Project 1: the nassCDS data

Contents

List of Figures

0 Notation

Throughout the project I will be refering to n as the length of the sample: $n = nrow(data)$; $B = 1000$ as the number of bootstrap samples; i as an index for objects in sample: $i = 1, \ldots, n$; b as an index for bootstrap samples: $b = 1, \ldots, B$. By “bootstrap replicate” I mean estimate of statistic computed in one bootstrap sample. By “resampling” I mean drawing objects from the observed sample with replacement.

1 Question 1

This question concentrates on 2 variables: *dead* (Y) and *ageOFocc* (X).

1.1

The task is to estimate parameters of GLM model:

$$g(P(Y_i = 1)) = \beta_0 + \beta_1 X_i.$$

Where g is *logit* function: $logit(p) = g(p) = \log\left(\frac{p}{1-p}\right)$. Estimating parameters β_i using *glm* function from package *stats* gives:

$$\hat{\beta_0}^{obs} = -3.91, \quad \hat{\beta_1}^{obs} = 0.02.$$

1.2

Let X_{10} be the age of occupant for which the probability to die is 0.1. Given equation $logit(0.1) = \beta_0 + \beta_1 X_{10}$ we can calculate X_{10}:

$$X_{10} = \frac{logit(p) - \beta_0}{\beta_1}.$$

The task is to use non-parametric bootstrap to estimate distribution of X_{10} and construct a 95% CI's. For $B = 1000$ we do the following:

1. Resample pairs (Y_i, X_i) and obtain sample $(Y, X)^{(b)}$ of the same length as original sample,
2. construct a GLM model to obtain $\hat{\beta_0}^{(b)}, \hat{\beta_1}^{(b)}$,
3. compute $\hat{X_{10}}^{(b)} = \frac{logit(0.1) - \hat{\beta_0}^{(b)}}{\hat{\beta_1}^{(b)}}$.

Distribution of X_{10} is presented on fig. 1.1. The confidence intervals are calculated as 2.5% and 97.5% quantiles from bootstrap replicates (bootstrap percentile interval): $[75.70, 87.34]$.

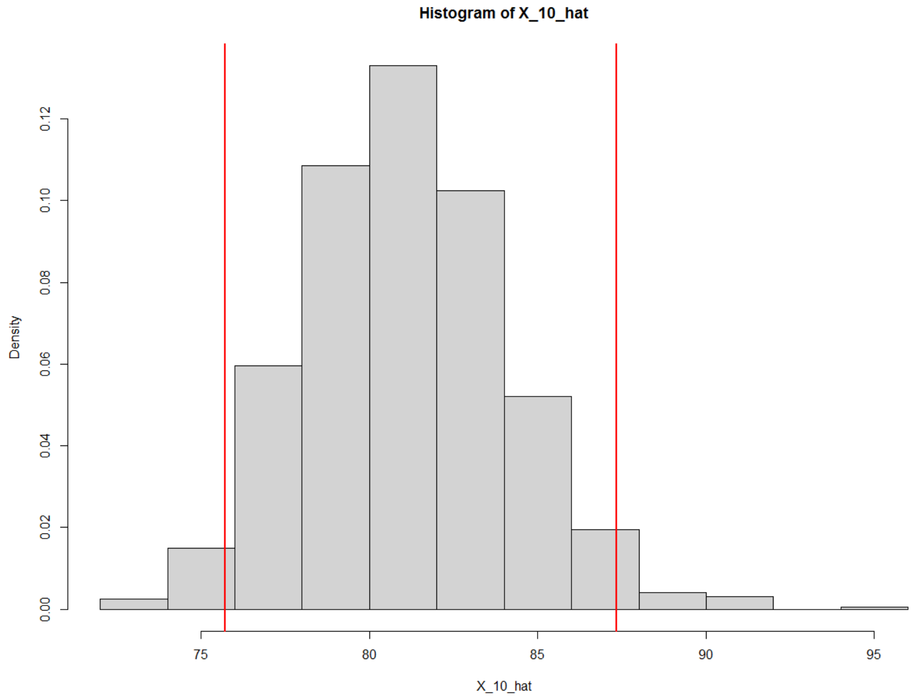

Figure 1.1: Histogram of X_{10}.

