Actividad Integradora 2

Eryk Elizondo González A01284899

2024-11-19

Bibliotecas

```
# Cargamos todas las librería en la lista "librerias"
librerias =
c('tidyverse','broom','ISLR','GGally','modelr','cowplot','rlang','modelr','ti
bble','Metrics','mice','visdat',"caret")

for (lib in librerias){
  library(lib,character.only=TRUE)}
```

```
## ── Attaching core tidyverse packages ──────────────────────── tidyverse
2.0.0 ──
## ✓ dplyr     1.1.4     ✓ readr     2.1.5
## ✓ forcats   1.0.0     ✓ stringr   1.5.1
## ✓ ggplot2   3.5.1     ✓ tibble    3.2.1
## ✓ lubridate 1.9.3     ✓ tidyr     1.3.1
## ✓ purrr     1.0.2
## ── Conflicts ────────────────────────────────────────
tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all
conflicts to become errors
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
##
##
## Attaching package: 'modelr'
##
##
## The following object is masked from 'package:broom':
##
##     bootstrap
##
##
##
## Attaching package: 'cowplot'
##
##
## The following object is masked from 'package:lubridate':
```

```
##
##     stamp
##
##
##
## Attaching package: 'rlang'
##
##
## The following objects are masked from 'package:purrr':
##
##     %@%, flatten, flatten_chr, flatten_dbl, flatten_int, flatten_lgl,
##     flatten_raw, invoke, splice
##
##
##
## Attaching package: 'Metrics'
##
##
## The following object is masked from 'package:rlang':
##
##     ll
##
##
## The following objects are masked from 'package:modelr':
##
##     mae, mape, mse, rmse
##
##
##
## Attaching package: 'mice'
##
##
## The following object is masked from 'package:stats':
##
##     filter
##
##
## The following objects are masked from 'package:base':
##
##     cbind, rbind
##
##
## Loading required package: lattice
##
##
## Attaching package: 'caret'
##
##
## The following objects are masked from 'package:Metrics':
##
```

```
##     precision, recall
##
##
## The following object is masked from 'package:purrr':
##
##     lift
```

Leyendo los datos:

```
M <- read.csv("Titanic.csv")
str(M)
```

```
## 'data.frame':    1309 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" ...
```

Las variables son:

- *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)

Preparación de la base de datos

Ajustando las variables

Variables de interés: Quita aquellas que de entrada no tengan que ver con la sobrevivencia del pasajero. Por ejemplo: Quitar variables 4, 9 y 11 (define si hay más)

Variables categóricas que deben aparecer como factores: define qué variables aparecerán como factores Por ejemplo: Survived, Pclass, Sex y Embarked (define si hay más)

```
# Eliminar variables:
M1 <- M[,c(-4,-9,-11)]

#Transformar a factores:
for(var in c('Survived','Pclass','Embarked','Sex'))
  M1[,var] <-as.factor(M1[,var])
```

Análisis de datos faltantes

Detectar si hay espacios vacíos en lugar de datos:

```
V = matrix(NA,ncol=1,nrow=9)
for(i in c(1:9)){
  V[i,] <- sum(with(M1,M1[,i])=="")}
V
```

```
  0
  0
  0
  0
NA
  0
  0
NA
```

NA

Ninguna variable contiene espacios vacíos, pero las variables 5 (Age), 8 (Fare) y 9 (Embarked) tienen datos faltantes.

Para contar los datos faltantes:

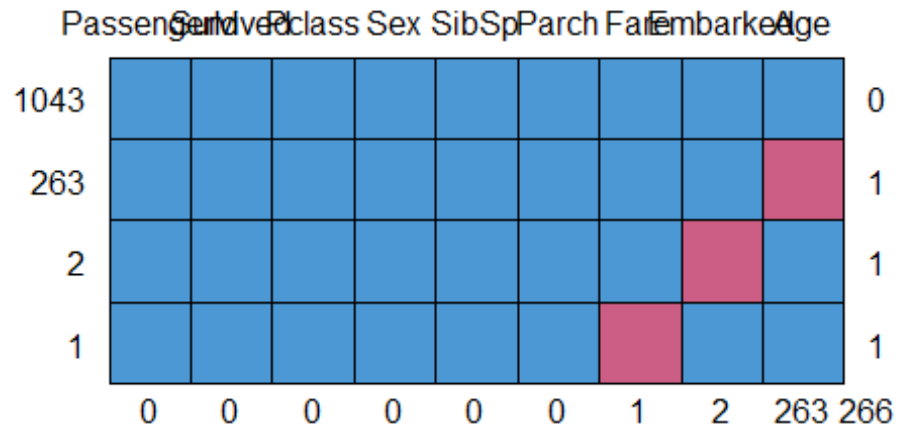
```
N = apply(X=is.na(M1),MARGIN = 2,FUN = sum)
P = round(100*N/length(M1[,2]),2)
NP = data.frame(as.numeric(N),as.numeric(P))
row.names(NP)= c("PassengerId", "Survived", "Pclass", "Sex", "Age", "SibSp",
"Parch", "Fare", "Embarked")
names(NP)=c("Número","Porcentaje")
t(NP)
```

	PassengerId	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
Número	0	0	0	0	263.00	0	0	1.00	2.00
Porcentaje	0	0	0	0	20.09	0	0	0.08	0.15

En edad hay muchos datos faltantes, el 20% de los datos.

