Mehmet Duman

Solution to Homework #4—MTH 522

Problem 1 (Chapter 5 Exercises 5):

In Chapter 4, we used logistic regression to predict the probability of default using income and balance on the Default data set. We will now estimate the test error of this logistic regression model using the validation set approach. Do not forget to set a random seed before beginning your analysis.

(a) Fit a logistic regression model that uses income and balance to predict default.

```
> library(ISLR)
> summary(Default)

default    student       balance           income
No :9667   No :7056   Min.   :   0.0   Min.   :  772
Yes: 333   Yes:2944   1st Qu.: 481.7   1st Qu.:21340
Median : 823.6   Median :34553
Mean   : 835.4   Mean   :33517
3rd Qu.:1166.3   3rd Qu.:43808
Max.   :2654.3   Max.   :73554


> attach(Default)
> set.seed(1)
> fit.glm = glm(default ~ income + balance, data = Default, family = "binomial")
> summary(fit.glm)

Call:
glm(formula = default ~ income + balance, family = "binomial",
data = Default)

Deviance Residuals:
Min       1Q   Median       3Q      Max
-2.4725  -0.1444  -0.0574  -0.0211   3.7245

Coefficients:
Estimate Std. Error z value Pr(>|z|)
(Intercept) -1.154e+01  4.348e-01 -26.545  < 2e-16 ***
income       2.081e-05  4.985e-06   4.174 2.99e-05 ***
balance      5.647e-03  2.274e-04  24.836  < 2e-16 ***
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1

(Dispersion parameter for binomial family taken to be 1)

Null deviance: 2920.6  on 9999  degrees of freedom
Residual deviance: 1579.0  on 9997  degrees of freedom
AIC: 1585

Number of Fisher Scoring iterations: 8
```

(b) Using the validation set approach, estimate the test error of this model. In order to do this, you must perform the following steps:

(b) i. Split the sample set into a training set and a validation set.

```
> train = sample(dim(Default)[1], dim(Default)[1] / 2)
```

(b) ii. Fit a multiple logistic regression model using only the training observations.

```
> fit.glm = glm(default ~ income + balance, data = Default, family = "binomial", subset = train)
> summary(fit.glm)

Call:
glm(formula = default ~ income + balance, family = "binomial",
data = Default, subset = train)

Deviance Residuals:
Min       1Q   Median       3Q      Max
-2.3583  -0.1268  -0.0475  -0.0165   3.8116

Coefficients:
Estimate Std. Error z value Pr(>|z|)
(Intercept) -1.208e+01  6.658e-01 -18.148   <2e-16 ***
income       1.858e-05  7.573e-06   2.454   0.0141 *
balance      6.053e-03  3.467e-04  17.457   <2e-16 ***
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1

(Dispersion parameter for binomial family taken to be 1)

Null deviance: 1457.0  on 4999  degrees of freedom
Residual deviance:  734.4  on 4997  degrees of freedom
AIC: 740.4

Number of Fisher Scoring iterations: 8
```

(b) iii. Obtain a prediction of default status for each individual in the validation set by computing the posterior probability of default for that individual, and classifying the individual to the default category if the posterior probability is greater than 0.5.

```
> probs = predict(fit.glm, newdata = Default[-train, ], type = "response")
> pred.glm = rep("No", length(probs))
> pred.glm[probs > 0.5] = "Yes"
```

(b) iv. Compute the validation set error, which is the fraction of the observations in the validation set that are misclassified.

```
> mean(pred.glm != Default[-train, ]$default)
[1] 0.0286
```

Using the validation set approach, the test error rate is 2.86%.

(c) Repeat the process in (b) three times, using three different splits of the observations into a training set and a validation set. Com- ment on the results obtained.

```
> train = sample(dim(Default)[1], dim(Default)[1] / 2)
>  fit.glm = glm(default ~ income+ balance, data=Default, family="binomial", subset=train)
> probs = predict(fit.glm, newdata = Default[-train, ], type = "response")
>  pred.glm = rep("No", length(probs))
>  pred.glm[probs > 0.5] = "Yes"
> mean(pred.glm != Default[-train, ]$default)
[1] 0.0236


> train = sample(dim(Default)[1], dim(Default)[1] / 2)
>  fit.glm = glm(default ~ income+ balance, data=Default, family="binomial", subset=train)
> probs = predict(fit.glm, newdata = Default[-train, ], type = "response")
>  pred.glm = rep("No", length(probs))
>  pred.glm[probs > 0.5] = "Yes"
> mean(pred.glm != Default[-train, ]$default)
[1] 0.028


> train = sample(dim(Default)[1], dim(Default)[1] / 2)
>  fit.glm = glm(default ~ income+ balance, data=Default, family="binomial", subset=train)
> probs = predict(fit.glm, newdata = Default[-train, ], type = "response")
>  pred.glm = rep("No", length(probs))
>  pred.glm[probs > 0.5] = "Yes"
> mean(pred.glm != Default[-train, ]$default)
[1] 0.0268
```

Each time we get different test error rate. The average test error rate is 2.613333%.

