Homework 1, STATS 315A

Stanford University, Winter 2019

Joe Higgins, Jessica Wetstone

Question 3: Bootstrap

Frequentist inference goes as follows. We have a random sample of N observations from a population distribution F, and we compute some statistic of interest (lets assume a scalar). The sampling distribution of that statistic is what we would get if we were able to repeat the process a large number of times: drawing samples of size N from F and computing the statistic each time. From this we could compute a quantity of interest, such as the standard deviation (standard error), which gives us an idea of the precision of our statistic. Many of our favorite models have convenient assumptions which allow us to derive theoretical expressions for say, the standard error, which can then be computed for our one and only sample. Sometimes these calculations are intractable.

The *bootstrap*, invented by our own Prof Brad Efron around 1980, is a cunning device for approximating the sampling distribution. Let $\hat{F}$ be the empirical distribution of our sample (the distribution that puts mass $1/N$ on each of the observed samples). We now draw a sample of size N from $\hat{F}$. This amounts to drawing from the original N observations a sample of size N with replacement. We compute our statistic on this bootstrap sample. This process is repeated a large number B times, and we use the bootstrap distribution of our statistic to approximate its sampling distribution.

(a) Assume the model $y = f(x) + \epsilon$ with $x \sim U[0, 1]$ and $\epsilon \sim N(0, (0.2)^2)$. For us $f(x) = 1 + x^2$. We have a sample of size 50 from this model. We fit a polynomial regression of degree 2 (we don't know the true model, so we include a linear term as well as a quadratic term), and build a prediction function $f(x)$ from the fitted model. We are interested in estimating $x_0 = \hat{f}^{-1}(1.3)$. (This kind of thing is done in dose-response experiments, and we are trying to estimate the dose that achieves a desired output). Write a function for taking a sample and delivering an estimate for $\hat{x}_0$.

```
rm(list = ls())
generate_model_from_pop <- function(n){
  X <- runif(n,0,1)
  X2 <- X^2
  y_noiseless <- 1+X2

  eps <- rnorm(n, 0, sqrt(0.2^2))
  y <- y_noiseless + eps
  return(lm(y ~ X + X2))
}

generate_model_from_boostrap <- function(n, X){
  X_bs <- sample(X, n, replace = TRUE)
  X2 <- X_bs^2
  y_noiseless <- 1+X2

  eps <- rnorm(n, 0, sqrt(0.2^2))
  y <- y_noiseless + eps
  return(lm(y ~ X_bs + X2))
}

f_hat <- function(x, model){
  features <- c(1, x, x^2)
  y_hat <- features %*% model$coefficients
  return(y_hat)
}

f_hat_inverse <- function(y, model){
  a <- model$coefficients[3]
  b <- model$coefficients[2]
```

```
  intcp <- model$coefficients[1]
  c <- intcp - y

  x_hat <- (-b + sqrt(b^2 - 4*a*c)) / (2*a)

  return(x_hat)
}

model <- generate_model_from_pop(50)
x_hat_0 <- f_hat_inverse(1.3, model)
x_hat_0
```

```
##         X
## 0.6052017
```

