Capítulo 5. Resampling Methods

Diego Felipe Bobadilla Restrepo
Karol Andrea Ujueta Rojas
Yosef Shmuel Guevara Salamanca

June 9, 2021

Ejercicios

1. Using basic statistical properties of the variance, as well as singlevariable calculus, derive (5.6). In other words, prove that α given by (5.6) does indeed minimize $Var(\alpha X + (1-\alpha)Y)$.

$$\alpha = \frac{\sigma_Y^2 - \sigma_{XY}}{\sigma_X^2 + \sigma_Y^2 - 2\sigma_{XY}} \qquad (5.6)$$

Se sabe que la varianza para 2 variables aleatorias esta dada por:

$$Var(aX + bY) = a^2 Var(X) + b^2 Var(Y) + 2abCov(X, Y)$$

Al aplicarlo en nuestro caso tenemos que:

$$Var(aX + (1-\alpha)Y) = \alpha^2 Var(X) + (1-\alpha)^2 Var(Y) + 2\alpha(1-\alpha)Cov(X, Y)$$

Expandiendo:

$$\alpha^2 Var(X) + Var(Y) - 2\alpha Var(Y) + \alpha^2 Var(Y) + 2\alpha Cov(X, Y) - 2\alpha^2 Cov(X, Y)$$

Derivando respecto a α:

$$\frac{d}{d\alpha}\left(\alpha^2 Var(X) + Var(Y) - 2\alpha Var(Y) + \alpha^2 Var(Y) + 2\alpha Cov(X, Y) - 2\alpha^2 Cov(X, Y)\right)$$

$$2\alpha Var(X) - 2Var(Y) + 2\alpha Var(Y) + 2Cov(X, Y) - 4\alpha Cov(X, Y)$$

Igualamos a ceros.

$$2\alpha Var(X) - 2Var(Y) + 2\alpha Var(Y) + 2Cov(X, Y) - 4\alpha Cov(X, Y) = 0$$

Agrupando términos semejantes y cambiar la notación tenemos que:

$$2\alpha(\sigma_X^2 + \sigma_Y^2 - 2\sigma_{XY}) - 2(\sigma_Y^2 - \sigma_{XY}) = 0$$

$$2\alpha(\sigma_X^2 + \sigma_Y^2 - 2\sigma_{XY}) = 2(\sigma_Y^2 - \sigma_{XY})$$

$$\alpha = \frac{2(\sigma_Y^2 - \sigma_{XY})}{2(\sigma_X^2 + \sigma_Y^2 - 2\sigma_{XY})}$$

Finalmente tenemos que:

$$\alpha = \frac{\sigma_Y^2 - \sigma_{XY}}{\sigma_X^2 + \sigma_Y^2 - 2\sigma_{XY}}$$

2. We will now derive the probability that a given observation is part of a bootstrap sample. Suppose that we obtain a bootstrap sample from a set of n observations.

   (a) What is the probability that the first bootstrap observation is not the jth observation from the original sample? Justify your answer.

   La probabilidad de que una observación jth este en el muestro original es $1/n$, todas las muestras $x_1, ...x_n$ tienen la misma probabilidad de ser seleccionadas, por lo tanto la probabilidad de una observación jth no ser seleccionada es:

   $$1 - \frac{1}{n}$$

   (b) What is the probability that the second bootstrap observation is not the jth observation from the original sample?

   Debido a que todas las muestras son independientes y a que se selecciona con reemplazo la posibilidad de que la segunda observación bustrapp no sea la jth es:

   $$1 - \frac{1}{n}$$

   Igual que el caso anterior.

   (c) Argue that the probability that the jth observation is not in the bootstrap sample is $(1 - 1/n)^n$.

   Gracias a los primeros 2 puntos podemos ver que la probabilidad de cualquier observiación de no ser sleccionada para el muestreo esta dada por:

   $$p_j(n) = \prod_{i=1}^{n} \pi_j = \left(1 - \frac{1}{n}\right)\left(1 - \frac{1}{n}\right)...\left(1 - \frac{1}{n}\right) = \left(1 - \frac{1}{n}\right)^n$$

   (d) When $n = 5$, what is the probability that the jth observation is in the bootstrap sample?

