Actividad_Integradora2

Rodolfo Jesús Cruz Rebollar

2024-11-19

1. Preparación de la base de datos de Titanic

```
# Importar base de datos de entrenamiento y prueba

titanic_train = read.csv("Titanic.csv")

titanic_test = read.csv("Titanic_test.csv")

# Analizar si hay datos faltantes en las bases de datos

# datos de entrenamiento (margin 2 indica que se realice conteo de datos
faltantes por
#columna)

apply(X = is.na(titanic_train), MARGIN = 2, FUN = sum)
```

```
## PassengerId    Survived      Pclass        Name         Sex
Age
##           0           0           0           0           0
263
##       SibSp       Parch      Ticket        Fare       Cabin
Embarked
##           0           0           0           1           0
2
```

```
# contar datos faltantes por columna en la base de datos de prueba
# margin = 2 para contar por columnas

apply(X = is.na(titanic_test), MARGIN = 2, FUN = sum)
```

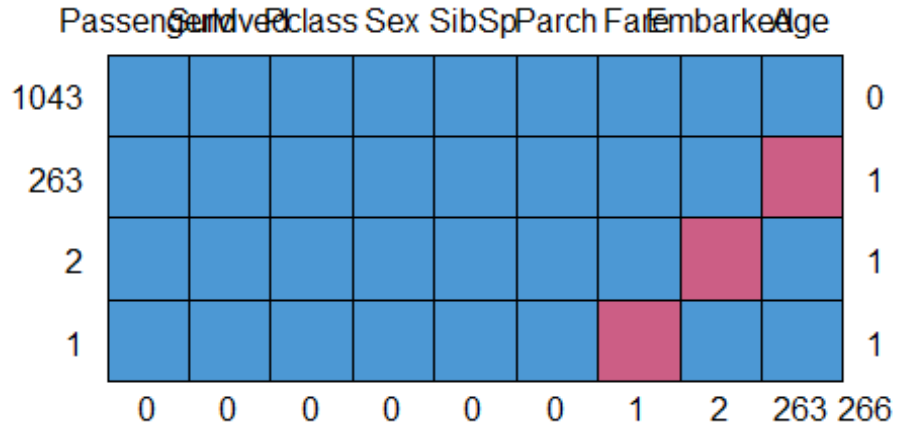
```
## PassengerId      Pclass        Name         Sex         Age
SibSp
##           0           0           0           0          86
0
##       Parch      Ticket        Fare       Cabin    Embarked
##           0           0           1           0           0
```

```
# Eliminar variables que no sean relevantes para el modelo

titanic_train = titanic_train[, c(-4,-9,-11)]

#Transformar variables categóricas a tipo factor
```

```
for(var in c('Survived','Pclass','Embarked','Sex'))

  titanic_train[, var] = as.factor(titanic_train[, var])
```

En la columna de edad se observa que existen 86 datos faltantes, mientras que en la columna Fare existe solamente 1 dato faltante, por lo que a continuación se procede a encontrar el patrón que siguen dichos datos faltantes:

```
# Patrón que siguen los datos faltantes

md.pattern(titanic_train)
```




```
##      PassengerId Survived Pclass Sex SibSp Parch Fare Embarked Age
## 1043           1        1      1   1     1     1    1        1   1   0
## 263            1        1      1   1     1     1    1        1   0   1
## 2              1        1      1   1     1     1    1        0   1   1
## 1              1        1      1   1     1     1    0        1   1   1
##                0        0      0   0     0     0    1        2 263 266

# Medidas estadísticas con datos faltantes

summary(titanic_train[, -1])

##  Survived Pclass      Sex           Age            SibSp
Parch
##  0:815    1:323   female:466   Min.   : 0.17   Min.   :0.0000   Min.
```

```
:0.000
##  1:494    2:277   male  :843   1st Qu.:21.00   1st Qu.:0.0000   1st
Qu.:0.000
##           3:709                Median :28.00   Median :0.0000   Median
:0.000
##                                Mean   :29.88   Mean   :0.4989   Mean
:0.385
##                                3rd Qu.:39.00   3rd Qu.:1.0000   3rd
Qu.:0.000
##                                Max.   :80.00   Max.   :8.0000   Max.
:9.000
##                                NA's   :263
##       Fare          Embarked
##  Min.   :  0.000   C   :270
##  1st Qu.:  7.896   Q   :123
##  Median : 14.454   S   :914
##  Mean   : 33.295   NA's:  2
##  3rd Qu.: 31.275
##  Max.   :512.329
##  NA's   :1
```

```
# Medidas sin datos faltantes

summary(na.omit(titanic_train)[, -1])
```

```
##  Survived Pclass      Sex           Age            SibSp
##  0:628    1:282   female:386   Min.   : 0.17   Min.   :0.0000
##  1:415    2:261   male  :657   1st Qu.:21.00   1st Qu.:0.0000
##           3:500                Median :28.00   Median :0.0000
##                                Mean   :29.81   Mean   :0.5043
##                                3rd Qu.:39.00   3rd Qu.:1.0000
##                                Max.   :80.00   Max.   :8.0000
##      Parch             Fare        Embarked
##  Min.   :0.0000   Min.   :  0.00   C:212
##  1st Qu.:0.0000   1st Qu.:  8.05   Q: 50
##  Median :0.0000   Median : 15.75   S:781
##  Mean   :0.4219   Mean   : 36.60
##  3rd Qu.:1.0000   3rd Qu.: 35.08
##  Max.   :6.0000   Max.   :512.33
```

