Bootstrapping and Confidence Intervals

October 5, 2023

Bootstrap Example

14 = chocolate 4 = vanilla

What is the true proportion of CSUEB masters students who prefer chocolate over vanilla?

Let $X_1, X_2, \cdots, X_n \overset{iid}{\sim} Bern(p = 14/18)$.

Then $E[X_i] = p$ and $Var[X_i] = p(1-p)$.

Population distribution of X

```r
library(ggplot2)
library(dplyr)

n <- 1 # n = 1 cuz Bernouli dist
n
```

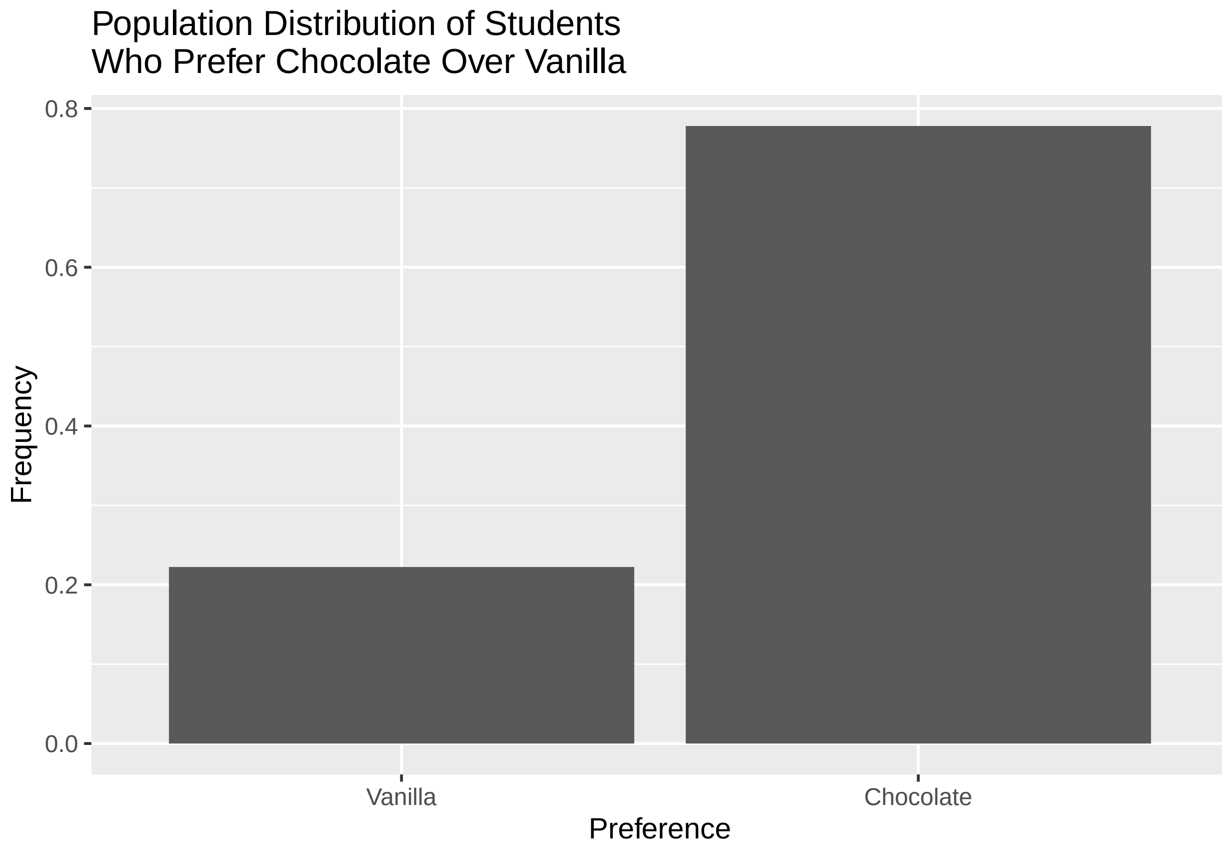
```
## [1] 1
```

```r
phat <- 14/18
phat
```

```
## [1] 0.7777778
```

```r
x <- c(0,1)
y <- dbinom(x, size = n, prob = phat)
pop <- data.frame(x = as.factor(x), y = y)

pop %>%
  ggplot(aes(x = x, y = y)) +
  geom_bar(stat = "identity") +
  scale_x_discrete(labels = c("Vanilla", "Chocolate")) +
  labs(x = "Preference",
       y = "Frequency",
       title = "Population Distribution of Students \nWho Prefer Chocolate Over Vanilla")
```

