



LAB ASSIGNMENT4

2. 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). (10 points) Fit a logistic regression model that uses income and balance to predict default.

```
Console ~/  
> library(ISLR)
> attach(Default)
> set.seed(1)
> fit.glm <- glm(default ~ income + balance, data = Default, family = "b
inomial")
> 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
```

(b) (10 points total) Using the validation set approach, estimate the test error of this model. In order to do this, you must perform the following steps:

1.(2.5 points) Split the sample set into a training set and a validation set.

(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

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

2.(2.5 points) Fit a multiple logistic regression model using only the training observations.

```
> train <- sample(dim(Default)[1], dim(Default)[1] / 2)
> 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
-2.5830	-0.1428	-0.0573	-0.0213	3.3395

Coefficients:

	Estimate	Std. Error	z value	Pr(> z)
(Intercept)	-1.194e+01	6.178e-01	-19.333	< 2e-16 ***
income	3.262e-05	7.024e-06	4.644	3.41e-06 ***
balance	5.689e-03	3.158e-04	18.014	< 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: 1523.8 on 4999 degrees of freedom
Residual deviance: 803.3 on 4997 degrees of freedom
AIC: 809.3

Number of Fisher Scoring iterations: 8

3. (2.5 points) 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.

Number of Fisher Scoring iterations: 8

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

4. Compute the validation set error, which is the fraction of the observations in the validation set that are misclassified.

```
> 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] 0.0254
>
```

(c). (10 points) Repeat the process in (b) three times, using three different splits of the observations into a training set and a validation set. Comment on the results obtained.

```
> mean(pred.glm != Default[-train, ]$default)
[1] 0.0254
> train <- sample(dim(Default)[1], dim(Default)[1] / 2)
> fit.glm <- glm(default ~ income + balance, data = Default, family = "binomial", subset = train)
> probs <- predict(fit.glm, newdata = Default[-train, ], type = "response")
> pred.glm <- rep("No", length(probs))
> pred.glm[probs > 0.5] <- "Yes"
> mean(pred.glm != Default[-train, ]$default)
[1] 0.0274
> 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, ]$default)
[1] 0.0244
> train <- sample(dim(Default)[1], dim(Default)[1] / 2)
> fit.glm <- glm(default ~ income + balance, data = Default, family = "binomial", subset = train)
> probs <- predict(fit.glm, newdata = Default[-train, ], type = "response")
> pred.glm <- rep("No", length(probs))
> pred.glm[probs > 0.5] <- "Yes"
> mean(pred.glm != Default[-train, ]$default)
[1] 0.0244
```

(d). (10 points) 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.

```
> train <- sample(dim(Default)[1], dim(Default)[1] / 2)
> fit.glm <- glm(default ~ income + balance + student, data = Default, family = "binomial", subset = train)
> pred.glm <- rep("No", length(probs))
> probs <- predict(fit.glm, newdata = Default[-train, ], type = "response")
> pred.glm[probs > 0.5] <- "Yes"
> mean(pred.glm != Default[-train, ]$default)
[1] 0.0278
>
```

3.(40 points) 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). (10 points) 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.

```

> 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) (10 points) 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) (10 points) 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, 1000)
```

ORDINARY NONPARAMETRIC BOOTSTRAP

Call:

```
boot(data = Default, statistic = boot.fn, R = 1000)
```

Bootstrap Statistics :

	original	bias	std. error
t1*	-1.154047e+01	-3.945460e-02	4.344722e-01
t2*	2.080898e-05	1.680317e-07	4.866284e-06
t3*	5.647103e-03	1.855765e-05	2.298949e-04

(d) (10 points) Comment on the estimated standard errors obtained using the `glm()` function and using your bootstrap function.

The standard errors obtained for above questions are approximately close.

4. (40 points) We will now consider the Boston housing data set, from the MASS library.

(a) (5 points) Based on this data set, provide an estimate for the population mean of `medv`. Call this estimate μ^{\wedge} .

```
> library(MASS)
> attach(Boston)
> mu.hat <- mean(medv)
> mu.hat
[1] 22.53281
```

(b) Provide an estimate of the standard error of μ^{\wedge} . 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.

```
> se.hat <- sd(medv) / sqrt(dim(Boston)[1])
> se.hat
[1] 0.4088611
```

(c) (5 points) Now estimate the standard error of μ^{\wedge} using the bootstrap. How does this compare to your answer from (b)?

```

> set.seed(1)
> 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	0.007650791	0.4106622

(d) (5 points) 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)`. Hint: You can approximate a 95% confidence interval using the formula $[\hat{\mu} - 2SE(\hat{\mu}), \hat{\mu} + 2SE(\hat{\mu})]$.

```
> 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.mu.hat <- c(22.53 - 2 * 0.4119, 22.53 + 2 * 0.4119)
> CI.mu.hat
[1] 21.7062 23.3538

```

(e) (5 points) Based on this data set, provide an estimate, $\hat{\mu}_{\text{med}}$, for the median value of medv in the population.

```

> med.hat <- median(medv)
> med.hat
[1] 21.2

```

(f) (5 points) We now would like to estimate the standard error of $\hat{\mu}_{\text{med}}$. 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	-0.0386	0.3770241

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

```
> percent.hat <- quantile(medv, c(0.1))
> percent.hat
10%
12.75
```

(h) (5 points) Use the bootstrap to estimate the standard error of $\hat{\mu}_{\text{med}}$. 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	0.0186	0.4925766