PH 716 Applied Survival Analysis

Part VI: Diagnostics of Cox Proportional Hazards Models

Zhiyang Zhou (zhou67@uwm.edu, zhiyanggeezhou.github.io)

2024/Mar/20 21:08:32

Types of residuals

- Cox-Snell residuals: assessing the overall fit of the final model
- Martingale residuals: determining the functional form of a covariate included in the model
- Deviance residuals: detecing outliers
- Schoenfeld residuals residuals: checking the appropriateness of the PH assumption

Cox-Snell residuals

- Inverse cdf theorem: arbitrary r.v. X with cdf $F_X(x) = \Pr(X \leq x) \Rightarrow F_X(X) \sim U(0,1)$
- It follows that $T_i \overset{\text{independent}}{\sim} S_{T_i}(\cdot) \Rightarrow S_{T_i}(T_i) \overset{\text{iid}}{\sim} U(0,1) \Rightarrow \Lambda_{T_i}(T_i) = -\ln S_{T_i}(T_i) \overset{\text{iid}}{\sim} \exp(1)$
- Cox-Snell residuals: $r_{i,\text{CS}} = \widehat{\Lambda}_{T_i}(\widetilde{T}_i)$
  - $\widehat{\Lambda}_{T_i}(\cdot)$: estimated $\Lambda_{T_i}(\cdot)$ given by the Cox PH model
  - $\{(r_{i,\text{CS}}, \Delta_i) : i = 1, \ldots, n\}$ is a right-censored dataset
    - $r_{i,\text{CS}} = \min(H_i, \widehat{\Lambda}_{T_i}(C_i)) \Leftarrow \widetilde{T}_i = \min(T_i, C_i)$ and monotonically ascending $\Lambda_{T_i}(\cdot)$
      - $H_i = \widehat{\Lambda}_{T_i}(T_i) \approx \Lambda_{T_i}(T_i) \Rightarrow \Lambda_{H_i}(t) \approx t$
  - $\widehat{\Lambda}_{H_i,\text{NA}}(t) \approx t$
    - $\widehat{\Lambda}_{H_i,\text{NA}}(\cdot)$: NA estimator of $\Lambda_{H_i}(\cdot)$ based on $\{(r_{i,\text{CS}}, \Delta_i) : i = 1, \ldots, n\}$
- Cox-Snell residual plot
  - For uncensored subjects
    - Compare $r_{i,\text{CS}}$ to exp(1) samples via Q-Q plot
    - Or, plot $\widehat{\Lambda}_{H_i,\text{NA}}(\tilde{t}_i)$ against $\tilde{t}_i$
  - Used to diagnose poor model fit
  - No insight into how model assumptions are violated

Ex. 6.1 [KM, Example 11.1]

- This multi-center acute leukemia study consists of 137 patients with acute myelocytic leukemia (AML) or acute lymphoblastic leukemia (ALL) aged 7 to 52 from March 1, 1984 to June 30, 1989 at four institutions.
- The disease-free survival time (`t2`) on study is defined as time (in days) to relapse or death
- `d3` is the disease free survival indicator: 1 - Dead or Relapsed, 0 - Alive Disease Free.
- Focus on effects of the following 9 covariates on disease-free survival:
  - `z1`: Patient age in years.

- `z2`: Donor age in years.
- `z3`: Patient sex: 1 - Male, 0 - Female.
- `z4`: Doner sex: 1 - Male, 0 - Female.
- `z5`: Patient Cytomegalovirus (CMV) status: 1 - CMV positive, 0 - CMV negative.
- `z6`: Donor CMV status: 1 - CMV positive, 0 - CMV negative.
- `z7`: Waiting time to transplant in days.
- `z8`: French–American–British classification (FAB): 1 - FAB Grade 4 or 5 and AML, 0 - otherwise.
- `z10`: Methotrexate (MTX): used as a Graft-Versus-Host-Prophylactic 1 - Yes, 0 - No.

```
options(digits=4)
library(survival)
# model fitting
data.ex55 = read.csv("bmt.csv")
fit.ex55 <- coxph(Surv(t2,d3) ~ z1+z2+z3+z4+z5+z6+z7+z8+z10, data=data.ex55)

# Cox-Snell residual
r.cs = data.ex55$d3-residuals(fit.ex55, type='martingale') # Cox-Snell

# Cox-Snell residual plot
set.seed(2024)
exp.rnd = rexp(10000)
qqplot(
  x = exp.rnd, y = r.cs[as.logical(data.ex55$d3)],
  xlab = "Theoretical Quantiles", ylab = "Sample Quantiles"
)
qqline(r.cs[as.logical(data.ex55$d3)], distribution = qexp)
# Or
cum.haz.r.cs <- basehaz(coxph(Surv(r.cs, d3)~1, data=data.ex55), centered = FALSE)
plot(
  x=cum.haz.r.cs[,2], y=cum.haz.r.cs[,1],
  xlab='t', ylab='Cumulative hazard of r.cs'
)
abline(a=0,b=1,col='red')
```