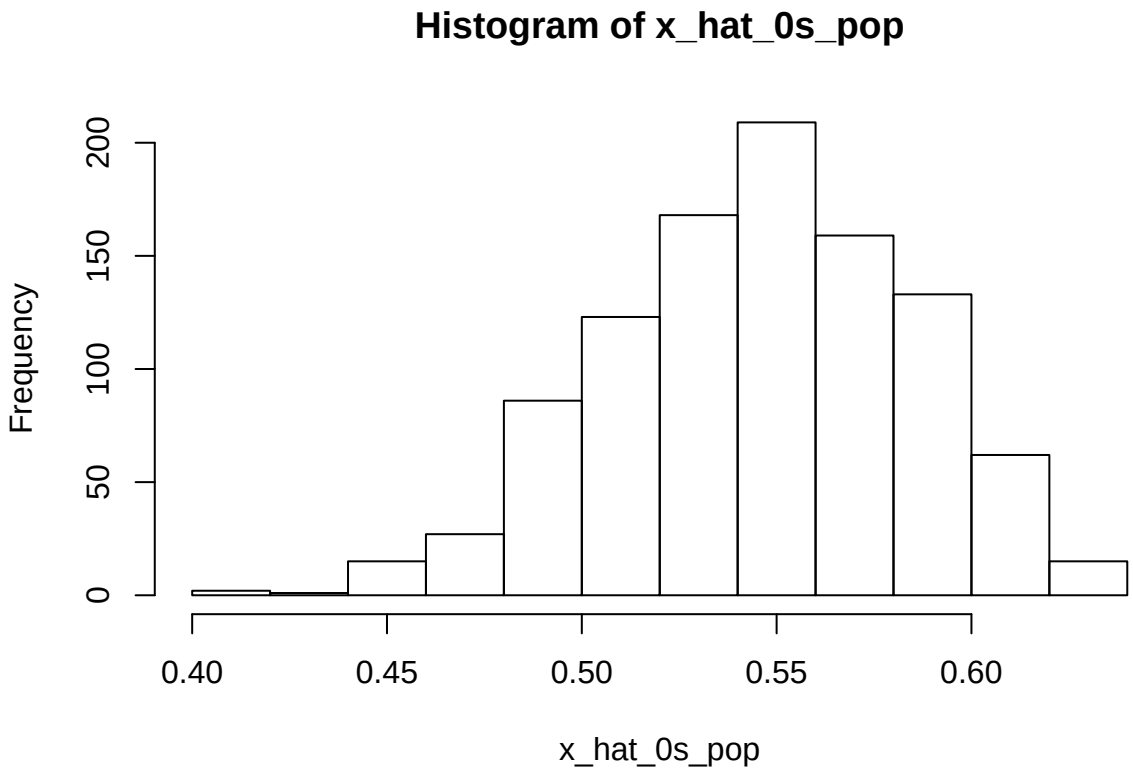
(b) Generate a sample from this model, and compute $\hat{x}_0$ for this sample.

```
model <- generate_model_from_pop(50)
x_hat_0 <- f_hat_inverse(1.3, model)
cat("Sample value: ", x_hat_0)
```

```
## Sample value:  0.5540935
```

