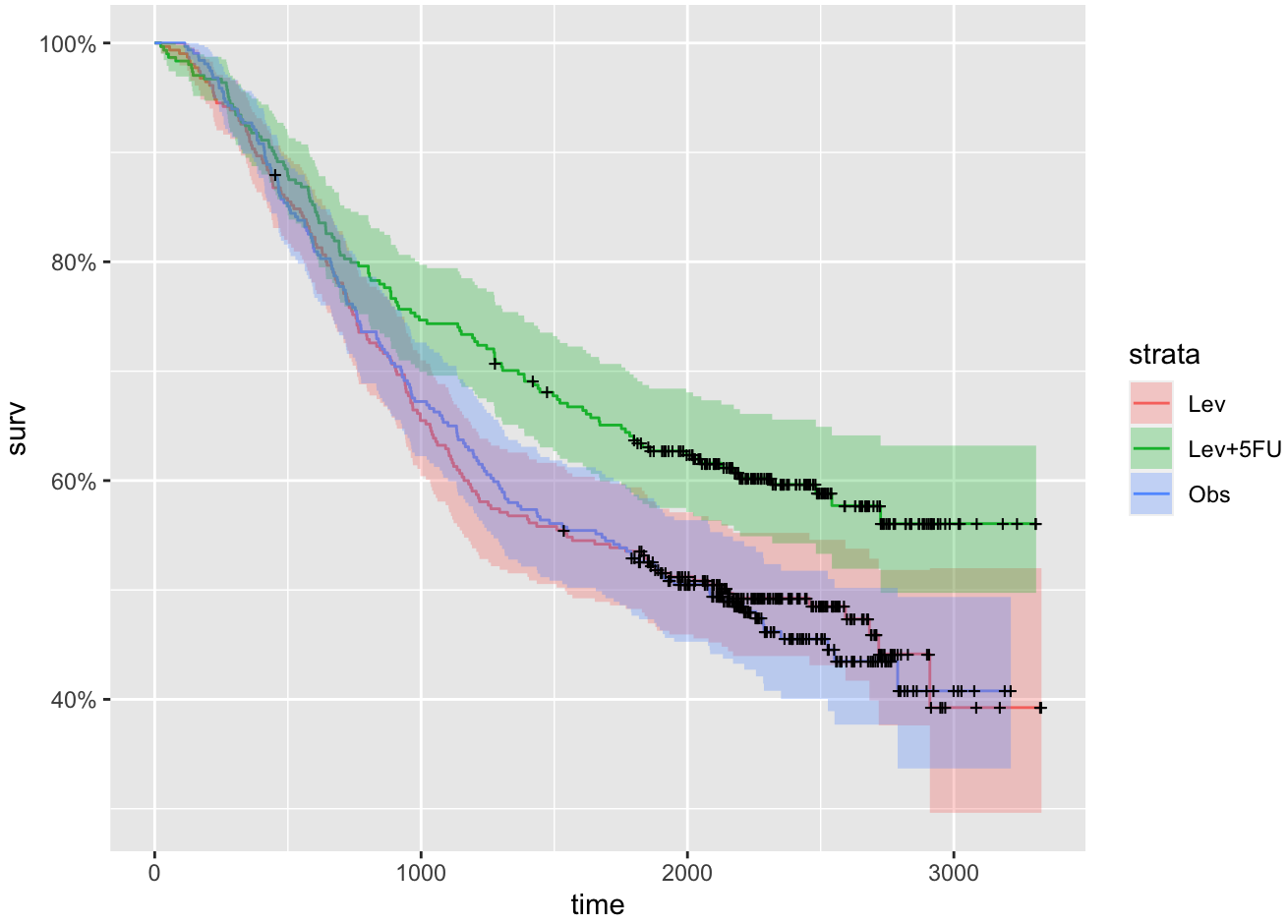
```
autoplot(km_fit)
```



Y-axis: Probability of survival
X-axis: Time

We can add some categorical factors to produce multiple survival curves in the same plot and compare the groups.