1.3

The task is to test the null hypothesis: $\beta_1 = 0$ using parametric bootstrap. To do that, we need to fit model under null:

$$\beta_1 = 0 \Rightarrow g(P(Y_i = 1)) = \beta_0^0, \quad \hat{\beta_0^0} = -3.05.$$

Then for $B = 1000$ we do the following:

1. Draw $Y_i^{(b)} \sim binom(1, \pi_i)$ where $\pi_i = \frac{e^{\hat{\beta_0^0}}}{1+e^{\hat{\beta_0^0}}}$ to obtain sample $(Y, X)^{(b)}$ of the same length as original sample,
2. construct a GLM model (of form $g(P(Y_i = 1)) = \beta_0 + \beta_1 X_i$) to obtain $\hat{\beta_0}^{(b)}, \hat{\beta_1}^{(b)}$.

To test hypothesis we use *Monte Carlo p-value*:

$$P = \frac{\#\left\{|\hat{\beta}_1^b| \geq |\hat{\beta_1}^{obs}|\right\} + 1}{B + 1} = 0.000999$$

P-value is small, so we reject the hypothesis $\beta_1 = 0$. It means, that deaths do depend on age of passenger.

2 Question 2

This question concentrates on variables *airbag* and *dead*.

2.1

The observation unit (X_i, Y_i) is a pair of binary variables *airbag* and *dead* that take value 1 when there was an airbag / death and 0 otherwise.

2.2

Given contingency table (tab. 2.1) we can calculate *oddsratio*:

	alive	dead
none	11058	669
airbag	13825	511

Table 2.1: Contingency table of variables *airbag* and *dead*.

$$\hat{OR}^{obs} = \frac{11058/669}{13825/511} = 0.61$$

Confidence intervals can be aproximated using log oddsratio ($L = \log OR$) as it is asymptoticly normal ($L \sim N(\log OR, \sigma^2)$, source [1]). Intervals for OR have form:

$$\exp(\hat{L}^{obs} \pm \hat{SE}^{obs} \cdot z_{\frac{\alpha}{2}}), \ \hat{SE}^{obs} = \sqrt{\frac{1}{11058} + \frac{1}{669} + \frac{1}{13825} + \frac{1}{511}}$$

$z_{\frac{\alpha}{2}}$ is a $\frac{\alpha}{2}$ quantile of normal distribution and $\hat{SE}^{obs}$ is an estimator of σ^2. The intervals are: $[0.543, 0.6874]$. We can conclude that accidents with airbags are less likely to have a death.

2.3

The task is to use parametric bootstrap to construct 95% confidence interval for OR. Then for $B = 100 =$ we do the following:

1. Draw $\hat{L}^{(b)} \sim N(\log \hat{OR}^{obs}, \hat{SE}^{obs})$,
2. transform $\hat{L}^{(b)}$ to $\hat{OR}^{(b)} = \exp(\hat{L}^{(b)})$.

The confidence intervals are calculated as 2.5% and 97.5% quantiles from bootstrap replicates (bootstrap percentile interval): $[0.5451, 0.6929]$. Those intervals are transformation respecting. This interval is wider than the theoretical.

2.4

The task is to use non-parametric bootstrap to test the hypothesis that airbags do not influence the accident outcome using a χ^2-square test. To do that we need to resample X_i and Y_i separately, so that X_i and Y_i are independent (null hypothesis). Then for $B = 1000$ we do the following:

1. Resample **only** X,
2. Resample **only** Y,
3. Obtain sample $(Y, X)^{(b)}$ of the same length as original sample,
4. compute test statistic:
$$\chi^2_{(b)} = \sum_{i=1}^{2}\sum_{j=1}^{2} \frac{(O_{ij} - E_{ij})^2}{E_{ij}} \overset{H_0}{\sim} \chi^2_1.$$

Where O_{ij} is the number of observed number of observations in i-th row and j-th column of contingency table and E_{ij} expected number of observations in the same cell. $E_{ij} \overset{H_0}{=} Np_{i.}p_{.j}$, where N is sum of all cells $p_{i.} = \sum_{j=1}^{2} \frac{O_{ij}}{N}$ and $p_{j.} = \sum_{i=1}^{2} \frac{O_{ij}}{N}$.