Martingale residuals

- Martingale residuals: $r_{i,\mathrm{M}} = \Delta_i - r_{i,\mathrm{CS}}$
  - Zero-sum: $\sum_i r_{i,\mathrm{M}} = 0$
  - Estimated excess number of events seen in the data but not predicted by the model
    - Positive $r_{i,\mathrm{M}}$: the patient died sooner than expected
    - Negative $r_{i,\mathrm{M}}$: the patient lived longer than expected (or were censored)
  - Analogous to the residuals in linear models
    - But asymmetric and unbounded from below
- Examining the best functional form for a given covariate
  1. Partition covariates into two parts:
     - $x_{i2}, \ldots, x_{ip}$: for which we know their proper functional form, say $f_2(\cdot), \ldots, f_p(\cdot)$, respectively
     - x_{i1}: a single covariate for which there is a potential functional form $f_1(\cdot)$
  2. If $f_1(\cdot)$ is best for x_{i1} and x_{i1} is independent of other covariates, then fit the Cox PH model without the jth covariate $\lambda_{T_i}(t) = \lambda_0(t)\exp\{\sum_{j=2}^{p} f_j(x_{ij})\beta_j\}$ and compute martingale residuals $r_{i,\mathrm{M}}$
  3. Confirm f_1 via the scatterplot of $r_{i,\mathrm{M}}$ against x_{i1} with a fitted loess (locally estimated scatterplot smoothing) line
     - If the fitted loess line is linear, then no transformation of x_{i1} is needed; otherwise, a discretized version/transformation of x_{i1} is indicated

4. Fitting $\lambda_{T_i}(t) = \lambda_0(t)\exp\{\sum_{j=2}^{p} f_j(x_{ij})\beta_j\}\exp\{f_1(x_{i1}\beta_1)\}$ and check the scatterplot of updated $r_{i,\mathrm{M}}$ against x_{i1} with a fitted loess (locally estimated scatterplot smoothing) line
   - If the fitted loess line is overlapping the x-axis, then no transformation of x_{i1} is needed.

- Theoretical notes:
  - Why would the martingale residuals reveal the correct functional forms of covariates?
    - Because $E(r_{i,\mathrm{M}}) \approx (n_D/n)\{f_1(x_{i1}) - C\}$ [KM, pp. 362]
      - n_D/n: the ratio of total number of events to total number of subjects
      - C: a constant
  - Why is the residual bearing such a name?
    - Martingale: a stochastic process $M(t)$ such that $\mathrm{E}\{M(t)\} = 0$ and $\mathrm{E}\{M(t) \mid M(s)\} = M(s)$ for all $s < t$
    - $r_{i,\mathrm{M}}$ obtained by evaluating a martingale at $\tilde{t}_i$
  - The zero-sum of martingale residuals
    - Specific for Cox PH model with the Breslow estimator for the baseline cumulative hazard
    - Proof: $\sum_i r_{i,\mathrm{CS}} = \sum_i \sum_{k:t_k \le \tilde{t}_i} \frac{d_k \exp(\sum_j x_{ij}\hat{\beta}_j)}{\sum_{\ell \in \mathcal{R}(t_k)} \exp(\sum_j x_{\ell j}\hat{\beta}_j)} = \sum_k \sum_{i \in \mathcal{R}(t_k)} \frac{d_k \exp(\sum_j x_{ij}\hat{\beta}_j)}{\sum_{\ell \in \mathcal{R}(t_k)} \exp(\sum_j x_{\ell j}\hat{\beta}_j)} = \sum_k d_k = \sum_i \delta_i$

