Group Project 1

Huanyu Chen, Shaolei Ma, Ruiqi Xue

2024-02-18

Proportional-Hazard Assumption

Under proportional-hazards assumption, the hazard function (Cox model) can be written as:

$$h(t|x) = h_0(t)exp(\beta' x)$$

where t is the time, x the vector of covariates, β the vector of regression coefficients, $h_0(t)$ is the baseline hazard function. Then, the survival function is

$$S(t|x) = exp[-H_0(t)exp(\beta' x)]$$

where

$$H_0(t) = \int_0^t h_0(u)du$$

Thus, the distribution function is

$$F(t|x) = 1 - exp[-H_0(t)exp(\beta' x)]$$

Let Y be a random variable with distribution function F, then $U = F(Y) \sim U[0,1]$, $(1-U) \sim U[0,1]$, i.e.

$$U = exp[-H_0(t)exp(\beta' x)] \sim U[0,1]$$

if $h_0(t) > 0$ for all t, then H_0 can be inverted and the survival time T of the model can be written as

$$T = H_0^{-1}[-log(U)exp(-\beta' x)]$$

where $U \sim U[0,1]$.

To simply the problem, here we only consider one covariate x, which indicates whether the sample belongs to the control arm ($x = 0$) or the treatment arm ($x = 1$), and set a negative β under the assumption that the treatment has a consistent positive effect.

Now, we only need to know H_0^{-1} to simulate the survival time. To do so, we consider two commonly used survival time distributions: exponential and Weibull distribution.

For exponential distribution with scale parameter $\lambda > 0$, the possibility density function is

$$f_0 = \lambda exp(-\lambda t)$$

Then,

$$F_0(t) = 1 - exp(-\lambda t)$$
$$S_0(t) = 1 - F_0(t) = exp(-\lambda t)$$
$$H_0(t) = -log(S_0(t)) = \lambda t$$
$$h(t) = H_0'(t) = \lambda > 0$$

$$H_0^{-1}(t) = \lambda^{-1}t$$

Thus,

$$T = -\lambda^{-1}log(U)exp(-\beta' x)$$

where $U \sim U[0, 1]$.

For Weibull distribution with the scale parameter λ, and is the shape parameter γ, the possibility density function is

$$f_0 = \lambda\gamma t^{\gamma-1}exp(-\lambda t^{\gamma})$$

Then,

$$F_0(t) = 1 - exp(-\lambda t^{\gamma})$$

$$S_0(t) = 1 - F_0(t) = exp(-\lambda t^{\gamma})$$

$$H_0(t) = -log(S_0(t)) = \lambda t^{\gamma}$$

$$h(t) = H_0'(t) = \lambda\gamma t^{(\gamma-1)} > 0$$

$$H_0^{-1}(t) = (\lambda^{-1}t)^{1/\gamma}$$

Thus,

$$T = (-\lambda^{-1}log(U)exp(-\beta' x))^{1/\gamma}$$

where $U \sim U[0, 1]$.

We can write the simulation process as follows:

```r
simulate_func = function(n, baseline, lambda, gamma = NULL, coveff)
{
  # Simulate treatment indicator variable
  x = rbinom(n = n, size = 1, prob = 0.5)
  # Draw from a U(0,1) random variable
  u = runif(n)
  # Simulate survival times depending on the baseline hazard
  if (baseline == "Exponential") {
    t = -log(u) / (lambda * exp(x * coveff))
    # Set the administrative censoring time to guarantee a censor rate of 0.2
    censor_time = qexp(0.8, rate = lambda)
  } else if(baseline == "Weibull") {
    t = (-log(u) / (lambda * exp(x * coveff)))^(1 / gamma)
    censor_time = qweibull(0.8, shape = gamma, scale = lambda)
  }
  # Make event indicator variable applying administrative censoring at t = 5
  d = as.numeric(t < censor_time)
  t = pmin(t, censor_time)
  # Return a tibble object
  if (baseline == "Exponential") {
    return(tibble(x, t, d, n, baseline, lambda, coveff))
  } else if(baseline == "Weibull") {
    return(tibble(x, t, d, n, baseline, lambda, gamma, coveff))
  }
}
```

To observe the potential relevance between test performance and number of samples (n), parameter value (λ, γ), and coefficient β, we set $n = 50$, 100, 200, $\lambda = 0$, 0.5, 1, $\gamma = 0$, 0.5, 1, and $\beta = 0$, 1, 2. We repeat 50 times for each value setting. The generation process is written as follows:

```r
exp_param_df = expand.grid(iteration = c(1:50), n = c(50, 100, 200),
            lambda = c(0.5, 0.8, 1), beta = c(-0.5, -1, -5))
wei_param_df = expand.grid(iteration = c(1:50), n = c(50, 100, 200),
            lambda = c(0.5, 0.8, 1), gamma = c(0.5, 0.8, 1),
            beta = c(-0.5, -1, -5))

exp_results =
  mapply(simulate_func, n = exp_param_df$n, baseline = "Exponential",
         lambda = exp_param_df$lambda, coveff = exp_param_df$beta)
wei_results =
  mapply(simulate_func, n = wei_param_df$n, baseline = "Weibull",
         lambda = wei_param_df$lambda, gamma = wei_param_df$gamma,
         coveff = wei_param_df$beta)

ph_exp_df = tibble()
ph_wei_df = tibble()

for(i in 1:ncol(exp_results))
{
  a = exp_results[, i]
  ph_exp_df = cbind.data.frame(x = a$x, t = a$t, d = a$d, n = a$n,
                               baseline = "Exponential", lambda = a$lambda,
                               beta = a$coveff) |> as_tibble() |>
    nest(data = c(x : d)) |> rbind(ph_exp_df)
}

for(i in 1:ncol(wei_results))
{
  a = wei_results[, i]
  ph_wei_df =
    cbind.data.frame(x = a$x, t = a$t, d = a$d, n = a$n, baseline = "Weibull",
          lambda = a$lambda, gamma = a$gamma, beta = a$coveff) |>
    as_tibble() |> nest(data = c(x : d)) |> rbind(ph_wei_df)
}

ph_exp_df = ph_exp_df |> nest(simulations = c(data))
ph_wei_df = ph_wei_df |> nest(simulations = c(data))
```