The empirical (histogram) and theoretical (black line) densities are presented in fig. 2.1. With $B = 1000$ bootstrap samples histogram resembles the theoretical density. To test hypothesis we use *Monte Carlo p-value*:

$$P = \frac{\#\left\{\hat{\chi^2_{(b)}} \geq \hat{\chi^2}_{obs}\right\} + 1}{B + 1} = 0.000999$$

Where $\hat{\chi^2_b}$ is a test statistic in b-th bootstrap sample and $\hat{\chi^2}$ is a test statistic in original sample. P-value is small, so we reject the hypothesis that airbags are independent of deaths. It means, that death in car accidents depend on having airbags in car.

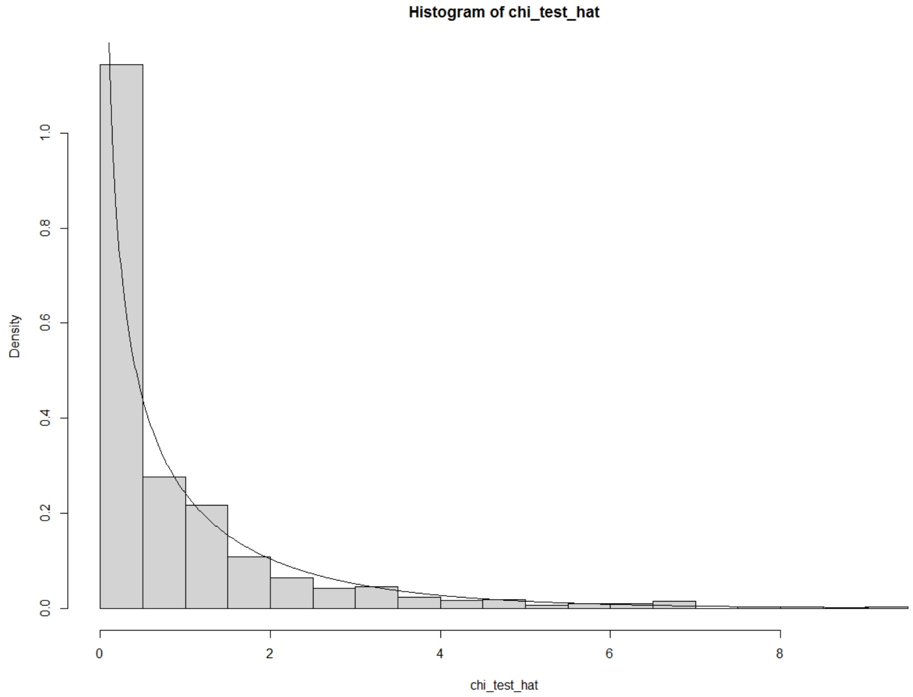

Figure 2.1: Histogram of χ^2 statistics.

3 Question 3

This question concentrates on variable *weight* and estimators of its mean: sample mean, median, trimmed mean (10%) and mid range ($\hat{\theta}_1, \hat{\theta}_2, \hat{\theta}_3, \hat{\theta}_4$ respectively).

3.1

The task is to use non-parametric bootstrap to estimate MSE of those estimators and find one with the smallest MSE. Variance and bias can be estimated using bootstrap. Then for $B = 1000$ we do the following:

1. Resample vector *weight* to obtain sample $X^{(b)}$ of the same length as original vector,
2. compute four estimators $\hat{\theta}_i^{(b)}$, $i = 1, \ldots, 4$,

Compute their sample variances and biases:

$$s^2(\hat{\theta}_i) = \frac{1}{B-1}\sum_{b=1}^{B}\left(\hat{\theta}_i^{(b)} - \hat{\theta}_i^{obs}\right)^2, \quad \hat{bias}(\hat{\theta}_i) = \frac{1}{B}\sum_{b=1}^{B}\left(\hat{\theta}_i^{(b)} - \hat{\theta}_i^{obs}\right) \quad i = 1, \ldots, 4$$

Estimate the MSE:

$$MSE(\hat{\theta}) = s^2(\hat{\theta}_i) + \hat{bias}(\hat{\theta}_i)^2$$

The estimates of MSE for different estimators is presented in table 3.1.

	mean	median	trimmed mean	midrange
MSE	95.72	0.291	6.87	2909892

Table 3.1: MSE of 4 estimators of mean.

