Práctica 1

Grupo 4

Iria Lago Portela
Mario Picáns Rey
Javier Kniffki
David Bamio Martínez

Ejercicios

```
##        rpart   rpart.plot        caret randomForest          pdp      kernlab
##         TRUE         TRUE         TRUE         TRUE         TRUE         TRUE
```

Preparación de los datos:

```
unis <- College4[, colnames(College4) != "Private"]

unis$Tipo <- factor(College4$Private == "Yes", labels = c("Pública", "Privada")) #Factor y cambio de et

#Proporción privada-pública
table(unis$Tipo)
```

```
##
## Pública Privada
##     143     357
```

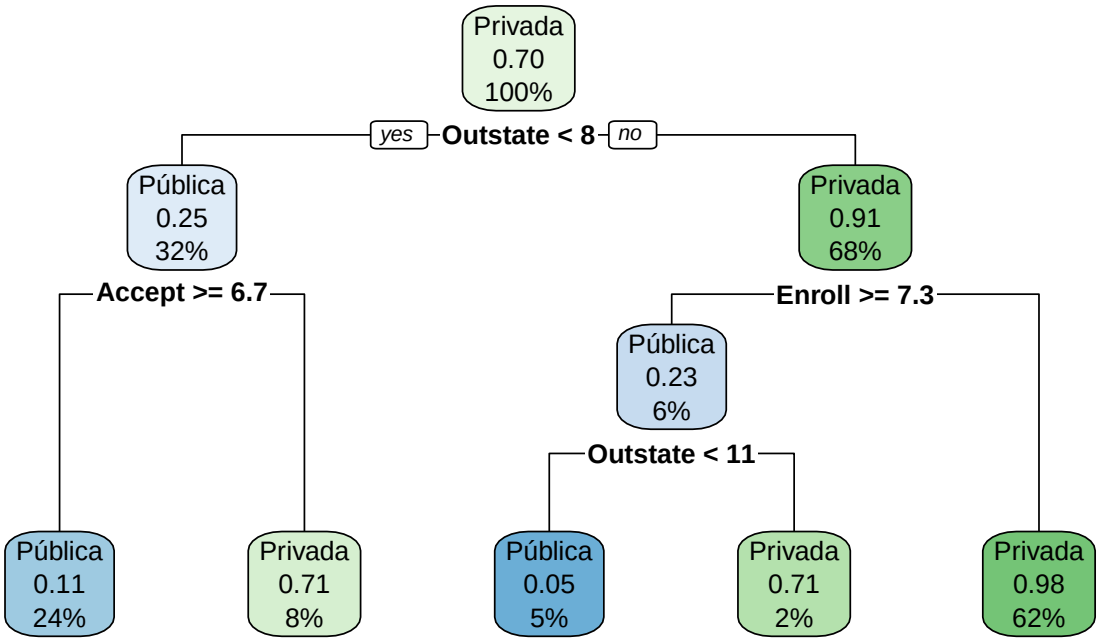
```
#Semilla
set.seed(40)

nobs <- nrow(unis) #Filas
itrain <- sample(nobs, 0.8 * nobs)
train <- unis[itrain, ] # M. Entrenamiento
test <- unis[-itrain, ] # M. Prueba

tree <- rpart(Tipo ~ ., data = train)

#Gráfico
rpart.plot(tree, main = "Árbol de clasificación Privada-Pública")
```

Árbol de clasificación Privada−Pública



1. Obtener un árbol de decisión que permita clasificar las observaciones (universidades)

en privadas (`Private="Yes"`) o públicas (`Private="No"`).

