Entrega 2

Bruno Olivera & Leandro Burastero

13/10/2019

Parte 1 - Laboratorio con variables simuladas

Sean X_1 y X_2 dos variables uniformes en [-4, 5] e Y una variable que se quiere predecir a partir de ellas.

1.a) Simulación de datos

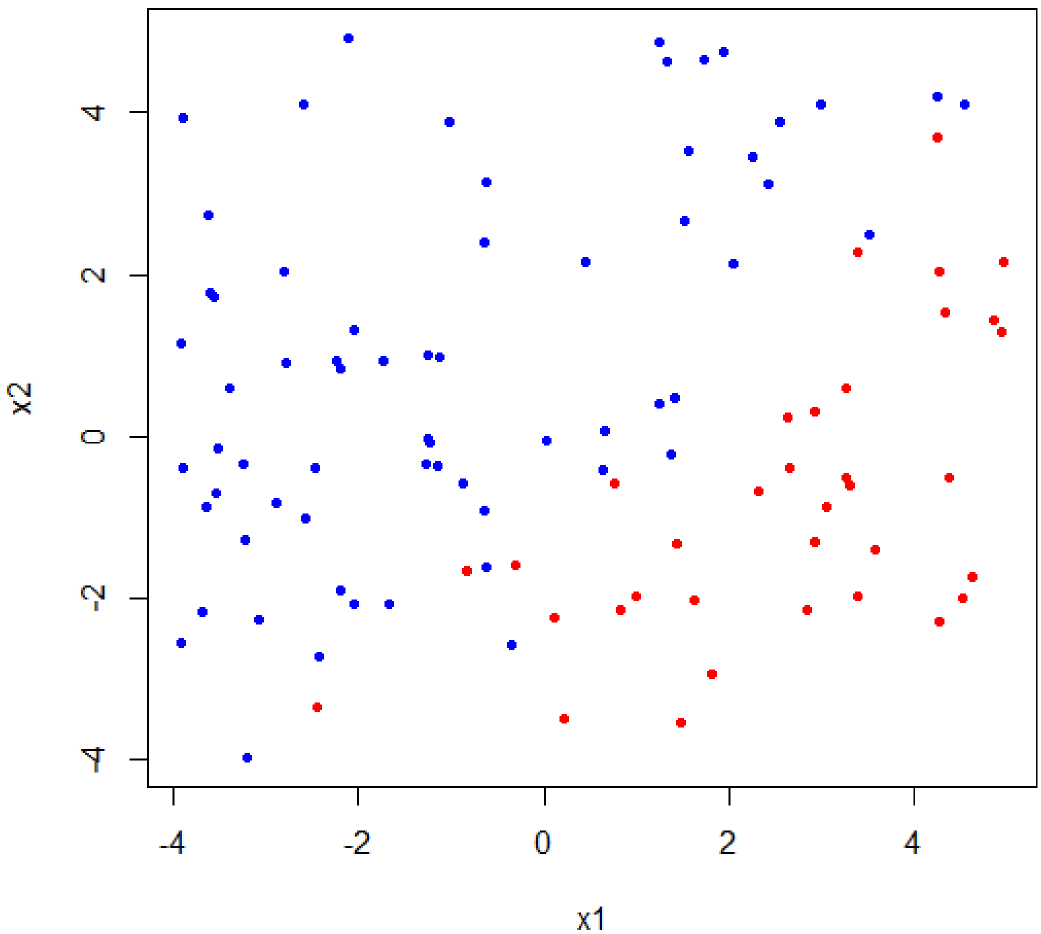
Simular una relación entre Y y (X_1, X_2).

```
set.seed(2019)

n=100
a=-2
b=2
c=3
x1=runif(n,-4,5)
x2=runif(n,-4,5)
y=exp(a*x1+b*x2+c + rnorm(n))
y=y/(1+y)
y=rbinom(n,1,y)
```