Observemos el patrón de los datos faltantes:

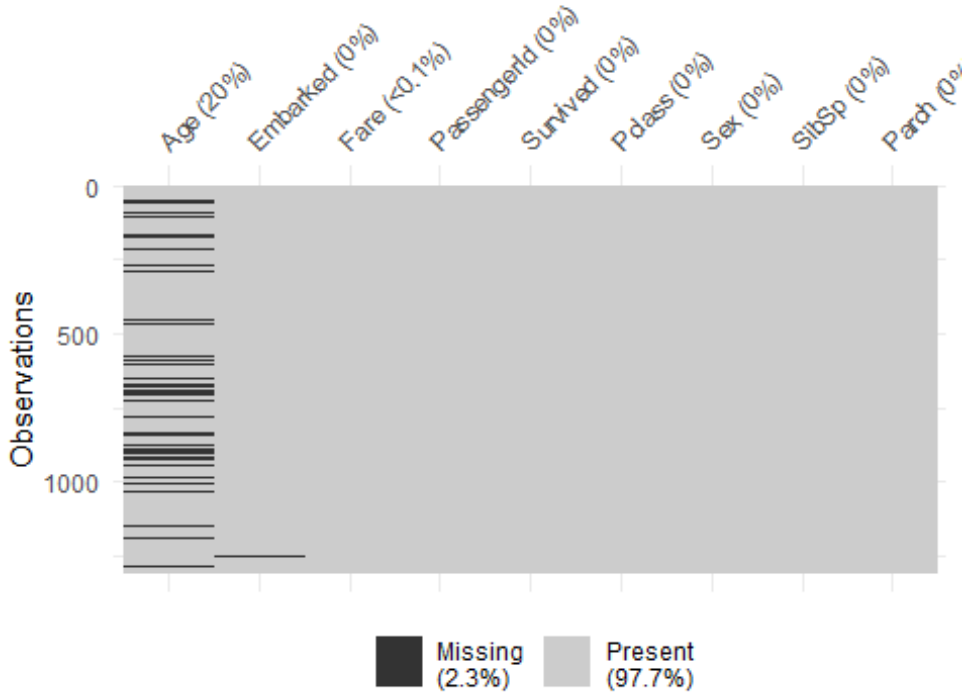
```
md.pattern(M1)
```



	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

Todos los datos faltantes son de distintos pasajeros (observaciones), por lo tanto, si se eliminan los NA, se eliminarían 266 observaciones y nos quedaríamos con 1043 observaciones.

```
vis_miss(M1,sort_miss = TRUE)
```



Análisis sobre datos faltantes

Medidas con datos faltantes

```
summary(M1[,-1])
```

Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0:815	1:323	female:466	Min. : 0.17	Min. :0.0000	Min. :0.000	Min. : 0.000	C :270
1:494	2:277	male :843	1st Qu.:21.00	1st Qu.:0.0000	1st Qu.:0.000	1st Qu.: 7.896	Q :123
NA	3:709	NA	Median :28.00	Median :0.0000	Median :0.000	Median : 14.454	S :914
NA	NA	NA	Mean :29.88	Mean :0.4989	Mean :0.385	Mean : 33.295	NA's: 2
NA	NA	NA	3rd Qu.:39.00	3rd Qu.:1.0000	3rd Qu.:0.000	3rd Qu.: 31.275	NA
NA	NA	NA	Max. :80.00	Max. :8.0000	Max. :9.000	Max. :512.329	NA
NA	NA	NA	NA's :263	NA	NA	NA's :1	NA

Medidas sin datos faltantes

```
M2 = na.omit(M1)
summary(M2[,-1])
```

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

¿Difieren las medidas con o sin datos faltantes?

Si difieren las medidas sin datos faltantes, por ejemplo, la media de la edad disminuye ligeramente de 29.88 a 29.81, la media de SibSP sube de 0.49 a 0.5, Parch sube de 0.385 a 0.4219 y Fare sube de 33.295 a 36.60.

¿cuáles son las variables que más se ven afectadas?

Vemos que las variables más afectadas son Fare y Parch, mientras que Age y SibSp no varían tanto.

Sobrevivientes

```
t2c = 100*prop.table(table(M1[,2]))
t2s = 100*prop.table(table(M2[,2]))
t2p = c(t2s[1]/t2c[1],t2s[2]/t2c[2])
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(M1[,3]))
t3s = 100*prop.table(table(M2[,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(M1[,4]))
t4s = 100*prop.table(table(M2[,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(M1[,9]))
t9s = 100*prop.table(table(M2[,9]))
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

Vemos que al remover los NAs aumenta la proporción de Sobrevivientes mientras que la proporción de Fallecidos disminuye. En Clase, aumenta la proporción de la Primera y Segunda clase mientras que disminuye en la Tercera. En Sexo aumenta la proporción de Mujer y disminuye en Hombre. En puerto de embarcación, se pierde la mitad de las

embarcaciones provenientes de Queenstown, disminuye ligeramente la proporción de Cherbourg y aumenta en Southampton.

Partición. Entrenamiento y prueba

Se toma el 70% de la muestra como entrenamiento y el 30% para prueba.

```
M_indice <- createDataPartition(M2$Survived, p = .7, list = FALSE, times = 1)

M_train <- M2[ M_indice,]  %>% as_tibble()
M_valid <- M2[-M_indice,] %>% as_tibble()
```

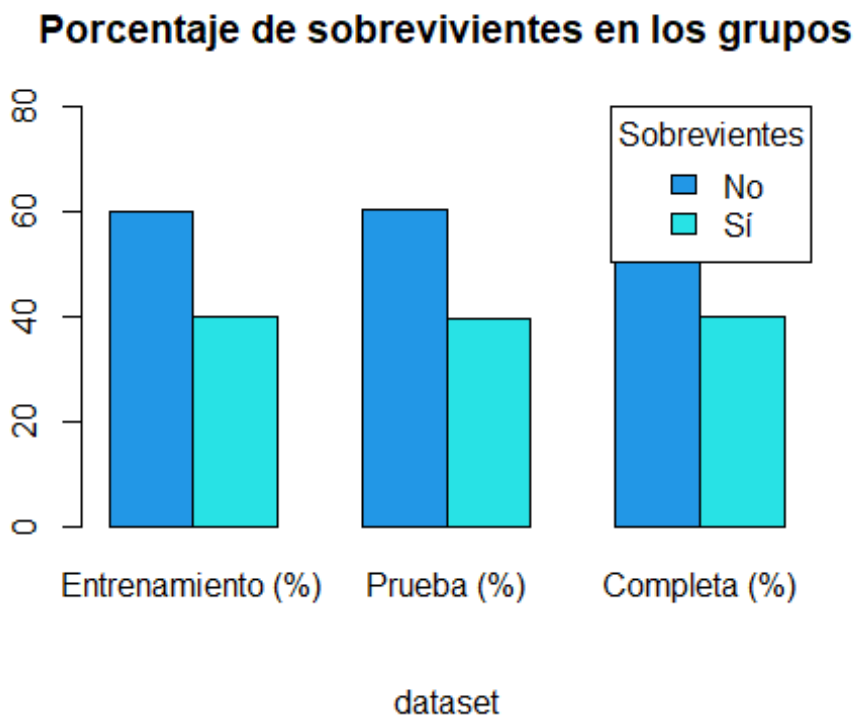
Proporciones de sobrevivientes en las tres bases de datos

- Calcula la proporción de sobrevivientes en cada base de datos: Entrenamiento, prueba y completa. Haz una tabla comparativa
- Haz un gráfico de barras que te ayude a comparar las tres bases de datos. Auxíliate del código:

