Actividad Integradora 2

Luis Maximiliano López Ramírez

2024-11-19

Utiliza los archivos del Titanic para detectar cuáles fueron las principales características que de las personas que sobrevivieron y elabora en modelo de predicción de sobrevivencia o no en el Titanic. Utiliza en las siguientes bases de datos:

Base de datos del Titanic: TitanicDownload Titanic Base de datos de prueba: Titanic_test Download Titanic_test

Las variables para la base de datos son las siguientes (excluye aquellas que no sean de interés para el análisis):

Name: Nombre del pasajero *PassengerId:* Ids del pasajero *Survived:* Si sobrevivió o no (No = 0, Sí = 1)
Ticket: Número de ticket
Cabin: Cabina en la que viajó
Pclass: Clase en la que viajó (1 = 1era, 2 = 2da, 3 = 3ra)
Sex: Masculino o Femenino (male/female)
Age: Edad
SibSp: Número de hermanos/conyuge a bordo
Parch: Número de padres/hijos a bordo
Fare: Tarifa que pagó
Embarked: Puerto de embarcación (C = Cherbourg, Q = Queenstown, S = Southampton)

 # 1. Prepara la base de datos Titanic

```
# Leer archivo CSV desde el directorio actual
titanic <- read.csv("Titanic.csv")

# Leer archivo CSV desde el directorio actual
titanic_test <- read.csv("Titanic_test.csv")
```

Analiza los datos faltantes

```
# Número de valores faltantes por columna en el conjunto de entrenamiento
colSums(is.na(titanic))
```

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

```
##           0           0           0           1           0
2

# Número de valores faltantes por columna en el conjunto de prueba
colSums(is.na(titanic_test))

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

# Porcentaje de valores faltantes en el conjunto de entrenamiento
colMeans(is.na(titanic)) * 100

## PassengerId    Survived      Pclass        Name         Sex
Age
##  0.00000000  0.00000000  0.00000000  0.00000000  0.00000000
20.09167303
##       SibSp       Parch      Ticket        Fare       Cabin
Embarked
##  0.00000000  0.00000000  0.00000000  0.07639419  0.00000000
0.15278839

# Porcentaje de valores faltantes en el conjunto de prueba
colMeans(is.na(titanic_test)) * 100

## PassengerId      Pclass        Name         Sex         Age
SibSp
##   0.0000000   0.0000000   0.0000000   0.0000000  20.5741627
0.0000000
##       Parch      Ticket        Fare       Cabin    Embarked
##   0.0000000   0.0000000   0.2392344   0.0000000   0.0000000

# Eliminar filas con valores nulos en el conjunto de entrenamiento
titanic_clean <- na.omit(titanic)

# Eliminar filas con valores nulos en el conjunto de prueba
titanic_test_clean <- na.omit(titanic_test)
```

