HW4

Sagar Ganapaneni

October 27, 2016

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

library(ISLR)

## Warning: package 'ISLR' was built under R version 3.2.5

attach(Default)  
## set random seed  
set.seed(12)  
glm\_model <- glm(default ~ income + balance, data = Default, family = "binomial")  
summary(glm\_model)

##   
## 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: i. Split the sample set into a training set and a validation set.

## split data  
split = sample(1:nrow(Default), size=nrow(Default)/2)  
Default\_train <- Default[split, ]  
Default\_test<- Default[-split,]

1. Fit a multiple logistic regression model using only the training observations.

glm\_model <- glm(default ~ income + balance, data = Default\_train, family = "binomial")  
summary(glm\_model)

##   
## Call:  
## glm(formula = default ~ income + balance, family = "binomial",   
## data = Default\_train)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.2757 -0.1335 -0.0531 -0.0184 3.7695   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -1.195e+01 6.431e-01 -18.575 <2e-16 \*\*\*  
## income 2.186e-05 7.231e-06 3.023 0.0025 \*\*   
## balance 5.875e-03 3.362e-04 17.476 <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: 1450.21 on 4999 degrees of freedom  
## Residual deviance: 756.78 on 4997 degrees of freedom  
## AIC: 762.78  
##   
## Number of Fisher Scoring iterations: 8

1. 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.

ods <- predict(glm\_model, Default\_test, type = "response")  
glm\_pred <- rep("No", length(ods))  
glm\_pred[ods > 0.5] <- "Yes"  
table(glm\_pred, Default\_train$default)

##   
## glm\_pred No Yes  
## No 4757 165  
## Yes 78 0

mean(glm\_pred != Default\_train$default)

## [1] 0.0486

(c)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.

# run 1  
split = sample(1:nrow(Default), size=nrow(Default)/2)  
Default\_train <- Default[split, ]  
Default\_test<- Default[-split,]  
glm\_model <- glm(default ~ income + balance, data = Default\_train, family = "binomial")  
ods <- predict(glm\_model, Default\_test, type = "response")  
glm\_pred <- rep("No", length(ods))  
glm\_pred[ods > 0.5] <- "Yes"  
table(glm\_pred, Default\_train$default)

##   
## glm\_pred No Yes  
## No 4743 171  
## Yes 86 0

mean(glm\_pred != Default\_train$default)

## [1] 0.0514

# run 2  
split = sample(1:nrow(Default), size=nrow(Default)/2)  
Default\_train <- Default[split, ]  
Default\_test<- Default[-split,]  
glm\_model <- glm(default ~ income + balance, data = Default\_train, family = "binomial")  
ods <- predict(glm\_model, Default\_test, type = "response")  
glm\_pred <- rep("No", length(ods))  
glm\_pred[ods > 0.5] <- "Yes"  
table(glm\_pred, Default\_train$default)

##   
## glm\_pred No Yes  
## No 4752 168  
## Yes 77 3

mean(glm\_pred != Default\_train$default)

## [1] 0.049

# run 3  
split = sample(1:nrow(Default), size=nrow(Default)/2)  
Default\_train <- Default[split, ]  
Default\_test<- Default[-split,]  
glm\_model <- glm(default ~ income + balance, data = Default\_train, family = "binomial")  
ods <- predict(glm\_model, Default\_test, type = "response")  
glm\_pred <- rep("No", length(ods))  
glm\_pred[ods > 0.5] <- "Yes"  
table(glm\_pred, Default\_train$default)

##   
## glm\_pred No Yes  
## No 4756 171  
## Yes 70 3

mean(glm\_pred != Default\_train$default)

## [1] 0.0482

Mention note on different test errors

(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.

## with dummy variable student  
  
split = sample(1:nrow(Default), size=nrow(Default)/2)  
Default\_train <- Default[split, ]  
Default\_test<- Default[-split,]  
glm\_model <- glm(default ~ income + balance+student, data = Default\_train, family = "binomial")  
ods <- predict(glm\_model, Default\_test, type = "response")  
glm\_pred <- rep("No", length(ods))  
glm\_pred[ods > 0.5] <- "Yes"  
table(glm\_pred, Default\_train$default)

##   
## glm\_pred No Yes  
## No 4785 162  
## Yes 51 2

mean(glm\_pred != Default\_train$default)

## [1] 0.0426

Problme 3 (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(12)  
attach(Default)

## The following objects are masked from Default (pos = 3):  
##   
## balance, default, income, student

glm\_model <- glm(default ~ income + balance, data = Default, family = "binomial")  
summary(glm\_model)

##   
## 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

Talk about coefficnet and standard errors

(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, split) {  
 glm\_model <- glm(default ~ income + balance, data = data, family = "binomial", subset = split)  
 return (coef(glm\_model))  
}

(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, 1000)

##   
## ORDINARY NONPARAMETRIC BOOTSTRAP  
##   
##   
## Call:  
## boot(data = Default, statistic = boot.fn, R = 1000)  
##   
##   
## Bootstrap Statistics :  
## original bias std. error  
## t1\* -1.154047e+01 -3.132754e-02 4.224566e-01  
## t2\* 2.080898e-05 1.195612e-07 4.791184e-06  
## t3\* 5.647103e-03 1.330443e-05 2.221212e-04

Talk about the beetas and SE

Problem 4:

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

library(MASS)  
attach(Boston)  
EMu\_medv<- mean(medv)  
EMu\_medv

## [1] 22.53281

(b)Provide an estimate of the standard error of ??^. Interpret this result.

# SE\_EMu\_mdev <- sd(medv) / sqrt(dim(Boston)[1])  
# SE\_EMu\_mdev

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

library(boot)  
set.seed(12)  
boot.fn <- function(data, split) {  
 EMu <- mean(data[split])  
 return (EMu)  
}  
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 0.009365217 0.4123366

compare with previous results

(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\_EMu <- c(22.53 - 2 \* 0.4119, 22.53 + 2 \* 0.4119)  
CI\_EMu

## [1] 21.7062 23.3538

(e)Based on this data set, provide an estimate, ??^med, for the median value of "medv" in the population.

EMed\_medv <- median(medv)  
EMed\_medv

## [1] 21.2

1. We now would like to estimate the standard error of ??med??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, split) {  
 Emedian <- median(data[split])  
 return (Emedian)  
}  
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 -0.01685 0.3775859

(g)Based on this data set, provide an estimate for the tenth percentile of "medv" in Boston suburbs. Call this quantity ??^0.1

Mu\_10p <- quantile(medv, c(0.1))  
Mu\_10p

## 10%   
## 12.75

(h)Use the bootstrap to estimate the standard error of ??^0.1. Comment on your findings.

boot.fn <- function(data, split) {  
 mu <- quantile(data[split], 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 -0.0095 0.5151797