Revisit Ex. 6.1

```
options(digits=4)
library(survival)
# [DM, pp. 208] a function to add the smooth curve and confidence limits
smoothSEcurve <- function(yy, xx) {
  # use after a call to "plot"
  # fit a lowess curve and 95% confidence interval curve
  # make list of x values
  xx.list <- min(xx) + ((0:100)/100)*(max(xx) - min(xx))
  # Then fit loess function through the points (xx, yy)
  # at the listed values
  yy.xx <- predict(loess(yy ~ xx), se=T,
  newdata=data.frame(xx=xx.list))
  lines(yy.xx$fit ~ xx.list, lwd=2)
  lines(yy.xx$fit -
  qt(0.975, yy.xx$df)*yy.xx$se.fit ~ xx.list, lty=2)
  lines(yy.xx$fit +
  qt(0.975, yy.xx$df)*yy.xx$se.fit ~ xx.list, lty=2)
}

# model fitting without z1
data.ex55 = read.csv("bmt.csv")
fit.ex55 <- coxph(Surv(t2,d3) ~ z2+z3+z4+z5+z6+z7+z8+z10, data=data.ex55, ties = 'exact')

# Martingale residual plot (for the model without z1) vs. z1
r.m = residuals(fit.ex55, type='martingale')
sum(r.m)
plot(
  x=data.ex55$z1, y=r.m,
  main = 'Martingale residuals \n (for the model without z1) \n versus z1')
smoothSEcurve(r.m, data.ex55$z1)
## indicating a cubic function?

# model fitting with a cubic function of z1
```

```r
fit.ex55.1 <- coxph(Surv(t2,d3) ~ z1+I(z1^2)+I(z1^3)+z2+z3+z4+z5+z6+z7+z8+z10, data=data.ex55)

# Martingale residual plot (for the model with a cubic function of z1) vs. z1
r.m.1 = residuals(fit.ex55.1, type='martingale')
plot(
  x=data.ex55$z1, y=r.m.1,
  main = 'Martingale residual \n (for the model with a cubic function of z1) \n versus z1')
smoothSEcurve(r.m.1, data.ex55$z1)
```

Deviance residuals

- Outlier: an observation for which the outcome is not sufficiently well predicted by the fitted model
- Deviance residuals: $r_{i,\mathrm{D}} = \mathrm{sign}(r_{i,M})\sqrt{-2\{r_{i,M} + \delta_i \ln(\delta_i - r_{i,M})\}}$
  - Symmetrically distributed with expected value 0 (if the fitted model is correct); de-skewed/transformed martingale residuals
    - $r_{i,\mathrm{D}} = 0 \Leftrightarrow r_{i,M} = 0$
    - Inflating $r_{i,\mathrm{D}}$ when $r_{i,M}$ is close to 1
    - Shrinking large negative $r_{i,M}$
  - Analogous to the deviance in GLMs
- Detecting outliers: plotting $r_{i,\mathrm{D}}$ against $\sum_{j=1}^{p} x_{ij}\hat{\beta}_j$ (called linear predictors or risk scores)
  - With moderate (or less) censoring, this plot should look like randomly-distributed noise without discernible pattern
  - Large absolute values of deviance residuals indicating observations that are poorly explained by the model, potentially pointing to outliers or influential points
    - 95% of absolute deviance residuals ≤ 2
    - 99.7% of absolute deviance residuals ≤ 3