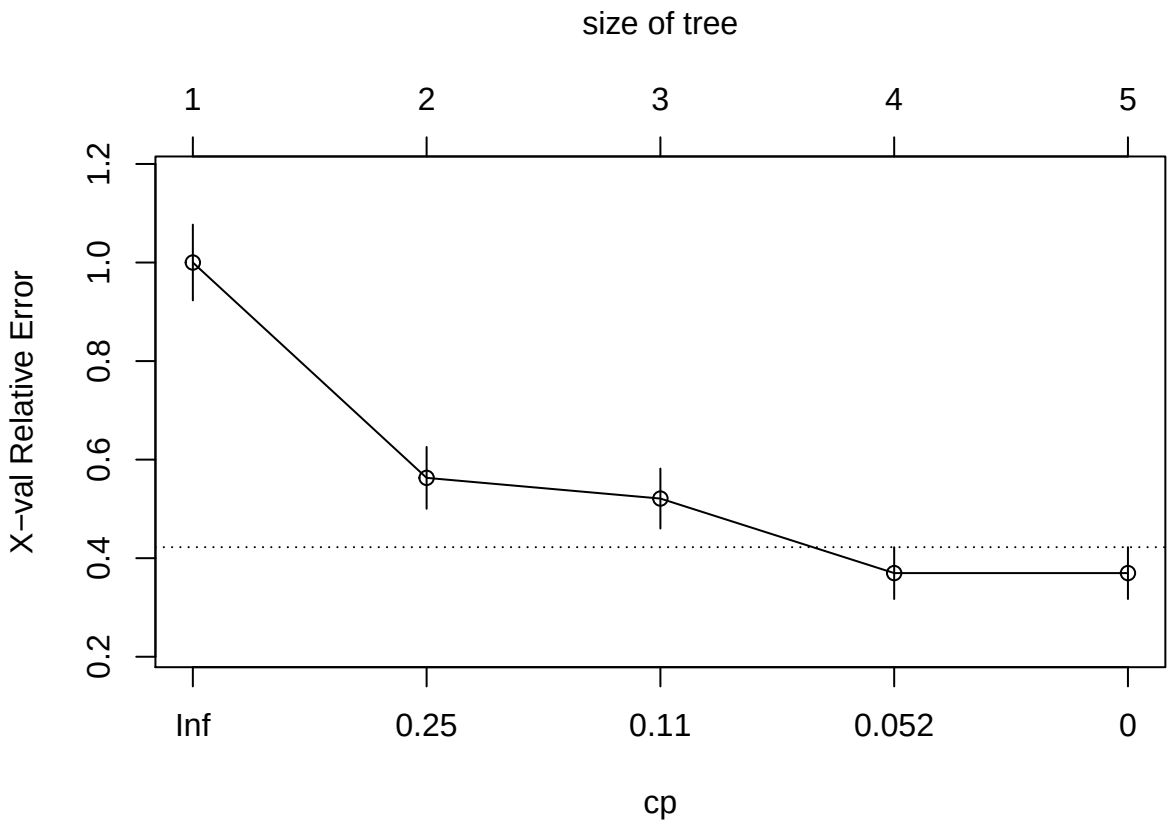
a. Seleccionar el parámetro de complejidad de forma automática, siguiendo el criterio de un error estándar de Breiman et al. (1984).

```
rpart.rules(tree, style = "tall")
```

```
## Tipo is 0.05 when
##     Outstate is 8 to 11
##     Enroll >= 7.3
##
## Tipo is 0.11 when
##     Outstate < 8
##     Accept >= 6.7
##
## Tipo is 0.71 when
##     Outstate < 8
##     Accept < 6.7
##
## Tipo is 0.71 when
##     Outstate >= 11
##     Enroll >= 7.3
##
```

```
## Tipo is 0.98 when
##     Outstate >= 8
##     Enroll < 7.3
```