The best estimator is sample median. In the distribution of *weight* there are frequently appearing 0, which means, that midrange will be just the half od maximum of the sample. There are also a lot of outliers, so the midrange is not an accurate estimator. Trimmed mean is performing better than ordinary sample mean, because its trimming those zeros and outliers.

3.2

The task is to use non-parametric bootstrap to estimate the distribution of sample median and construct 95% confidence intervals. The bootstrap algorithm for obtaining distribution of sample median is the same as in previous task. Distribution of sample median is presented on fig. 3.1. The confidence intervals are calculated as quantiles (bootstrap percentile interval): $[86.22, 88.48]$.

3.3

The task is to use jackknife method to estimate MSE of every estimator and select the best one. The jackknife method calculates estimator for every replication of original sample, but removes one observation. So given our vector *weight*, there will be 26063 replications of form: $X_{(i)} = (x_1, x_2, \ldots, x_{i-1}, x_{i+1}, \ldots, x_{26063})$. Based on those we can compute sample mean and variance to obtain estimates for bias and then MSE.

In jackknife method to estimate variance *sample variance* needs to be multiplied by inflation factor of $\frac{(n-1)^2}{n}$ and estimated bias needs to be multiplied by $(n-1)$.

The jackknife method does not work for sample median and midrange (bias and variance is equal zero), because in our original sample median, maxima and minima are repeated more than once, so for every jackknife replicate there is always the same value of estimator.

The MSE of sample mean is lower compared to the previous task. The sample trimmed mean is large, because it has more bias than sample mean and it's enlarged by factor $(n-1)^2$. The sample median and midrange cannot be compared using this technique, so the best estimator is now sample mean.

	mean	median	trimmed mean	midrange
MSE	89.56	0	6193.1	0

Table 3.2: MSE of 4 estimators of mean (jackknife method).

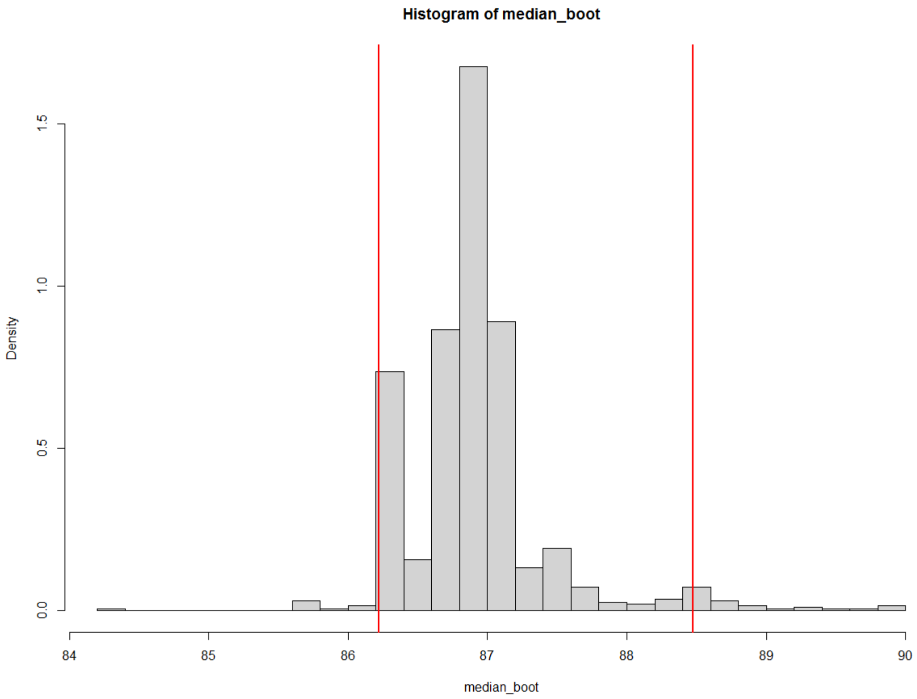

Figure 3.1: Histogram of sample median.