```
pst = 100*prop.table(table(M_train[,2]))
psv = 100*prop.table(table(M_valid[,2]))
psf = 100*prop.table(table(M2[,2]))
ps3 = data.frame(as.numeric(pst),as.numeric(psv),as.numeric(psf))

row.names(ps3) = c("Murió","Sobrevivió")
names(ps3) = c("Entrenamiento (%)","Prueba (%)","Completa (%)")
round(ps3,2)
```

	Entrenamiento (%)	Prueba (%)	Completa (%)
Murió	60.19	60.26	60.21
Sobrevivió	39.81	39.74	39.79

```
barplot(as.matrix(ps3), col=4:5, beside=TRUE, main="Porcentaje de
sobrevivientes en los grupos", sub="dataset",ylim=c(0,80))
legend("topright",legend = c("No","Sí"), title = "Sobrevientes",fill = 4:5)
```



Define si la proporción de no sobrevientes se mantiene en las tres bases de datos.

Vemos que la proporción de no sobrevivientes se mantiene en las 3 bases de datos con una variación mínima de 0.05.

Modelación (entrenamiento)

Comienza con el modelo completo, incluyendo las variables categóricas (factores). Aplica el comando *step* para poder encontrar el mejor modelo.

step utiliza el criterio de Aikaike (AIC) para definir el mejor modelo, sin embargo también proporciona la desviación residual del modelo completo. Un menor AIC y una menor *Deviance* indicarán un mejor modelo.

```
A = glm(Survived ~ ., data = M_train, family = "binomial")

step(A, direction="both", trace=1 )

## Start:  AIC=563.77
## Survived ~ PassengerId + Pclass + Sex + Age + SibSp + Parch +
##     Fare + Embarked
##
##               Df Deviance    AIC
## - Embarked     2   542.68 560.68
## - Fare         1   541.79 561.79
## - PassengerId  1   541.89 561.89
```

```
## - Parch        1   542.49 562.49
## - SibSp        1   543.65 563.65
## <none>             541.77 563.77
## - Age          1   568.77 588.77
## - Pclass       2   590.51 608.51
## - Sex          1   863.87 883.87
##
## Step:  AIC=560.68
## Survived ~ PassengerId + Pclass + Sex + Age + SibSp + Parch +
##     Fare
##
##               Df Deviance    AIC
## - Fare         1   542.77 558.77
## - PassengerId  1   542.81 558.81
## - Parch        1   543.38 559.38
## - SibSp        1   544.66 560.66
## <none>             542.68 560.68
## + Embarked     2   541.77 563.77
## - Age          1   570.71 586.71
## - Pclass       2   595.37 609.37
## - Sex          1   867.18 883.18
##
## Step:  AIC=558.77
## Survived ~ PassengerId + Pclass + Sex + Age + SibSp + Parch
##
##               Df Deviance    AIC
## - PassengerId  1   542.90 556.90
## - Parch        1   543.38 557.38
## - SibSp        1   544.70 558.70
## <none>             542.77 558.77
## + Fare         1   542.68 560.68
## + Embarked     2   541.79 561.79
## - Age          1   571.14 585.14
## - Pclass       2   624.61 636.61
## - Sex          1   870.39 884.39
##
## Step:  AIC=556.9
## Survived ~ Pclass + Sex + Age + SibSp + Parch
##
##               Df Deviance    AIC
## - Parch        1   543.51 555.51
## - SibSp        1   544.89 556.89
## <none>             542.90 556.90
## + PassengerId  1   542.77 558.77
## + Fare         1   542.81 558.81
## + Embarked     2   541.91 559.91
## - Age          1   571.23 583.23
## - Pclass       2   624.98 634.98
## - Sex          1   870.64 882.64
##
```

```
## Step:  AIC=555.51
## Survived ~ Pclass + Sex + Age + SibSp
##
##               Df Deviance    AIC
## <none>             543.51 555.51
## - SibSp        1   546.33 556.33
## + Parch        1   542.90 556.90
## + PassengerId  1   543.38 557.38
## + Fare         1   543.51 557.51
## + Embarked     2   542.62 558.62
## - Age          1   571.98 581.98
## - Pclass       2   626.26 634.26
## - Sex          1   877.79 887.79

##
## Call:  glm(formula = Survived ~ Pclass + Sex + Age + SibSp, family =
"binomial",
##     data = M_train)
##
## Coefficients:
## (Intercept)      Pclass2      Pclass3      Sexmale          Age
SibSp
##     4.85709     -1.43767     -2.66513     -3.64604     -0.04619     -
0.22224
##
## Degrees of Freedom: 730 Total (i.e. Null);  725 Residual
## Null Deviance:       982.8
## Residual Deviance: 543.5     AIC: 555.5
```

- Identifica el mejor modelo de acuerdo con el AIC

El mejor modelo es Survived ~ Pclass + Sex + Age + SibSp

- Selecciona la última variable que elminó el comando *step*. Prueba dos modelos, uno con esa variable y otro sin ella.

Survived ~ Pclass + Sex + Age + SibSp + Fare

Modelo B

- Prueba el modelo incluyendo la última variable que eliminó el comando *step*.
- Indica cuáles son las variables que incluye.
- Interpreta la significancia global (de todo el modelo) y la individual (de cada una de las variables)

