

mis_indep_of_Y

2022-07-26

2) Probability of Missing data doesn't depend on Response Y

- Similarly to above, suppose scientific interest focuses on determining $\beta_1, \beta_2, \beta_3, \beta_4$ in the linear model $y_i = \alpha + x_1\beta_1 + x_2\beta_2 + x_3\beta_3 + x_4\beta_4 + \epsilon_i$.
- Here, $\epsilon_i \sim N(0, \sigma^2)$.
- The missingness model gives observations with $X_2 > \text{median}(X_2)$ a different probability of missingness in X_1, X_3, X_4 than for observations with $X_2 \leq \text{median}(X_2)$

```
# Draw data from artificial model specified above
create.data <- function(alpha = 20, beta_1 = 1, beta_2 = 2, beta_3 = 3, beta_4 = 4,
                        sigma2 = 1, n = 200, run = 1) {
  set.seed(seed = run)
  x_1 <- rnorm(n)
  x_2 <- rnorm(n)
  x_3 <- rnorm(n)
  x_4 <- rnorm(n)
  y <- beta_1 * x_1 + beta_2 * x_2 + beta_3 * x_3 + beta_4 * x_4 + alpha + rnorm(n, sd = sqrt(sigma2))
  as.data.frame(cbind("Y" = y, "X_1" = x_1, "X_2" = x_2, "X_3" = x_3, "X_4" = x_4))
}
```

```
data <- create.data()
```

```
head(data)
```

```
##           Y           X_1           X_2           X_3           X_4
## 1 20.96450 -0.6264538  0.4094018  1.0744410 -0.3410670
## 2 33.43197  0.1836433  1.6888733  1.8956548  1.5024245
## 3 23.63707 -0.8356286  1.5865884 -0.6029973  0.5283077
## 4 21.91777  1.5952808 -0.3309078 -0.3908678  0.5421914
## 5 13.36405  0.3295078 -2.2852355 -0.4162220 -0.1366734
## 6 18.32300 -0.8204684  2.4976616 -0.3756574 -1.1367339
```

```
MNAR.make.missing <- function(data, prob_missing_larger = 0.2,
                              prob_missing_smaller = 0.8){
  # Setting up the randomness categories for missingness in X_1, X_3, X_4
  higher <- data %>% filter(X_2 > median(X_2)) %>% select(X_1)
  rx1_larger <- rbinom(nrow(higher), 1, prob_missing_larger)
  rx1_smaller <- rbinom(nrow(data) - nrow(higher), 1, prob_missing_smaller)
  rx3_larger <- rbinom(nrow(higher), 1, prob_missing_larger)
  rx3_smaller <- rbinom(nrow(data) - nrow(higher), 1, prob_missing_smaller)
  rx4_larger <- rbinom(nrow(higher), 1, prob_missing_larger)
  rx4_smaller <- rbinom(nrow(data) - nrow(higher), 1, prob_missing_smaller)
  rx1 <- c(rx1_larger, rx1_smaller)
  rx3 <- c(rx3_larger, rx3_smaller)
  rx4 <- c(rx4_larger, rx4_smaller)
  data <- data %>%
```

```

    arrange(desc(X_2)) %>%
    cbind(rx1, rx3, rx4)
# Implementing the missingness in X_1, X_3, X_4
data <- data %>% mutate(X_1 = case_when(rx1 == 1 ~ as.numeric(NA),
    rx1 == 0 ~ as.numeric(data$X_1))) %>%
    mutate(X_3 = case_when(rx3 == 1 ~ as.numeric(NA),
    rx3 == 0 ~ as.numeric(data$X_3))) %>%
    mutate(X_4 = case_when(rx4 == 1 ~ as.numeric(NA),
    rx4 == 0 ~ as.numeric(data$X_4)))
# Remove setup variables
data <- select(data, -c(rx1, rx3, rx4))
data
}

data_mis <- MNAR.make.missing(data, 0.2, 0.8)