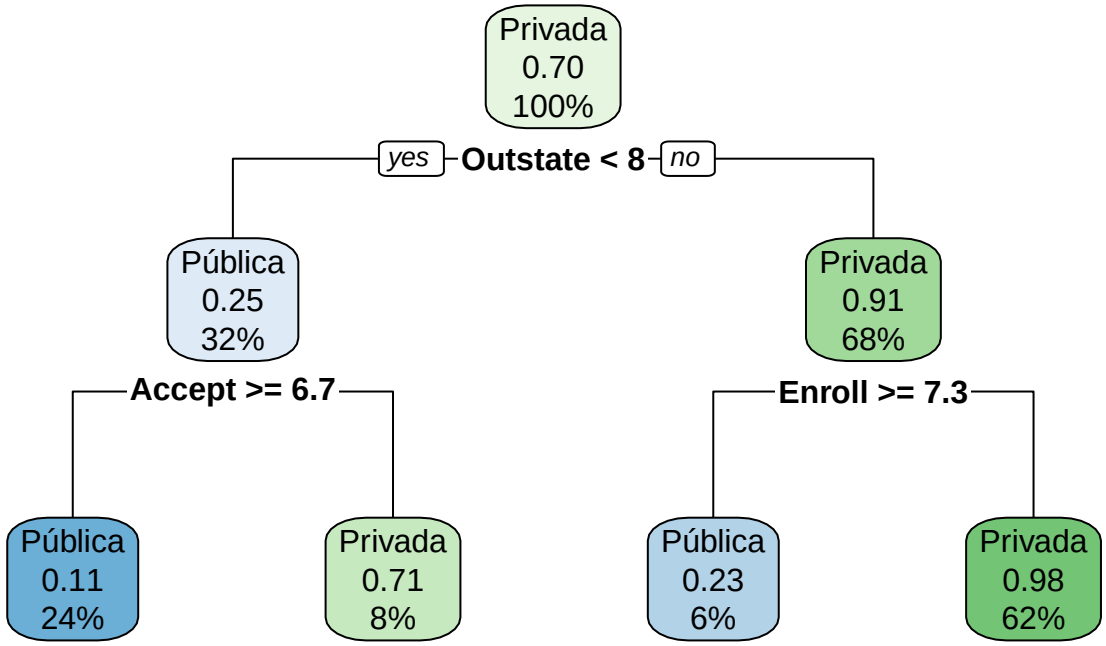
```
tree <- rpart(Tipo ~ ., data = train, cp = 0)
plotcp(tree)
```




```
xerror <- tree$cptable[,"xerror"]
imin.xerror <- which.min(xerror)
upper.xerror <- xerror[imin.xerror] + tree$cptable[imin.xerror, "xstd"]
icp <- min(which(xerror <= upper.xerror))
cp <- tree$cptable[icp, "CP"]
tree <- prune(tree, cp = cp)
```

b. Representar e interpretar el árbol resultante.

```
rpart.plot(tree, main = "Árbol de clasificación privada-pública")
```



Interpretación:

c. Evaluar la precisión, de las predicciones y de las estimaciones de la probabilidad, en la muestra de test.

```
#Predicciones
obs <- test$Tipo
head(predict(tree, newdata = test))
```

```
##                                Pública   Privada
## University of San Francisco 0.02016129 0.9798387
## Clarkson University         0.02016129 0.9798387
## Marymount University        0.02016129 0.9798387
## West Virginia Wesleyan College 0.02016129 0.9798387
## Salem-Teikyo University     0.02016129 0.9798387
## Loyola Marymount University 0.02016129 0.9798387
```

```
pred <- predict(tree, newdata = test, type = "class")
table(obs, pred)
```

```
##          pred
## obs       Pública Privada
##   Pública      17       7
##   Privada       6      70
```

```
confusionMatrix(pred,obs)
```

```
## Confusion Matrix and Statistics
```

```
##
##           Reference
## Prediction Pública Privada
##    Pública      17       6
##    Privada       7      70
##
##                Accuracy : 0.87
##                  95% CI : (0.788, 0.9289)
##     No Information Rate : 0.76
##     P-Value [Acc > NIR] : 0.004749
##
##                   Kappa : 0.6385
##
##  Mcnemar's Test P-Value : 1.000000
##
##             Sensitivity : 0.7083
##             Specificity : 0.9211
##          Pos Pred Value : 0.7391
##          Neg Pred Value : 0.9091
##              Prevalence : 0.2400
##          Detection Rate : 0.1700
##    Detection Prevalence : 0.2300
##       Balanced Accuracy : 0.8147
##
##        'Positive' Class : Pública
##
```

2. Realizar la clasificación anterior empleando Bosques Aleatorios mediante

el método `"rf"` del paquete `caret`.

a. Considerar 300 árboles y seleccionar el número de predictores empleados en cada división `mtry = c(1, 2, 4, 6)` mediante validación cruzada, con 10 grupos y empleando el criterio de un error estándar de Breiman.