Análisis de influencia de datos faltantes por cada variable

Sobrevivientes

```
t2c = 100*prop.table(table(titanic_train[, 2]))

t2s = 100*prop.table(table(na.omit(titanic_train)[,2]))

t2p = c(t2s[1]/t2c[1],t2s[2]/t2c[2])
```

```r
t2 = data.frame(as.numeric(t2c),as.numeric(t2s),as.numeric(t2p))

row.names(t2) = c("Murió","Sobrevivió")

names(t2) = c("Con NA (%)","Sin NA (%)","Pérdida (prop)")

round(t2,2)
```

```
##            Con NA (%) Sin NA (%) Pérdida (prop)
## Murió           62.26      60.21           0.97
## Sobrevivió      37.74      39.79           1.05
```

Clase en que viajó

```r
t3c = 100*prop.table(table(titanic_train[,3]))

t3s = 100*prop.table(table(na.omit(titanic_train)[,3]))

t3p = c(t3s[1]/t3c[1],t3s[2]/t3c[2],t3s[3]/t3c[3])

t3 = data.frame(as.numeric(t3c),as.numeric(t3s),as.numeric(t3p))

row.names(t3) = c("Primera","Segunda","Tercera")

names(t3) = c("Con NA (%)","Sin NA (%)","Pérdida (prop)")

round(t3,2)
```

```
##         Con NA (%) Sin NA (%) Pérdida (prop)
## Primera      24.68      27.04           1.10
## Segunda      21.16      25.02           1.18
## Tercera      54.16      47.94           0.89
```

Sexo

```r
t4c = 100*prop.table(table(titanic_train[,4]))
t4s = 100*prop.table(table(na.omit(titanic_train)[,4]))
t4p = c(t4s[1]/t4c[1],t4s[2]/t4c[2])
t4 = data.frame(as.numeric(t4c),as.numeric(t4s),as.numeric(t4p))
row.names(t4) = c("Mujer","Hombre")
names(t4) = c("Con NA (%)","Sin NA (%)","Pérdida (prop)")
round(t4,2)
```

```
##        Con NA (%) Sin NA (%) Pérdida (prop)
## Mujer        35.6      37.01           1.04
## Hombre       64.4      62.99           0.98
```

Puerto de embarcación

```r
t9c = 100*prop.table(table(titanic_train[,9]))
t9s = 100*prop.table(table(na.omit(titanic_train)[,9]))
```

```r
t9p = c(t9s[1]/t9c[1],t9s[2]/t9c[2],t9s[3]/t9c[3])
t9 = data.frame(as.numeric(t9c),as.numeric(t9s),as.numeric(t9p))
row.names(t9) = c("Cherbourg","Queenstown","Southampton")
names(t9) = c("Con NA (%)","Sin NA (%)","Pérdida (prop)")
round(t9,2)
```

```
##             Con NA (%) Sin NA (%) Pérdida (prop)
## Cherbourg        20.66      20.33           0.98
## Queenstown        9.41       4.79           0.51
## Southampton      69.93      74.88           1.07
```

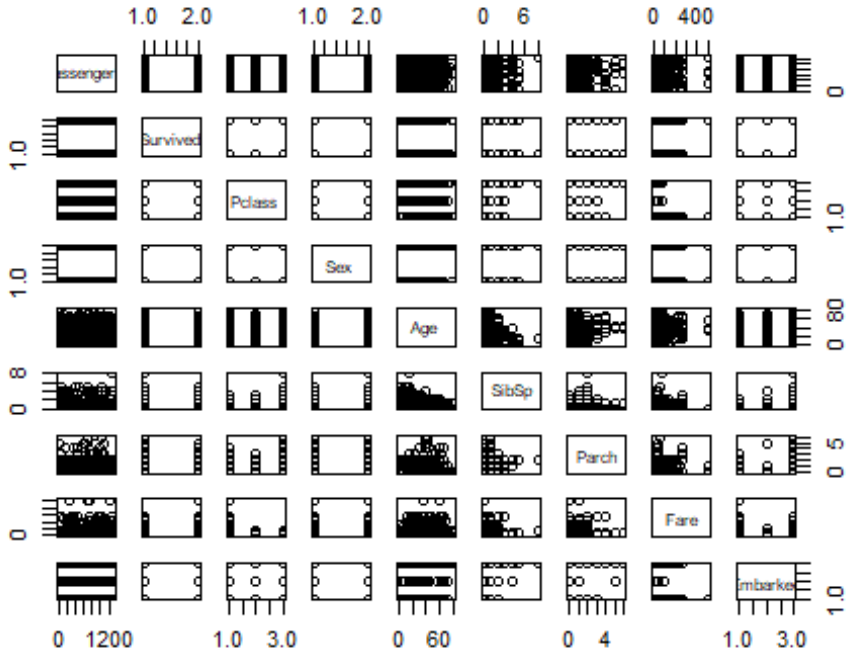
Anteriormente se aprecia que la variables más afectada al eliminar datos faltantes será la edad (Age), principalmente debido a que dicha columna en la base de datos de entrenamiento, cuenta con un total de 263 datos faltantes, mientras que en la base de datos de prueba, esa misma columna posee un total de 86 valores faltantes o desconocidos, mientras que el resto de variables tanto en prueba como en entrenamiento no poseen datos faltantes o son muy pocos, por lo que al eliminar 263 datos de edades desconocidas, los datos se sesgarán, lo que podría disminuir la precisión del modelo logístico al momento de entrenarlo.

```r
# Remover los datos faltantes para tener únicamente registros que sí
tengan datos conocidos

titanic_train = na.omit(titanic_train)

titanic_test = na.omit(titanic_test)

# Obtener medidas estadísticas de la base de datos para entrenar el
modelo logístico

summary(titanic_train)
```

```
##   PassengerId     Survived Pclass      Sex           Age
##  Min.   :   1.0   0:628    1:282   female:386   Min.   : 0.17
##  1st Qu.: 326.5   1:415    2:261   male  :657   1st Qu.:21.00
##  Median : 662.0            3:500                Median :28.00
##  Mean   : 655.4                                 Mean   :29.81
##  3rd Qu.: 973.5                                 3rd Qu.:39.00
##  Max.   :1307.0                                 Max.   :80.00
##      SibSp           Parch             Fare        Embarked
##  Min.   :0.0000   Min.   :0.0000   Min.   :  0.00   C:212
##  1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:  8.05   Q: 50
##  Median :0.0000   Median :0.0000   Median : 15.75   S:781
##  Mean   :0.5043   Mean   :0.4219   Mean   : 36.60
##  3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.: 35.08
##  Max.   :8.0000   Max.   :6.0000   Max.   :512.33
```

```r
# Realizar gráficos de dispersión de las variables que conforman a los
datos de
#entrenamiento
```

```r
plot(titanic_train)
```