   Usando la ecuación en (c), la probabilidad que la observación sea seleccionada por el muestreo boostrap es:

   $$1 - \left(1 - \frac{1}{5}\right)^5 = 0.67232$$

   (e) When $n = 100$, what is the probability that the jth observation is in the bootstrap sample?

   $$1 - \left(1 - \frac{1}{100}\right)^{100} = 0.63396$$

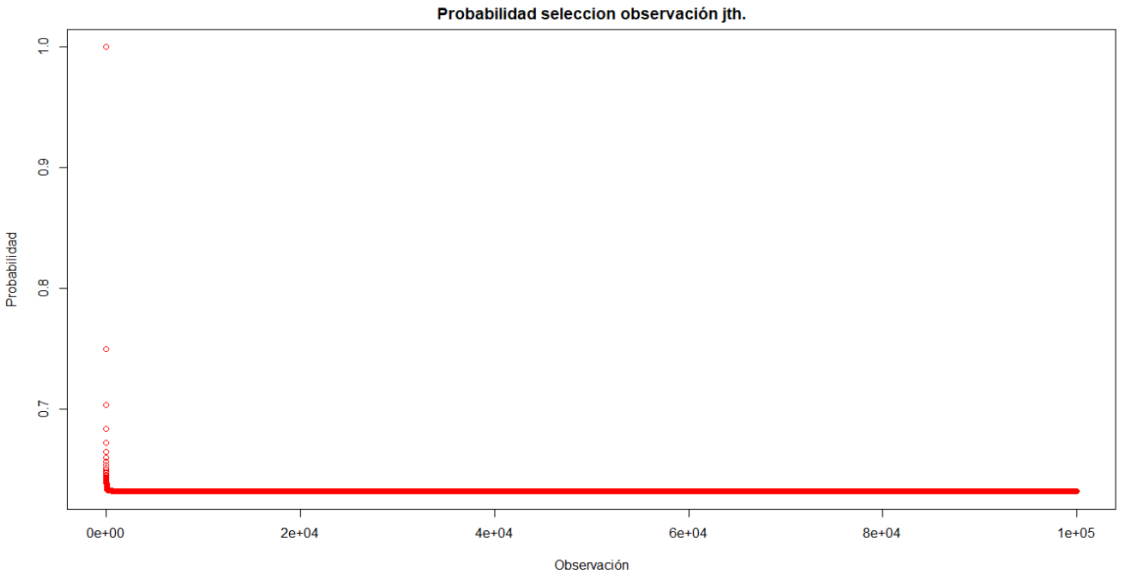
(f) When $n = 10,000$, what is the probability that the jth observation is in the bootstrap sample?

$$1 - \left(1 - \frac{1}{10000}\right)^{10000} = 0.63214$$

(g) Create a plot that displays, for each integer value of n from 1 to $100,000$, the probability that the jth observation is in the bootstrap sample. Comment on what you observe.

```
## Esta funcion calcula la probabilidad de que una observación jth
## sea seleccionada por el muestreo boostrap

probabilidad <- function(n){

  1 - (1-1/n)^n

}
```

```
n <- rep(1:100000)

prob <- sapply(n, probabilidad)

plot(prob,main="jth sea seleccionada por el muestreo boostrap.",
     xlab="observation", ylab="probability", col="blue")
```