```

Multiple Imputation

```

# Simulate multiple imputation, obtaining estimates and 95% confidence interval.
simulate_MI2 <- function(runs = 100) {
  res <- array(NA, dim = c(5, runs, 3))
  times <- array(NA, dim = c(100, 1, 1))
  dimnames(res) <- list(c("Intercept", "X_1", "X_2", "X_3", "X_4"),
    as.character(1:runs), c("estimate", "2.5%", "97.5%"))
  sim_dataset <- as.data.frame(create.data(n = 200))
  for (run in 1:runs){
    # Note that time is only measured for the MI/imp steps
    # (i.e. filtering, predicting)
    missingness_sim_dataset <- MNAR.make.missing(sim_dataset, 0.2, 0.8)
    start_time <- Sys.time()
    imp_MI <- mice(missingness_sim_dataset, print = FALSE)
    fit <- with(imp_MI, lm(Y ~ X_1 + X_2 + X_3 + X_4))
    end_time <- Sys.time()
    tab <- summary(pool(fit), "all", conf.int = TRUE)
    res[1, run, ] <- as.numeric(tab[1, c("estimate", "2.5 %", "97.5 %")])
    res[2, run, ] <- as.numeric(tab[2, c("estimate", "2.5 %", "97.5 %")])
    res[3, run, ] <- as.numeric(tab[3, c("estimate", "2.5 %", "97.5 %")])
    res[4, run, ] <- as.numeric(tab[4, c("estimate", "2.5 %", "97.5 %")])
    res[5, run, ] <- as.numeric(tab[5, c("estimate", "2.5 %", "97.5 %")])

    times[run, 1, 1] <- as.numeric(end_time - start_time)
  }
  list(res, times)
}

# Run 100 iterations
res_MI2 <- simulate_MI2(100)

# Obtain confidence intervals & estimates for all coefficients, intercept.
apply(res_MI2[[1]], c(1, 3), mean, na.rm = TRUE)

```

```

##           estimate      2.5%      97.5%
## Intercept 20.132687 19.5481163 20.717258
## X_1       1.093180  0.6973546  1.489006

```

| | Raw_Bias | Percent_Bias | Coverage_Rate | Average_Width | RMSE |
|-----------|----------|--------------|---------------|---------------|-------|
| Intercept | 0.133 | 0.663 | 0.95 | 1.169 | 0.233 |
| X_1 | 0.093 | 9.318 | 0.98 | 0.792 | 0.161 |
| X_2 | -0.025 | 1.265 | 0.97 | 1.182 | 0.180 |
| X_3 | 0.077 | 2.554 | 0.97 | 0.661 | 0.133 |
| X_4 | 0.043 | 1.065 | 0.98 | 0.622 | 0.110 |

```
## X_2      1.974707  1.3837934  2.565620
## X_3      3.076629  2.7460809  3.407177
## X_4      4.042583  3.7316625  4.353504

# Mean time for the multiple imputation instances
times <- res_MI2[[2]]
Multiple_Imputation <- mean(times)

# Evaluating imputation method performance for estimating
# all parameters of interest.
res <- res_MI2[[1]]
true <- c(20, 1, 2, 3, 4)

Raw_Bias <- rowMeans(res[, "estimate"]) - true
Percent_Bias <- 100 * abs((rowMeans(res[, "estimate"]) - true) / true)
Coverage_Rate <- rowMeans(res[, "2.5%"] < true & true < res[, "97.5%"])
Average_Width <- rowMeans(res[, "97.5%"] - res[, "2.5%"])
RMSE <- sqrt(rowMeans((res[, "estimate"] - true)^2))
MI_measures <- data.frame(Raw_Bias, Percent_Bias, Coverage_Rate, Average_Width, RMSE)
knitr::kable(round(MI_measures, 3), align = "cccc") %>% kable_styling()
```