References

[1] Wikipedia contributors. Odds ratio — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=Odds_ratio&oldid=1057111761, 2021. [Online; accessed 30-December-2021].

Appendices

Appendix 1 Preparing data

```
#Project 1
set.seed(297759)
library("DAAG")
data(nassCDS)
names(nassCDS)
nassCDS = na.omit(nassCDS)
attach(na.omit(nassCDS))

#Converting factor to logical
levels(dead) = c(FALSE,TRUE)
dead = as.logical(dead)
```

Appendix 2 Bootstrap algorithm

```
airbag = as.logical(airbag)

#######################################################################################
#############################BOOTSTRAP################################################
```

```
#BOOTSTRAP ALGORITHM FOR A VECTOR
resample_vector_nonparam = function(X, n=length(X)) sample(X, size = n,
    replace = TRUE)

resample_vector_param = function(X, rdist, n=length(X)) rdist(n)

bootstrap_vector = function(B=1000, X, theta_est, param=FALSE, rdist){
  if(param)
    X_boot = sapply(1:B, function(n)resample_vector_param(X, rdist, length(
        X)))
  else
    X_boot = sapply(1:B, function(n)resample_vector_nonparam(X, length(X)))

  theta_hat = apply(X_boot, 2, theta_est)
  return(theta_hat)
}

#BOOTSTRAP ALGORITHM FOR A DATAFRAME
resample_dataframe_nonparam = function(data, n=nrow(data)){
  id_boot = sample(1:n, size = n, replace = TRUE)
  return(data[id_boot,])
}

resample_dataframe_param = function(data, rdist_list, cols=1, n=nrow(data))
    {
  for (col in cols)
    data[,col] = rdist_list[[col]](n)

  return(data)
}

bootstrap_dataframe = function(B=1000, data, theta_est, param=FALSE, rdist_
    list, cols=1){
  if(param)
    data_boot = lapply(1:B, function(n)resample_dataframe_param(data, rdist
        _list, cols, nrow(data)))
  else
    data_boot = lapply(1:B, function(n)resample_dataframe_nonparam(data,
        nrow(data)))

  theta_hat = sapply(data_boot, theta_est)
  return(theta_hat)
}
```

Appendix 3 Question 1

```
##########################PROJECT 1###########################################

#Question 1
#Q1.1
#Estimating the model using the classical GLM approach
fit_binom = glm(dead~ageOFocc, family = "binomial")
fit_binom$coefficients
plot(ageOFocc, predict.glm(fit_binom, type = "response"))

#Q1.2
#Contingency table
cont = round(prop.table(table(dead, ageOFocc), margin = 2), 2)
```

```
#Table sorted ascending by probability of death
cont[, order(cont[2,])]
#Random variable X_10 as an inverse of link function
logit = function(p) log(p/(1-p))
X_10 = function(beta_0, beta_1) (logit(0.1)-beta_0)/beta_1
#Estimator of X_10
X_10_est = function(data){
  beta = glm(data$dead~data$ageOFocc, family = "binomial")$coefficients
  return(X_10(beta[1], beta[2]))
}
#Bootstrap
X_10_hat = bootstrap_dataframe(B=1000, data.frame(dead, ageOFocc), X_10_est
    )
hist(X_10_hat, freq = FALSE)
abline(v = quantile(X_10_hat, c(0.025, 0.975)), col="red", lwd=2)
save(X_10_hat, file="CIM_project1_1_2_X_10.RData")

#Q1.3
#Fit the null model
fit_binom_0 = glm(dead~1, family = "binomial")
#Resampling under null hypothesis: beta_1 = 0 (fixing ageOFocc, sampling
    from dead)
sigmoid = function(x) exp(x) / (1 + exp(x))
#Sampling from binomial distribution with pi_j
rdist = function(n) rbinom(n=n, size=1, prob = sigmoid(fit_binom_0$
    coefficients))
#Estimator of betas
beta_est = function(data)glm(data[,1]~data[,2], family = "binomial")$
    coefficients
#bootstrap algorithm
beta = bootstrap_dataframe(B=1000, data = data.frame(dead, ageOFocc),
                               theta_est = beta_est, param=TRUE, cols=1, rdist_
                                   list=list(rdist))


hist(beta[1,], freq = FALSE, main = "Histogram␣of␣beta_0", xlab = "beta_0")
hist(beta[2,], freq = FALSE, main = "Histogram␣of␣beta_1", xlab = "beta_1")
save(beta, file="CIM_project1_1_3_beta.RData")
#Monte Carlo p-value
```