Es notable que las probabilidades convergen hacia cierto punto. Podemos calcular este punto sabiendo que la función exponencial satisface que:

$$e^x = \lim_{n\to\infty}\left(1 + \frac{x}{n}\right)$$

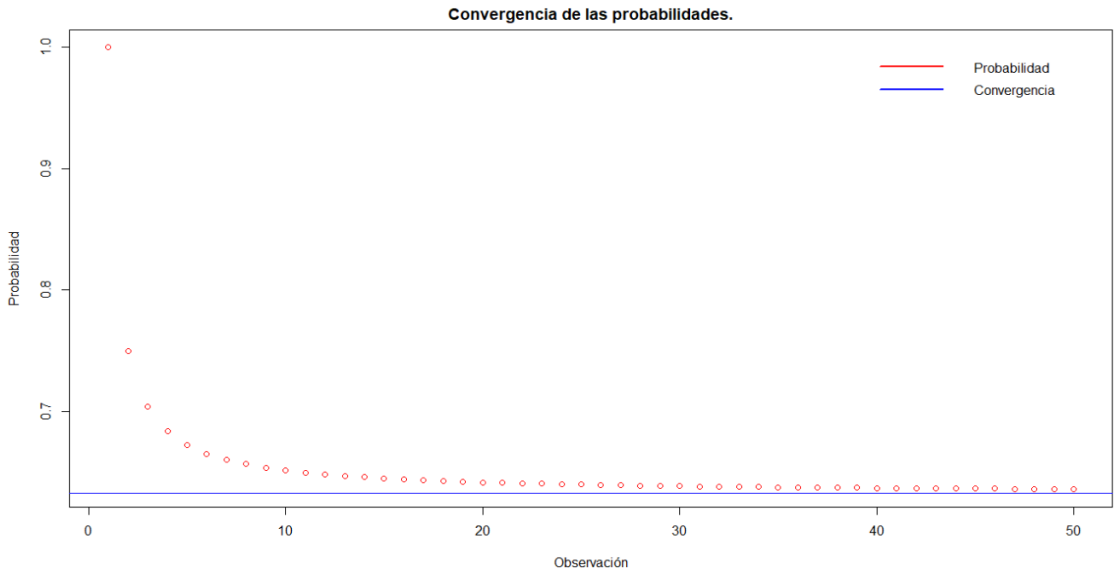
Se puede ver que:

$$p_j := \lim_{n\to\infty} p_j(n) = e^{-1} \approx 0.3678$$

Por lo tanto el limite de la probabilidad jth de ser seleccionada por el muestreo bootstrap cuando n incrementa es:

$$1 - 0.3678 = 0.6321$$

Al revisar solo las primeras 50 observaciones tenemos que.



(h) We will now investigate numerically the probability that a bootstrap sample of size $n = 100$ contains the jth observation. Here $j = 4$. We repeatedly create bootstrap samples, and each time we record whether or not the fourth observation is contained in the bootstrap sample.

```
store=rep (NA , 10000)

for (i in 1:10000) {
  store[i]=sum(sample (1:100 , rep =TRUE)==4) >0
}

mean(store)
```

[1] 0.6324

Comment on the results obtained.

Este codigo genera 10.000 muestras, del conjunto de entre el rango $\{1... 100\}$, siendo el tamaño de la muestra 100, produciendo un remplazo de cada muestreo $rep = True$,

```
> sample (1:100 , rep =TRUE)
  [1]  41  93  70  27  64   7  77   7  46  49  84   6  64  96  62  15   6  80  85  77  62  18  58
 [24]  88  59  17  55  18  58  37  62  84  28  12  27  28  10   5  54  73  71   5  78  75  87   8
 [47]  53  62  50  13  74  26  62  58  35  37  46  46  51  66  88  16  68   8  30  37  84  32  66
 [70]  62   9  23  22  54  10  68  87  43  80  14  56  50  46  56  56  19  46  63   5  31  41  75
 [93]  87 100  29  39   5  53   7  41
```

Luego $sum(sample(1:100, rep = TRUE) == 4) > 0$, Sumara una unidad cada vez que en el dentro de las 100 muestras seleccionadas se encuentre el 4,

```
> sum(sample (1:100 , rep =TRUE)==4)
[1] 2
```

Y se almacena un $TRUE$,en la posición $[i]$ del vector *store* cuando la suma de números 4 sea mayor a 0.

```
> sum(sample (1:100 , rep =TRUE)==4)>0
[1] TRUE
```

Finalmente se promedian la cantidad de $TRUE$ en el vector *store*, siendo El resultado muy cercano al valor de la convergencia calculada en el punto (g) como era lo esperado al acercarse $n \to \infty$.