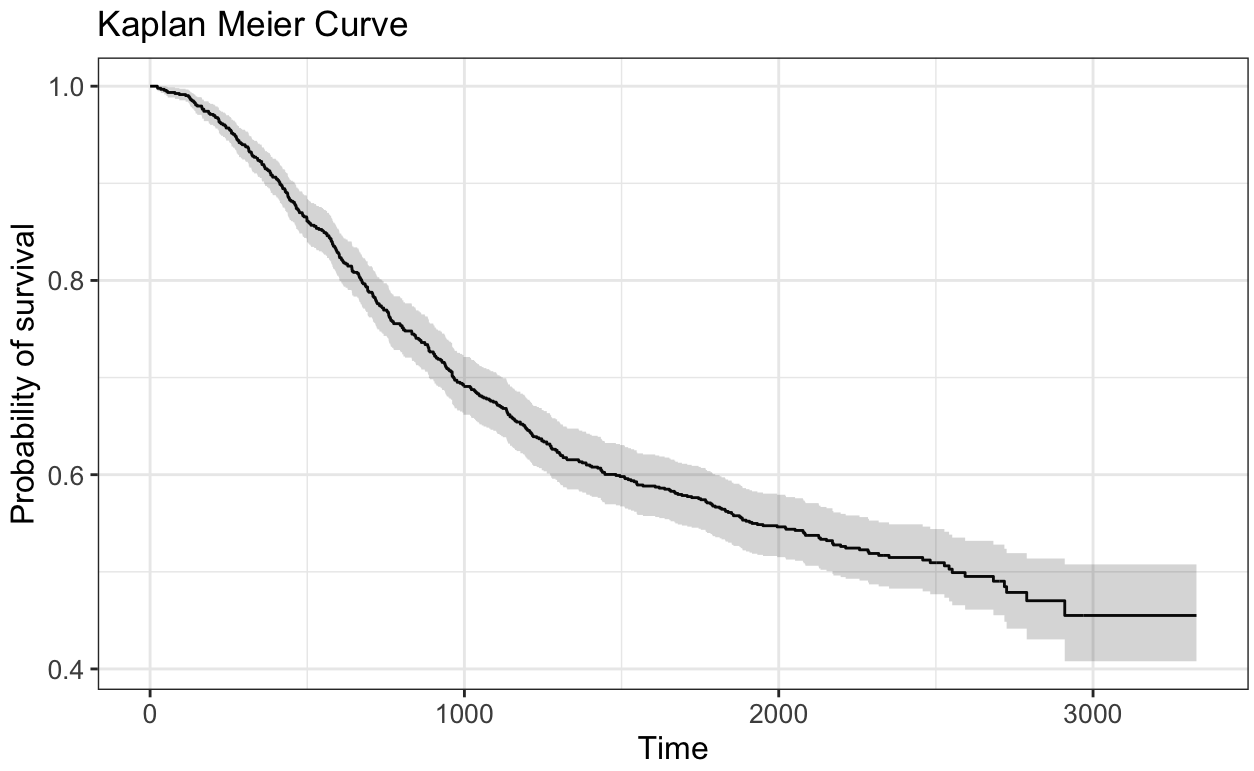
```
km_rx_fit = survfit(Surv(time, status) ~ rx, data=colon)
autoplot(km_rx_fit)
```



From the graph, we can see that patients using Lev+5FU treatment has higher probability of survival within certain time. Maybe infer effective treatment?

To customize your survival curve, we can use `survfit2()` function in the `{ggsurvfit}` package. The methods are same as `ggplot2` . You can also the numbers at risk (same interpretation as the summary table above) in a table below the x-axis.

```
survfit2(Surv(time, status) ~ 1, data = colon) %>%
  ggsurvfit() +
  labs(
    x = "Time",
    y = "Probability of survival",
    title = "Kaplan Meier Curve"
  ) +
  add_confidence_interval() +
  add_risktable()
```