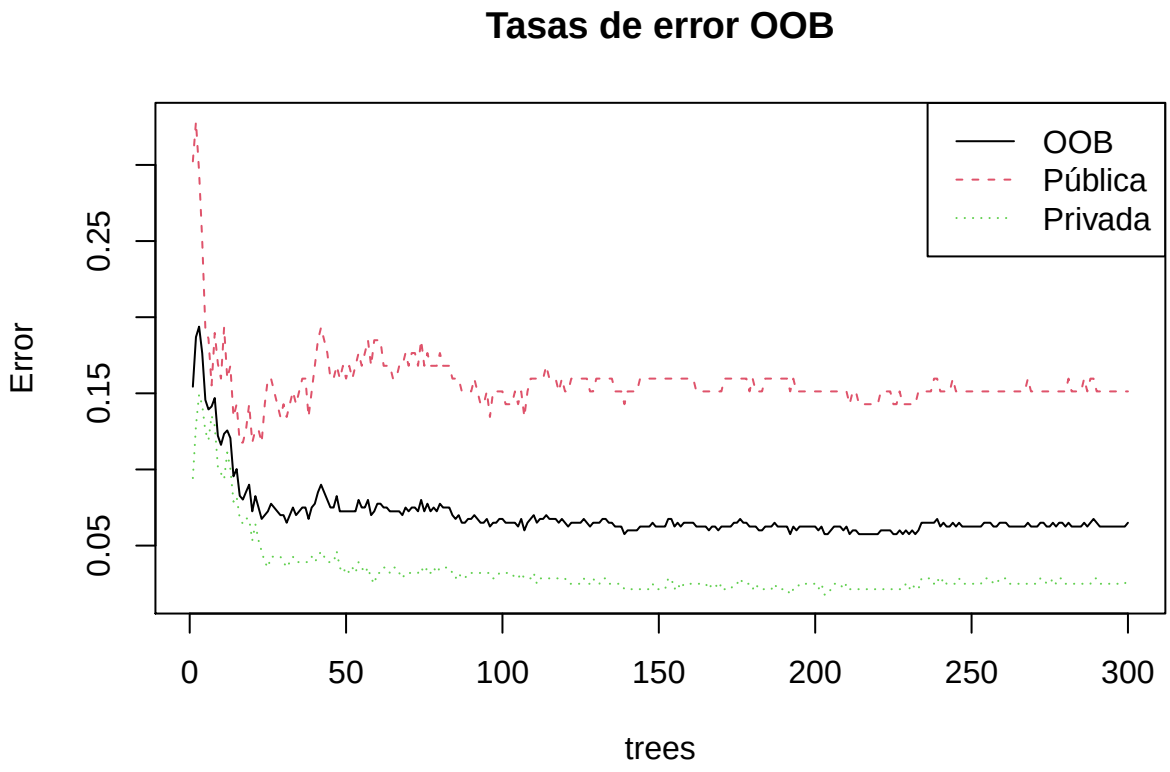
```r
tuneGrid <- data.frame(mtry = c(1, 2, 4, 6))

rf.caret <-
    train(
        Tipo ~ .,
        data = train,
        method = "rf",
        ntree = 300,
        tuneGrid = tuneGrid,
        trControl = trainControl(
            method = "cv",
            number = 10,
            selectionFunction = "oneSE"
        )
    )

final <- rf.caret$finalModel
```

b. Representar la convergencia del error en las muestras OOB en el modelo final.

```
plot(final, main = "Tasas de error OOB")
legend("topright",
       colnames(final$err.rate),
       lty = 1:5,
       col = 1:6)
```