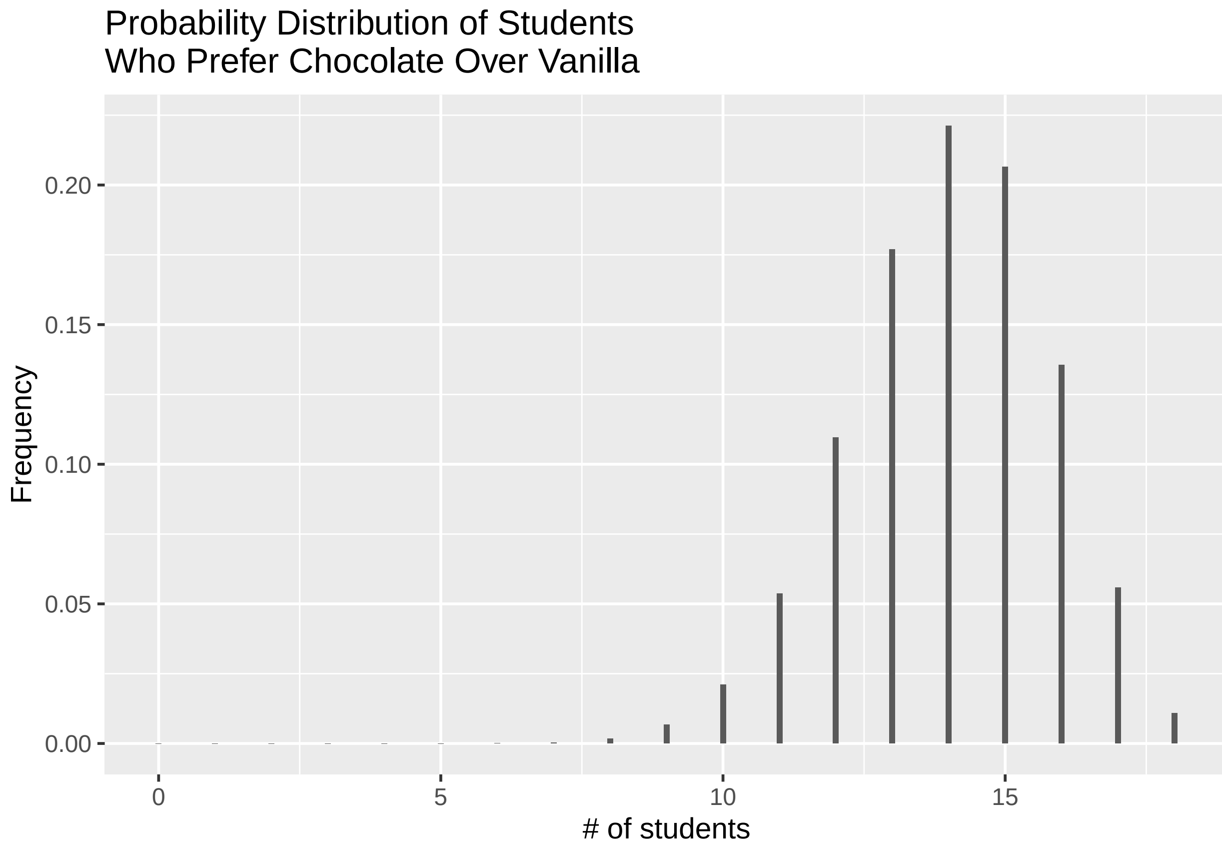



```
# n <- 1
# phat <- 14/18
# x <- c(0,1)
# y <- dbinom(x, n, prob = phat)
# pop <- data.frame(x = as.factor(x), y = y)
#
# pop %>%
#   ggplot(aes(x = x, y = y))+
#   geom_bar(stat = "identity")+
#   scale_x_discrete(labels = c("Vanilla", "Chocolate"))+
#   labs(x = "",
#        y = "",
#        title = "")

n_1 <- 18 #n=18 cuz Binomial Dist
p_1 <- phat # pretend this is the truth
x_1 <- 0:n_1
y_1 <- dbinom(x_1, size = n_1, prob = p_1)
pop_1 <- data.frame(x = x_1, y = y_1)

pop_1 %>%
  ggplot(aes(x = x_1, y = y_1)) +
  geom_bar(stat = "identity",
           width = 0.1) +
  labs(x = "# of students",
       y = "Frequency",
```

```
     title = "Probability Distribution of Students \nWho Prefer Chocolate Over Vanilla")
```