```
B = glm(formula = Survived ~ Pclass + Sex + Age + SibSp + Fare, family =
"binomial", data = M_train)
summary(B)

##
## Call:
```

```
## glm(formula = Survived ~ Pclass + Sex + Age + SibSp + Fare, family =
"binomial",
##     data = M_train)
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)  4.8405125  0.5593656   8.654  < 2e-16 ***
## Pclass2     -1.4266243  0.3529450  -4.042 5.30e-05 ***
## Pclass3     -2.6518459  0.3714576  -7.139 9.40e-13 ***
## Sexmale     -3.6441437  0.2484379 -14.668  < 2e-16 ***
## Age         -0.0461486  0.0090480  -5.100 3.39e-07 ***
## SibSp       -0.2235388  0.1349372  -1.657   0.0976 .
## Fare         0.0001743  0.0024766   0.070   0.9439
## ---
## 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: 543.51  on 724  degrees of freedom
## AIC: 557.51
##
## Number of Fisher Scoring iterations: 5
```

Las variables que se incluyen son Pclass, Sex, Age, SibSp y Fare.

Se observa que todas las variables, excepto la última variable que es la que fue eliminada por el comando de Step son significativas lo que indica que la recomendación de eliminar la última variable parece ser correcta.

Modelo C

- Prueba el modelo tal como te lo recomendó el comando *step*.
- Indica cuáles son las variables que incluye.
- Interpreta la significancia global (de todo el modelo) y la individual (de cada una de las variables)

```
C = glm(formula = Survived ~ Pclass + Sex + Age + SibSp, family = "binomial",
data = M_train)
summary(C)
```

```
##
## Call:
## glm(formula = Survived ~ Pclass + Sex + Age + SibSp, family = "binomial",
##     data = M_train)
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)  4.857088   0.507610   9.569  < 2e-16 ***
## Pclass2     -1.437669   0.316190  -4.547 5.45e-06 ***
```

```
## Pclass3     -2.665133   0.320046  -8.327  < 2e-16 ***
## Sexmale     -3.646043   0.247006 -14.761  < 2e-16 ***
## Age         -0.046186   0.009033  -5.113 3.17e-07 ***
## SibSp       -0.222242   0.133683  -1.662   0.0964 .
## ---
## 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: 543.51  on 725  degrees of freedom
## AIC: 555.51
##
## Number of Fisher Scoring iterations: 5
```

Las variables que se incluyen son Pclass, Sex, Age y SibSp.

Se observa que todas las variables son significativas y el modelo posee un menor AIC que el anterior, entonces podemos concluir que la recomendación del Step de eliminar la última variable (Fare) es correcta.

Análisis de los modelos B y C

Resumen de los indicadores importantes de los modelos B y C

Compara el AIC, la *Null Deviance* y la *Residual Deviance* de los modelos B y C. Extrae los valores con los modelos con los comandos:

- B$aic
- B$deviance
- B$null.deviance

Elabora una tabla comparativa

```
AIC1 = data.frame(as.numeric(
B$aic),as.numeric(B$deviance),as.numeric(B$null.deviance))
row.names(AIC1) = c("Modelo B")
names(AIC1) = c("AIC","Residual Deviance","Null Deviance")
round(AIC1,2)
```

	AIC	Residual Deviance	Null Deviance
Modelo B	557.51	543.51	982.8

```
AIC2 = data.frame(as.numeric(
C$aic),as.numeric(C$deviance),as.numeric(C$null.deviance))
row.names(AIC2) = c("Modelo C")
names(AIC2) = c("AIC","Residual Deviance","Null Deviance")
round(AIC2,2)
```

	AIC	Residual Deviance	Null Deviance

	AIC	Residual Deviance	Null Deviance
Modelo C	555.51	543.51	982.8

```
rbind(AIC1, AIC2)
```

	AIC	Residual Deviance	Null Deviance
Modelo B	557.5079	543.5079	982.7966
Modelo C	555.5128	543.5128	982.7966

¿Cómo se comporta la *Null Deviance*? ¿por qué?

Vemos que se comporta igual en ambos modelos porque la desviación nula se calcula solo con la variable de intercepto la cual es similar en ambos modelos.

¿Qué pasa con el AIC y la *Residual Deviance*?

Vemos que el AIC disminuye con el Modelo C pero su desviación residual aumenta.

Cálculo de la Desviación explicada ($pseudor^2$)

Calcula la desviación explicada para cada modelo. Recuerda que es igual a:

pseudo r^2 = 1-Desviación residual/Desviación nula

```
pseudoB <- 1 - as.numeric(B$deviance)/as.numeric(B$null.deviance)
pseudoB

## [1] 0.4469783

pseudoC <- 1 - as.numeric(C$deviance)/as.numeric(C$null.deviance)
pseudoC

## [1] 0.4469732
```

Compara los resultados obtenidos por ambos modelos

Vemos que la desviación explicada es muy similar en ambos modelos pero es lijeramente menor en el modelo C.

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)

Se calcula el estadístico de χ^2 para la razón de verosimilutud a partir de las *Deviance* de los modelos.

