Bootstraping and Monte Carlo Simulation

2025-01-22

```
library(ggplot2)
library(readxl)
library(dplyr)
```

```
##
## 다음의 패키지를 부착합니다: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(MASS)
```

```
##
## 다음의 패키지를 부착합니다: 'MASS'
```

```
## The following object is masked from 'package:dplyr':
##
##     select
```

```
data <- read_excel("C:/Users/sjh50/OneDrive/문서/UC Davis/Winter/Advanced stat/Class_2a/Prefere
nces 2025.xlsx") %>%
  rename(
    PreferenceRank = `Preference Rank`,
    Screen75Inch = `Screen 75 inch`,
    Screen85Inch = `Screen 85 inch`,
    Resolution4K = `Resolution 4K = 1`,
    Sony = `Sony = 1`,
    PriceLowHigh = `Price ₩r₩n(low = 0; high =1)`
  ) %>%
  mutate(
    PreferenceRank = as.numeric(PreferenceRank),
    Screen75Inch = as.numeric(Screen75Inch),
    Screen85Inch = as.numeric(Screen85Inch),
    Resolution4K = as.numeric(Resolution4K),
    Sony = as.numeric(Sony),
    PriceLowHigh = as.numeric(PriceLowHigh)
  )

head(data)
```

```
## # A tibble: 6 × 8
##   `Profile Nos` Profiles   PreferenceRank Screen75Inch Screen85Inch Resolution4K
##           <dbl> <chr>               <dbl>        <dbl>        <dbl>        <dbl>
## 1            19 75, 1K, S…              1            1            0            0
## 2             6 65, 4K, S…              2            0            0            1
## 3             9 65, 1K, S…              3            0            0            0
## 4            12 65, 4K, S…              4            0            0            1
## 5             3 65, 1K, S…              5            0            0            0
## 6            13 75, 1K, S…              6            1            0            0
## # ℹ 2 more variables: Sony <dbl>, PriceLowHigh <dbl>
```

```
# Fit the initial model
lm_fit <- lm(PreferenceRank ~ Screen75Inch + Screen85Inch + Resolution4K + Sony + PriceLowHigh, data = data)
estimates <- coef(lm_fit)
residuals <- lm_fit$residuals
y_hat <- predict(lm_fit)


# Compute dollars per utility for WTP calculation
cost_per_feature <- list(Price = 500) # set the lowest price as 2000 and the highest price as 2500
dollars_per_util <- abs(cost_per_feature[["Price"]] / estimates["PriceLowHigh"])


# Function to compute WTP from coefficients
compute_wtp <- function(coefs) {
  c(
    Screen85Inch = coefs["Screen85Inch"] * dollars_per_util,
    Resolution4K = coefs["Resolution4K"] * dollars_per_util,
    Sony = coefs["Sony"] * dollars_per_util
  )
}

# Residual Bootstrap
resid_boot_wtp <- replicate(1000, {
  y_star <- y_hat + sample(residuals, replace = TRUE)
  coefs <- coef(lm(y_star ~ Screen75Inch + Screen85Inch + Resolution4K + Sony + PriceLowHigh, data = data))
  compute_wtp(coefs)
})

resid_boot_ci <- apply(resid_boot_wtp, 1, function(x) quantile(x, c(0.025, 0.975)))




# Data Bootstrap
data_boot_wtp <- replicate(1000, {
  boot_data <- data[sample(nrow(data), replace = TRUE), ]
  coefs <- coef(lm(PreferenceRank ~ Screen75Inch + Screen85Inch + Resolution4K + Sony + PriceLowHigh, data = boot_data))
  compute_wtp(coefs)
})
data_boot_ci <- apply(data_boot_wtp, 1, function(x) quantile(x, c(0.025, 0.975)))




# Monte Carlo Simulations
cov_matrix <- vcov(lm_fit)


mc_wtp <- replicate(1000, {
  coefs <- mvrnorm(1, estimates, cov_matrix)
  compute_wtp(coefs)
})
```

```
mc_ci <- apply(mc_wtp, 1, function(x) quantile(x, c(0.025, 0.975)))



# Combine results into tables
attributes <- c("Screen85Inch", "Resolution4K", "Sony")
resid_boot_table <- data.frame(Attribute = attributes, t(resid_boot_ci))
data_boot_table <- data.frame(Attribute = attributes, t(data_boot_ci))
mc_table <- data.frame(Attribute = attributes, t(mc_ci))



colnames(resid_boot_table) <- colnames(data_boot_table) <- colnames(mc_table) <- c("Attribute", "Lower 95%", "Upper 95%")
print("Residual Bootstrap Results:")
```

```
## [1] "Residual Bootstrap Results:"
```

```
print(resid_boot_table)
```

```
##                              Attribute Lower 95% Upper 95%
## Screen85Inch.Screen85Inch Screen85Inch  485.2877  612.2552
## Resolution4K.Resolution4K Resolution4K  639.2122  742.6672
## Sony.Sony                         Sony  211.9627  319.9283
```

```
print("Data Bootstrap Results:")
```

```
## [1] "Data Bootstrap Results:"
```

```
print(data_boot_table)
```

```
##                              Attribute Lower 95% Upper 95%
## Screen85Inch.Screen85Inch Screen85Inch  484.6591  615.3131
## Resolution4K.Resolution4K Resolution4K  636.2211  742.6449
## Sony.Sony                         Sony  213.2794  315.3907
```

```
print("Monte Carlo Simulation Results:")
```

```
## [1] "Monte Carlo Simulation Results:"
```

```
print(mc_table)
```

```
##                              Attribute Lower 95% Upper 95%
## Screen85Inch.Screen85Inch Screen85Inch  480.4644  613.2949
## Resolution4K.Resolution4K Resolution4K  635.0133  743.2207
## Sony.Sony                         Sony  211.0090  319.2307
```