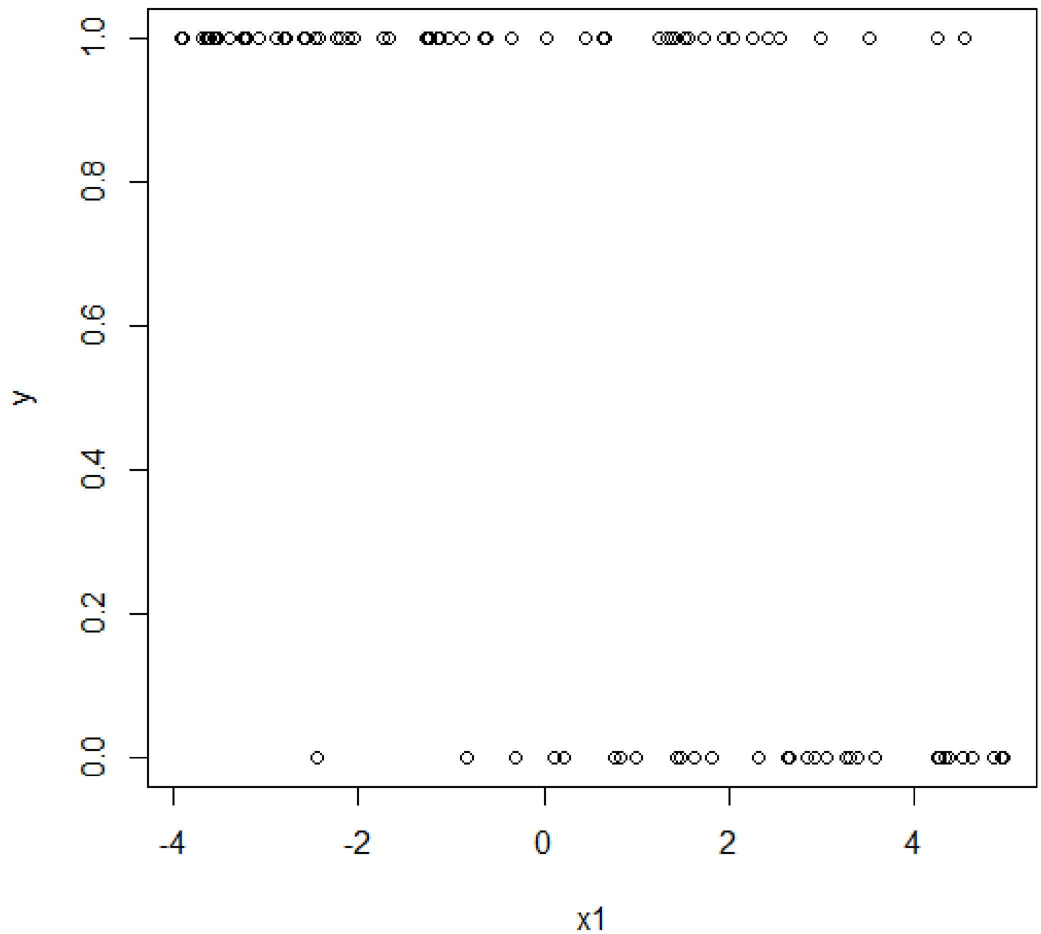
1.b) Graficar relaciones entre variables

Representar graficamente la nube de puntos formada por las variables explicativas, representando los puntos con colores distintos según la modalidad de Y. Representar Y en función de X_1 e Y en función de X_2.