(d) Now consider a logistic regression model that predicts the probability of default using income, balance, and a dummy variable for student. Estimate the test error for this model using the validation set approach. Comment on whether or not including a dummy variable for student leads to a reduction in the test error rate.

using :income, balance, student

```
> train = sample(dim(Default)[1], dim(Default)[1] / 2)
>  fit.glm=glm(default~income+balance+student,data=Default,family="binomial",subset=train)
> probs = predict(fit.glm, newdata = Default[-train, ], type = "response")
>  pred.glm = rep("No", length(probs))
>  pred.glm[probs > 0.5] = "Yes"
> mean(pred.glm != Default[-train, ]$default)

[1] 0.0264
```

The test error rate is 2.64%, including a dummy variable for student didn't leads to a reduction in the test error rate.

Problem 2 (Chapter 5 Exercises 6):

We continue to consider the use of a logistic regression model to predict the probability of default using income and balance on the Default data set. In particular, we will now compute estimates for the standard errors of the income and balance logistic regression coefficients in two different ways:
]

(1) using the bootstrap, and

(2) using the standard formula for computing the standard errors in the glm() function.

Do not forget to set a random seed before beginning your analysis.

(a) Using the summary() and glm() functions, determine the estimated standard errors for the coefficients associated with income and balance in a multiple logistic regression model that uses both predictors.

```
>   set.seed(1)
> attach(Default)
The following objects are masked from Default (pos = 3):

balance, default, income, student




> fit.glm = glm(default ~ income + balance, data = Default, family = "binomial")
>  summary(fit.glm)

Call:
glm(formula = default ~ income + balance, family = "binomial",
data = Default)

Deviance Residuals:
Min            1Q        Median           3Q           Max
-2.472542783  -0.144435943  -0.057366212  -0.021063920   3.724454436

Coefficients:
                   Estimate      Std. Error   z value   Pr(>|z|)
(Intercept) -1.15404684e+01  4.34756357e-01 -26.54468 < 2.22e-16 ***
income       2.08089755e-05  4.98516718e-06   4.17418 2.9906e-05 ***
balance      5.64710294e-03  2.27373142e-04  24.83628 < 2.22e-16 ***
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1

(Dispersion parameter for binomial family taken to be 1)

Null deviance: 2920.649711  on 9999  degrees of freedom
Residual deviance: 1578.966270  on 9997  degrees of freedom
AIC: 1584.96627

Number of Fisher Scoring iterations: 8
```

The glm() functions estimated standard errors for the coefficients are 4.34756357e-01 , 4.98516718e-06 and 2.27373142e-04

(b) Write a function, boot.fn(), that takes as input the Default data set as well as an index of the observations, and that outputs the coefficient estimates for income and balance in the multiple logistic regression model.

```
> boot.fn = function(data, index) {
+ fit = glm(default ~ income + balance, data = data, family = "binomial", subset = index)
+ return (coef(fit))
+ }
```

(c) Use the boot() function together with your boot.fn() function to estimate the standard errors of the logistic regression coefficients for income and balance.

```
> library(boot)
> boot(Default, boot.fn, 300)

ORDINARY NONPARAMETRIC BOOTSTRAP


Call:
boot(data = Default, statistic = boot.fn, R = 300)


Bootstrap Statistics :
         original        bias     std. error
t1* -1.154047e+01 -6.310201e-02 4.424387e-01
t2*  2.080898e-05 -1.442808e-07 4.886839e-06
t3*  5.647103e-03  3.762129e-05 2.301732e-04
```

The boot.fn() function to estimates the standard errors of the logistic regression coefficients are 4.424387e-01, 4.886839e-06 and 2.301732e-04

d) Comment on the estimated standard errors obtained using the glm() function and using your bootstrap function.

The estimated standard errors obtained using the glm() function and using the bootstrap function are pretty close.

