hmwrk 8-Beimnet Taye

2023-03-15

Worked with Joan Shim and Lucas Yoshida

P1

1.

- First let:
- $Z_i = \frac{1}{X_i}$ and $Z = \frac{1}{X}$
- Then we get:
- $\frac{1}{n}\sum_i (Z_i)$
- Since the CLT always holds no matter how Z is distributed, with a large enough sample the plug in estimator for Z will have a sampling distribution distributed in the following fashion:
- final: $\frac{1}{n}\sum_i Z_i \sim \mathcal{N}(E[Z], \sqrt{\frac{V[Z]}{n}})$

2.

- The asymptotic variance is $V[\frac{1}{X}]$

3.

- Plug in: $\frac{1}{n}\sum_i (\frac{1}{X_i} - \hat{\mu})^2$

4.

- Plug in: $\sqrt{\frac{\frac{1}{n}\sum_i (\frac{1}{X_i} - \hat{\mu})^2}{n}}$

5.

```
CI <- function(X) {
  inv <- 1/X
  n <- length(inv)
  mu <- mean(inv)
  SE <- sd(inv)/sqrt(n)
  CI <- c(mu - 1.96*SE, mu + 1.96*SE)
  return(CI)

}

CI(rnorm(500))
```

```
## [1] -1.306485  1.128827
```

P2

1.

- First, The expectation,and thus its plug-in estimator, the mean, of a binary variable is just its probability. So:
- $\mu_1 = E[Y|W = 1] = P(Y|W = 1)$
- From here we can use this conditional probability identity:
- $P(Y|W = 1) = \frac{P(W=1,Y)}{P(W=1)}$
- Since we already defined for binary variables Y and W that their probabilities equal their expected values we arrive at:
- $E[Y|W = 1] = \frac{E[WY]}{E[W]}$
- Using plug in estimators for the expectations we arrive at:
- $= \frac{\frac{1}{n}\sum_i WY}{\frac{1}{n}\sum_i W}$
- The generalized notation with indicator functions work since the indicator function of a binary value for value w: $1_w(W)$ returns 1 if inputted W equals the set condition w and returns 0 if that statement is not true. Thus the average of an indicator function of a binary variable is just the probability of the set condition, in this case w.
- $\frac{1}{n}\sum_i 1_w(W) = P(W = w)$

2.

- We can use the CLT theorem to demonstrate that both the numerator and denominators are asymptotically normal since they are both just individual means.

3.

- We can use the delta method since it involves a function of means. $\hat{\mu}_w$ is a ratio of means and thus a function of means so it is appropriate to use the delta method.

4.

- We can use the delta method since it involves a ratio of a ratio of means.$\hat{\psi}$ is a ratio of a ratio of means so we can try to use the delta method to demonstrate asymptotic normality.

P3.

1.

```r
estimator = function(data) {
n = nrow(data)
# estimate required sample averages
data %>%
mutate( # to handle rare case where all W=0 or =1
W = factor(W, levels=c(0,1))
) %>%
```

```
group_by(W, .drop=F) %>%
summarize(
EY_W = mean(Y),
EW = n()/n
) %$%
{
  EY_W[W==1] / EY_W[W==0]
}
}
```

```
bootstrap_estimator = function(data, esti, B=100) {
  estimate = esti(data)
  size <- nrow(data)
  bootstraps <- map_df(1:B,function(.x){
  boot_ratio <- slice_sample(data,n = size, replace = TRUE) %>%
     esti()
  return(tibble(
    error = estimate - boot_ratio
    )
  )
    }

  )

tibble(
estimate = estimate, # the point estimate
ci_a = estimate - quantile(bootstraps$error, probs = 0.975), # lower bound of the CI
ci_b = estimate - quantile(bootstraps$error, probs = 0.025) # upper bound of the CI
)
  }
```

2.

```
DGP_ratio <- function(n) {
  tibble(W = rbern(n, 0.75),
         Y = rbern(n,0.75))
}

test_DGP <- DGP_ratio(100)
```

3.

```
true_ratio <- estimator(DGP_ratio(100000)) #should be 1
true_ratio
```

```
## [1] 1.002352
```

4.

```r
coverage <- function(data, esti,n=100, rep = 10){
  true_ratio <- esti(data(100000))
  samples <- map_df(1:rep,function(.x){
    dataset <- slice_sample(data(n),n = nrow(data(n)),replace = T)
    return(bootstrap_estimator(data= dataset,esti, B=100))
  }
  )
  samples %>%
    mutate(cover = ifelse(true_ratio >= ci_a & true_ratio <= ci_b,1,0)) %$% {
      mean(cover)
    }
}

test <- coverage(DGP_ratio,estimator, rep = 500)
test
```

```
## [1] 0.78
```

5.

```r
estimator_delta = function(data) {
n = nrow(data)
# estimate required sample averages
data %>%
mutate( # to handle rare case where all W=0 or =1
W = factor(W, levels=c(0,1))
) %>%
group_by(W, .drop=F) %>%
summarize(
EY_W = mean(Y),
EW = n()/n
) %$%
c(EY_W[W==0], EY_W[W==1], EW[W==0], EW[W==1]) %->%
c(mu0, mu1, pi0, pi1)
# construct the plugins
rr = mu1/mu0
avar = rr^2 * ((1-mu0)/(mu0*pi0) + (1-mu1)/(mu1*pi1)) # from math
se = sqrt(avar / n)
# return result
tibble(
estimate=rr,
ci_a=rr-2*se,
ci_b=rr+2*se
)
}

coverage_delta <- function(data,esti=estimator_delta,rep = 10, n = 100) {
  true_ratio <- estimator(data(10000))
  map_df(1:rep,function(.x){
```

```
    esti(data(n))
    }
  ) %>%
    mutate(cover = ifelse(true_ratio >= ci_a & true_ratio <= ci_b,1,0)) %$% {
      mean(cover)
    }

}

test_delta <- coverage_delta(DGP_ratio, rep = 500)
test_delta
```

```
## [1] 0.966
```

6.

- Deriving the formula has better coverage on average of the true value while the bootstrap has worse coverage of the true value but is easier to implement since you don't have to do any mathematical derivations.