3. We now review k-fold cross-validation.

   (a) Explain how k-fold cross-validation is implemented.

   En la validación cruzada se divide aleatoriamente el conjunto de observaciones en k grupos aproximadamente del mismo tamaño, el primer grupo se trata como un conjunto de validación y se ajusta para los restantes k-1 grupos. El error cuadrático medio de aplica sobre las observaciones sobre el grupo seleccionado es decir sobre el grupo de entrenamiento y se repite tantas veces como k seleccionados en primera instancia para el grupo de prueba.

   (b) What are the advantages and disadvantages of k-fold crossvalidation relative to:

      i. The validation set approach?
      La estimación de validación de la tasa de error de la prueba puede ser muy variable. Este método tiene menos varianza, pero más sesgo. Además, como se debe seleccionar un conjunto de las observaciones, se puede sobrestimar la tasa de error de prueba en los casos donde el conjunto seleccionado contenga pocos datos.

      ii. LOOCV?
      El método de validación cruzada de LOOCV puede dar estimaciones aproximadamente insesgadas del error de prueba, ya que cada conjunto de entrenamiento contiene n 1 observaciones; sin embargo, este método tiene una varianza más alta debido al promedio de resultados de n modelos ajustados con respecto al k-fold.

4. Suppose that we use some statistical learning method to make a prediction for the response Y for a particular value of the predictor X. Carefully describe how we might estimate the standard deviation of our prediction.

   Podemos hacer esto mediante boostrap. Es decir debemos generar una gran cantidad de estimaciones para α repitiendo este procdeimiento B veces, para algún valor grande de B, para producir B conjunto de datos diferentes $Z_1, ...Z_B$ entrenar nuestro modelo para obtener las estimaciones $\hat{\alpha}_1, ...\hat{\alpha}_b$ ver como cambian las estimaciones, esto lo podemos calcular mediante la ecuación (5.8).

$$SE_B(\hat{\alpha}) = \sqrt{\frac{1}{B-1}\sum_{r=1}^{B}\left(\hat{\alpha}^{*r} - \frac{1}{B}\sum_{r'=1}^{B}\hat{\alpha}^{*r'}\right)^2}$$

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.

```
attach(Default)
names(Default)
```

```
## [1] "default" "student" "balance" "income"
```

```
set.seed(1)
mod_1 <- glm(default ~ income + balance, data = Default, family = "binomial")
summary(mod_1)
```

```
##
## 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.

```
train <- sample(nrow(Default), nrow(Default)*0.5)
```

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

```
mod_1 <- glm(default ~ income + balance, data = Default, family = "binomial",
subset = train)
summary(mod_1)
```

```
##
## 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
```

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(mod_1, newdata = Default[-train, ], type = "response")
pred.mod1 <- rep("No", length(probs))
pred.mod1[probs > 0.5] <- "Yes"
```

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

```
pred <- mean(pred.mod1 != Default[-train, ]$default)
pred
```

```
## [1]



 0.0254
```