β_0		β_1		β_2
4.34756357e-01	,	4.98516718e-06	and	2.27373142e-04
4.424387e-01	,	4.886839e-06	and	2.301732e-04

Problem 3 (Chapter 5 Exercises 7):

In Sections 5.3.2 and 5.3.3, we saw that the cv.glm() function can be used in order to compute the LOOCV test error estimate. Alternatively, one could compute those quantities using just the glm() and predict.glm() functions, and a for loop. You will now take this approach in order to compute the LOOCV error for a simple logistic regression model on the Weekly data set. Recall that in the context of classification problems, the LOOCV error is given in (5.4).

```
> library(ISLR)
> set.seed(1)
> attach(Weekly)
> summary(Weekly)
Year           Lag1               Lag2               Lag3
Min.   :1990   Min.   :-18.1950   Min.   :-18.1950   Min.   :-18.1950
1st Qu.:1995   1st Qu.: -1.1540   1st Qu.: -1.1540   1st Qu.: -1.1580
Median :2000   Median :  0.2410   Median :  0.2410   Median :  0.2410
Mean   :2000   Mean   :  0.1506   Mean   :  0.1511   Mean   :  0.1472
3rd Qu.:2005   3rd Qu.:  1.4050   3rd Qu.:  1.4090   3rd Qu.:  1.4090
Max.   :2010   Max.   : 12.0260   Max.   : 12.0260   Max.   : 12.0260
Lag4               Lag5               Volume             Today              Direction
Min.   :-18.1950   Min.   :-18.1950   Min.   :0.08747   Min.   :-18.1950   Down:484
1st Qu.: -1.1580   1st Qu.: -1.1660   1st Qu.:0.33202   1st Qu.: -1.1540   Up  :605
Median :  0.2380   Median :  0.2340   Median :1.00268   Median :  0.2410
Mean   :  0.1458   Mean   :  0.1399   Mean   :1.57462   Mean   :  0.1499
3rd Qu.:  1.4090   3rd Qu.:  1.4050   3rd Qu.:2.05373   3rd Qu.:  1.4050
Max.   : 12.0260   Max.   : 12.0260   Max.   :9.32821   Max.   : 12.0260
```

(a) Fit a logistic regression model that predicts Direction using Lag1 and Lag2.

```
> fit.glm = glm(Direction ~ Lag1 + Lag2, data = Weekly, family = "binomial")
> summary(fit.glm)

Call:
glm(formula = Direction ~ Lag1 + Lag2, family = "binomial", data = Weekly)

Deviance Residuals:
Min      1Q  Median      3Q     Max
-1.623  -1.261   1.001   1.083   1.506

Coefficients:
Estimate Std. Error z value Pr(>|z|)
(Intercept)  0.22122    0.06147   3.599 0.000319 ***
Lag1        -0.03872    0.02622  -1.477 0.139672
Lag2         0.06025    0.02655   2.270 0.023232 *
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1

(Dispersion parameter for binomial family taken to be 1)

Null deviance: 1496.2  on 1088  degrees of freedom
Residual deviance: 1488.2  on 1086  degrees of freedom
AIC: 1494.2

Number of Fisher Scoring iterations: 4
```

(b) Fit a logistic regression model that predicts Direction using Lag1 and Lag2 using all but the first observation.

```
> fit.glm.b = glm(Direction ~ Lag1 + Lag2, data = Weekly[-1, ], family = "binomial")
> summary(fit.glm.b)

Call:
glm(formula = Direction ~ Lag1 + Lag2, family = "binomial", data = Weekly[-1,
])

Deviance Residuals:
Min       1Q   Median       3Q      Max
-1.6258  -1.2617   0.9999   1.0819   1.5071

Coefficients:
Estimate Std. Error z value Pr(>|z|)
(Intercept)  0.22324    0.06150   3.630 0.000283 ***
Lag1        -0.03843    0.02622  -1.466 0.142683
Lag2         0.06085    0.02656   2.291 0.021971 *
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1

(Dispersion parameter for binomial family taken to be 1)

Null deviance: 1494.6  on 1087  degrees of freedom
Residual deviance: 1486.5  on 1085  degrees of freedom
AIC: 1492.5

Number of Fisher Scoring iterations: 4
```

(c) Use the model from (b) to predict the direction of the first observation. You can do this by predicting that the first observation will go up if P(Direction="Up"—Lag1, Lag2) ¿ 0.5. Was this observation correctly classified?

```
> predict.glm(fit.glm.b, Weekly[1, ], type = "response") > 0.5
1
TRUE
```

Prediction for the observation is Up. This observation was not correctly classified;true direction is Down.