Revisit Ex. 6.1

```r
options(digits=4)
library(survival)
# [DM, pp. 208] a function to add the smooth curve and confidence limits
smoothSEcurve <- function(yy, xx) {
  # use after a call to "plot"
  # fit a lowess curve and 95% confidence interval curve
  # make list of x values
  xx.list <- min(xx) + ((0:100)/100)*(max(xx) - min(xx))
  # Then fit loess function through the points (xx, yy)
  # at the listed values
  yy.xx <- predict(loess(yy ~ xx), se=T,
  newdata=data.frame(xx=xx.list))
  lines(yy.xx$fit ~ xx.list, lwd=2)
  lines(yy.xx$fit -
  qt(0.975, yy.xx$df)*yy.xx$se.fit ~ xx.list, lty=2)
  lines(yy.xx$fit +
  qt(0.975, yy.xx$df)*yy.xx$se.fit ~ xx.list, lty=2)
}

# model fitting
fit.ex55.1 <- coxph(
  Surv(t2,d3) ~ z1+I(z1^2)+I(z1^3)+z2+z3+z4+z5+z6+z7+z8+z10,
```

```
  data=data.ex55,
  x = T
)

# Two ways to calculate linear predictors
risk.score.1 = fit.ex55.1$x %*% coef(fit.ex55.1)
risk.score.2 = fit.ex55.1$linear.predictors
sum((risk.score.1-risk.score.2)^2) # seems distinct?

# Deviance residual plot vs. risk scores
r.d = residuals(fit.ex55.1, type='deviance')
plot(
  x=risk.score.1, y=r.d,
  main = 'Deviance residuals \n versus risk scores')
smoothSEcurve(yy=r.d, xx=risk.score.1)

# Potential outliers
(1:nrow(data.ex55))[abs(r.d) > 2]
sum(abs(r.d) > 2)/nrow(data.ex55)
sum(abs(r.d) > 3)/nrow(data.ex55)
```

Schoenfeld residuals

- Schoenfeld residuals: for uncensored subject i and the jth covariate,

$$r_{ij,\mathrm{S}} = x_{ij} - \bar{x}_{\cdot j}$$

  - $\bar{x}_{\cdot j} = \sum_{k \in \text{uncensored subjects}} w_{kj} z_{kj}$ with weights $w_{kj} = \frac{\exp(\sum_{j=1}^{p} x_{kj}\beta_j)}{\sum_{\ell \in \mathcal{R}(\tilde{t}_k)} \exp(\sum_{j=1}^{p} x_{\ell j}\beta_j)}$
- Investigating the PH assumption: plotting $r_{ij,\mathrm{S}}$ versus the covariate x_{ij} for the j covariate
  - Points are centered at zero if the PH assumption holds
- Theoretical note [DM, Sec. 7.2.2]:
  - Schoenfeld residuals are components of the score function
  - $\Rightarrow \sum_{i \in \text{uncensored subjects}} r_{ij,\mathrm{S}} = 0$ for each j

Revisit Ex. 6.1

```
options(digits=4)
library(survival)
# [DM, pp. 208] a function to add the smooth curve and confidence limits
smoothSEcurve <- function(yy, xx) {
  # use after a call to "plot"
  # fit a lowess curve and 95% confidence interval curve
  # make list of x values
  xx.list <- min(xx) + ((0:100)/100)*(max(xx) - min(xx))
  # Then fit loess function through the points (xx, yy)
  # at the listed values
  yy.xx <- predict(loess(yy ~ xx), se=T,
  newdata=data.frame(xx=xx.list))
  lines(yy.xx$fit ~ xx.list, lwd=2)
  lines(yy.xx$fit -
  qt(0.975, yy.xx$df)*yy.xx$se.fit ~ xx.list, lty=2)
  lines(yy.xx$fit +
  qt(0.975, yy.xx$df)*yy.xx$se.fit ~ xx.list, lty=2)
```

```
}

# model fitting
fit.ex55.1 <- coxph(
  Surv(t2,d3) ~ z1+I(z1^2)+I(z1^3)+z2+z3+z4+z5+z6+z7+z8+z10,
  data=data.ex55,
  x=T
)

# Schoenfeld residual plot
r.s = residuals(fit.ex55.1, type='schoenfeld')
par(mfrow=c(2,2))
for (j in c(1,4,9)){
  plot(
    x=fit.ex55.1$x[data.ex55$d3==1,j], y=r.s[,j],
    main = paste0('Schoenfeld residuals \n versus ', colnames(fit.ex55.1$x)[j]),
    xlab = paste0(colnames(fit.ex55.1$x)[j]),
    ylab = 'Schoenfeld residuals'
  )
  smoothSEcurve(yy=r.s[,j], xx=fit.ex55.1$x[data.ex55$d3==1,j])
}
```