Partición de los datos en entrenamiento y prueba

```r
data_indice = createDataPartition(titanic_train$Survived, p = 0.7, list = FALSE, times = 1)

data_train = titanic_train[ data_indice,]  %>% as_tibble()
data_valid = titanic_train[-data_indice,] %>% as_tibble()

data_train$Pclass = as.integer(data_train$Pclass)
data_valid$Pclass = as.integer(data_valid$Pclass)
```

2. Entrenamiento del modelo logístico

```r
# Modelo completo con todas las variables predictoras

model1 = glm(Survived ~., data = data_train, family = "binomial")

# Utilizar criterio AIC para definir cuál es el mejor modelo

step(model1, direction="both", trace = 1)
```

```
## Start:  AIC=585.95
## Survived ~ PassengerId + Pclass + Sex + Age + SibSp + Parch +
##     Fare + Embarked
```

```
##
##               Df Deviance    AIC
## - Embarked     2   566.79 582.79
## - Parch        1   566.04 584.04
## - PassengerId  1   566.04 584.04
## - Fare         1   566.13 584.13
## <none>             565.95 585.95
## - SibSp        1   571.38 589.38
## - Age          1   586.82 604.82
## - Pclass       1   601.12 619.12
## - Sex          1   880.54 898.54
##
## Step:  AIC=582.79
## Survived ~ PassengerId + Pclass + Sex + Age + SibSp + Parch +
##     Fare
##
##               Df Deviance    AIC
## - PassengerId  1   566.86 580.86
## - Parch        1   566.97 580.97
## - Fare         1   567.17 581.17
## <none>             566.79 582.79
## + Embarked     2   565.95 585.95
## - SibSp        1   572.61 586.61
## - Age          1   587.94 601.94
## - Pclass       1   603.11 617.11
## - Sex          1   888.35 902.35
##
## Step:  AIC=580.86
## Survived ~ Pclass + Sex + Age + SibSp + Parch + Fare
##
##               Df Deviance    AIC
## - Parch        1   567.02 579.02
## - Fare         1   567.23 579.23
## <none>             566.86 580.86
## + PassengerId  1   566.79 582.79
## + Embarked     2   566.04 584.04
## - SibSp        1   572.62 584.62
## - Age          1   587.98 599.98
## - Pclass       1   603.13 615.13
## - Sex          1   888.36 900.36
##
## Step:  AIC=579.02
## Survived ~ Pclass + Sex + Age + SibSp + Fare
##
##               Df Deviance    AIC
## - Fare         1   567.31 577.31
## <none>             567.02 579.02
## + Parch        1   566.86 580.86
## + PassengerId  1   566.97 580.97
## + Embarked     2   566.12 582.12
```

```
## - SibSp        1   573.81 583.81
## - Age          1   588.10 598.10
## - Pclass       1   604.79 614.79
## - Sex          1   897.32 907.32
##
## Step:  AIC=577.31
## Survived ~ Pclass + Sex + Age + SibSp
##
##               Df Deviance    AIC
## <none>             567.31 577.31
## + Fare         1   567.02 579.02
## + Parch        1   567.23 579.23
## + PassengerId  1   567.25 579.25
## + Embarked     2   566.26 580.26
## - SibSp        1   573.82 581.82
## - Age          1   588.86 596.86
## - Pclass       1   627.37 635.37
## - Sex          1   902.78 910.78

##
## Call:  glm(formula = Survived ~ Pclass + Sex + Age + SibSp, family =
"binomial",
##     data = data_train)
##
## Coefficients:
## (Intercept)       Pclass      Sexmale          Age        SibSp
##     5.35280     -1.10477     -3.54775     -0.03953     -0.31361
##
## Degrees of Freedom: 730 Total (i.e. Null);  726 Residual
## Null Deviance:       982.8
## Residual Deviance: 567.3     AIC: 577.3
```