At Risk	929	641	456	17
Events	0	287	420	452

Cox Proportional Hazards Model

The model is based on hazard function (https://en.wikipedia.org/wiki/Failure_rate#hazard_function (https://en.wikipedia.org/wiki/Failure_rate#hazard_function)), here we denote as $\lambda(t)$. Let $X_i = (X_{i1}, \ldots, X_{ip})$ be the realized values of the covariates for subject i. The hazard function for the Cox proportional hazards model at time t for subject i with covariate vector (explanatory variables) X_i is:

$$\lambda(t|X_i) = \lambda_0(t)\exp(\beta_1 X_{i1} + \ldots + \beta_p X_{ip}) = \lambda_0(t)\exp(X_i \cdot \beta)$$

For more knowledge about the model: https://en.wikipedia.org/wiki/Proportional_hazards_model (https://en.wikipedia.org/wiki/Proportional_hazards_model)

Fit a model with all the possible covariates:

```
cox = coxph(Surv(time, status) ~ ., data = colon[-2]) #delete study column b/c it is a constant
summary(cox)
```

```
## Call:
## coxph(formula = Surv(time, status) ~ ., data = colon[-2])
##
##   n= 888, number of events= 430
##    (41 observations deleted due to missingness)
##
##                 coef  exp(coef)   se(coef)      z Pr(>|z|)
## id         0.0001492  1.0001492  0.0001841  0.810 0.417829
## rxLev+5FU -0.3167534  0.7285103  0.1247648 -2.539 0.011123 *
## rxObs      0.0456791  1.0467385  0.1144918  0.399 0.689913
## sex        0.0045586  1.0045690  0.0975020  0.047 0.962710
## age        0.0075640  1.0075926  0.0041787  1.810 0.070277 .
## obstruct   0.2726471  1.3134367  0.1205327  2.262 0.023696 *
## perfor     0.0177149  1.0178727  0.2700752  0.066 0.947702
## adhere     0.1666497  1.1813404  0.1319186  1.263 0.206490
## nodes      0.0438518  1.0448275  0.0153894  2.849 0.004379 **
## differ     0.1373679  1.1472501  0.1009110  1.361 0.173426
## extent     0.4471670  1.5638755  0.1185709  3.771 0.000162 ***
## surg       0.2443003  1.2767277  0.1063515  2.297 0.021613 *
## node4      0.6762320  1.9664542  0.1434093  4.715 2.41e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##           exp(coef) exp(-coef) lower .95 upper .95
## id           1.0001     0.9999    0.9998    1.0005
## rxLev+5FU    0.7285     1.3727    0.5705    0.9303
## rxObs        1.0467     0.9553    0.8363    1.3101
## sex          1.0046     0.9955    0.8298    1.2161
## age          1.0076     0.9925    0.9994    1.0159
## obstruct     1.3134     0.7614    1.0371    1.6634
## perfor       1.0179     0.9824    0.5995    1.7282
## adhere       1.1813     0.8465    0.9122    1.5299
## nodes        1.0448     0.9571    1.0138    1.0768
## differ       1.1473     0.8716    0.9414    1.3981
## extent       1.5639     0.6394    1.2396    1.9730
## surg         1.2767     0.7833    1.0365    1.5726
## node4        1.9665     0.5085    1.4846    2.6047
##
## Concordance= 0.672  (se = 0.013 )
## Likelihood ratio test= 140.3  on 13 df,   p=<2e-16
## Wald test            = 148.7  on 13 df,   p=<2e-16
## Score (logrank) test = 159.2  on 13 df,   p=<2e-16
```

`exp(coef)` : the harzard ratio ($\exp(\beta_i)$ in the model)

Testing the assumption of the model

```
print(cox.zph(cox))
```

```
##             chisq df       p
## id        0.61426  1 0.43319
## rx        2.78908  2 0.24795
## sex       0.13030  1 0.71812
## age       0.96503  1 0.32592
## obstruct  6.49947  1 0.01079
## perfor    0.69777  1 0.40353
## adhere    0.01999  1 0.88757
## nodes     0.41030  1 0.52182
## differ   13.28508  1 0.00027
## extent    4.14389  1 0.04179
## surg      0.00247  1 0.96033
## node4     4.80493  1 0.02838
## GLOBAL   38.20776 13 0.00027
```