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

```
set.seed(2)
train <- sample(nrow(Default), nrow(Default)*0.5)
mod_2 <- glm(default ~ income + balance, data = Default, family = "binomial",
subset = train)
summary(mod_2)

##
## Call:
## glm(formula = default ~ income + balance, family = "binomial",
##     data = Default, subset = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.3702  -0.1628  -0.0673  -0.0259   3.6470
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.090e+01  5.749e-01 -18.955   <2e-16 ***
## income       1.622e-05  6.891e-06   2.354   0.0186 *
## balance      5.365e-03  3.049e-04  17.598   <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: 1483.83  on 4999  degrees of freedom
## Residual deviance:  854.49  on 4997  degrees of freedom
## AIC: 860.49
##
## Number of Fisher Scoring iterations: 8

probs <- predict(mod_2, newdata = Default[-train, ], type = "response")
pred.mod2 <- rep("No", length(probs))
pred.mod2[probs > 0.5] <- "Yes"

pred2 <- mean(pred.mod2 != Default[-train, ]$default)
pred2

## [1] 0.0238

set.seed(3)
train <- sample(nrow(Default), nrow(Default)*0.5)
mod_3 <- glm(default ~ income + balance, data = Default, family = "binomial",
subset = train)
summary(mod_3)
## Call:
## glm(formula = default ~ income + balance, family = "binomial",
```

```
##     data = Default, subset = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.5804  -0.1390  -0.0524  -0.0180   3.7789
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.204e+01  6.326e-01 -19.031  < 2e-16 ***
## income       2.462e-05  6.941e-06   3.547  0.00039 ***
## balance      5.894e-03  3.287e-04  17.929  < 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: 1536.99  on 4999  degrees of freedom
## Residual deviance:  793.34  on 4997  degrees of freedom
## AIC: 799.34
##
## Number of Fisher Scoring iterations: 8
```

```
probs <- predict(mod_3, newdata = Default[-train, ], type = "response")
pred.mod3 <- rep("No", length(probs))
pred.mod3[probs > 0.5] <- "Yes"

pred3 <- mean(pred.mod3 != Default[-train, ]$default)
pred3
```

```
## [1] 0.0264
```

Se logra observar que la tasa de error es variable dependiendo del conjunto de observaciones de los datos de entrenamiento y del conjunto de observaciones para la validación. Sin embargo, las variaciones no son muy grandes, todas se encuentran entre 2.3% y 2.6%

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.

```
set.seed(1)
train <- sample(nrow(Default), nrow(Default)*0.5)
mod_4 <- glm(default ~ income + balance + student, data = Default, family = "
binomial", subset = train)
pred.mod4 <- rep("No", length(probs))
probs <- predict(mod_4, newdata = Default[-train, ], type = "response")
pred.mod4[probs > 0.5] <- "Yes"
mean(pred.mod4 != Default[-train, ]$default)
```

```
## [1] 0.026
```

Incluir la variable Student dentro del modelo no modifica la tasa de error de prueba.

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.

```
library("ISLR")
data(Default)
attach(Default)
```

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.

```
glm.fit = glm(default ~ income + balance, family=binomial)
summary(glm.fit)
```

```
##
## Call:
## glm(formula = default ~ income + balance, family = binomial)
##
## 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
```

Se puede ver con claridad que tanto **income** como **balance** son variables predictoras significativas para este modelo pues p-value < .05; sin embargo tanto la **desviación residual** como el **AIC**, son muy grandes, lo cual es un indicativo que el modelo entrenado no es el más el adecuado.

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.

```
##Se establece una semill para que los resultados siempre sean replicables.

set.seed(13)
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.

```
set.seed(13)
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 -2.509758e-02 4.106349e-01
## t2*  2.080898e-05  2.341013e-08 4.571344e-06
## t3*  5.647103e-03  1.483680e-05 2.162846e-04
```

Los estimados de los errores estándar generados por el boostrap son:

- Std. error para $\widehat{\beta_0} = 4.11e-01$
- Std. error para $\widehat{\beta_1} = 4.57e-06$
- Std. error para $\widehat{\beta_2} = 2.16e-04$

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

Al comparar los resultados de los errores estándar estimados para los puntos (c) y (d),vemos que son muy cercanos entre sí, por lo cual se puede decir que el modelo boostrap es eficiente.

La cercanía entre estos valores es cada vez más notoria al incrementar el número de observaciones utilizadas en el boostrap como se puede ver a continuación.