Realiza un análisis descriptivo

```
# Tipos de datos en el conjunto limpio
str(titanic_clean)

## 'data.frame':    1043 obs. of  12 variables:
##  $ PassengerId: int  892 893 894 895 896 897 898 899 900 901 ...
##  $ Survived   : int  0 1 0 0 1 0 1 0 1 0 ...
##  $ Pclass     : int  3 3 2 3 3 3 3 2 3 3 ...
##  $ Name       : chr  "Kelly, Mr. James" "Wilkes, Mrs. James (Ellen
Needs)" "Myles, Mr. Thomas Francis" "Wirz, Mr. Albert" ...
```

```
##  $ Sex        : chr  "male" "female" "male" "male" ...
##  $ Age        : num  34.5 47 62 27 22 14 30 26 18 21 ...
##  $ SibSp      : int  0 1 0 0 1 0 0 1 0 2 ...
##  $ Parch      : int  0 0 0 0 1 0 0 1 0 0 ...
##  $ Ticket     : chr  "330911" "363272" "240276" "315154" ...
##  $ Fare       : num  7.83 7 9.69 8.66 12.29 ...
##  $ Cabin      : chr  "" "" "" "" ...
##  $ Embarked   : chr  "Q" "S" "Q" "S" ...
##  - attr(*, "na.action")= 'omit' Named int [1:266] 11 23 30 34 37 40 42
48 55 59 ...
##   ..- attr(*, "names")= chr [1:266] "11" "23" "30" "34" ...
```

```
str(titanic_test_clean)
```

```
## 'data.frame':    331 obs. of  11 variables:
##  $ PassengerId: int  892 893 894 895 896 897 898 899 900 901 ...
##  $ Pclass     : int  3 3 2 3 3 3 3 2 3 3 ...
##  $ Name       : chr  "Kelly, Mr. James" "Wilkes, Mrs. James (Ellen
Needs)" "Myles, Mr. Thomas Francis" "Wirz, Mr. Albert" ...
##  $ Sex        : chr  "male" "female" "male" "male" ...
##  $ Age        : num  34.5 47 62 27 22 14 30 26 18 21 ...
##  $ SibSp      : int  0 1 0 0 1 0 0 1 0 2 ...
##  $ Parch      : int  0 0 0 0 1 0 0 1 0 0 ...
##  $ Ticket     : chr  "330911" "363272" "240276" "315154" ...
##  $ Fare       : num  7.83 7 9.69 8.66 12.29 ...
##  $ Cabin      : chr  "" "" "" "" ...
##  $ Embarked   : chr  "Q" "S" "Q" "S" ...
##  - attr(*, "na.action")= 'omit' Named int [1:87] 11 23 30 34 37 40 42
48 55 59 ...
##   ..- attr(*, "names")= chr [1:87] "11" "23" "30" "34" ...
```

```
# Resumen estadístico de las variables numéricas
summary(titanic_clean)
```

```
##   PassengerId        Survived          Pclass          Name
##  Min.   :   1.0   Min.   :0.0000   Min.   :1.000   Length:1043
##  1st Qu.: 326.5   1st Qu.:0.0000   1st Qu.:1.000   Class :character
##  Median : 662.0   Median :0.0000   Median :2.000   Mode  :character
##  Mean   : 655.4   Mean   :0.3979   Mean   :2.209
##  3rd Qu.: 973.5   3rd Qu.:1.0000   3rd Qu.:3.000
##  Max.   :1307.0   Max.   :1.0000   Max.   :3.000
##      Sex                 Age            SibSp            Parch
##  Length:1043        Min.   : 0.17   Min.   :0.0000   Min.   :0.0000
##  Class :character   1st Qu.:21.00   1st Qu.:0.0000   1st Qu.:0.0000
##  Mode  :character   Median :28.00   Median :0.0000   Median :0.0000
##                     Mean   :29.81   Mean   :0.5043   Mean   :0.4219
##                     3rd Qu.:39.00   3rd Qu.:1.0000   3rd Qu.:1.0000
##                     Max.   :80.00   Max.   :8.0000   Max.   :6.0000
##     Ticket               Fare           Cabin             Embarked
##  Length:1043        Min.   :  0.00   Length:1043        Length:1043
##  Class :character   1st Qu.:  8.05   Class :character   Class
```

```
:character
##  Mode  :character   Median : 15.75   Mode  :character   Mode
:character
##                     Mean   : 36.60
##                     3rd Qu.: 35.08
##                     Max.   :512.33
```

```
summary(titanic_test_clean)
```

```
##   PassengerId         Pclass          Name               Sex
##  Min.   : 892.0   Min.   :1.000   Length:331         Length:331
##  1st Qu.: 992.5   1st Qu.:1.000   Class :character   Class :character
##  Median :1100.0   Median :2.000   Mode  :character   Mode  :character
##  Mean   :1100.2   Mean   :2.142
##  3rd Qu.:1210.5   3rd Qu.:3.000
##  Max.   :1307.0   Max.   :3.000
##       Age            SibSp            Parch           Ticket
##  Min.   : 0.17   Min.   :0.0000   Min.   :0.0000   Length:331
##  1st Qu.:21.00   1st Qu.:0.0000   1st Qu.:0.0000   Class :character
##  Median :27.00   Median :0.0000   Median :0.0000   Mode  :character
##  Mean   :30.18   Mean   :0.4834   Mean   :0.3988
##  3rd Qu.:39.00   3rd Qu.:1.0000   3rd Qu.:1.0000
##  Max.   :76.00   Max.   :8.0000   Max.   :6.0000
##       Fare           Cabin             Embarked
##  Min.   :  0.00   Length:331         Length:331
##  1st Qu.:  8.05   Class :character   Class :character
##  Median : 16.00   Mode  :character   Mode  :character
##  Mean   : 40.98
##  3rd Qu.: 40.63
##  Max.   :512.33
```

```
# Frecuencia de 'Sex'
table(titanic_clean$Sex)
```

```
##
## female   male
##    386    657
```

```
table(titanic_test_clean$Sex)
```

```
##
## female   male
##    127    204
```

```
# Proporciones
prop.table(table(titanic_clean$Sex))
```

```
##
##    female      male
## 0.3700863 0.6299137
```