Appendix 4 Question 2

```
mean(beta[2,])
sd(beta[2,])


#Question 2
table(airbag, dead)

#Q.2.2
oddsratio = function(X, Y){
  cont = table(X, Y)
  return( (cont[1,1]*cont[2,2]) / (cont[1,2]*cont[2,1]) )
}
oddsratio(airbag, dead)
#Standard error of logOR
SE = sqrt(1/11058 + 1/669 + 1/13825 + 1/511)
#Confidence intervals
```

```
exp(log(oddsratio(airbag, dead)) + SE * 1.96)
exp(log(oddsratio(airbag, dead)) - SE * 1.96)

#Q2.3 (lecture 1c, page 15)
#logOR_hat is asymptoticly N(logOR, SE)
log_theta_hat = rnorm(1000, mean = log(oddsratio(airbag, dead)), sd = SE)
#Bootstrap CI
quantile(exp(log_theta_hat), probs = c(0.025, 0.975))

#Q2.4
#Nonparametric bootstrap
chi_test_est = function(data) chisq.test(data[,1], data[,2])$statistic
chi_test_hat = NULL

#resampling independetly
for (b in 1:1000) {
  dead_boot = resample_vector_nonparam(dead)
  airbag_boot = resample_vector_nonparam(airbag)
  chi_test_hat[b] = chi_test_est(data.frame(dead_boot, airbag_boot))
}

hist(chi_test_hat, freq = FALSE, breaks = 20)
x = seq(min(chi_test_hat), max(chi_test_hat), by=0.01)
lines(x, dchisq(x, df=1))
```

Appendix 5 Question 3

```
#testing independence
p_value_boot(chi_test_hat, chisq.test(dead, airbag)$statistic)

#Question 3
#Q3.1
MSE = function(theta_boot, theta_obs) var(theta_boot) + (mean(theta_boot)-
    theta_obs)^2
#MSE of mean
mean_boot = bootstrap_vector(B=1000, X=weight, theta_est = mean)
MSE(mean_boot, mean(weight))
#MSE of median
median_boot = bootstrap_vector(B=1000, X=weight, theta_est = median)
MSE(median_boot, median(weight))
#MSE of trimmed mean
t_mean = function(x) mean(x, trim = 0.1)
t_mean_boot = bootstrap_vector(B=1000, X=weight, theta_est = t_mean)
MSE(t_mean_boot, t_mean(weight))
#MSE of mid range
midrange = function(x) (min(x) + max(x))/2
midrange_boot = bootstrap_vector(B=1000, X=weight, theta_est = midrange)
MSE(midrange_boot, midrange(weight))

#Q3.2
hist(median_boot, freq = FALSE, breaks=30)
abline(v=quantile(median_boot, probs = c(0.025, 0.975)), col="red", lwd=2)

#Q3.3
jackknife = function(X, theta_est){
  theta_hat = NULL
  for (i in 1:length(X)) {
    theta_hat[i] = theta_est(X[-i])
  }
  return(theta_hat)
}
#MSE needs to include inflation factor
MSE_jackknife = function(theta_boot, theta_obs){
  n = length(theta_boot)
  (n-1)^2 / n * var(theta_boot) +((n-1)*(mean(theta_boot)-theta_obs))^2
}
#MSE of mean
mean_jack = jackknife(weight, mean)
MSE_jackknife(mean_jack, mean(weight))
#MSE of median
median_jack = jackknife(weight, median)
MSE_jackknife(median_jack, median(weight))
#MSE of trimmed mean
t_mean_jack = jackknife(weight, t_mean)
```