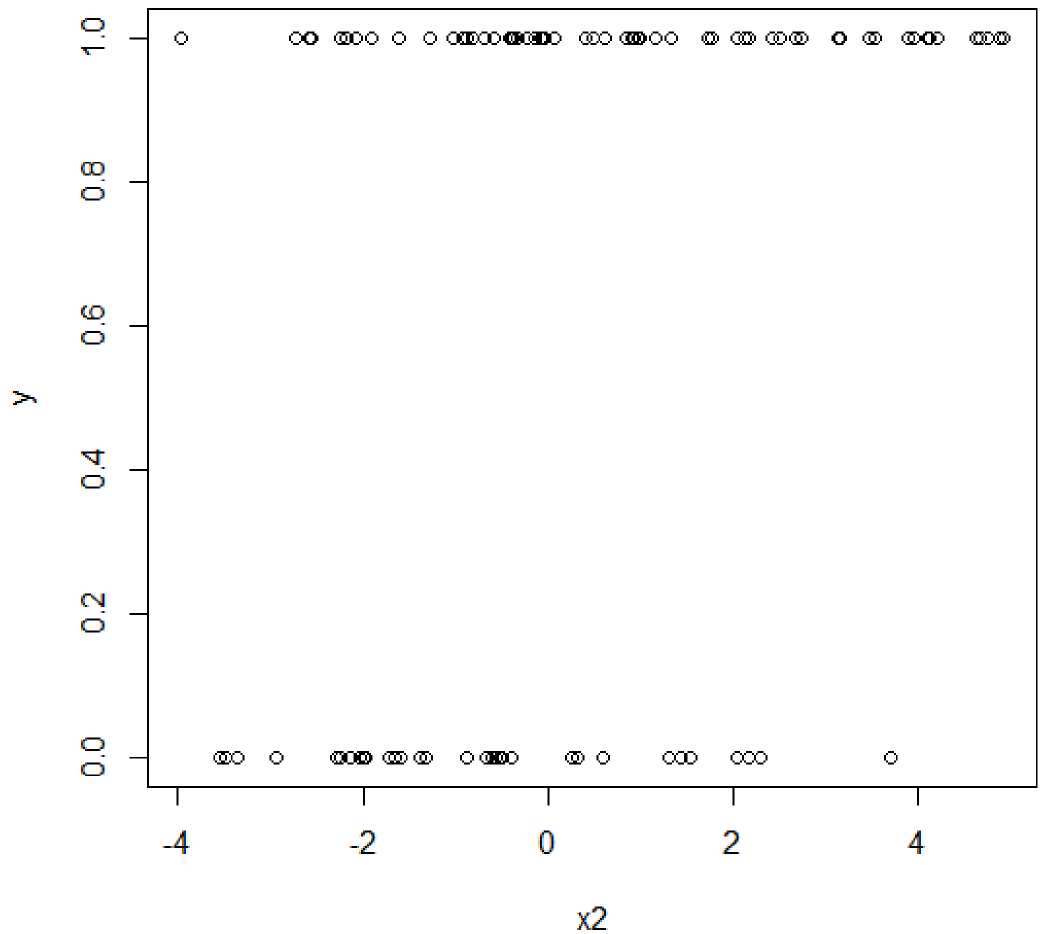
```
group <- NA
group[y == 0] = 1
group[y == 1] = 2

plot(x1,x2, col = c("red", "blue")[group], pch=20)
```




```
plot(x1,y)
```




```
plot(x2,y)
```



1.c) Regresión logística y resultados

Estimar el modelo de regresiónn logística a través de la función glm.