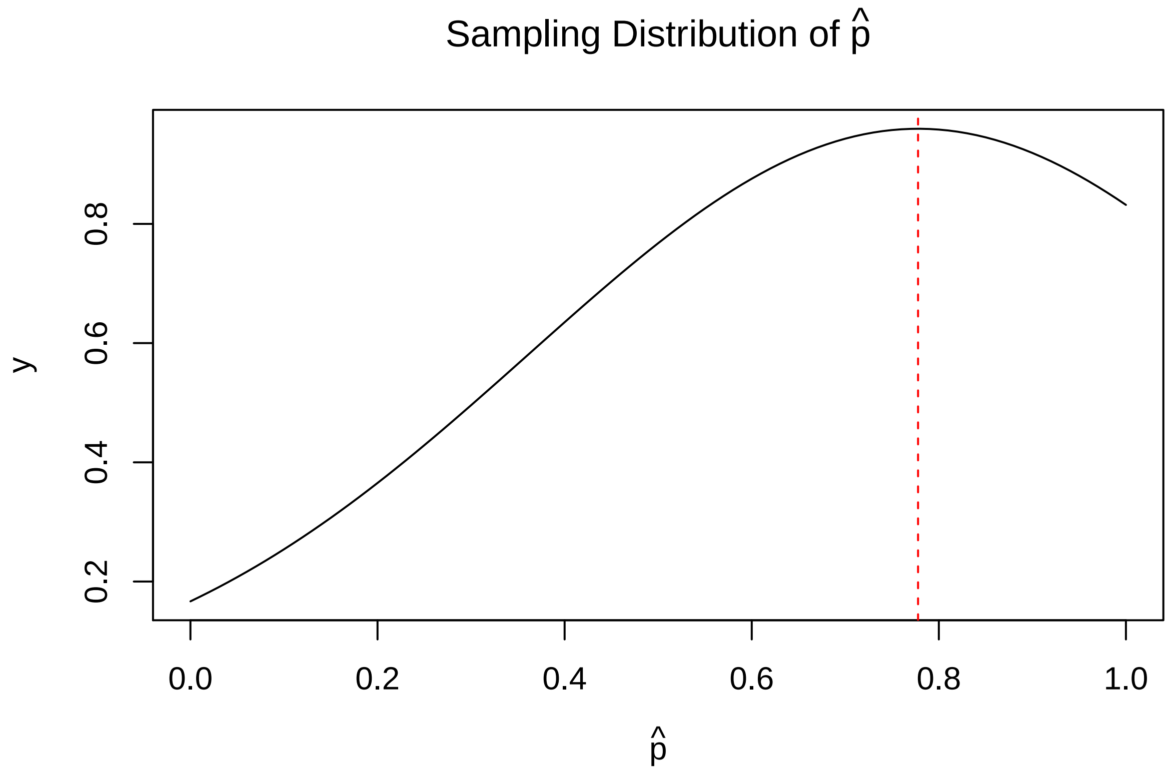


Sampling Distribution of $\hat{p}$

```
p <- phat
mu_phat <- p
sd_phat <- sqrt(p*(1 - p)/n)

x <- seq(0, 1, 0.01)
y <- dnorm(x, mean = mu_phat, sd = sd_phat)

plot(x, y, type = "l",
     xlab = expression(hat(p)),
     main = expression(paste("Sampling Distribution of ", hat(p)) ))
abline(v = p, col = "red", lty = 2)
legend(0.55, 3, paste("Mean = ", round(mu_phat,4)))
```



Bootstrap Distribution

```
B <- 5000
n_boot <- n_1

orig_samp <- c(rep(0,4 ), rep(1,14 ))
orig_samp
```

```
##  [1] 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