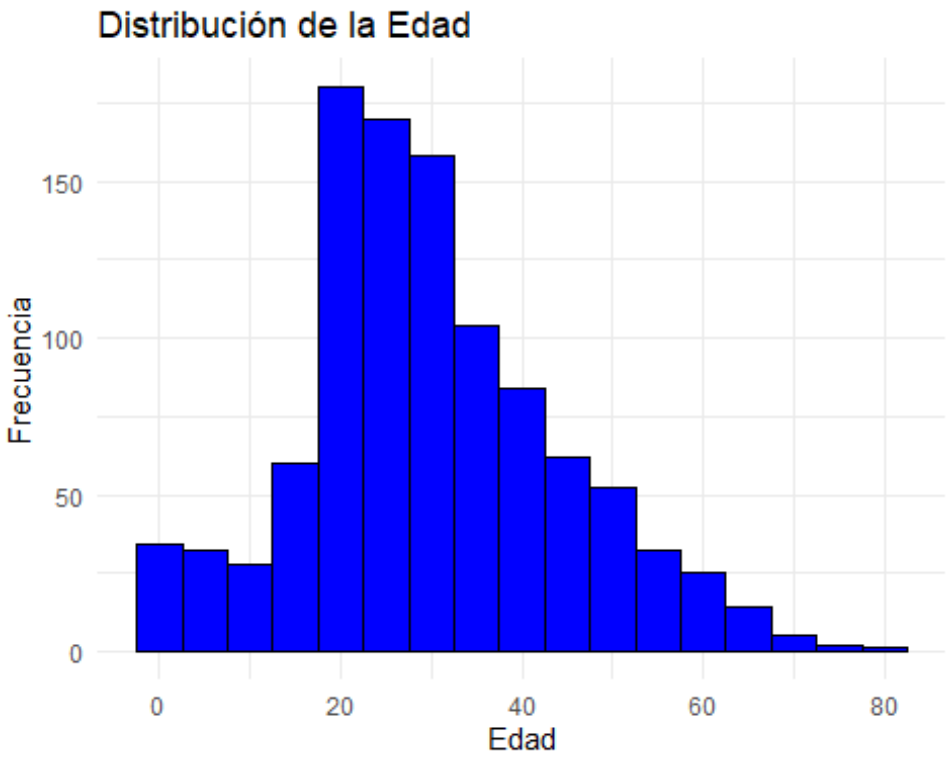
```
prop.table(table(titanic_test_clean$Sex))
```

```
##
##    female      male
## 0.3836858 0.6163142
```

```
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 4.3.2
```

```
# Histograma de la edad
ggplot(titanic_clean, aes(x = Age)) +
  geom_histogram(binwidth = 5, fill = "blue", color = "black") +
  theme_minimal() +
  labs(title = "Distribución de la Edad", x = "Edad", y = "Frecuencia")
```




```
# Convertir en el conjunto de entrenamiento
titanic_clean$Sex <- ifelse(titanic_clean$Sex == "male", 1, 0)

# Convertir en el conjunto de prueba
titanic_test_clean$Sex <- ifelse(titanic_test_clean$Sex == "male", 1, 0)
```

Haz una partición de los datos (70-30) para el entrenamiento y la validación. Revisa la proporción de sobrevivientes para la partición y la base original.

```
library(caret)
```

```
## Warning: package 'caret' was built under R version 4.3.2
```

```
## Loading required package: lattice
```

```r
set.seed(123)  # Fijar la semilla para reproducibilidad

# Crear partición para el conjunto de entrenamiento (70%)
train_index <- createDataPartition(titanic_clean$Survived, p = 0.7, list
= FALSE)

# Dividir los datos
train_data <- titanic_clean[train_index, ]  # Conjunto de entrenamiento
validation_data <- titanic_clean[-train_index, ]  # Conjunto de
validación

# Proporción en la base original
prop_original <- prop.table(table(titanic_clean$Survived))

# Proporción en el conjunto de entrenamiento
prop_train <- prop.table(table(train_data$Survived))

# Proporción en el conjunto de validación
prop_validation <- prop.table(table(validation_data$Survived))

# Mostrar las proporciones
list(
  Original = prop_original,
  Entrenamiento = prop_train,
  Validación = prop_validation
)
```

```
## $Original
##
##         0         1
## 0.6021093 0.3978907
##
## $Entrenamiento
##
##         0         1
## 0.6073871 0.3926129
##
## $Validación
##
##         0         1
## 0.5897436 0.4102564
```

 # 2. Con la base de datos de entrenamiento, encuentra un modelo logístico para encontrar el mejor conjunto de predictores que auxilien a clasificar la dirección de cada observación.