Having very small p values (less than our threshold of 0.05) indicates that there are time dependent coefficients which you need to take care of. Here, we choose to delete these variables in our final model.

```
cox_final = coxph(Surv(time, status) ~ ., data = colon[-c(2, 6, 11, 14)])
summary(cox_final)
```

```
## Call:
## coxph(formula = Surv(time, status) ~ ., data = colon[-c(2, 6,
##     11, 14)])
##
##   n= 911, number of events= 441
##    (18 observations deleted due to missingness)
##
##                 coef  exp(coef)   se(coef)      z Pr(>|z|)
## id         0.0001402  1.0001402  0.0001829  0.766   0.4435
## rxLev+5FU -0.3061796  0.7362544  0.1227803 -2.494   0.0126 *
## rxObs      0.0815726  1.0849920  0.1120090  0.728   0.4664
## sex       -0.0300386  0.9704081  0.0958598 -0.313   0.7540
## age        0.0048852  1.0048971  0.0040840  1.196   0.2316
## perfor     0.0374835  1.0381949  0.2674761  0.140   0.8886
## adhere     0.2045510  1.2269740  0.1294437  1.580   0.1141
## nodes      0.0925424  1.0969596  0.0092099 10.048  < 2e-16 ***
## extent     0.5182960  1.6791639  0.1132642  4.576 4.74e-06 ***
## surg       0.2589425  1.2955593  0.1052988  2.459   0.0139 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##           exp(coef) exp(-coef) lower .95 upper .95
## id           1.0001     0.9999    0.9998    1.0005
## rxLev+5FU    0.7363     1.3582    0.5788    0.9366
## rxObs        1.0850     0.9217    0.8711    1.3514
## sex          0.9704     1.0305    0.8042    1.1710
## age          1.0049     0.9951    0.9969    1.0130
## perfor       1.0382     0.9632    0.6146    1.7537
## adhere       1.2270     0.8150    0.9520    1.5813
## nodes        1.0970     0.9116    1.0773    1.1169
## extent       1.6792     0.5955    1.3449    2.0965
## surg         1.2956     0.7719    1.0540    1.5925
##
## Concordance= 0.661  (se = 0.013 )
## Likelihood ratio test= 121.6  on 10 df,   p=<2e-16
## Wald test            = 145.9  on 10 df,   p=<2e-16
## Score (logrank) test = 151.5  on 10 df,   p=<2e-16
```

```
print(cox.zph(cox_final))
```

```
##          chisq df     p
## id     1.14216  1 0.285
## rx     2.39977  2 0.301
## sex    0.58862  1 0.443
## age    0.60568  1 0.436
## perfor 0.70397  1 0.401
## adhere 0.01202  1 0.913
## nodes  0.00315  1 0.955
## extent 3.18878  1 0.074
## surg   0.01896  1 0.890
## GLOBAL 9.04847 10 0.528
```

Now we are good to go!

References

[1] Zabor, E. C. (2018, August 30). Survival analysis in R. Retrieved March 26, 2023, from https://www.emilyzabor.com/tutorials/survival_analysis_in_r_tutorial.html (https://www.emilyzabor.com/tutorials/survival_analysis_in_r_tutorial.html)

[2] Rickert, J. (2017, September 25). Survival analysis with R. R Views. Retrieved March 26, 2023, from https://rviews.rstudio.com/2017/09/25/survival-analysis-with-r/ (https://rviews.rstudio.com/2017/09/25/survival-analysis-with-r/)

````
---
title: "demo"
author: "Tristal Li"
date: "`r Sys.Date()`"
output: html_document
---

```{r setup, include=FALSE}
knitr::opts_chunk$set(echo = TRUE)
```

# Import packages and dataset

```{r}
# Uncomment the following line to install the packages
#install.packages(c("tidyverse", "survival", "ggsurvfit", "condSURV", "ggfortify"))
library(tidyverse)
library(survival)
library(ggsurvfit)
library(condSURV)
library(ggfortify)
colon_orig = read.csv("colon.csv")
```

## Data summary

Chemotherapy for Stage B/C colon cancer

```{r}
head(colon_orig)
```

Columns:
![data_description](data_dspt.png)

```{r}
# Only include death event data
colon = filter(colon_orig, etype==2)[-16]
```


Important data for survival analysis:\
1. `status`: *censor* data, censoring occurs when the event of interest is not observed for some subjects before the study is terminated.\
2. `time`: *time to event* or censoring

# Kaplan Meier Analysis

First, use Surv() to build the standard survival object.