De acuerdo con el criterio AIC, el mejor modelo es aquel que tiene como variables predictoras Pclass, Sex, Age y SibSp, y que al mismo tiempo, posee un valor de AIC igual a 579.53, siendo el AIC más bajo de todos los modelos generados antes del último modelo (el que tiene el menor AIC), no obstante se realizará un segundo modelo que incluirá a la última variable eliminada por el método `step`, la cual es `Parch`:

```
# Segundo modelo que incluye la última variable eliminada por step
(Parch)

model2 = glm(Survived ~ Pclass + Sex + Age + SibSp + Parch, data =
data_train,
             family = "binomial")

# Modelo 3 para formular un modelo con la sugerencia del criterio AIC

model3 = glm(Survived ~ Pclass + Sex + Age + SibSp, data = data_train,
family = "binomial")
```

```r
# Resumen del modelo con la variable parch

summary(model2)
```

```
##
## Call:
## glm(formula = Survived ~ Pclass + Sex + Age + SibSp + Parch,
##     family = "binomial", data = data_train)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  5.373854   0.586780   9.158  < 2e-16 ***
## Pclass      -1.104178   0.151705  -7.278 3.38e-13 ***
## Sexmale     -3.560371   0.241475 -14.744  < 2e-16 ***
## Age         -0.039637   0.008776  -4.517 6.28e-06 ***
## SibSp       -0.302679   0.131109  -2.309    0.021 *
## Parch       -0.035824   0.126485  -0.283    0.777
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 982.80  on 730  degrees of freedom
## Residual deviance: 567.23  on 725  degrees of freedom
## AIC: 579.23
##
## Number of Fisher Scoring iterations: 5
```

```r
# Resumen del modelo sin la variable Parch

summary(model3)
```

```
##
## Call:
## glm(formula = Survived ~ Pclass + Sex + Age + SibSp, family =
"binomial",
##     data = data_train)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  5.352795   0.581675   9.202  < 2e-16 ***
## Pclass      -1.104773   0.151716  -7.282 3.29e-13 ***
## Sexmale     -3.547751   0.237000 -14.969  < 2e-16 ***
## Age         -0.039533   0.008761  -4.512 6.41e-06 ***
## SibSp       -0.313609   0.125358  -2.502   0.0124 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
```

```
##     Null deviance: 982.80  on 730  degrees of freedom
## Residual deviance: 567.31  on 726  degrees of freedom
## AIC: 577.31
##
## Number of Fisher Scoring iterations: 5
```

Los 2 modelos anteriores son los que se consideran como los mejores modelos para predecir la sobrevivencia en el Titanic, debido a que poseen valores de AIC de 579.72 y 579.53, respectivamente, los cuales son bastante cercanos entre sí, por lo que eso indica que ambos modelos poseen grados de precisión similares en cuanto a sus predicciones, mientras que el resto de posibles modelos generados poseen AIC más elevados, por lo que son menos adecuados para predecir la supervivencia que los 2 anteriormente descritos.

3. Análisis de modelos

Desviación residual de modelos

```
cat("Desviación residual del modelo 1 (con la variable Parch): ",
model2$deviance, "\n")

## Desviación residual del modelo 1 (con la variable Parch):  567.2251

cat("Desviación residual del modelo 2 (sin Parch): ", model3$deviance)

## Desviación residual del modelo 2 (sin Parch):  567.3053
```

Desviación nula de los modelos

```
cat("Desviación nula del modelo 1 (con la variable Parch): ",
model2$null.deviance, "\n")

## Desviación nula del modelo 1 (con la variable Parch):  982.7966

cat("Desviación nula del modelo 2 (sin Parch): ", model3$null.deviance)

## Desviación nula del modelo 2 (sin Parch):  982.7966
```

Desviación explicada

```
# Desviación explicada por el primer modelo (con Parch)

cat("Desviación explicada modelo 1: ", 1-
model2$deviance/model2$null.deviance, "\n")

## Desviación explicada modelo 1:  0.4228459

# Desviación explicada por el segundo modelo (sin Parch)

cat("Desviación explicada modelo 2: ", 1 - model3$deviance /
model3$null.deviance)

## Desviación explicada modelo 2:  0.4227643
```

Prueba de razón de verosimilitud

H_0: El modelo con predictores explica mejor la variable respuesta: $log\left(\frac{p}{1-p}\right)$ que el modelo nulo.

H_1: El modelo nulo explica mejor la variable respuesta: $log\left(\frac{p}{1-p}\right)$ (la probabilidad es constante).

```
dif_devianceM1 = model2$null.deviance - model2$deviance

gl_M1 = model2$df.null - model2$df.deviance

pchisq(dif_devianceM1, gl_M1, lower.tail = FALSE)

## numeric(0)

dif_devianceM2 = model3$null.deviance - model3$deviance

gl_M2 = model3$df.null - model3$df.deviance

pchisq(dif_devianceM2, gl_M2, lower.tail = FALSE)

## numeric(0)
```

Se observa que para ambos modelos, el p valor es igual a 0, lo cual al ser menor que 0.05, provoca que se rechace la hipótesis nula H_0, lo cual a su vez implica que el modelo nulo (sin predictores) explica de una mejor manera la variable respuesta, que en este caso es el logaritmo de los nomios, lo que significa que la probabilidad de supervivencia en realidad se mantiene constante, sin embargo, nos quedamos con el modelo 1 que tiene una desviación explicada de 0.4243, lo cual es sutilmente mayor a la del modelo 2, misma que es de 0.4225, por lo cual, en resumen, se elige el modelo 1 (que incluye a la variable Parch).

Ecuación del mejor modelo elegido:

$$log\left(\frac{p}{1-p}\right) = 4.53 - 1.28 * Pclass2 - 2.28 * Pclass3 - 3.599 * Sexmale - 0.0397 * Age - 0.3457 * SibSp - 0.1642 * Parch$$

En la ecuación anterior, se aprecia que por cada unidad que incrementan las variables Pclass2, Pclass3, Sexmale, Age, SibSp y Parch, el logaritmo de los nomios (cociente de la probabilidad de supervivencia y la probabilidad de no supervivencia), experimenta un cambio de -1.28, -2.28, -3.599, -0.0397, -0.3457 y -0.1642 unidades, respectivamente.