Auxiliate del criterio de AIC para determinar cuál es el mejor modelo.

```r
# Identificar las columnas de tipo carácter
char_vars <- sapply(train_data, is.character)
```

```r
# Eliminar las columnas de tipo carácter del conjunto de entrenamiento
train_data_clean <- train_data[, !char_vars]

# Hacer lo mismo para el conjunto de validación
char_vars_val <- sapply(validation_data, is.character)
validation_data_clean <- validation_data[, !char_vars_val]

# Eliminar la columna PassengerId del conjunto de entrenamiento
train_data_clean <- train_data_clean[, !colnames(train_data_clean) %in%
"PassengerId"]

# Eliminar la columna PassengerId del conjunto de validación
validation_data_clean <- validation_data_clean[,
!colnames(validation_data_clean) %in% "PassengerId"]


# Modelo logístico inicial con todas las variables predictoras
modelo_completo <- glm(Survived ~ ., data = train_data_clean, family =
binomial)

# Instalar y cargar la librería MASS si no está instalada
library(MASS)

# Selección del mejor modelo basado en AIC
modelo_optimo <- stepAIC(modelo_completo, direction = "both")
```

```
## Start:  AIC=590.06
## Survived ~ Pclass + Sex + Age + SibSp + Parch + Fare
##
##          Df Deviance    AIC
## - Parch   1   576.36 588.36
## - Fare    1   577.00 589.00
## <none>        576.06 590.06
## - SibSp   1   579.89 591.89
## - Age     1   597.37 609.37
## - Pclass  1   613.88 625.88
## - Sex     1   880.45 892.45
##
## Step:  AIC=588.36
## Survived ~ Pclass + Sex + Age + SibSp + Fare
##
##          Df Deviance    AIC
## - Fare    1   577.11 587.11
## <none>        576.36 588.36
## + Parch   1   576.06 590.06
## - SibSp   1   581.40 591.40
## - Age     1   597.48 607.48
## - Pclass  1   615.83 625.83
```

```
## - Sex     1   886.13 896.13
##
## Step:  AIC=587.11
## Survived ~ Pclass + Sex + Age + SibSp
##
##          Df Deviance    AIC
## <none>        577.11 587.11
## + Fare    1   576.36 588.36
## + Parch   1   577.00 589.00
## - SibSp   1   581.68 589.68
## - Age     1   598.42 606.42
## - Pclass  1   638.76 646.76
## - Sex     1   890.69 898.69
```

```
# Mostrar el resumen del modelo óptimo
summary(modelo_optimo)
```

```
##
## Call:
## glm(formula = Survived ~ Pclass + Sex + Age + SibSp, family =
binomial,
##     data = train_data_clean)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  5.341614   0.580188   9.207  < 2e-16 ***
## Pclass      -1.110879   0.150124  -7.400 1.36e-13 ***
## Sex         -3.438301   0.233736 -14.710  < 2e-16 ***
## Age         -0.037868   0.008423  -4.496 6.94e-06 ***
## SibSp       -0.259574   0.124230  -2.089   0.0367 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 979.40  on 730  degrees of freedom
## Residual deviance: 577.11  on 726  degrees of freedom
## AIC: 587.11
##
## Number of Fisher Scoring iterations: 5
```

Propón por lo menos los dos que consideres mejores modelos.

Modelo 1: Survived ~ Pclass + Sex + Age + SibSp + Fare con AIC = 588.36

Modelo 2: Survived ~ Pclass + Sex + Age + SibSp con AIC = 587.11

 # 3. Analiza los modelos a través de:

Identificación de la Desviación residual de cada modelo

```r
# Modelo 1: Survived ~ Pclass + Sex + Age + SibSp + Fare
modelo_1 <- glm(Survived ~ Pclass + Sex + Age + SibSp + Fare,
                family = binomial,
                data = train_data_clean)

# Modelo 2: Survived ~ Pclass + Sex + Age + SibSp
modelo_2 <- glm(Survived ~ Pclass + Sex + Age + SibSp,
                family = binomial,
                data = train_data_clean)

# Desviación residual del Modelo 1
desviacion_modelo_1 <- modelo_1$deviance
cat("Desviación Residual del Modelo 1:", desviacion_modelo_1, "\n")

## Desviación Residual del Modelo 1: 576.3587

# Desviación residual del Modelo 2
desviacion_modelo_2 <- modelo_2$deviance
cat("Desviación Residual del Modelo 2:", desviacion_modelo_2, "\n")

## Desviación Residual del Modelo 2: 577.1135
```

Identificación de la Desviación nula

```r
# Identificación de la desviación nula
desviacion_nula_modelo_1 <- modelo_1$null.deviance
desviacion_nula_modelo_2 <- modelo_2$null.deviance

# Imprimir las desviaciones nulas
cat("Desviación Nula del Modelo 1:", desviacion_nula_modelo_1, "\n")

## Desviación Nula del Modelo 1: 979.3975

cat("Desviación Nula del Modelo 2:", desviacion_nula_modelo_2, "\n")

## Desviación Nula del Modelo 2: 979.3975
```

Cálculo de la Desviación Explicada

```r
# Cálculo de la desviación explicada
desviacion_explicada_1 <- 1 - (modelo_1$deviance /
modelo_1$null.deviance)
desviacion_explicada_2 <- 1 - (modelo_2$deviance /
modelo_2$null.deviance)

# Imprimir los resultados
cat("Desviación Explicada del Modelo 1:", desviacion_explicada_1, "\n")

## Desviación Explicada del Modelo 1: 0.4115171

cat("Desviación Explicada del Modelo 2:", desviacion_explicada_2, "\n")
```

```
## Desviación Explicada del Modelo 2: 0.4107464
```

Prueba de la razón de verosimilitud

```
# Modelo completo: Survived ~ Pclass + Sex + Age + SibSp + Fare
modelo_completo <- glm(Survived ~ Pclass + Sex + Age + SibSp + Fare,
                       family = binomial,
                       data = train_data_clean)

# Modelo reducido: Survived ~ Pclass + Sex + Age + SibSp
modelo_reducido <- glm(Survived ~ Pclass + Sex + Age + SibSp,
                       family = binomial,
                       data = train_data_clean)

# Calcular la estadística de la prueba
D <- modelo_reducido$deviance - modelo_completo$deviance

# Grados de libertad
df <- modelo_reducido$df.residual - modelo_completo$df.residual

# p-valor
p_value <- pchisq(D, df, lower.tail = FALSE)

# Resultados
cat("Estadística D:", D, "\n")

## Estadística D: 0.7548594

cat("Grados de libertad:", df, "\n")

## Grados de libertad: 1

cat("p-valor:", p_value, "\n")

## p-valor: 0.3849421
```

Define cuál es el mejor modelo

Aunque el *Modelo 1* tiene un ajuste ligeramente mejor (menor desviación residual y mayor desviación explicada), la diferencia no es estadísticamente significativa según la prueba de la razón de verosimilitud. Además, el *Modelo 2* es más simple, lo que favorece su uso en términos de interpretabilidad y parsimonia.