```{r}
km = with(colon, Surv(time, status))
head(km,80)
```

Note: "+" after the time in the print out of `km` indicates censoring.

Next, produce the Kaplan-Meier estimates of the probability of survival over time.
````

```{r}
km_fit = survfit(Surv(time, status) ~ 1, data=colon)
# Print the estimate of survival probability for some selected time
summary(km_fit, times = c(8,24,62,100*(1:10)))
```

Interpretation of this summary table:\
1. `time`: time point t\
2. `n.risk`: the number of subjects at risk immediately before the time point t. Being "at risk" at time t means that the subject has not had an event before time t, and is n censored before or at time t.\
3. `n.event`: the number of subjects who have events at time t.\
4. `survival`: the proportion of survival (probability)\
5. `std.err`: the standard error of the estimated survival. \
6. `lower 95% CI` and `upper 95% CI`: lower and upper 95% confidence bounds for the proportion of survival.

Now, we can have a simplest Kaplan Meier curve using `autoplot()` function in `{ggfortify}` package

```{r}
autoplot(km_fit)
```

Y-axis: Probability of survival\
X-axis: Time

We can add some categorical factors to produce multiple survival curves in the same plo and compare the groups.

```{r}
km_rx_fit = survfit(Surv(time, status) ~ rx, data=colon)
autoplot(km_rx_fit)
```

From the graph, we can see that patients using Lev+5FU treatment has higher probability of survival within certain time. Maybe infer effective treatment?

To customize your survival curve, we can use `survfit2()` function in the `{ggsurvfit}` package. The methods are same as `ggplot2`. You can also the numbers at risk (same interpretation as the summary table above) in a table below the x-axis.

```{r}
survfit2(Surv(time, status) ~ 1, data = colon) %>%
  ggsurvfit() +
  labs(
    x = "Time",
    y = "Probability of survival",
    title = "Kaplan Meier Curve"
  ) +
  add_confidence_interval() +
  add_risktable()
```

Cox Proportional Hazards Model

The model is based on hazard function (https://en.wikipedia.org/wiki/Failure_rate#hazard_function), here we denote as $\lambda(t)$. Let $X_i = (X_{i1}, … , X_{ip})$ be the realized values of the covariates for subject i. The hazard function $\lambda(t)$ for the Cox proportional hazards model at time t for subject i with covariate vector (explanatory variables) X_i is:

$$\lambda(t|X_i)=\lambda_0(t)\exp(\beta_1X_{i1}+...+\beta_pX_{ip})=\lambda_0(t)\exp(X_i\cdot\beta)$$

For more knowledge about the model:
https://en.wikipedia.org/wiki/Proportional_hazards_model

Fit a model with all the possible covariates:

```{r}
cox = coxph(Surv(time, status) ~ ., data = colon[-2]) #delete study column b/c it is a constant
summary(cox)
```

`exp(coef)`: the harzard ratio ($\exp(\beta_i)$ in the model)

Testing the assumption of the model

```{r}
print(cox.zph(cox))
```

Having very small p values (less than our threshold of 0.05) indicates that there are time dependent coefficients which you need to take care of. Here, we choose to delete these variables in our final model.

```{r}
cox_final = coxph(Surv(time, status) ~ ., data = colon[-c(2, 6, 11, 14)])
summary(cox_final)
```

```{r}
print(cox.zph(cox_final))
```

Now we are good to go!

References

[1] Zabor, E. C. (2018, August 30). Survival analysis in R. Retrieved March 26, 2023, from https://www.emilyzabor.com/tutorials/survival_analysis_in_r_tutorial.html

[2] Rickert, J. (2017, September 25). Survival analysis with R. R Views. Retrieved March 26, 2023, from https://rviews.rstudio.com/2017/09/25/survival-analysis-with-r/