4. Análisis de las predicciones para datos de entrenamiento

Matriz de confusión

```
library(vcd)
```

```
## Loading required package: grid
```

```
##
## Attaching package: 'vcd'
```

```
## The following object is masked from 'package:ISLR':
##
##     Hitters
```

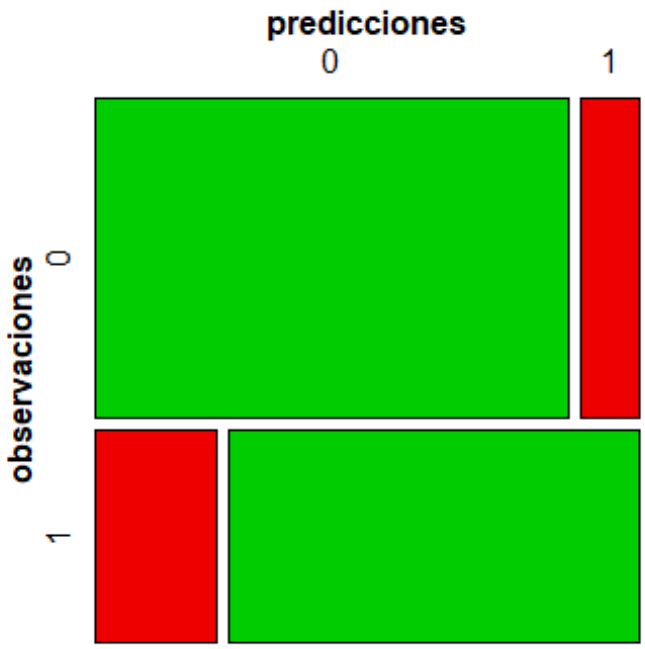
```
pred = ifelse(test = model2$fitted.values > 0.5, yes = 1, no = 0)
```

```
conf_mat = table(data_train$Survived, pred, dnn = c("observaciones",
"predicciones"))
```

```
conf_mat
```

```
##              predicciones
## observaciones   0   1
##             0 392  48
##             1  66 225
```

```
mosaic(conf_mat, shade = T, colorize = T,
       gp = gpar(fill = matrix(c("green3", "red2", "red2", "green3"), 2,
2)))
```




```
acc = (conf_mat[1,1] + conf_mat[2,2])/sum(conf_mat)
cat("La Exactitud (accuracy) del modelo es", acc,"\n")

## La Exactitud (accuracy) del modelo es 0.8440492

sens = conf_mat[1,1]/sum(conf_mat[1,])
cat("La Sensibilidad del modelo es", sens,"\n")

## La Sensibilidad del modelo es 0.8909091

specif = conf_mat[2,2]/sum(conf_mat[2,])
cat("La Especificidad del modelo es", specif,"\n")

## La Especificidad del modelo es 0.7731959

precis = conf_mat[1,1]/sum(conf_mat[,1])
cat("La Precisión del modelo es", precis,"\n")

## La Precisión del modelo es 0.8558952
```

Curva ROC

```
pred_training = predict(model2, data = M_train, type = 'response')

library(pROC)

## Type 'citation("pROC")' for a citation.

##
## Attaching package: 'pROC'
```

```
## The following object is masked from 'package:Metrics':
##
##     auc

## The following objects are masked from 'package:stats':
##
##     cov, smooth, var

curva_ROC = roc(response = data_train$Survived, predictor =
pred_training)

## Setting levels: control = 0, case = 1

## Setting direction: controls < cases

curva_ROC

##
## Call:
## roc.default(response = data_train$Survived, predictor = pred_training)
##
## Data: pred_training in 440 controls (data_train$Survived 0) < 291
cases (data_train$Survived 1).
## Area under the curve: 0.8926

ggroc(curva_ROC, color = "blue", size = 2) +
  geom_abline(slope = 1, intercept = 1, linetype ='dashed') +
  labs(title = "Curva ROC") +
  theme_bw()
```

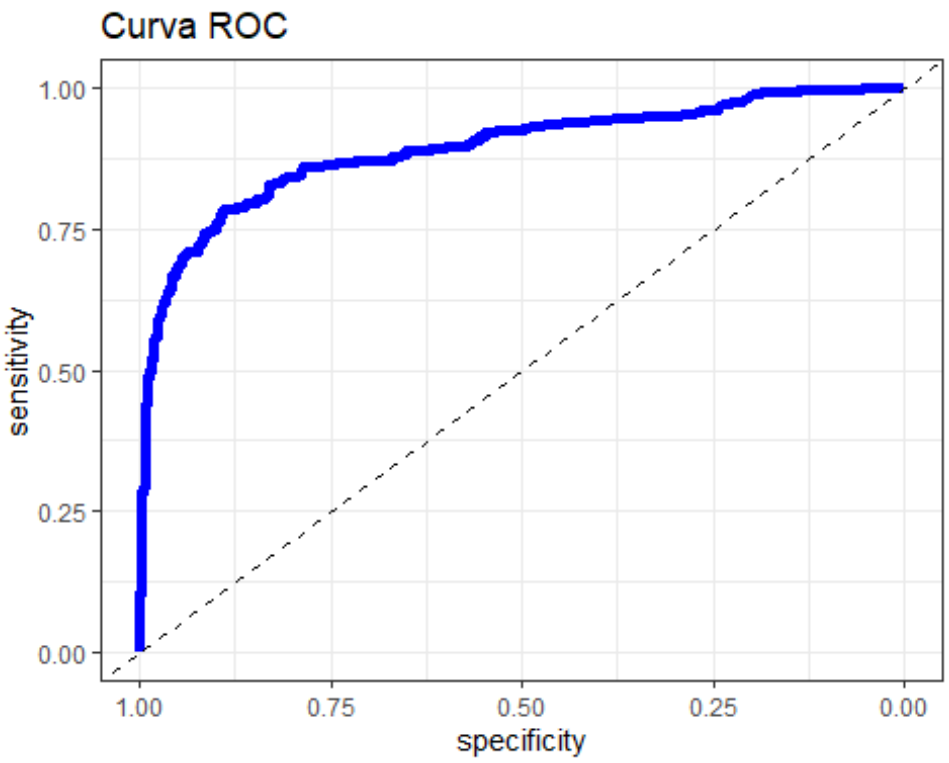
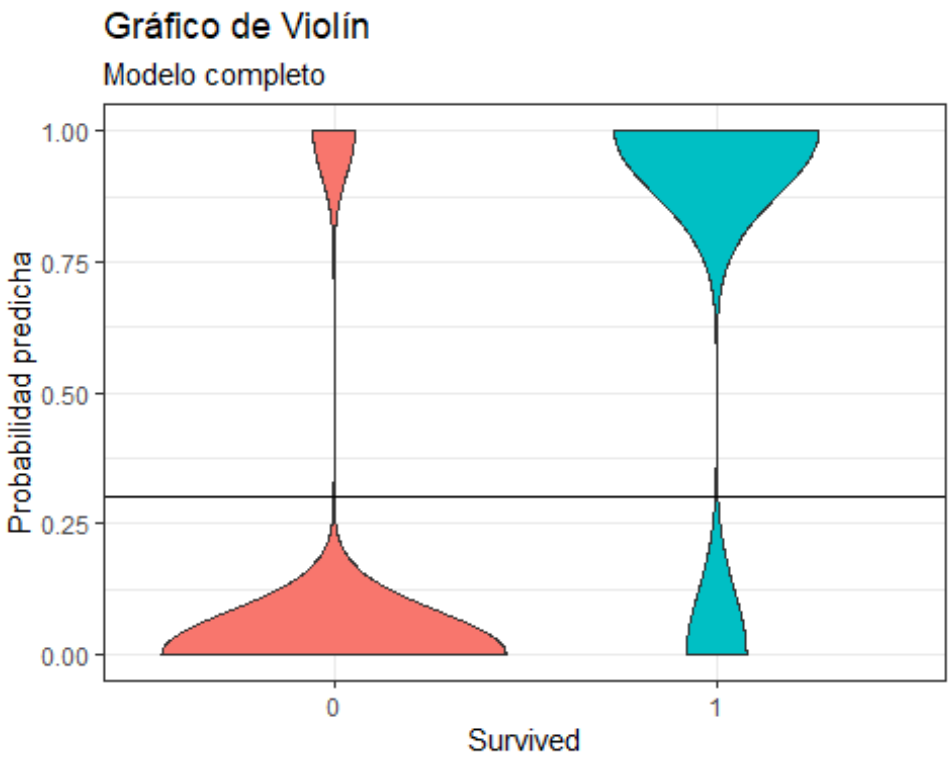