Por lo que el mejor modelo es el **Modelo 2**

Escribe su ecuación, analiza sus coeficientes y detecta el efecto de cada predictor en la clasificación.

```
# Ajustar el modelo logístico
modelo_2 <- glm(Survived ~ Pclass + Sex + Age + SibSp,
                family = binomial,
                data = train_data_clean)
```

```r
# Resumen del modelo
summary(modelo_2)
```

```
##
## Call:
## glm(formula = Survived ~ Pclass + Sex + Age + SibSp, family =
binomial,
##     data = train_data_clean)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  5.341614   0.580188   9.207  < 2e-16 ***
## Pclass      -1.110879   0.150124  -7.400 1.36e-13 ***
## Sex         -3.438301   0.233736 -14.710  < 2e-16 ***
## Age         -0.037868   0.008423  -4.496 6.94e-06 ***
## SibSp       -0.259574   0.124230  -2.089   0.0367 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 979.40  on 730  degrees of freedom
## Residual deviance: 577.11  on 726  degrees of freedom
## AIC: 587.11
##
## Number of Fisher Scoring iterations: 5
```

```r
# Extraer los coeficientes
coeficientes <- coef(modelo_2)
cat("Coeficientes del Modelo:\n")
```

```
## Coeficientes del Modelo:
```

```r
print(coeficientes)
```

```
## (Intercept)      Pclass         Sex         Age       SibSp
##  5.34161445 -1.11087878 -3.43830103 -0.03786791 -0.25957417
```

```r
# Escribir la ecuación
cat("\nEcuación del modelo:\n")
```

```
##
## Ecuación del modelo:
```

```r
cat("logit(P(Survived)) = ", coeficientes["(Intercept)"],
    " + ", coeficientes["Pclass"], "*Pclass",
    " + ", coeficientes["Sex"], "*Sex",
    " + ", coeficientes["Age"], "*Age",
    " + ", coeficientes["SibSp"], "*SibSp\n")
```

```
## logit(P(Survived)) =  5.341614  +  -1.110879 *Pclass  +  -3.438301
*Sex  +  -0.03786791 *Age  +  -0.2595742 *SibSp
```

El efecto de cada predictor en la clasificación puede interpretarse desde los coeficientes del modelo logístico. El sexo (Sex) tiene el impacto más fuerte: ser hombre (Sex=1) reduce drásticamente la probabilidad de supervivencia en comparación con ser mujer (Sex=0), con una disminución significativa en el logit de -3.438. Esto indica que el género es el factor más determinante en la clasificación. La clase del boleto (Pclass) también tiene un efecto importante: por cada aumento de clase (de 1ª a 3ª), el logit disminuye en -1.111, mostrando que los pasajeros en clases más bajas tienen menor probabilidad de sobrevivir. Ambos predictores son los más influyentes en la clasificación.

Otros predictores como la edad (Age) y el número de hermanos/cónyuges a bordo (SibSp) tienen efectos menores pero significativos. Por cada año adicional de edad, el logit disminuye ligeramente (-0.038), lo que sugiere que los pasajeros mayores tienen menor probabilidad de sobrevivir. Asimismo, tener más familiares cercanos abordo reduce la probabilidad de supervivencia, aunque el efecto es más moderado (-0.260 por cada unidad de SibSp). En resumen, mientras que todos los predictores contribuyen al modelo, el género y la clase social dominan la capacidad del modelo para clasificar correctamente.