Under different settings, we want to test the H_0 : there is no difference in survival between the treatment and control arm. Therefore, we use three different log-rank tests and compare the test power at the 0.05 significance level.

```r
pwr_func = function(list_df, n = 50)
{
  test1_reject = 0
  test2_reject = 0
  test3_reject = 0
  for(j in 1:nrow(list_df)) {
    dat = list_df |> slice(j) |> unnest(cols = c(data))
    test_results = logrank.maxtest(dat$t, dat$d, dat$x)
    test1_reject = test1_reject +
      ((test_results$tests |> filter(Test == 1) |> pull(p)) < 0.05)
    test2_reject = test2_reject +
```

```r
      ((test_results$tests |> filter(Test == 2) |> pull(p)) < 0.05)
    test3_reject = test3_reject +
      ((test_results$tests |> filter(Test == 3) |> pull(p)) < 0.05)
  }
  return(
    tibble(
      test1_power = test1_reject / n,
      test2_power = test2_reject / n,
      test3_power = test3_reject / n
    )
  )
}
(ph_exp_df =
  ph_exp_df |> mutate(power = map(simulations, pwr_func)) |> unnest(power))
```

```
## # A tibble: 27 x 8
##        n baseline   lambda  beta simulations test1_power test2_power test3_power
##    <dbl> <chr>       <dbl> <dbl> <list>            <dbl>       <dbl>       <dbl>
##  1   200 Exponenti~    1      -5 <tibble>              1        1              1
##  2   100 Exponenti~    1      -5 <tibble>              1        1              1
##  3    50 Exponenti~    1      -5 <tibble>              1        1              1
##  4   200 Exponenti~    0.8    -5 <tibble>              1        1              1
##  5   100 Exponenti~    0.8    -5 <tibble>              1        1              1
##  6    50 Exponenti~    0.8    -5 <tibble>              1        1              1
##  7   200 Exponenti~    0.5    -5 <tibble>              1        1              1
##  8   100 Exponenti~    0.5    -5 <tibble>              1        1              1
##  9    50 Exponenti~    0.5    -5 <tibble>              1        1              1
## 10   200 Exponenti~    1      -1 <tibble>              1        0.98           1
## # i 17 more rows
```

```r
(ph_wei_df =
  ph_wei_df |> mutate(power = map(simulations, pwr_func)) |> unnest(power))
```

```
## # A tibble: 81 x 9
##        n baseline lambda gamma  beta simulations       test1_power test2_power
##    <dbl> <chr>     <dbl> <dbl> <dbl> <list>                  <dbl>       <dbl>
##  1   200 Weibull     1     1      -5 <tibble [50 x 1]>        1           1
##  2   100 Weibull     1     1      -5 <tibble [50 x 1]>        1           1
##  3    50 Weibull     1     1      -5 <tibble [50 x 1]>        1           1
##  4   200 Weibull     0.8   1      -5 <tibble [50 x 1]>        1           1
##  5   100 Weibull     0.8   1      -5 <tibble [50 x 1]>        1           1
##  6    50 Weibull     0.8   1      -5 <tibble [50 x 1]>        1           1
##  7   200 Weibull     0.5   1      -5 <tibble [50 x 1]>        1           1
##  8   100 Weibull     0.5   1      -5 <tibble [50 x 1]>        1           1
##  9    50 Weibull     0.5   1      -5 <tibble [50 x 1]>        0.98        0.9
## 10   200 Weibull     1     0.8    -5 <tibble [50 x 1]>        1           1
## # i 71 more rows
## # i 1 more variable: test3_power <dbl>
```

(reference link: https://spia.uga.edu/faculty_pages/rbakker/pols8501/OxfordThreeNotes.pdf)

Cox proportional hazards model

does not assume a particular baseline hazard function $h_0(t)$ but the proportional relation between groups and baseline.

$$h_i(t) = h_0(t) \exp[\beta_1 X_{i1} + \beta_2 X_{i2} + \ldots + \beta_p X_{ip}]$$

Weibull proportional hazards model

assumes a specific functional form for the hazard rate, which can either increase or decrease over time. Its shape parameter distinctly describes whether the hazard rate is increasing, decreasing, or constant.

$$h_i(t) = \lambda \gamma t^{\gamma-1} \exp[\beta_1 X_{i1} + \beta_2 X_{i2} + \ldots + \beta_p X_{ip}]$$

where λ is the scale parameter, and γ is the shape parameter.

Non-Proportional-Hazard Assumption

Weibull accelerated failure time model

References

Bender, R., Augustin, T., & Blettner, M. (2005). Generating survival times to simulate Cox proportional hazards models. *Statistics in medicine*, 24(11), 1713–1723. https://doi.org/10.1002/sim.2059