(d) Write a for loop from i=1 to i=n,where n is the number of observations in the data set, that performs each of the following steps:

i. Fit a logistic regression model using all but the ith observation to predict Direction using Lag1 and Lag2.
ii. Compute the posterior probability of the market moving up for the ith observation.
iii. Use the posterior probability for the ith observation in order to predict whether or not the market moves up.
iv. Determine whether or not an error was made in predicting the direction for the ith observation. If an error was made, then indicate this as a 1, and otherwise indicate it as a 0.

```
> error = rep(0, dim(Weekly)[1])
> for (i in 1:dim(Weekly)[1]) {
+      fit.glm = glm(Direction ~ Lag1 + Lag2, data = Weekly[-i, ],  family = "binomial")
+      pred.up = predict.glm(fit.glm, Weekly[i, ], type = "response") > 0.5
+      true.up = Weekly[i, ]$Direction == "Up"
+      if (pred.up != true.up)
+         error[i] = 1
+ }
> error
[1] 1 1 0 1 0 1 0 0 0 1 1 0 1 0 1 0 1 0 1 0 0 0 1 1 1 1 1 1 0 1 1 1 1 0 1 0 0
[38] 0 1 0 1 0 0 1 0 1 1 1 0 1 0 0 0 1 0 0 1 1 0 0 0 0 1 0 1 1 0 0 1 0 1 1 0 0
[75] 0 1 0 1 1 0 0 1 1 0 1 1 0 0 1 0 0 1 1 1 0 0 0 0 0 1 0 1 1 0 0 1 0 1 0 0 1
[112] 1 0 0 1 0 0 1 0 0 1 1 1 1 0 0 0 1 0 1 0 1 1 0 0 0 1 1 1 0 0 0 1 0 0 0 0 0
[149] 0 1 1 1 0 1 0 0 1 1 0 1 0 0 1 1 0 0 1 0 0 1 0 0 1 1 1 0 1 0 1 0 0 0 0 0 0
[186] 0 0 1 1 0 1 0 1 0 1 0 1 0 0 1 0 0 1 0 0 1 0 1 0 1 1 1 0 0 1 1 0 1 0 0 1 1
[223] 0 0 0 1 1 1 0 1 0 1 0 1 0 0 0 1 1 0 1 0 1 0 1 0 1 0 1 1 0 1 0 0 1 0 0 1 0
[260] 0 0 0 0 1 0 0 0 1 0 0 1 0 0 0 1 0 0 1 0 0 1 0 0 1 0 1 1 0 0 0 0 0 1 0 1 0
[297] 0 1 0 0 0 1 0 0 1 1 0 0 1 0 0 0 0 1 0 1 1 0 0 1 0 1 0 1 1 0 0 0 1 0 1 0 0
[334] 1 1 1 1 0 1 0 0 1 0 0 0 1 0 1 0 1 0 0 0 0 0 1 1 0 0 1 0 0 1 0 0 0 1 1 0 1
[371] 1 1 1 1 0 0 0 1 0 0 0 0 0 0 1 0 1 1 0 0 1 1 0 0 0 0 0 1 0 0 1 1 1 0 1 0 1
[408] 0 1 1 1 0 1 0 0 0 0 0 0 0 0 0 0 1 0 1 0 1 0 1 0 0 1 0 1 0 0 0 0 0 1 1 1 1
[445] 0 1 1 0 1 0 1 1 0 1 0 0 1 0 0 1 1 0 0 0 0 1 1 0 0 1 0 1 0 0 0 1 0 0 1 0 0
[482] 0 1 1 1 0 1 0 0 0 1 0 1 1 1 0 0 0 0 1 1 1 0 1 1 0 1 0 0 0 1 0 1 0 0 0 1 0
[519] 1 1 0 0 1 1 0 0 0 1 1 0 1 0 1 1 1 1 1 0 0 0 1 0 0 0 1 1 0 1 0 0 0 1 1 1 1
[556] 1 1 0 1 0 1 0 0 1 0 0 1 1 1 0 0 0 1 1 1 1 1 1 1 1 1 1 0 1 0 0 1 0 0 1 0 1
[593] 0 0 1 0 0 1 0 1 1 0 1 1 1 0 1 0 1 0 1 0 0 0 1 0 1 0 1 0 1 1 0 1 1 0 1 0 1
[630] 0 1 1 1 1 0 1 1 0 0 0 1 1 1 1 0 1 1 1 0 1 0 0 0 1 1 1 1 1 1 0 1 0 0 1 0 0
[667] 0 1 1 0 1 0 1 1 1 1 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 1 0 1 1
[704] 0 0 0 0 1 0 1 0 1 0 1 0 0 1 1 0 0 0 0 0 0 0 0 0 1 0 0 1 0 0 1 1 1 0 1 1 0
[741] 1 1 1 1 0 0 0 1 1 1 1 1 1 0 1 0 0 0 0 0 0 1 0 1 1 1 0 0 0 1 0 0 1 0 0 0 1
[778] 1 1 0 0 0 1 0 0 1 1 1 0 0 1 0 1 0 1 0 0 1 0 0 1 0 0 0 0 0 1 0 1 1 0 0 1 1
[815] 0 1 1 1 0 0 0 0 0 1 1 0 0 1 0 0 1 0 1 0 0 0 1 1 0 1 1 0 1 0 1 0 1 1 0 0 1
[852] 1 1 0 1 1 0 0 0 1 0 1 0 1 0 1 0 0 0 0 0 1 0 0 1 1 0 0 1 0 1 0 1 1 0 1 0 1
[889] 1 0 0 0 1 0 0 0 0 0 0 0 1 0 1 0 1 0 0 0 1 1 1 1 1 0 1 1 0 0 0 0 0 1 0 0 1
[926] 0 0 0 0 1 0 1 1 1 0 0 1 1 0 1 1 1 1 0 1 0 1 0 1 0 1 0 1 0 0 1 1 1 1 1 0 1
[963] 0 0 0 1 1 1 0 1 1 1 1 0 0 0 0 1 1 0 0 0 0 1 0 0 1 1 1 0 0 1 1 1 0 1 0 0 0
[1000] 0 1 0 0 1 0 1 0 0 1 1 1 1 0 1 0 0 1 0 0 1 0 0 1 1 0 1 1 1 0 1 1 0 0 0 1 0
[1037] 1 0 1 1 1 1 0 0 1 0 0 0 0 0 0 1 0 1 1 0 0 0 0 0 1 1 1 0 0 0 1 0 1 1 1 0 0
[1074] 0 0 1 0 0 0 0 0 1 0 1 0 0 0 0 0

> sum(error)
[1] 490
```