```r
Diferencia = as.numeric(B$null.deviance)-as.numeric(B$deviance)

gl = as.numeric(B$df.null) - as.numeric(B$df.residual)

pchisq(Diferencia,gl,lower.tail = FALSE)
```

```
## [1] 9.909065e-92
```

```r
DiferenciaC = as.numeric(C$null.deviance)-as.numeric(C$deviance)

glC = as.numeric(C$df.null) - as.numeric(C$df.residual)

pchisq(DiferenciaC,glC,lower.tail = FALSE)
```

```
## [1] 1.006108e-92
```

Vemos que en ambos rechazamos la hipotesis nula por la alternativa, entonces el modelo nulo explica mejor la variable respuesta.

Interpreta en el contexto del problema

Ambos modelos con sus respectivas variables ayudan a predecir.

Comparación entre los modelos B y C

Se pueden comparar los modelo B y C para ver si hay una diferencia significativa entre ambos con la misma razón de verosimilitud utilizando el comando ANOVA y la prueba LR.

```r
library(car)
```

```
## Loading required package: carData
```

```
##
## Attaching package: 'car'
```

```
## The following object is masked from 'package:dplyr':
##
##     recode
```

```
## The following object is masked from 'package:purrr':
##
##     some
```

```r
anova(B,C,test="LR")
```

Resid. Df	Resid. Dev	Df	Deviance	Pr(>Chi)
724	543.5079	NA	NA	NA
725	543.5128	-1	-0.0049621	0.9438418

Con esto y principalmente porque en modelo B tiene una variable no significativa, el mejor modelo es el C el cual se desplegará subsecuentemente:

Modelo Seleccionado

```
c0 = round(C$coefficients[1],3)
c1 = round(C$coefficients[2],3)
c2 = round(C$coefficients[3],3)
c3 = round(C$coefficients[4],4)
c4 = round(C$coefficients[5],5)
c5 = round(C$coefficients[6],6)

cat("logit(Pr(Y)) =", c0, c1, "* Pclass2", c2, "* Pclass3", c3, "* Sex", c4,
"* Age", c5, "* SibSp")

## logit(Pr(Y)) = 4.857 -1.438 * Pclass2 -2.665 * Pclass3 -3.646 * Sex -
0.04619 * Age -0.222242 * SibSp
```

$\text{logit}(\Pr(Y)) =c0} + 0.05810 \cdot \text{Lag2}$ ### Gráfica el modelo

```
C

##
## Call:  glm(formula = Survived ~ Pclass + Sex + Age + SibSp, family =
"binomial",
##     data = M_train)
##
## Coefficients:
## (Intercept)      Pclass2      Pclass3      Sexmale          Age
SibSp
##     4.85709     -1.43767     -2.66513     -3.64604     -0.04619     -
0.22224
##
## Degrees of Freedom: 730 Total (i.e. Null);  725 Residual
## Null Deviance:       982.8
## Residual Deviance: 543.5     AIC: 555.5
```

Para percibir el efecto de cada variable, grafica cada variable contra los valores predichos por el modelo. Aunque en el modelo, la variable respuesta es:

$$\hat{y} = log\left(\frac{p}{1-p}\right)$$

con el subcomando: *fitted.values* del comando *glm* se obtienen las probabilidades estimadas para los valores datos. R despeja las probabilidades:

$$\hat{p} = \left(\frac{e^{\hat{y}}}{1+e^{\hat{y}}}\right)$$

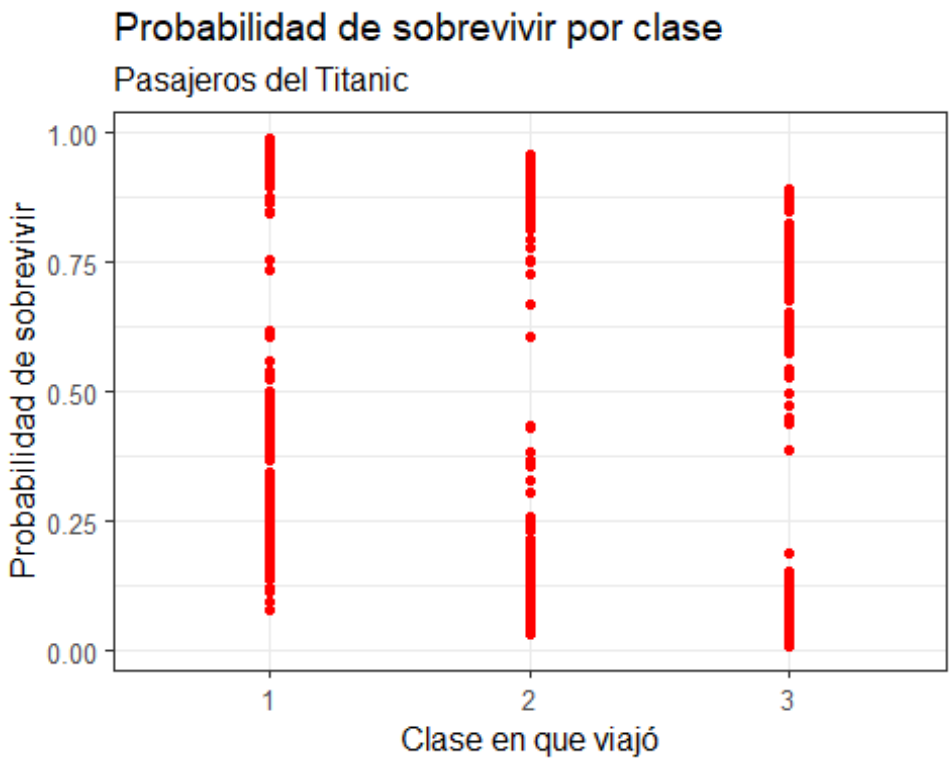
Así que interpretar el efecto de cada variable, se grafica cada una de ellas contra los valores predichos para la probabilidad de sobrevivencia.

Para hacer los gráficos se ejemplifica con:

Clase en que viajó el pasajero

```
p_pred = C$fitted.values
M_pred = data.frame(M_train[,c(2,3,4,5,6)],p_pred)

ggplot(M_pred, aes( x = Pclass)) +
geom_point(aes(y=M_pred$p_pred), size=1.5,color="red") +
  labs(x="Clase en que viajó", y="Probabilidad de sobrevivir",
       title="Probabilidad de sobrevivir por clase",
       subtitle="Pasajeros del Titanic",
       col="")+
  theme_bw(base_size = 12)
```