```
## Boostrap con 100 datos

set.seed(13)
library(boot)
boot(Default, boot.fn, 100)

##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = Default, statistic = boot.fn, R = 100)
##
##
## Bootstrap Statistics :
##          original        bias     std. error
## t1* -1.154047e+01 -3.496958e-02 4.214075e-01
## t2*  2.080898e-05  6.619300e-07 5.012127e-06
## t3*  5.647103e-03  9.342470e-06 2.173372e-04

## Boostrap con 5000 datos
set.seed(13)
library(boot)
boot(Default, boot.fn, 5000)

##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = Default, statistic = boot.fn, R = 5000)
##
##
## Bootstrap Statistics :
##          original        bias     std. error
## t1* -1.154047e+01 -2.999144e-02 4.343384e-01
## t2*  2.080898e-05 -1.531876e-09 4.795164e-06
## t3*  5.647103e-03  1.769232e-05 2.289305e-04
```

La siguiente tabla resumen nos ayuda a visualizar resultados anteriores.

Std. Error para	100	1000	5000	Todos los datos
$\hat{\beta}_0$	4.21e-01	4.11e-01	4.34e-0.1	4.34e-01
$\hat{\beta}_1$	5.01e-06	4.57e-06	4.79e-06	4.98e-06
$\hat{\beta}_2$	2.17e-04	2.16e-04	2.28e-04	2.27e-04

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).

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

```
data(Weekly)
attach(Weekly)
names(Weekly)
```

```
## [1] "Year"      "Lag1"      "Lag2"      "Lag3"      "Lag4"      "Lag5"
## [7] "Volume"    "Today"     "Direction"
```

```
set.seed(1)
mod1 <- glm(Direction ~ Lag1 + Lag2, data=Weekly, family=binomial)
summary(mod1)
```

```
##
## 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.

```
set.seed(1)
mod2 <- glm(Direction ~ Lag1 + Lag2, data = Weekly[-1, ], family = "binomial"
)
summary(mod2)
```

```
##
## 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?

```
ifelse(predict(mod2, Weekly[1,], type="response")>0.5, "Up", "Down")

##    1
## "Up"

Weekly[1,]$Direction

## [1] Down
## Levels: Down Up
```

La primera predicción estpa clasificada de forma incorrecta ya que debería ser UP según los niveles o categorías establecidas.

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.

```
set.seed(1)


loocv.err <- rep(0,nrow(Weekly))


for (i in 1:nrow(Weekly)) {


  mod <- glm(Direction ~ Lag1 + Lag2, data=Weekly[-i,], family=binomial)


  mod_pred <- ifelse(predict(mod, Weekly[1,], type="response")>0.5, "Up", "Do
wn")


  loocv.err[i] <- ifelse(Weekly[i,]$Direction==mod_pred, 0, 1)


}


str(loocv.err)

##  num [1:1089] 1 1 0 0 0 1 0 0 0 1 ...
```

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(loocv.err)

## [1] 0.4444444
```

La estimación para la tase de error de prueba es de 44.4%.

8. We will now perform cross-validation on a simulated data set.

(a) Generate a simulated data set as follows

```
set.seed(1)
y=rnorm (100)
x=rnorm (100)
y=x-2*x^2+ rnorm (100)
```

In this data set, what is **n** and what is **p**? Write out the model used to generate the data in equation form.

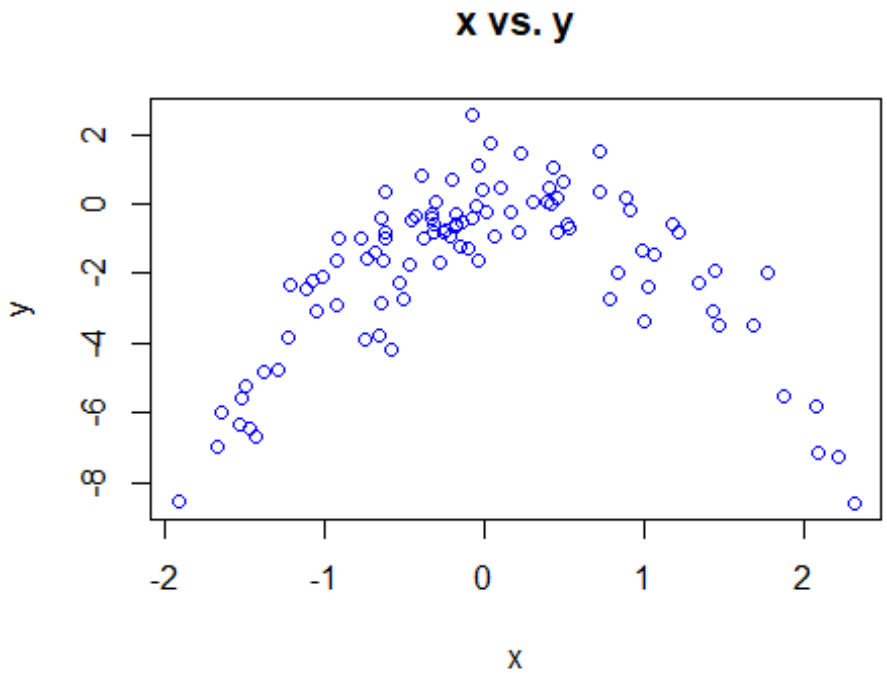
Tenemos que $n = 100$ y $p = 2$, el modelo usado para generar esta data es:

$$y = x - 2x^2 + \varepsilon$$

$$\varepsilon \sim N(0,1)$$

(b) Create a scatterplot of X against Y . Comment on what you find.

```
plot(x,y, main="x vs. y", col="blue")
```



Se observa claramente una relación curva entre **X** y **Y**.

(c) Set a random seed, and then compute the LOOCV errors that result from fitting the following four models using least squares:

```
library(boot)
### se crea un data frame para facilitar la manipulación de los datos
Data <- as.data.frame(cbind(x, y))
set.seed(1)
```

$$i. Y = \beta_0 + \beta_1 X + \varepsilon$$

```
glm.fit1 <- glm(y ~ x)
cv.error1<-cv.glm(Data, glm.fit1)$delta
cv.error1[1]
```

```
## [1] 5.890979
```

La función cv.error<-cv.glm() produce una lista con varios componentes. El primer valor del vector delta contiene los resultados de la validación cruzada, siendo esta el promedio del error cuadrático medio de los **n** errores del conjunto de prueba estimados.

$$ii. Y = \beta_0 + \beta_1 X + \beta_2 X^2 + \varepsilon$$

```
glm.fit2 <- glm(y ~ poly(x, 2))
cv.error2<-cv.glm(Data, glm.fit2)$delta
cv.error2[1]
```

```
## [1] 1.086596
```

$$iii. Y = \beta_0 + \beta_1 X + \beta_2 X^2 + \beta_3 X^3 + \varepsilon$$

```
glm.fit3 <- glm(y ~ poly(x, 3))
cv.error3<-cv.glm(Data, glm.fit3)$delta
cv.error3[1]
```

```
## [1] 1.102585
```

$$iv. Y = \beta_0 + \beta_1 X + \beta_2 X^2 + \beta_3 X^3 + \beta_4 X^4 + \varepsilon$$

```
glm.fit4 <- glm(y ~ poly(x, 4))
cv.error4<-cv.glm(Data, glm.fit4)$delta
cv.error4[1]
```

```
## [1] 1.114772
```

(d) Repeat (c) using another random seed and report your results. Are your results the same as what you got in (c)? Why?

```
set.seed(10)
glm.fit1 <- glm(y ~ x)
cv.error1<-cv.glm(Data, glm.fit1)$delta
cv.error1[1]

## [1] 5.890979

glm.fit2 <- glm(y ~ poly(x, 2))
cv.error2<-cv.glm(Data, glm.fit2)$delta
cv.error2[1]

## [1] 1.086596

glm.fit3 <- glm(y ~ poly(x, 3))
cv.error3<-cv.glm(Data, glm.fit3)$delta
cv.error3[1]

## [1] 1.102585

glm.fit4 <- glm(y ~ poly(x, 4))
cv.error4<-cv.glm(Data, glm.fit4)$delta
cv.error4[1]

## [1] 1.114772
```

Los resultados son idénticos debido a que LOOCV solo usa una observación para validar el modelo, las demás observaciones son usadas para el entrenamiento.

(e) Which of the models in (c) had the smallest LOOCV error? Is this what you expected? Explain your answer.