```
table(orig_samp)
```

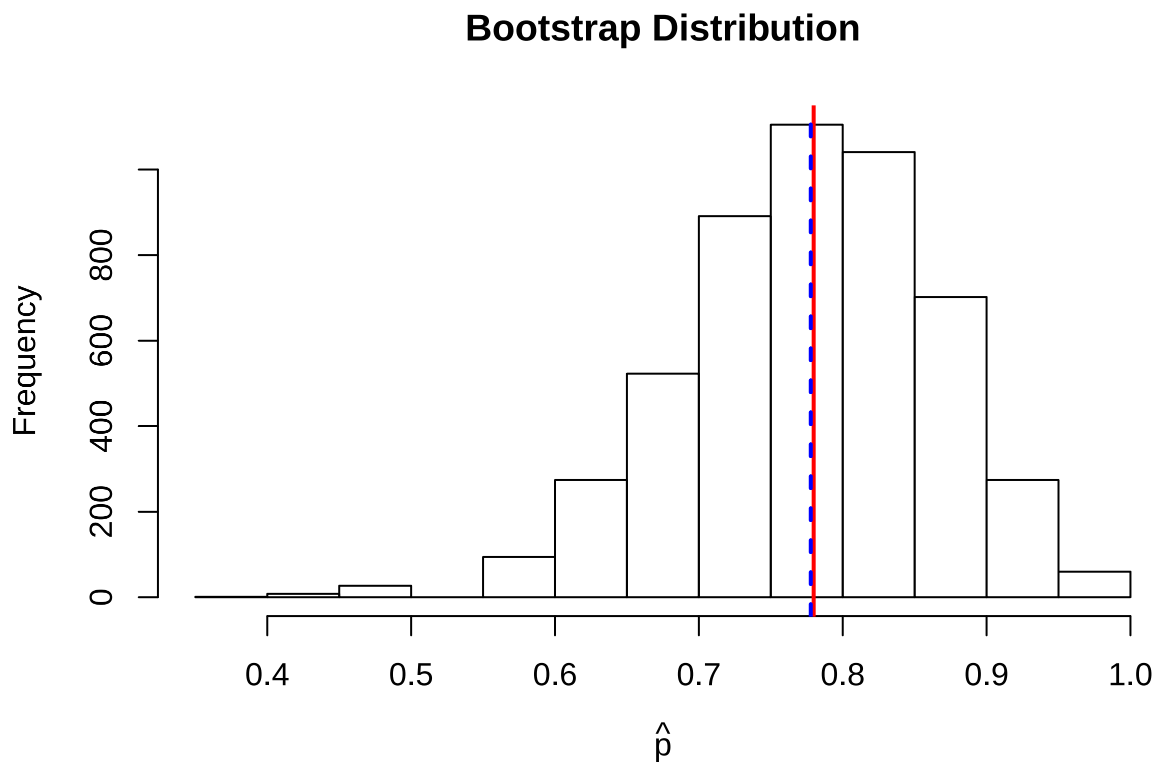
```
## orig_samp
##  0  1
##  4 14
```

```
phats <- rep(NA, B)

for(i in 1:B){
  boot_samp <- sample(orig_samp, size = n_boot, replace = TRUE)
  phats[i] <- mean(boot_samp)
}

hist(phats, breaks = 10,
     xlab = expression(hat(p)),
     main = "Bootstrap Distribution")
abline(v = mean(phats), col = "red", lwd = 2)
```

```r
abline(v = p, col = "blue", lwd = 2, lty = 2)
legend(0.01, 1500, paste("Mean = ", round(mean(phats), 3)))
```




```r
sqrt((p*(1-p))/n)
```

```
## [1] 0.4157397
```

```r
sd(phats)
```

```
## [1] 0.09698041
```

Discussion Questions (in groups):

1. Why is knowing $\hat{p}$ not enough?

- Need to know variability

1. Why do we want to know the distribution of $\hat{p}$?

- Inference

Bootstrap Percentile Confidence Intervals

To compute a 95% bootstrap confidence interval for some parameter, we compute the 2.5th and 97.5th percentiles of the bootstrap distribution.

We can use the `quantile()` function in R.

```r
boot_ci <- quantile(phats, c(0.025, 0.975))
boot_ci
```

```
##      2.5%     97.5%
## 0.5555556 0.9444444
```

Bootstrap CI: We are 95% confident that the true proportion of CSUEB master's students who prefer chocolate over vanilla is between 0.56 and 0.94.

Let's compare this to our confidence interval based on our original sample of data.

```r
ci_low <- phat - qnorm(.975) * sqrt(phat*(1-phat)/n)
ci_low
```

```
## [1] -0.03705708
```

```r
sqrt(phat*(1-phat)/n)
```

```
## [1] 0.4157397
```

```r
phat
```

```
## [1] 0.7777778
```

```r
phat*(1-phat)/n
```

```
## [1] 0.1728395
```

```r
1-phat
```

```
## [1] 0.2222222
```

```r
ci_high <- phat + qnorm(.975) * sqrt(phat*(1-phat)/n)
ci_high
```

```
## [1] 1.592613
```

CI from original sample: We are 95% confident that the true proportion of CSUEB master's students who prefer chocolate over vanilla is between -0.04 and 1.59.

Confidence Intervals (via Simulation)

```r
n_samp <- 100

phats <- rbinom(n_samp, n, p) / n


ci_low  <- phats - qnorm(0.975) * sqrt(phats*(1 - phats) / n)
ci_high <- phats + qnorm(0.975) * sqrt(phats*(1 - phats) / n)

color <- rep(NA, n_samp)

for(i in 1:n_samp){
  if(p > ci_low[i] & p < ci_high[i]){
    color[i] <- "aquamarine3"
  }
  else color[i] <- "coral"
```

```
}
table(color)
```

```
## color
## coral
##   100
```

```
x <- 1:n_samp
plot(x, phats, ylim = c(0,1),
     pch = 16, cex = 0.5, col = color,
     main = "100 CI's for p",
     xlab = "",
     ylab = "Proportion")
segments(x, ci_low, x, ci_high,
         col = color)
abline(h = p, col = "black")
```