```
## Warning: Use of `M_pred$p_pred` is discouraged.
## ℹ Use `p_pred` instead.
```



Grafica y concluye cómo cambia la probabilidad predicha con cada variable que resultó significativa

Predicciones

Se hace el análisis con el modelo seleccionado, en el ejemplo suponemos que se seleccionó el modelo C.

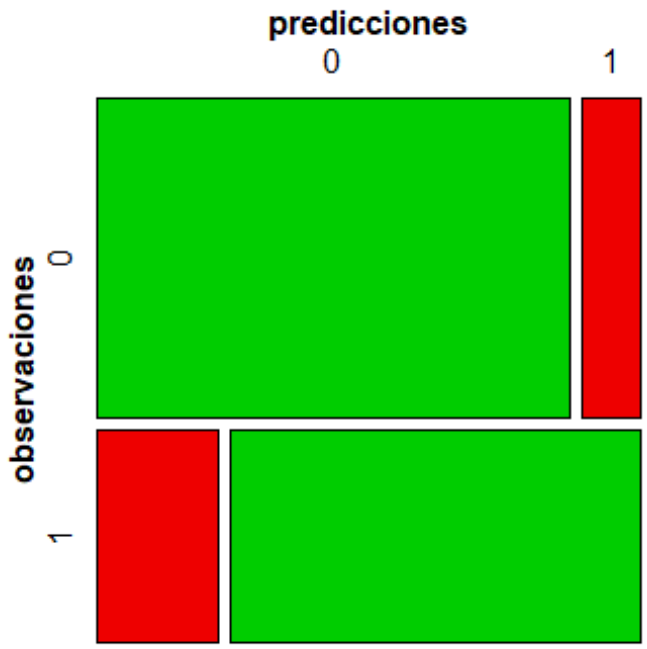
Matriz de confusión

```
library(vcd)
```

```
## Loading required package: grid

##
## Attaching package: 'vcd'

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

predicciones <- ifelse(test = C$fitted.values > 0.5, yes = 1, no = 0)
M_C <- table(C$model$Survived, predicciones, dnn = c("observaciones",
"predicciones"))
M_C
```

observaciones/predicciones	0	1
0	392	48
1	66	225

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




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

## La Exactitud (accuracy) del modelo es 0.8440492

Se = M_C[1,1]/sum(M_C[1,])
cat("La Sensibilidad del modelo es", Se,"\n")
```

```
## La Sensibilidad del modelo es 0.8909091
```

```
Sp = M_C[2,2]/sum(M_C[2,])
cat("La Especificidad del modelo es", Sp,"\n")
```

```
## La Especificidad del modelo es 0.7731959
```

```
P = M_C[1,1]/sum(M_C[,1])
cat("La Precisión del modelo es", P,"\n")
```

```
## La Precisión del modelo es 0.8558952
```

Define si el modelo es bueno o no.

Si, el modelo es bueno con valores altos en todas sus metricas.

Curva ROC

Para hacer la curva, es necesario crear las predicciones para el data set de entrenamiento. El comando *roc* cálculará la sensibilidad y la especificidad para los datos obtenidos.

```
pred = predict(C, 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
```

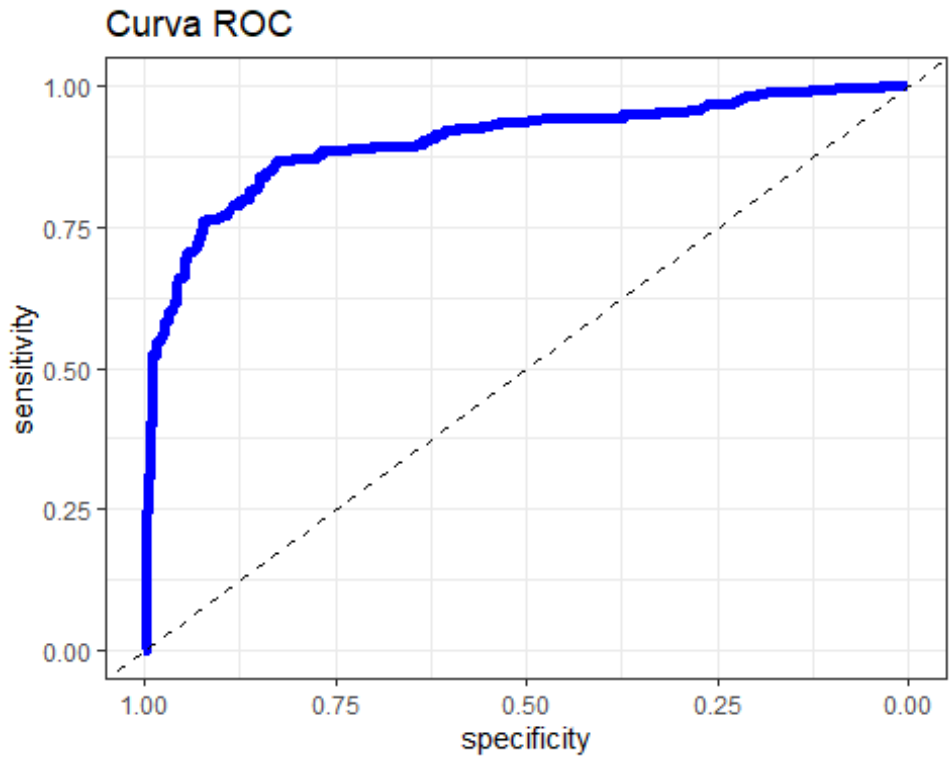
```
ROC <- roc(response=M_train$Survived, predictor=pred)
```