Gráfico de violín

```
violin_data = data.frame(Survived = data_train$Survived, predicted =
pred)

ggplot(data = violin_data, aes(x = Survived, y = pred, group = Survived,
                               fill=factor(Survived))) + geom_violin() +
  geom_abline(aes(intercept = 0.3, slope = 0)) + theme_bw() + guides(fill
= FALSE) + labs(title ='Gráfico de Violín', subtitle = 'Modelo completo',
y = 'Probabilidad predicha')

## Warning: The `<scale>` argument of `guides()` cannot be `FALSE`. Use
"none" instead as
## of ggplot2 3.3.4.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning
was
## generated.
```



Finalmente, es posible concluir en base a las predicciones para los datos de entrenamiento, que el modelo logístico seleccionado resulta ser mayormente adecuado para predecir la supervivencia de los pasajeros abordo del titanic, esto debido a que al momento de emplear el modelo elegido para realizar predicciones de la supervivencia de los pasajeros del dataset de entrenamiento, el modelo resulta tener en general niveles considerablemente elevados en sus diferentes métricas de desempeño, por ejemplo, el modelo seleccionado tiene un nivel de accuracy (exactitud) en entrenamiento de 0.8399, lo que significa que de todos los datos de entrenamiento clasificados por el modelo, el 83.66% de ellos fueron clasificados de forma correcta, mientras que solamente un 16.44% se clasificaron incorrectamente por el mismo modelo, además, de forma similar, el modelo tiene un nivel de precisión del 84.73%, que hace referencia al hecho de que de todos los pasajeros que realmente sobrevivieron al desastre en el Titanic, el modelo clasificó al 84.73% de ellos como que sí sobrevivieron cuando realmente es así, por lo que considerando lo anterior, es posible afirmar que el modelo logístico elegido es adecuado para predecir la supervivencia de los pasajeros en general.