Comentar el resultado obtenido. ¿Cual es el aporte de cada variable explicativa?

```
glm.com=glm(y~x1+x2,family=binomial)
summary(glm.com)

##
## Call:
## glm(formula = y ~ x1 + x2, family = binomial)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -1.78134  -0.05205   0.00661   0.10076   2.01694
##
```

```
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   2.9576     0.8567   3.452 0.000556 ***
## x1           -2.3476     0.6567  -3.575 0.000350 ***
## x2            2.2012     0.6456   3.409 0.000651 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 129.489  on 99  degrees of freedom
## Residual deviance:  27.336  on 97  degrees of freedom
## AIC: 33.336
##
## Number of Fisher Scoring iterations: 8
```

```
glm.res1=glm(y~x1,family=binomial)
summary(glm.res1)
```

```
##
## Call:
## glm(formula = y ~ x1, family = binomial)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.4600  -0.6142   0.2497   0.5753   2.0187
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   1.2616     0.3358   3.757 0.000172 ***
## x1           -0.6967     0.1373  -5.076 3.86e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 129.489  on 99  degrees of freedom
## Residual deviance:  83.463  on 98  degrees of freedom
## AIC: 87.463
##
## Number of Fisher Scoring iterations: 5
```

```
glm.res2=glm(y~x2,family=binomial)
summary(glm.res2)
```

```
##
## Call:
## glm(formula = y ~ x2, family = binomial)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
```

```
## -2.1081  -1.0947   0.5326   0.9469   1.6198  
## 
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)    
## (Intercept)   0.6086     0.2282   2.666 0.007666 ** 
## x2            0.4058     0.1180   3.439 0.000583 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## (Dispersion parameter for binomial family taken to be 1)
## 
##     Null deviance: 129.49  on 99  degrees of freedom
## Residual deviance: 114.67  on 98  degrees of freedom
## AIC: 118.67
## 
## Number of Fisher Scoring iterations: 4
```

```r
anova(glm.com,test='Chisq')
```

```
## Analysis of Deviance Table
## 
## Model: binomial, link: logit
## 
## Response: y
## 
## Terms added sequentially (first to last)
## 
## 
##      Df Deviance Resid. Df Resid. Dev  Pr(>Chi)    
## NULL                    99    129.489              
## x1    1   46.026        98     83.463 1.167e-11 ***
## x2    1   56.127        97     27.336 6.795e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
anova(glm.res1,test='Chisq')
```

```
## Analysis of Deviance Table
## 
## Model: binomial, link: logit
## 
## Response: y
## 
## Terms added sequentially (first to last)
## 
## 
##      Df Deviance Resid. Df Resid. Dev  Pr(>Chi)    
## NULL                    99    129.489              
## x1    1   46.026        98     83.463 1.167e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
anova(glm.res2,test='Chisq')
```

```
## Analysis of Deviance Table
##
## Model: binomial, link: logit
##
## Response: y
##
## Terms added sequentially (first to last)
##
##
##      Df Deviance Resid. Df Resid. Dev  Pr(>Chi)
## NULL                    99     129.49
## x2    1   14.819        98     114.67 0.0001184 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
anova(glm.res1, glm.res2, glm.com, test='Chisq')
```

```
## Analysis of Deviance Table
##
## Model 1: y ~ x1
## Model 2: y ~ x2
## Model 3: y ~ x1 + x2
##   Resid. Df Resid. Dev Df Deviance  Pr(>Chi)
## 1        98     83.463
## 2        98    114.671  0  -31.208
## 3        97     27.336  1   87.335 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
AIC(glm.res1)-AIC(glm.res2)
```

```
## [1] -31.20797
```

```r
AIC(glm.res1)-AIC(glm.com)
```

```
## [1] 54.12664
```

```r
OR_Beta1 = exp(coef(glm.com)[2])
OR_Beta1
```

```
##         x1
## 0.09560086
```

```r
OR_Beta2 = exp(coef(glm.com)[3])
OR_Beta2
```

```
##       x2
## 9.035584
```

Conclusión punto 1.c)

En base a los test realizados, el modelo completo ($y \sim X_1 + X_2$), predice mejor que los modelos reducidos, donde el test ANOVA arroja un p-valor muy inferior al 5%, rechazando así la hipótesis nula de que las verosimilitudes son iguales (ver análisis tabla de análisis: anova(glm.res1, glm.res2, glm.com, test='Chisq')). Además, el coeficiente AIC del modelo completo es muy inferior al obtenido en el modelo reducido (AIC($y \sim X_1$)-AIC($y \sim X_1 + X_2$)=54) con una sola variable ($y \sim X_1$).

Dados los X_1 y X_2 simulados, es posible concluir que ambas variables son relevantes a la variable explicada Y. La variable X_1 influye "negativamente", o incrementa la probabilidad de que la variable Y sea igual a 0 en la medida que sea mayor (β_1 es -2.3476). Por otra parte, la variable X_2 influye "positivamente", o incrementa la probabilidad de que la variable Y sea igual a 1 en la medida que sea mayor (β_1 es 2,2012).

Asimismo, la variable X_2 es más influyente que X_1 en el resultado de Y. Esto se logra apreciar en el resultado del OR, ya que el mismo en X_1 es de 0.0956, y el asociado a X_2 es de 9.04.