(e) Take the average of the n numbers obtained in (d)iv in order to obtain the LOOCV estimate for the test error. Comment on the results.

```
> mean(error)
[1] 0.4499541
```

The LOOCV estimate for the test error rate is 44.99541%.

Problem 4 (Chapter 5 Exercises 9):

We will now consider the Boston housing data set, from the MASS library.

```
> library(MASS)
> set.seed(1)
> attach(Boston)
> summary(Boston)
crim                zn              indus            chas
Min.   : 0.00632   Min.   :  0.00   Min.   : 0.46   Min.   :0.00000
1st Qu.: 0.08204   1st Qu.:  0.00   1st Qu.: 5.19   1st Qu.:0.00000
Median : 0.25651   Median :  0.00   Median : 9.69   Median :0.00000
Mean   : 3.61352   Mean   : 11.36   Mean   :11.14   Mean   :0.06917
3rd Qu.: 3.67708   3rd Qu.: 12.50   3rd Qu.:18.10   3rd Qu.:0.00000
Max.   :88.97620   Max.   :100.00   Max.   :27.74   Max.   :1.00000
nox               rm              age              dis
Min.   :0.3850   Min.   :3.561   Min.   :  2.90   Min.   : 1.130
1st Qu.:0.4490   1st Qu.:5.886   1st Qu.: 45.02   1st Qu.: 2.100
Median :0.5380   Median :6.208   Median : 77.50   Median : 3.207
Mean   :0.5547   Mean   :6.285   Mean   : 68.57   Mean   : 3.795
3rd Qu.:0.6240   3rd Qu.:6.623   3rd Qu.: 94.08   3rd Qu.: 5.188
Max.   :0.8710   Max.   :8.780   Max.   :100.00   Max.   :12.127
rad              tax            ptratio          black
Min.   : 1.000   Min.   :187.0   Min.   :12.60   Min.   :  0.32
1st Qu.: 4.000   1st Qu.:279.0   1st Qu.:17.40   1st Qu.:375.38
Median : 5.000   Median :330.0   Median :19.05   Median :391.44
Mean   : 9.549   Mean   :408.2   Mean   :18.46   Mean   :356.67
3rd Qu.:24.000   3rd Qu.:666.0   3rd Qu.:20.20   3rd Qu.:396.23
Max.   :24.000   Max.   :711.0   Max.   :22.00   Max.   :396.90
lstat            medv
Min.   : 1.73   Min.   : 5.00
1st Qu.: 6.95   1st Qu.:17.02
Median :11.36   Median :21.20
Mean   :12.65   Mean   :22.53
3rd Qu.:16.95   3rd Qu.:25.00
Max.   :37.97   Max.   :50.00
```