5. Validación del modelo

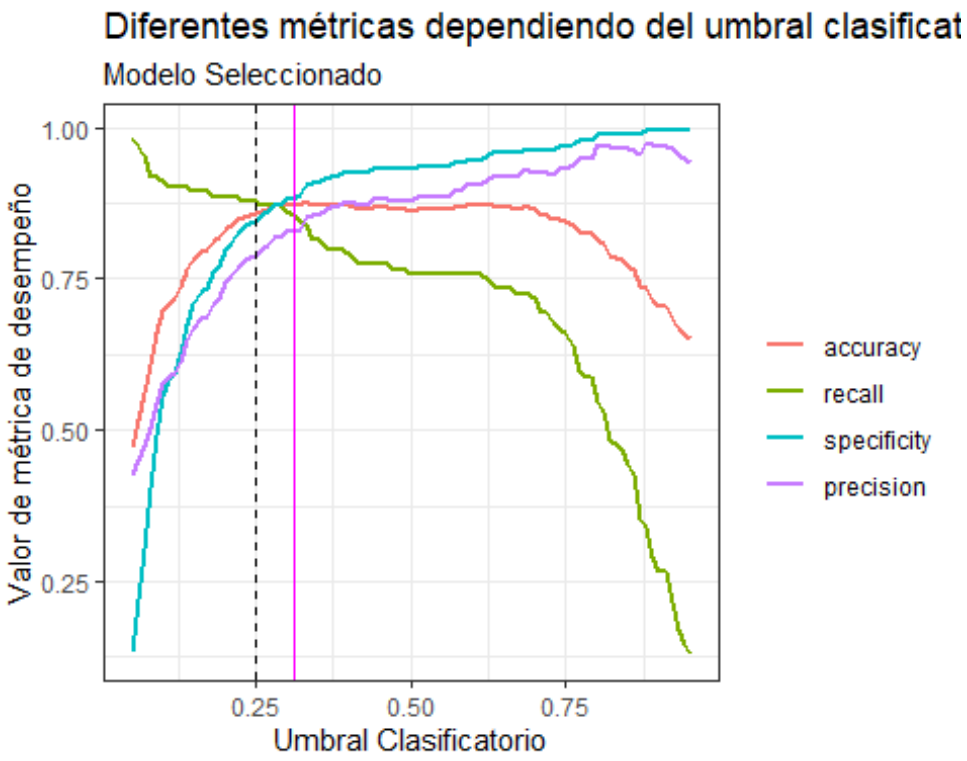
Selección del umbral de clasficación óptimo

```
pred_val = predict(model2, newdata = data_valid, type = "response")

clase_real = data_valid$Survived
```

```r
datosV = data.frame(accuracy = NA, recall = NA, specificity = NA,
precision = NA)

for (i in 5:95){
  clase_predicha = ifelse(pred_val>i/100, 1, 0)

##Creamos la matriz de confusión

cm = table(clase_predicha, clase_real)

## Accuracy (exactitud): Proporción de correctamente predichos

datosV[i,1] = (cm[1,1] + cm[2,2]) / (cm[1,1] + cm[1,2] + cm[2,1] +
cm[2,2])

## Recall: Tasa de positivos correctamente predichos

datosV[i,2] = (cm[2,2]) / (cm[1,2] + cm[2,2])

## Specificity: Tasa de negativos correctamente predichos

datosV[i,3] = cm[1,1] / (cm[1,1] + cm[2,1])

## Precision: Tasa de bien clasificados entre los clasificados como
positivos

datosV[i,4] = cm[2,2] / (cm[2,1] + cm[2,2])

}

## Limpieza del conjunto de datos utilizado

datosV = na.omit(datosV)

datosV$umbral = seq(0.05,0.95,0.01)

# Agregar la columna "métrica" a los datos para almacenar los valores de
las métricas de
# desempeño del modelo

library(reshape2)

##
## Attaching package: 'reshape2'

## The following object is masked from 'package:tidyr':
##
##     smiths
```

```r
datosV_m = melt(datosV,id.vars=c("umbral"))

colnames(datosV_m)[2] = c("Métrica")

library(ggplot2)

# Comenzar con un umbral de clasificación de 0.25

umbral = 0.25

ggplot(data = datosV_m, aes(x = umbral, y = value, color = Métrica)) +
  geom_line(size = 1) + theme_bw() +
  labs(title= "Diferentes métricas dependiendo del umbral
clasificatorio",
       subtitle= 'Modelo Seleccionado',
       color="", x = "Umbral Clasificatorio", y = "Valor de métrica de
desempeño") +
  geom_vline(xintercept = umbral, linetype = "dashed", color = "black") +
  geom_vline(xintercept = mean(c(0.25, 0.375)), linetype = "solid", color
= "magenta")
```

```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2
3.4.0.
## ℹ Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning
was
## generated.
```



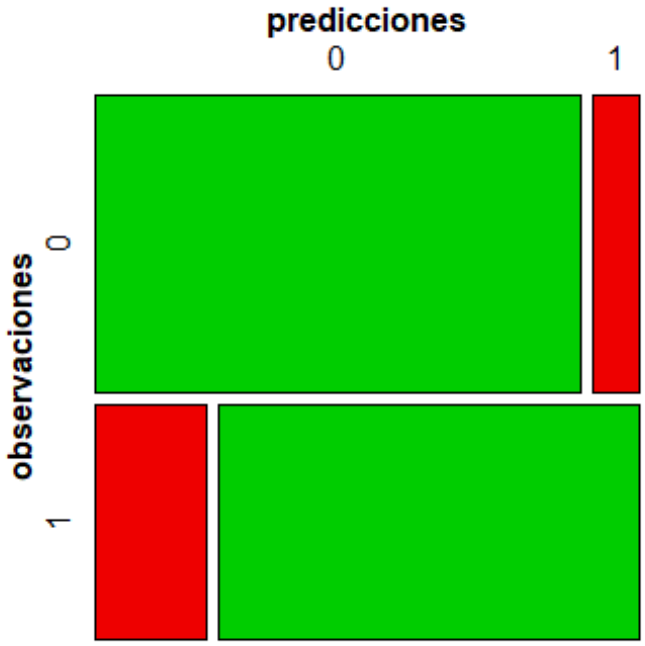
De acuerdo a la gráfica mostrada anteriormente, es posible apreciar que casi todas las métricas alcanzan un punto máximo en común cuando el umbral de clasificación del modelo logístico se ubica entre 0.25 y 0.5, más específicamente entre 0.25 y 0.375, lo cual es un indicativo de que el modelo tiene el mejor rendimiento en general, cuando el punto de corte o umbral para la clasificación es aproximadamente igual a 0.3125, por lo cual, cuando el umbral es aproximadamente igual a dicho valor, el modelo es capaz de realizar las predicciones más precisas y confiables posible, sin descuidar el resto de las métricas de desempeño, motivo por el cual, en resumen, el umbral óptimo elegido es de 0.3125.