 # 4. Analiza las predicciones para los datos de entrenamiento

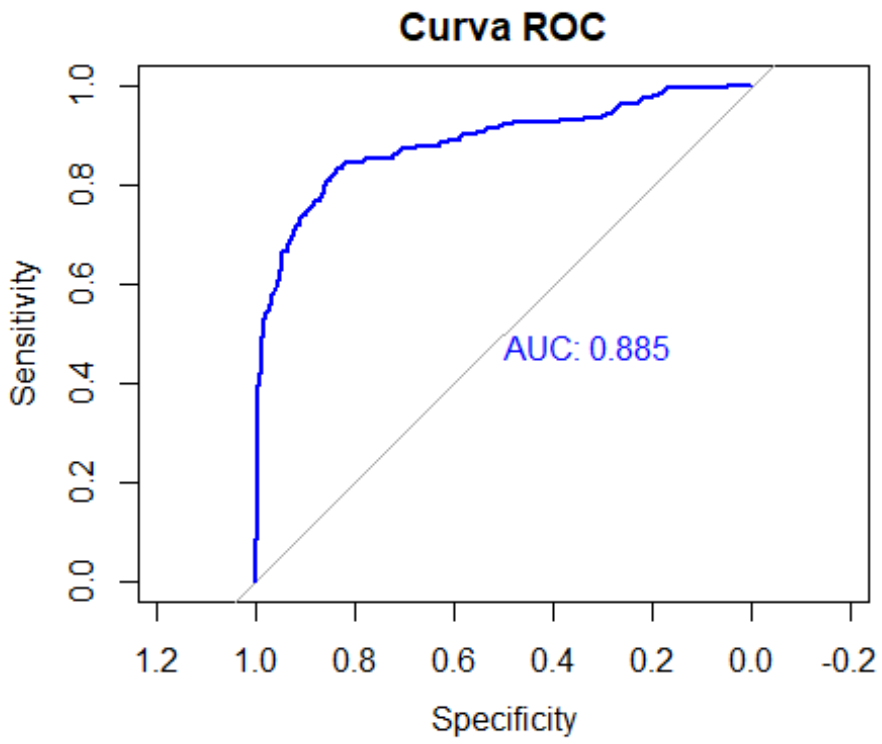
Elabora la matriz de confusión

```
# Calcular las probabilidades predichas en el conjunto de entrenamiento
train_data_clean$Probabilidad <- predict(modelo_2, type = "response")

# Clasificar las observaciones con un umbral de 0.5
train_data_clean$Prediccion <- ifelse(train_data_clean$Probabilidad >
0.5, 1, 0)

# Crear la matriz de confusión
matriz_confusion <- table(Predicted = train_data_clean$Prediccion, Actual
= train_data_clean$Survived)

# Mostrar la matriz de confusión
cat("Matriz de Confusión:\n")

## Matriz de Confusión:

print(matriz_confusion)

##          Actual
## Predicted   0   1
##         0 398  74
##         1  46 213
```

```r
# Calcular el Accuracy
accuracy <- sum(diag(matriz_confusion)) / sum(matriz_confusion)
cat("Accuracy del modelo en los datos de entrenamiento:", accuracy, "\n")

## Accuracy del modelo en los datos de entrenamiento: 0.8358413
```

Elabora la Curva ROC

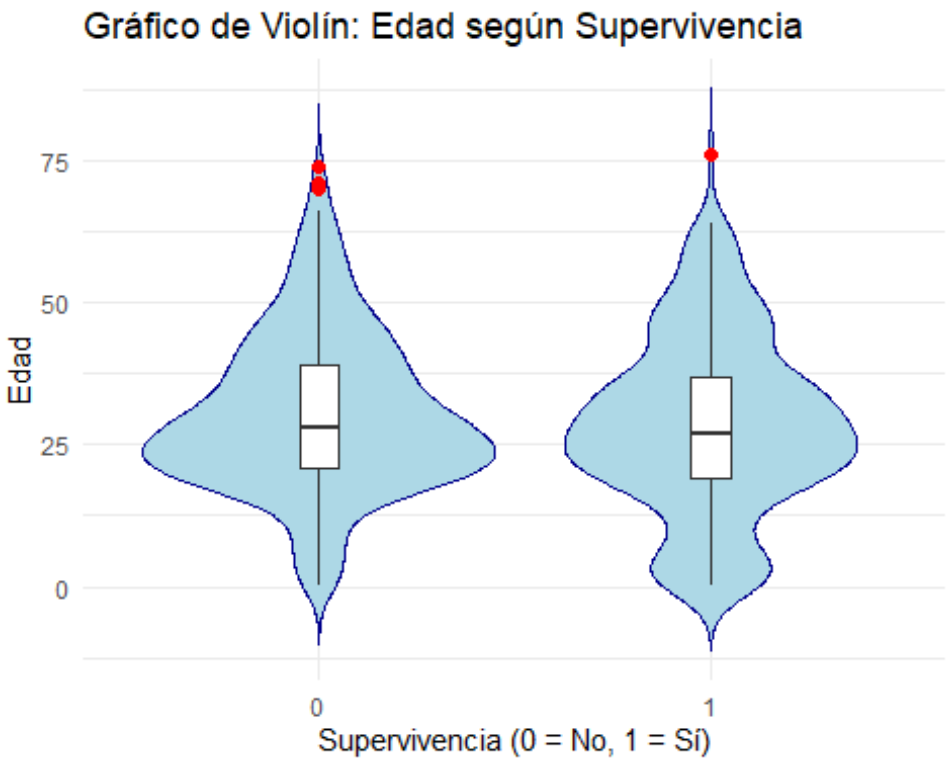
```r
# Instalar la librería si no está instalada
library(pROC)

## Warning: package 'pROC' was built under R version 4.3.2

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

##
## Attaching package: 'pROC'

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

# Crear el objeto ROC
roc_obj <- roc(train_data_clean$Survived, train_data_clean$Probabilidad)

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

## Setting direction: controls < cases

# Graficar la Curva ROC
plot(roc_obj, col = "blue", main = "Curva ROC", print.auc = TRUE)
```




```
# Mostrar el AUC (Área bajo la curva)
cat("Área bajo la curva (AUC):", auc(roc_obj), "\n")

## Área bajo la curva (AUC): 0.8846878
```