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

## Setting direction: controls < cases
```

```
ROC
```

```
##
## Call:
## roc.default(response = M_train$Survived, predictor = pred)
##
## Data: pred in 440 controls (M_train$Survived 0) < 291 cases
(M_train$Survived 1).
## Area under the curve: 0.9006
```

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



Nota: Se grafica Especificidad, pero en realidad se está graficando 1 - Especificidad.

Interpreta el gráfico y la salida que da el comando *roc*

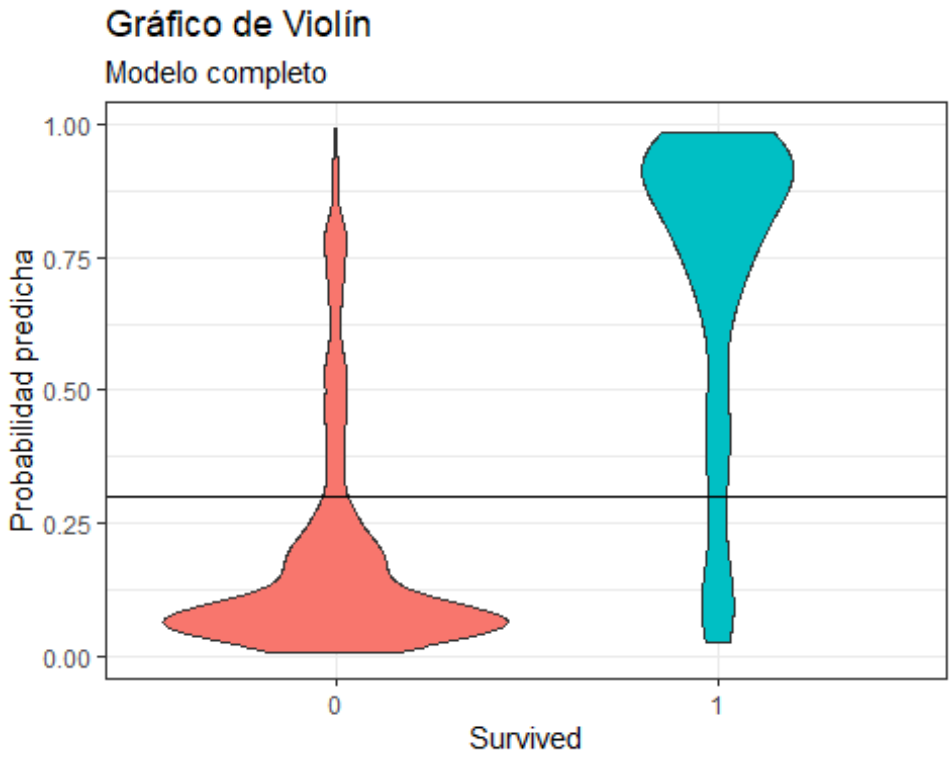
Vemos que el punto óptimo es 0.8936 y como es positivo es mejor ya que en el caso perfecto seria 1, 1. Como se encuentra arriba de la línea es un modelo bueno con buenas metricas.

Gráfico de violín

Se crea la base de datos para el gráfico, se usan las predicciones ya elaboradas para el gráfico ROC y las clasificaciones originales (*train$M_Survived*).

```
v_d = data.frame(Survived=M_train$Survived,pred=pred)

ggplot(data=v_d, 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.
```



Interpreta

Vemos que la distribución de los datos entre survived y no se encuentra posiscionada correctamente donde la mayoria de los datos debajo del ubral pertenecientes de no survived y lla mayoria de survived arriba, lo que nos indica que las predicciones son correctas.

Validación

Elección de un umbral de clasificación optimo.

Elección del umbral de clasificación (punto de corte)

Se trabaja con la base de datos de validación (*M_valid*) y se realiza el gráfico de la Exactitud, Sensibilidad, Especificidad y Precisión para distintos valores del umbral de clasificación. Se siguen los siguientes pasos:

1. Predicción en los datos de validación con el modelo elegido (en el ejemplo, el C)

2. Se definen los umbrales de clasificación: irán desde 0.05 hasta 0.95.
3. Se definen las métricas de la matriz de confusión para cada umbral de clasificación
4. Se prepara el conjunto de datos: se quitan los NA y se agrega la columna de umbrales de clasificaición
5. Se le da un formato a la base de datos para que pueda ser graficada más fácilmente.

Generación de base de datos para graficar

```
pred_val = predict(C, newdata=M_valid, type='response')
clase_real = M_valid$Survived

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: 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])
}

## Se limpia el conjunto de datos
datosV = na.omit(datosV)
datosV$umbral = seq(0.05,0.95,0.01)
```

Formato de datos

- Se crea la variable *métrica* que será una variable categórica para las métricas (Exactitud, Sensibilidad, Especificidad y Precisión)
- Los valores de las métricas se ponen en una sola columna.

- Se identifican las métricas para los distintos umbrales con la variable ‘umbral’.

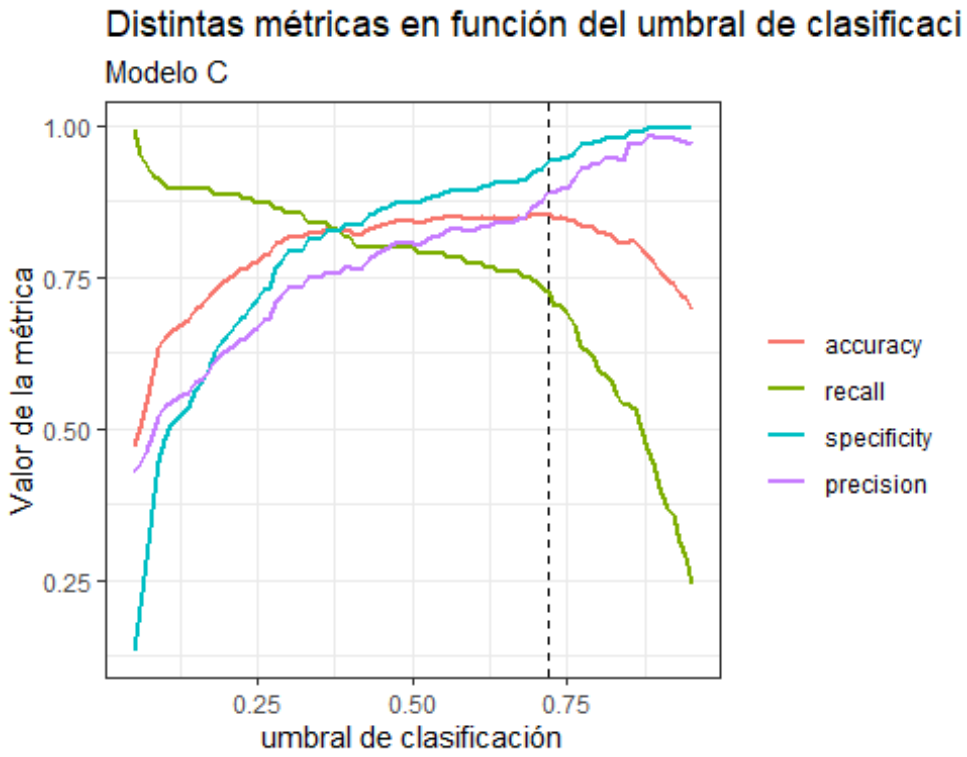
```
library(reshape2)