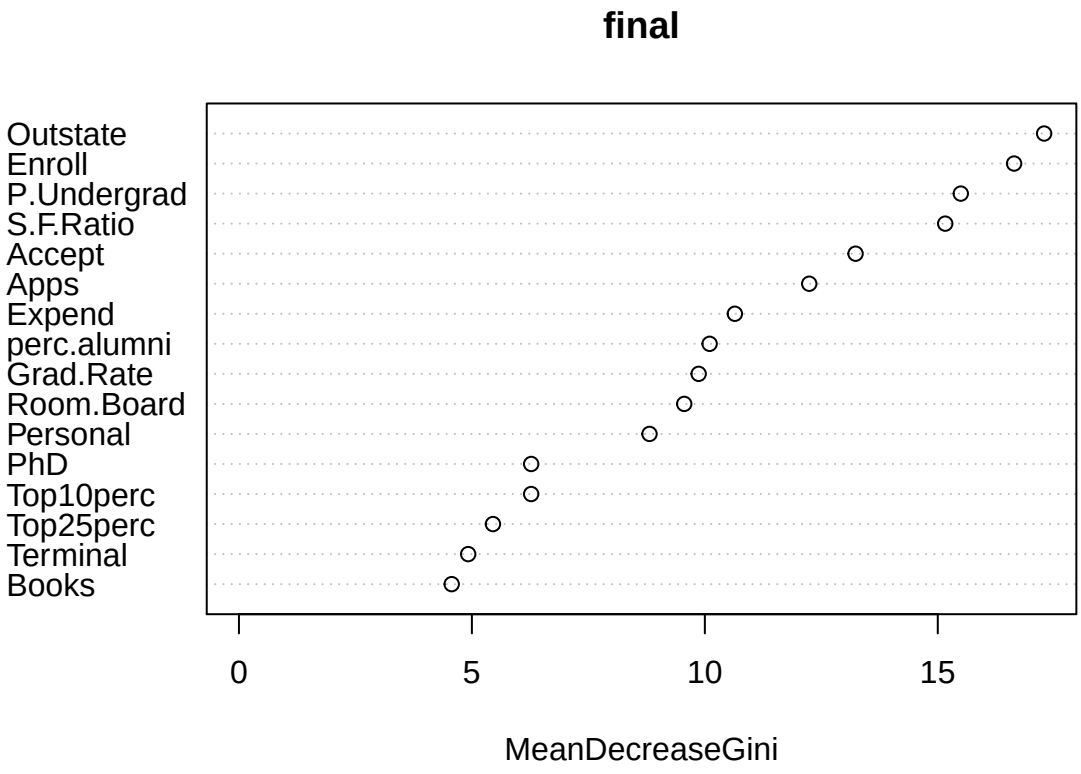
c. Estudiar la importancia de las variables y el efecto de las principales empleando algún método gráfico (para la interpretación del modelo).

```
importance(final)
```

```
##             MeanDecreaseGini
## Apps                12.241090
## Accept              13.235416
## Enroll              16.638087
## Top10perc            6.272712
## Top25perc            5.452311
## P.Undergrad         15.493902
## Outstate            17.283349
## Room.Board           9.557924
## Books                4.566409
## Personal             8.812833
## PhD                  6.273321
## Terminal             4.920698
```