Elabora el gráfico de violín

```
# Gráfico de violín para Age según Survived
ggplot(train_data_clean, aes(x = factor(Survived), y = Age)) +
  geom_violin(trim = FALSE, fill = "lightblue", color = "darkblue") +
  geom_boxplot(width = 0.1, fill = "white", outlier.colour = "red",
outlier.size = 2) +
  labs(title = "Gráfico de Violín: Edad según Supervivencia",
       x = "Supervivencia (0 = No, 1 = Sí)",
       y = "Edad") +
  theme_minimal()
```



Concluye sobre el modelo basándote en las predicciones de los datos de entrenamiento.

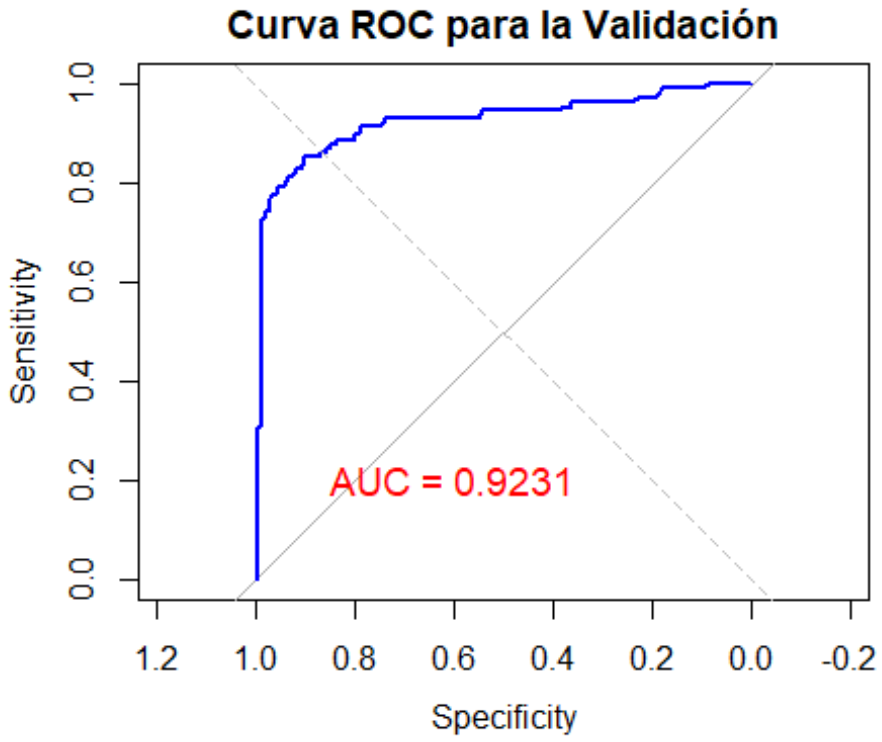
El modelo logístico muestra un desempeño sólido en los datos de entrenamiento, con un accuracy de 83.58% y un AUC de 0.8847, indicando una buena capacidad para distinguir entre las clases (supervivientes y no supervivientes). La matriz de confusión revela que el modelo clasifica correctamente la mayoría de las observaciones, aunque presenta 74 falsos negativos (supervivientes clasificados como no supervivientes) y 46 falsos positivos (no supervivientes clasificados como supervivientes). Estos resultados sugieren que el modelo es más preciso en identificar correctamente a los no supervivientes, pero podría mejorar en la detección de los supervivientes. En general, el modelo captura bien las relaciones clave entre las características y la probabilidad de supervivencia.

 # 5. Validación del modelo con la base de datos de validación

Elije un umbral de clasificación óptimo

```
# Calcular las probabilidades predichas para los datos de validación
validation_data_clean$Probabilidad <- predict(modelo_2, newdata =
validation_data_clean, type = "response")

library(pROC)

# Crear el objeto ROC
roc_obj <- roc(validation_data_clean$Survived,
validation_data_clean$Probabilidad)
```

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

## Setting direction: controls < cases

# Calcular el AUC
auc_value <- auc(roc_obj)

# Encontrar el umbral óptimo
optimal_coords <- coords(roc_obj, "best", ret = c("threshold",
"specificity", "sensitivity"))
optimal_threshold <- optimal_coords[["threshold"]]

# Graficar la Curva ROC con el valor de AUC
plot(roc_obj, col = "blue", main = "Curva ROC para la Validación")
text(0.6, 0.2, paste("AUC =", round(auc_value, 4)), col = "red", cex =
1.2)
abline(a = 0, b = 1, col = "gray", lty = 2)  # Línea de referencia
```