Matriz de confusión con umbral óptimo elegido

```
pred_optimo = ifelse(pred_val > mean(0.25, 0.375), yes = 1, no = 0)

mat_opt = table(pred_optimo, data_valid$Survived, dnn =
c("observaciones", "predicciones"))

mat_opt
```

```
##              predicciones
## observaciones   0   1
##             0 159  15
##             1  29 109
```

```
mosaic(mat_opt, shade = TRUE, colorize = TRUE,
       gp = gpar(fill = matrix(c("green3", "red2", "red2", "green3"), 2,
2)))
```




```
# Mostrar nuevas métricas con el umbral de clasificación modificado

ex_optimo = (mat_opt[1, 1] + mat_opt[2, 2]) / sum(mat_opt)
cat("Exactitud (accuracy) del modelo = ", ex_optimo,"\n")

## Exactitud (accuracy) del modelo =  0.8589744

sens_opt = mat_opt[1, 1] / sum(mat_opt[1, ])
cat("Sensibilidad del modelo = ", sens_opt,"\n")

## Sensibilidad del modelo =  0.9137931

esp_opt = mat_opt[2, 2] / sum(mat_opt[2, ])
cat("La Especificidad del modelo es", esp_opt,"\n")

## La Especificidad del modelo es 0.7898551

precision_opt = mat_opt[1, 1] / sum(mat_opt[, 1])
cat("La Precisión del modelo es", precision_opt,"\n")

## La Precisión del modelo es 0.8457447
```

En las métricas del nuevo modelo, es posible notar que el valor de las mismas disminuye ligeramente al modificar el umbral configurado originalmente (0.5), no obstante, el desempeño general del modelo sigue siendo mayormente adecuado, aunque el modelo elegido originalmente tiene métricas ligeramente más altas, lo cual garantiza que el modelo escogido originalmente arroje las predicciones más

confiables y precisas posibles, por lo que al final, se escoge el modelo original en vez de éste nuevo.

6. Testeo con la base de datos de prueba

```
# Importar base de datos de prueba

data_test = na.omit(read.csv("Titanic_test.csv"))

data_test$Pclass = as.integer(data_test$Pclass)

# Calcular las predicciones de supervivencia de pasajeros a partir de la
base de datos de
# prueba y con el umbral óptimo elegido

pred_test = ifelse(predict(model2, newdata = data_test,
                           type = "response") > mean(c(0.25, 0.375)), yes
= 1, no = 0)

data_test$Prediction = pred_test

clase_real_test = data_test$Survived

print("Predicciones para la base de datos de prueba:")

## [1] "Predicciones para la base de datos de prueba:"

head(pred_test, 20)

##  1  2  3  4  5  6  7  8  9 10 12 13 14 15 16 17 18 19 20 21
##  0  1  0  0  1  0  1  0  1  0  0  1  0  1  1  0  0  1  1  0
```

7. Conclusión en el contexto del problema

En conclusión, las principales características de aquellos pasajeros que sobrevivieron radican principalmente en que en su gran mayoría fueron personas jóvenes, con una edad comprendida entre los 20 y los 50 años (media de 35 años), lo cual sugiere que las personas jóvenes poseen un mayor grado de resistencia contra las adversidades a diferencia de las personas de edad avanzada, además de que también la mayor parte de aquellos pasajeros que sí sobrevivieron eran de la tercera clase, lo cual es un indicativo de que el estar en la primera clase (la más alta) no era garantía de que dichos pasajeros sobrevivieran, puesto que la supervivencia en sí dependía de factores adicionales como la edad que ya se mencionó anteriormente, motivo por el cual, es posible que los pasajeros de la tercera clase hayan podido escapar más rápido del barco antes de que éste mismo se hundiera, que aquellos otros pasajeros de primera y segunda clase, por lo que se aprecia una mayor cantidad de pasajeros que sí sobrevivieron y pertenecían precisamente a la tercera clase.

Por otro lado, la ecuación que representa al modelo elegido para predecir la supervivencia de los pasajeros es:

$$log\left(\frac{p}{1-p}\right) = 4.53 - 1.28 * Pclass2 - 2.28 * Pclass3 - 3.599 * Sexmale - 0.0397 * Age - 0.3457 * SibSp - 0.1642 * Parch$$

En el que el coeficiente -1.28 significa que si el pasajero pertenece a la segunda clase, el logaritmo de los nomios experimenta un cambio de -1.28 unidades, mientras que el coeficiente -2.28 señala que si el pasajero pertenecía a la tercera clase, el logaritmo de odds disminuye en 2.28 unidades, mientras que a su vez, el coeficviente -3.599 indica que si el pasajero es hombre, el logaritmo de odds disminuye en 3.599 unidades, mientras que de manera similar, por cada año que cumpla un pasajero determinado, el logaritmo de odds disminuye en 0.0397 unidades, ya que conforme la edad avanza, se reducen las capacidades físicas para hacer frente a eventos de intensa magnitud. Además, el coeficiente 0.3457 indica que por cada hermano o pariente que tengan los pasajeros, el logaritmo de los nomios experimenta un decremento de 0.3457 unidades y finalmente, el coeficiente 0.1642 señala que si la variable predictora `Parch` es igual a 1, entonces el logaritmo de odds disminuye en 0.1642 unidades.

Por último, el mejor umbral de clasificación fue de 0.3125, puesto que cuando el umbral es igual a ese valor, prácticamente se logra clasificar correctamente a una gran mayoría de los pasajeros abordo del Titanic, mientras que con umbrales mayores, el modelo sería cada vez menos capaz de detectar de forma acertada pasajeros que en realidad sobrevivieron pero que el modelo los considere como fallecidos y viceversa, por lo que un umbral relativamente pequeño como el anteriormente mencionado, permite al modelo tener un adecuado grado de sensibilidad para clasificar, es decir, que no será necesario que la probabilidad de supervivencia predicha para un cierto pasajero sea muy alta para clasificarlo como superviviente, permitiendo incluso abarcar pasajeros cuya probabilidad de supervivencia sea mayormente baja pero suficiente para considerar que sí sobrevivió, por lo que para que el modelo pueda tomar en cuenta a dichos pasajeros, se requieren umbrales clasificatorios mayormente pequeños.