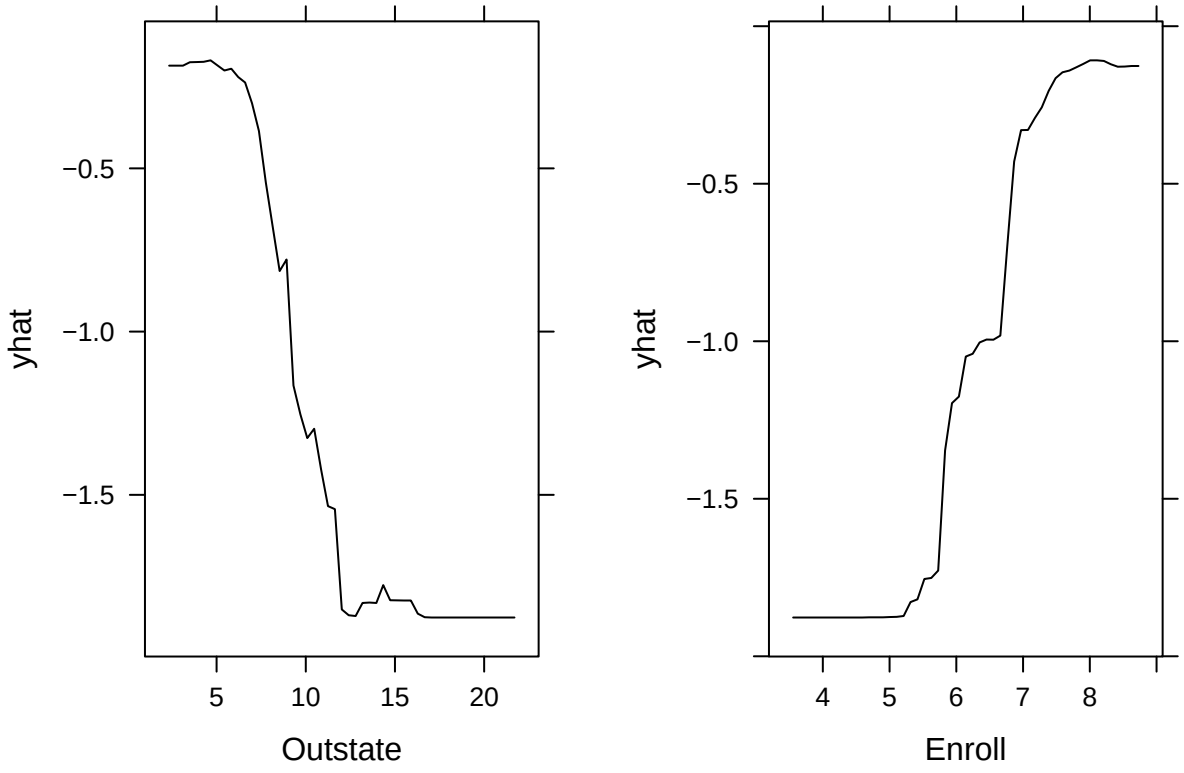
```
## S.F.Ratio          15.160058
## perc.alumni        10.102554
## Expend             10.644972
## Grad.Rate           9.867373
```

```
varImpPlot(final)
```




```
pdp1 <- partial(final, "Outstate", train = train)
p1 <- plotPartial(pdp1)

pdp2 <- partial(final, "Enroll", train = train)
p2 <- plotPartial(pdp2)
grid.arrange(p1, p2, ncol = 2)
```



d. Evaluar la precisión de las predicciones en la muestra de test y comparar los resultados con los obtenidos con el modelo del ejercicio anterior.

```
obs <- test$Tipo
head(predict(final, newdata = test))
```

```
##    University of San Francisco            Clarkson University
##                        Privada                        Privada
##           Marymount University West Virginia Wesleyan College
##                        Privada                        Privada
##        Salem-Teikyo University    Loyola Marymount University
##                        Privada                        Privada
## Levels: Pública Privada
```

```
pred <- predict(final, newdata = test, type = "class")
table(obs, pred)
```

```
##          pred
## obs       Pública Privada
##   Pública      18       6
##   Privada       4      72
```

```
confusionMatrix(pred, obs)
```

```
## Confusion Matrix and Statistics
##
##           Reference
```

```
## Prediction Pública Privada
##    Pública      18       4
##    Privada       6      72
##
##                Accuracy : 0.9
##                  95% CI : (0.8238, 0.951)
##     No Information Rate : 0.76
##     P-Value [Acc > NIR] : 0.0003075
##
##                   Kappa : 0.7178
##
##  Mcnemar's Test P-Value : 0.7518296
##
##             Sensitivity : 0.7500
##             Specificity : 0.9474
##          Pos Pred Value : 0.8182
##          Neg Pred Value : 0.9231
##              Prevalence : 0.2400
##          Detection Rate : 0.1800
##    Detection Prevalence : 0.2200
##       Balanced Accuracy : 0.8487
##
##        'Positive' Class : Pública
##
```