```
# Mostrar el umbral óptimo y el AUC
cat("Umbral óptimo:", optimal_threshold, "\n")

## Umbral óptimo: 0.5141739

cat("Área bajo la curva (AUC):", auc_value, "\n")

## Área bajo la curva (AUC): 0.9230639
```

Elabora la matriz de confusión con el umbral de clasificación óptimo

```r
# Clasificar usando el umbral óptimo
validation_data_clean$Prediccion <-
ifelse(validation_data_clean$Probabilidad > optimal_threshold, 1, 0)

# Crear la matriz de confusión
matriz_confusion_validacion <- table(Predicted =
validation_data_clean$Prediccion, Actual =
validation_data_clean$Survived)

# Mostrar la matriz de confusión
cat("Matriz de Confusión en la Validación:\n")

## Matriz de Confusión en la Validación:

print(matriz_confusion_validacion)

##          Actual
## Predicted   0   1
##         0 166  19
##         1  18 109

# Calcular el Accuracy
accuracy_validacion <- sum(diag(matriz_confusion_validacion)) /
sum(matriz_confusion_validacion)
cat("Accuracy en la Validación:", accuracy_validacion, "\n")

## Accuracy en la Validación: 0.8814103
```

 # 6. Elabora el testeo con la base de datos de prueba.

```r
# Calcular las probabilidades predichas para los datos de prueba
titanic_test_clean$Probabilidad <- predict(modelo_2, newdata =
titanic_test_clean, type = "response")

# Usar el umbral óptimo obtenido anteriormente
titanic_test_clean$Prediccion <- ifelse(titanic_test_clean$Probabilidad >
optimal_threshold, 1, 0)
```

 # 7. Concluye en el contexto del problema:

Define las principales características que influyen en el modelo seleccionado e interpretalas: ¿qué características tuvieron las personas que sobrevivieron?

Mayor probabilidad de ser mujeres: La probabilidad de supervivencia para las mujeres fue significativamente mayor debido a las políticas de evacuación prioritarias ("mujeres y niños primero"). Esto se alinea con el fuerte impacto del predictor Sex.

Pasajeros de clases altas: Los pasajeros de 1ª clase tenían acceso más rápido y directo a los botes salvavidas, además de encontrarse físicamente más cerca de las áreas de evacuación. Este privilegio contribuyó significativamente a su mayor probabilidad de supervivencia.

Personas jóvenes: Los niños y jóvenes tuvieron prioridad en los rescates, posiblemente debido a la percepción de vulnerabilidad y a las normas sociales de cuidado hacia los más pequeños.

Pasajeros sin familias numerosas: Las personas viajando solas o con pocos familiares tuvieron una mayor probabilidad de reaccionar rápidamente y salvarse, a diferencia de aquellos que intentaron salvar a múltiples familiares.

Las personas que sobrevivieron fueron, en su mayoría, **mujeres, pasajeros de 1ª clase, más jóvenes y con pocos familiares cercanos a bordo**. Estas características reflejan tanto factores estructurales (clase social) como decisiones de rescate basadas en normas sociales ("mujeres y niños primero").

Interpreta los coeficientes del modelo

El efecto de cada predictor en la clasificación puede interpretarse desde los coeficientes del modelo logístico. El sexo (Sex) tiene el impacto más fuerte: ser hombre (Sex=1) reduce drásticamente la probabilidad de supervivencia en comparación con ser mujer (Sex=0), con una disminución significativa en el logit de -3.438. Esto indica que el género es el factor más determinante en la clasificación. La clase del boleto (Pclass) también tiene un efecto importante: por cada aumento de clase (de 1ª a 3ª), el logit disminuye en -1.111, mostrando que los pasajeros en clases más bajas tienen menor probabilidad de sobrevivir. Ambos predictores son los más influyentes en la clasificación.

Otros predictores como la edad (Age) y el número de hermanos/cónyuges a bordo (SibSp) tienen efectos menores pero significativos. Por cada año adicional de edad, el logit disminuye ligeramente (-0.038), lo que sugiere que los pasajeros mayores tienen menor probabilidad de sobrevivir. Asimismo, tener más familiares cercanos abordo reduce la probabilidad de supervivencia, aunque el efecto es más moderado (-0.260 por cada unidad de SibSp). En resumen, mientras que todos los predictores contribuyen al modelo, el género y la clase social dominan la capacidad del modelo para clasificar correctamente.

Define cuál es el mejor umbral de clasificación y por qué

Umbral óptimo: 0.5141739

El umbral óptimo se determina porque es el valor que mejor balancea la sensibilidad (True Positive Rate, TPR) y la especificidad (True Negative Rate, TNR) del modelo. Este balance se logra a través de métodos como la Curva ROC, que evalúa la relación entre TPR y FPR (False Positive Rate).

En el contexto de la Curva ROC, el umbral óptimo es el punto más cercano a la esquina superior izquierda del gráfico. También sirve usar la Curva ROC ya que la predicción de ambas clases tienen la misma importancia por lo que los falsos positivos y falsos negativos tienen la misma importancia.