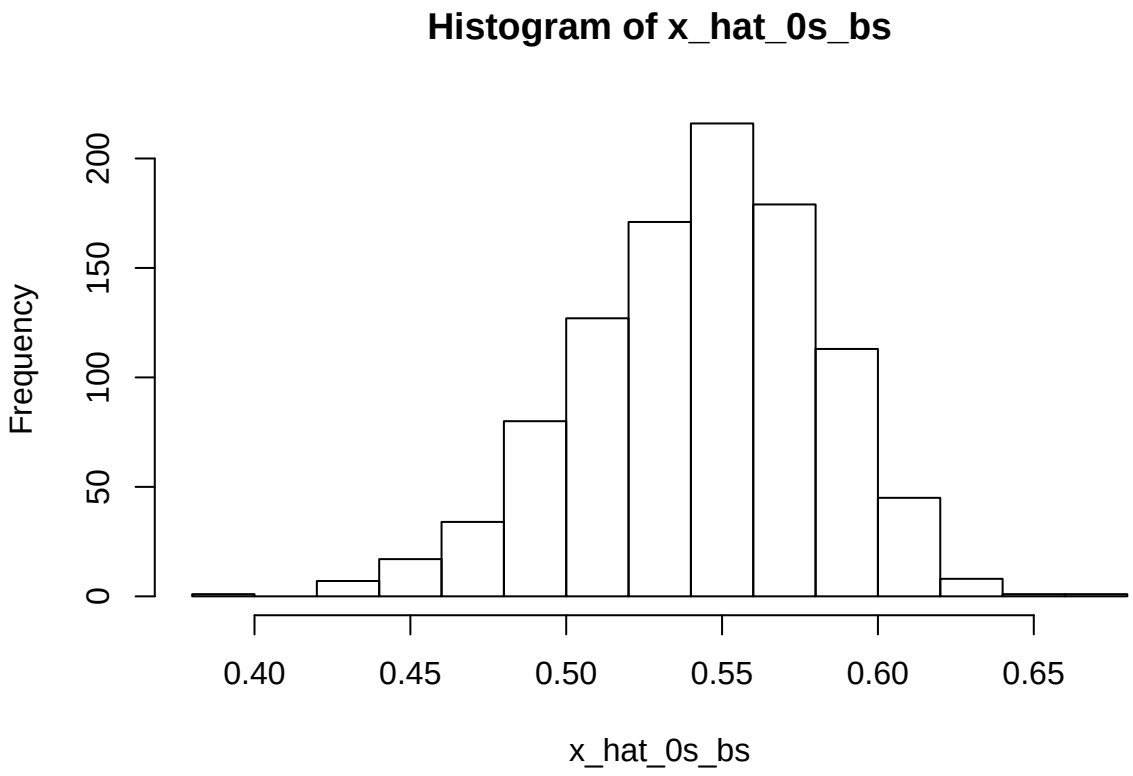
(c) Simulate a 1000 realizations from this model, to approximate the sampling distribution of $\hat{x}_0$.

```
realizations <- 1000
x_hat_0s_pop <- c()
for(i in 1:realizations){
  model <- generate_model_from_pop(50)
  x_hat_0s_pop <- c(x_hat_0s_pop, f_hat_inverse(1.3, model))
}

hist(x_hat_0s_pop)
```



(d) Using just your original sample, compute the bootstrap distribution of $\hat{x}_0$.

```
X <- runif(50,0,1)
x_hat_0s_bs <- c()
for(i in 1:realizations){
  model <- generate_model_from_boostrap(50, X)
  x_hat_0s_bs <- c(x_hat_0s_bs, f_hat_inverse(1.3, model))
}

hist(x_hat_0s_bs)
```



(e) Compare these two distributions, and in particular their standard deviations.

The means are the same, but the standard deviation from the bootstrap is slightly larger than from the population.

```
cat("Mean resampling from population: ", mean(x_hat_0s_pop),"\n")
cat("Mean resampling from bootstrap: ", mean(x_hat_0s_bs),"\n")
cat("STD resampling from population: ", sd(x_hat_0s_pop),"\n")
cat("STD resampling from bootstrap: ", sd(x_hat_0s_bs),"\n")
```

```
## Mean resampling from population:  0.545927
## Mean resampling from bootstrap:  0.5431767
## STD resampling from population:  0.0388089
## STD resampling from bootstrap:  0.03881879
```

Question 4

(b) Consider a special instance of this model with $X \in R^1$, and let $X \sim U[-0.5, 1.5]$, and $f(x) = x + 3x^2 - 1.5x^3$. Take $N = 15$, and run $B = 10$ simulations from the model. Produce a figure showing the true function f, and superimpose the 10 lines for $XE(\hat{\beta}|X)$.

```
rm(list = ls())
x_min <- -0.5
x_max <-  1.5
linspace <- seq(x_min, x_max, 0.1)

xform <- function(x){
  return(x + 3*x^2 - 1.5*x^3)
}

truth <- xform(linspace)

plot(linspace, truth, ylim=c(-2.0, 3.5))

N <- 15
B <- 10

for(i in 1:B){
  X <- runif(N, x_min, x_max)
  Y <- xform(X)

  model <- lm(Y~X)
  intcp <- as.numeric(model$coefficients[1])
  slope <- as.numeric(model$coefficients[2])
  lines(linspace,intcp + slope*linspace, col="blue")
}

legend("topleft", legend = c("truth","simulation"), fill = c("black","blue"))
```