3. Realizar la clasificación anterior empleando SVM mediante la función `ksvm()` del paquete `kernlab`,

a. Ajustar el modelo con las opciones por defecto.

```
set.seed(40)
svm <- ksvm(Tipo ~ ., data = train)
svm
```

```
## Support Vector Machine object of class "ksvm"
##
## SV type: C-svc  (classification)
##  parameter : cost C = 1
##
## Gaussian Radial Basis kernel function.
##  Hyperparameter : sigma =  0.0543868376683745
##
## Number of Support Vectors : 114
##
## Objective Function Value : -66.589
## Training error : 0.0425
```

```
pred <- predict(svm, newdata = test)
confusionMatrix(pred, test$Tipo)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction Pública Privada
##    Pública      19       3
```

```
##    Privada       5      73
##
##                Accuracy : 0.92
##                  95% CI : (0.8484, 0.9648)
##     No Information Rate : 0.76
##     P-Value [Acc > NIR] : 3.001e-05
##
##                   Kappa : 0.7743
##
##  Mcnemar's Test P-Value : 0.7237
##
##             Sensitivity : 0.7917
##             Specificity : 0.9605
##          Pos Pred Value : 0.8636
##          Neg Pred Value : 0.9359
##              Prevalence : 0.2400
##          Detection Rate : 0.1900
##    Detection Prevalence : 0.2200
##       Balanced Accuracy : 0.8761
##
##        'Positive' Class : Pública
##
```

b. Ajustar el modelo empleando validación cruzada con 10 grupos para seleccionar los valores "óptimos" de los hiperparámetros, considerando las posibles combinaciones de `sigma = c(0.01, 0.05, 0.1)` y `C = c(0.5, 1, 10)` (sin emplear el paquete `caret`; ver Ejercicio 3.1 en *03-bagging_boosting-ejercicios.html*).

```
tune.grid <- expand.grid(
    sigma = c(0.01, 0.05, 0.1),
    C = c(0.5, 1, 10),
    error = NA
)

best.err <- Inf
set.seed(40)
for (i in 1:nrow(tune.grid)) {
    fit <-
        ksvm(
            Tipo ~ .,
            data = train[, ],
            cross = 10,
            C = tune.grid$C[i],
            kpar = list(tune.grid$sigma[i])
        )
    fit.error <- fit@cross
    tune.grid$error[i] <- fit.error
    if (fit.error < best.err) {
        final.model <- fit
        best.err <- fit.error
        best.tune <- tune.grid[i,]
    }
}
```

```
final.model
```

```
## Support Vector Machine object of class "ksvm"
##
## SV type: C-svc  (classification)
##  parameter : cost C = 0.5
##
## Gaussian Radial Basis kernel function.
##  Hyperparameter : sigma =  0.01
##
## Number of Support Vectors : 145
##
## Objective Function Value : -53.98
## Training error : 0.055
## Cross validation error : 0.0525
```

```
pred2 <- predict(final.model, newdata = test)
confusionMatrix(pred2, test$Tipo)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction Pública Privada
##    Pública      19       3
##    Privada       5      73
##
##                Accuracy : 0.92
##                  95% CI : (0.8484, 0.9648)
##     No Information Rate : 0.76
##     P-Value [Acc > NIR] : 3.001e-05
##
##                   Kappa : 0.7743
##
##  Mcnemar's Test P-Value : 0.7237
##
##             Sensitivity : 0.7917
##             Specificity : 0.9605
##          Pos Pred Value : 0.8636
##          Neg Pred Value : 0.9359
##              Prevalence : 0.2400
##          Detection Rate : 0.1900
##    Detection Prevalence : 0.2200
##       Balanced Accuracy : 0.8761
##
##        'Positive' Class : Pública
##
```

c. Evaluar la precisión de las predicciones de ambos modelos en la muestra de test y comparar también los resultados con los obtenidos en el ejercicio anterior.