Listwise Deletion

```
# Simulate listwise deletion, obtaining estimates and 95% confidence interval.

simulate_LD <- function(runs = 100){
  res <- array(NA, dim = c(5, runs, 3))
  dimnames(res) <- list(c("Intercept", "X_1", "X_2", "X_3", "X_4"),
                        as.character(1:runs), c("estimate", "2.5%", "97.5%"))
  times <- array(NA, dim = c(runs, 1, 1))
  sim_dataset <- as.data.frame(create.data(n = 200))
  # Note that time is only measured for the LD/imp steps (i.e. filtering, predicting)
  for (run in 1:runs){
    missingness_sim_dataset <- MNAR.make.missing(sim_dataset, 0.2, 0.8)
    start_time <- Sys.time()
    filtered_sim_dataset <- missingness_sim_dataset %>%
      select(Y, X_1, X_2, X_3, X_4) %>%
      filter(!is.na(X_1), !is.na(X_3), !is.na(X_4))
    fit <- with(filtered_sim_dataset, lm(Y ~ X_1 + X_2 + X_3 + X_4))
    end_time <- Sys.time()
    times[run, 1, 1] <- as.numeric(end_time - start_time)
    # loop over each variable. Note we do the imputation just ONCE b/c LD is
    # deterministic.
    for (var in 1:5){
      edges <- as.numeric((confint(fit)[var,]))
      estimate <- as.numeric(fit$coefficients)[var]
```

| | average-runtime |
|---------------------|-----------------|
| Multiple Imputation | 0.1891701 |
| ListwiseDeletion | 0.0083455 |

```

    interval <- c(estimate, edges)
    res[var, run, ] <- interval
  }

}
list(res, times)
}

```

```
result_LD <- simulate_LD()
```

```

# Obtain confidence intervals & estimates for all coefficients, intercept.
apply(result_LD[[1]], c(1, 3), mean, na.rm = TRUE)

```

```

##           estimate      2.5%      97.5%
## Intercept 20.114295 19.5798252 20.648765
## X_1       1.064777  0.7446178  1.384937
## X_2       1.966311  1.4257592  2.506864
## X_3       3.014668  2.7245870  3.304749
## X_4       3.982521  3.7122312  4.252812

```

```

# Evaluating imputation method performance for estimating
# all parameters of interest.

```

```

res <- result_LD[[1]]
true <- c(20, 1, 2, 3, 4)
Raw_Bias <- rowMeans(res[, "estimate"]) - true
Percent_Bias <- 100 * abs((rowMeans(res[, "estimate"]) - true) / true)
Coverage_Rate <- rowMeans(res[, "2.5%"] < true & true < res[, "97.5%"])
Average_Width <- rowMeans(res[, "97.5%"] - res[, "2.5%"])
RMSE <- sqrt(rowMeans((res[, "estimate"] - true)^2))
LD_measures <- data.frame(Raw_Bias, Percent_Bias, Coverage_Rate, Average_Width, RMSE)

```

```

# Mean time for 100 instances of LD
times_LD <- result_LD[[2]]
mean(times_LD)

```

```
## [1] 0.008345499
```

```
ListwiseDeletion <- mean(times_LD)
```

```

mean_times <- as.data.frame(rbind(Multiple Imputation, ListwiseDeletion), col.names = "average_runtime")
colnames(mean_times) <- "average-runtime"
mean_times

```

```

##           average-runtime
## Multiple Imputation      0.189170134
## ListwiseDeletion         0.008345499

```

```
knitr::kable(mean_times) %>% kable_styling(full_width = F)
```

```
knitr::kable(round(LD_measures, 3), align = "ccccc") %>% kable_styling()
```

| | Raw_Bias | Percent_Bias | Coverage_Rate | Average_Width | RMSE |
|-----------|----------|--------------|---------------|---------------|-------|
| Intercept | 0.114 | 0.571 | 0.98 | 1.069 | 0.216 |
| X_1 | 0.065 | 6.478 | 1.00 | 0.640 | 0.134 |
| X_2 | -0.034 | 1.684 | 1.00 | 1.081 | 0.176 |
| X_3 | 0.015 | 0.489 | 1.00 | 0.580 | 0.097 |
| X_4 | -0.017 | 0.437 | 1.00 | 0.541 | 0.086 |