El error cuadrático medio **MSE** más pequeño corresponde estimado por el LOOCV el modelo **glm.fit2**, puesto que como se establecido al durante la simulación del ejercicio la relación entre las variable es cuadrática.

(f) Comment on the statistical significance of the coefficient estimates that results from fitting each of the models in (c) using least squares. Do these results agree with the conclusions drawn based on the cross-validation results?

```
summary(glm.fit4)

##
## Call:
## glm(formula = y ~ poly(x, 4))
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.8914  -0.5244   0.0749   0.5932   2.7796
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -1.8277     0.1041 -17.549   <2e-16 ***
## poly(x, 4)1   2.3164     1.0415   2.224   0.0285 *
## poly(x, 4)2 -21.0586     1.0415 -20.220   <2e-16 ***
## poly(x, 4)3  -0.3048     1.0415  -0.293   0.7704
## poly(x, 4)4  -0.4926     1.0415  -0.473   0.6373
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 1.084654)
##
##     Null deviance: 552.21  on 99  degrees of freedom
## Residual deviance: 103.04  on 95  degrees of freedom
## AIC: 298.78
##
## Number of Fisher Scoring iterations: 2
```

Gracia al p-value se identifica que solo los coeficientes β_1 y β_2, que acompañan a los términos linear y cuadratico respectivamente son significativos. Tanto el **AIC** como la **deviación residual muestran** un buen ajuste del modelo dado que sus valores no son muy grandes.

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

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

```
data("Boston")
attach(Boston)
names(Boston)

##  [1] "crim"    "zn"      "indus"   "chas"    "nox"     "rm"      "age"
##  [8] "dis"     "rad"     "tax"     "ptratio" "black"   "lstat"   "medv"

mu.hat <- mean(medv)
mu.hat

## [1] 22.53281
```

b) Provide an estimate of the standard error of μ. Interpret this result.

```
medv.se <- sd(Boston$medv)/sqrt(nrow(Boston))
medv.se

## [1] 0.4088611
```

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

```
set.seed(1)
mean.fn <- function(var, id) {
  return(mean(var[id]))
}
boot.res <- boot(Boston$medv, mean.fn, R=100)
boot.res

##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = Boston$medv, statistic = mean.fn, R = 100)
##
##
## Bootstrap Statistics :
##     original      bias    std. error
## t1* 22.53281 0.009027668   0.3482331
```

El error estimado Boostrap es 0.36042 el cual es menor que el encontrado en el literal (b) (pasa de 0.408 a 0.360 con el método Boostrap)

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

IC.mu.hat <- c(22.53 - 2 * 0.360419, 22.53 + 2 * 0.360419)
IC.mu.hat

## [1] 21.80916 23.25084
```

El intervalo con un nivel de confianza del 95% será (21.809, 23.251)

e) Based on this data set, provide an estimate, μmed, for the median value of medv in the population.

```
med.hat <- median(medv)
med.hat

## [1] 21.2
```

f) We now would like to estimate the standard error of ˆ μmed. 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.

```
set.seed(1)
median.fn <- function(var, id) {
  return(median(var[id]))
}
(boot.res <- boot(Boston$medv, median.fn, R=100))

##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = Boston$medv, statistic = median.fn, R = 100)
##
##
## Bootstrap Statistics :
##     original  bias    std. error
## t1*     21.2  -0.029   0.3461316
```

Se evidencia que el resiltado es 21.2 igual que en el literal anterior, con un error estandar de 0.36.

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.)

```
medv.qt <- quantile(Boston$medv, 0.1)
medv.qt
```

```
##   10%
## 12.75
```

h) Use the bootstrap to estimate the standard error of $\mu 0.1$. Comment on your findings.

```
set.seed(1)
quantil_10.fn <- function(var, id) {
  return(quantile(var[id], 0.1))
}
(boot.res <- boot(Boston$medv, quantil_10.fn, R=100))
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = Boston$medv, statistic = quantil_10.fn, R = 100)
##
##
## Bootstrap Statistics :
##     original  bias    std. error
## t1*    12.75   0.008   0.5370477
```

El error estimado es de 0.537