##
## Attaching package: 'reshape2'
```

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

datosV_m <- reshape2::melt(datosV,id.vars=c('umbral'))
colnames(datosV_m)[2] <- c('Metrica')
```

Gráfica

En la gráfica se define cuál es el mejor umbral de clasificación dependiendo de cuál métrica es más importante en el contexto del problema (Exactitud, Sensibilidad, Especificidad o Precisión). Si no hay una métrica de preferencia, se opta por escoger el máximo valor de que pueden tener estás métricas en conjunto. En cualquier caso da valores a u para mover el umbral de clasificación y observar como se comporta con respecto a las métricas.

```
library(ggplot2)

u = 0.72 #Se dio un valor arbitrario, tú modificalo de acuerdo al criterio
que selecciones.

ggplot(data=datosV_m, aes(x=umbral,y=value,color=Metrica)) +
geom_line(size=1) + theme_bw() +
  labs(title= 'Distintas métricas en función del umbral de clasificación',
       subtitle= 'Modelo C',
       color="", x = 'umbral de clasificación', y = 'Valor de la métrica') +
  geom_vline(xintercept=u, linetype="dashed", color = "black")

## 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.
```



Define cuál es el mejor umbral en donde se obtienen las mejores métricas Recall, Accuracy, Sensitivity y Specificity.

El mejor umbral es 0.72 ya que es el punto donde todas las metricas se maximisan entre si.

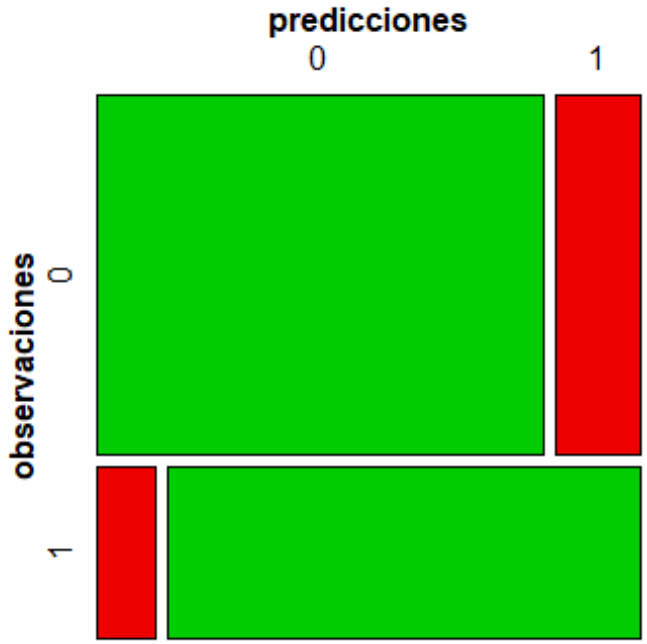
Matriz de confusión con el umbral de clasificación optimo

De acuerdo al umbral seleccionado, calcula la matriz de confusión y las métricas obtenidas. Indica si mejora la predicción con respecto al umbral de u = 0.5, que es el que se maneja por default.

```
prediccionesV = ifelse(pred_val > 0.72, yes = 1, no = 0)
M_Cv <- table(prediccionesV, M_valid$Survived, dnn = c("observaciones",
"predicciones"))
M_Cv
```

observaciones/predicciones	0	1
0	177	34
1	11	90

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




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

## La Exactitud (accuracy) del modelo es 0.8557692

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

## La Sensibilidad del modelo es 0.8388626

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

## La Especificidad del modelo es 0.8910891

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

## La Precisión del modelo es 0.9414894
```

Si comparamos un umbral de 0.5 vs el propuesto que es 0.72 se dan mejores métricas.

Conclusiones

Concluye definiendo cuáles fueron las principales características de las personas que sobrevivieron e indica cuáles son los coeficientes de cada variable en el modelo de

predicción de sobrevivencia. Interpreta los coeficientes de predicción de cada variable. Indica cómo influyó en la sobrevivencia.

Las principales caracteristucas de las personas que sobrevivieron fueron su sexo, donde el ser hombre disminuye las probabilidades con una pendiente de -3.6. También, La clase, donde la tercera clase tiene una reducción de sobrevivencia con una pendiente de -2.5 y de segunda clase con una pendiente de -1.5. El resto de las variables no tienen un peso significativo.

Indica cuál es el mejor umbral de clasificación y por qué.

El mejor umbral de clasificación es 0.72 el cual es mejor que el default de 0.5 lo cual se respalda por la grafica de violin donde al definir un umbral de 0.5 no se capta la suficiente información para la clase de no sobrevivencia.

Nota: Lo entregado fue lo realizado durante el tiempo del examen sin modificaciones extras lo cual puede que hayan varios errores en el documento pero para mantener el avance durante el examen se entrega sin modificaciones extras.