(a) Based on this data set, provide an estimate for the population mean of medv. Call this estimate μ.

```
> estimate.mu = mean(medv)
> estimate.mu

[1] 22.53281
```

(b) Provide an estimate of the standard error of μ . Interpret this result.
Hint: We can compute the standard error of the sample mean by dividing the sample standard deviation by the square root of the number of observations.

```
> estimate.mu <- sd(medv) / sqrt(dim(Boston)[1])
> estimate.mu

[1] 0.4088611
```

(c) Now estimate the standard error of μ using the bootstrap. How does this compare to your answer from (b)?

```
> boot.fn = function(data, index) {
+      mu = mean(data[index])
+      return (mu)
+ }
> boot(medv, boot.fn, 1000)

ORDINARY NONPARAMETRIC BOOTSTRAP


Call:
boot(data = medv, statistic = boot.fn, R = 1000)


Bootstrap Statistics :
original    bias    std. error
t1* 22.53281 0.0027583   0.4120131
```

Estimate the standard error of μ using the bootstrap is 0.4120131 and from(b) estimate of the standard error of μ is 0.4088611. They are similar.

(d) Based on your bootstrap estimate from (c), provide a 95 % confidence interval for the mean of medv. Compare it to the results obtained using t.test(Boston$medv).

```
> t.test(medv)

One Sample t-test

data:  medv
t = 55.111, df = 505, p-value < 2.2e-16
alternative hypothesis: true mean is not equal to 0
95 percent confidence interval:
21.72953 23.33608
sample estimates:
mean of x
22.53281




> CI.estimate.mu = c(22.53 - 2 * 0.4119, 22.53 + 2 * 0.4119)
>  CI.estimate.mu

[1] 21.7062 23.3538
```

```
The bootstrap confidence interval estimate :  21.72953 23.33608
                           t.test estimate :  21.7062  23.3538
            Two estimates are only 0.02 away from each other
```

(e) Based on this data set, provide an estimate,μ_{med}, for the median value of medv in the population.

```
> med.estimate <- median(medv)
>  med.estimate

[1] 21.2
```

(f) We now would like to estimate the standard error of μ_{med},.Unfortunately, there is no simple formula for computing the standard error of the median. Instead, estimate the standard error of the median using the bootstrap. Comment on your findings.

```
> boot.fn = function(data, index) {
+      mu = median(data[index])
+      return (mu)
+  }
>  boot(medv, boot.fn, 1000)

ORDINARY NONPARAMETRIC BOOTSTRAP


Call:
boot(data = medv, statistic = boot.fn, R = 1000)


Bootstrap Statistics :
    original  bias    std. error
t1*     21.2 -0.0119   0.3856515
```

Estimate median value is 21.2. It is same value from(e). The standart error 0.3856515 is to small compared to median value.

(g) Based on this data set, provide an estimate for the tenth percentile of medv in Boston suburbs. Call this quantity $\mu_{0.1}$. (You can use the quantile() function.)

```
> percent.estimate <- quantile(medv, c(0.1))
>  percent.estimate

10%
12.75
```

(h) Use the bootstrap to estimate the standard error of $\mu_{0.1}$. Comment on your findings.

```
> boot.fn = function(data, index) {
+        mu = quantile(data[index], c(0.1))
+        return (mu)
+   }
>   boot(medv, boot.fn, 1000)

ORDINARY NONPARAMETRIC BOOTSTRAP

Call:
boot(data = medv, statistic = boot.fn, R = 1000)

Bootstrap Statistics :
   original  bias    std. error
t1*    12.75  0.0041   0.5098364
```

Estimate tenth percentile value is 12.75. It is same value from (g) estimate for the tenth percentile of medv. The standart error 0.5098364 is to small compared to percentile value.