1.d) Predicción y matriz de confusión

Realizar las predicciones de Y para la muestra de entrenamiento, y dar los resultados con una matriz de confusión.

```
yhat=predict(glm.com,data.frame(x1=x1,x2=x2),type='response')

yhat=ifelse(yhat<=0.5,0,1)

table(yhat,y)

##      y
## yhat  0  1
##    0 30  2
##    1  5 63
```

1.e) Evaluación de modelo

Simular una nueva muestra de tamaño 100. Calcular la sensibilidad y la especificidad para seq(0,1,0.01). Trazar la curva ROC (como función escalera).

```
#Simulación de muestra de validación

x1_new=runif(n,-4,5)
x2_new=runif(n,-4,5)
y_new=exp(a*x1_new+b*x2_new+c + rnorm(n))
y_new=y_new/(1+y_new)
y_new=rbinom(n,1,y_new)

#Cálculo de sensibilidad y especificidad
```

```r
sens_array = c()
speci_array = c()
yhat_simprimero=predict(glm.com,data.frame(x1=x1,x2=x2),type='response')

for (i in c(seq(0,1,0.01)))
  {
  yhat_sim=ifelse(yhat_simprimero<=i,0,1)
  confmatrix_sim = table(factor(yhat_sim,c(0,1)),y)
  sens_sim = confmatrix_sim[2,2]/(confmatrix_sim[2,2]+confmatrix_sim[1,2])
  speci_sim = confmatrix_sim[1,1]/(confmatrix_sim[1,1]+confmatrix_sim[2,1])
  sens_array = cbind(sens_array,sens_sim[1])
  speci_array = cbind(speci_array,speci_sim[1])
}

sens_array #Sensibilidad en el rango solicitado

##      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12] [,13]
## [1,]    1    1    1    1    1    1    1    1    1     1     1     1     1
##      [,14]     [,15]     [,16]     [,17]     [,18]     [,19]     [,20]
## [1,]     1 0.9846154 0.9846154 0.9846154 0.9846154 0.9846154 0.9846154
##          [,21]     [,22]     [,23]     [,24]     [,25]     [,26]     [,27]
## [1,] 0.9846154 0.9846154 0.9846154 0.9846154 0.9846154 0.9846154 0.9846154
##          [,28]     [,29]     [,30]     [,31]     [,32]     [,33]     [,34]
## [1,] 0.9846154 0.9846154 0.9846154 0.9846154 0.9846154 0.9846154 0.9692308
##          [,35]     [,36]     [,37]     [,38]     [,39]     [,40]     [,41]
## [1,] 0.9692308 0.9692308 0.9692308 0.9692308 0.9692308 0.9692308 0.9692308
##          [,42]     [,43]     [,44]     [,45]     [,46]     [,47]     [,48]
## [1,] 0.9692308 0.9692308 0.9692308 0.9692308 0.9692308 0.9692308 0.9692308
##          [,49]     [,50]     [,51]     [,52]     [,53]     [,54]     [,55]
## [1,] 0.9692308 0.9692308 0.9692308 0.9692308 0.9692308 0.9692308 0.9692308
##          [,56]     [,57]     [,58]     [,59]     [,60]     [,61]     [,62]
## [1,] 0.9692308 0.9692308 0.9538462 0.9538462 0.9538462 0.9538462 0.9538462
##          [,63]     [,64]     [,65]     [,66]     [,67]     [,68]     [,69]
## [1,] 0.9538462 0.9384615 0.9384615 0.9384615 0.9384615 0.9384615 0.9230769
##          [,70]     [,71]     [,72]     [,73]     [,74]     [,75]     [,76]
## [1,] 0.9230769 0.9230769 0.9076923 0.8923077 0.8923077 0.8923077 0.8923077
##          [,77]     [,78]     [,79]     [,80]     [,81]     [,82]     [,83]
## [1,] 0.8923077 0.8923077 0.8923077 0.8923077 0.8769231 0.8769231 0.8769231
##          [,84]     [,85]     [,86]     [,87]     [,88]     [,89]     [,90]
## [1,] 0.8615385 0.8615385 0.8615385 0.8461538 0.8461538 0.8461538 0.8461538
##          [,91]     [,92] [,93] [,94]     [,95]     [,96]     [,97]
## [1,] 0.8461538 0.8307692   0.8   0.8 0.7846154 0.7538462 0.7538462
##          [,98]     [,99]    [,100] [,101]
## [1,] 0.7384615 0.7230769 0.6923077      0

speci_array #Especificidad en el rango solicitado

##      [,1]      [,2]      [,3]      [,4]      [,5]      [,6]      [,7] [,8]
## [1,]    0 0.4857143 0.5714286 0.6571429 0.7428571 0.7428571 0.7428571  0.8
```

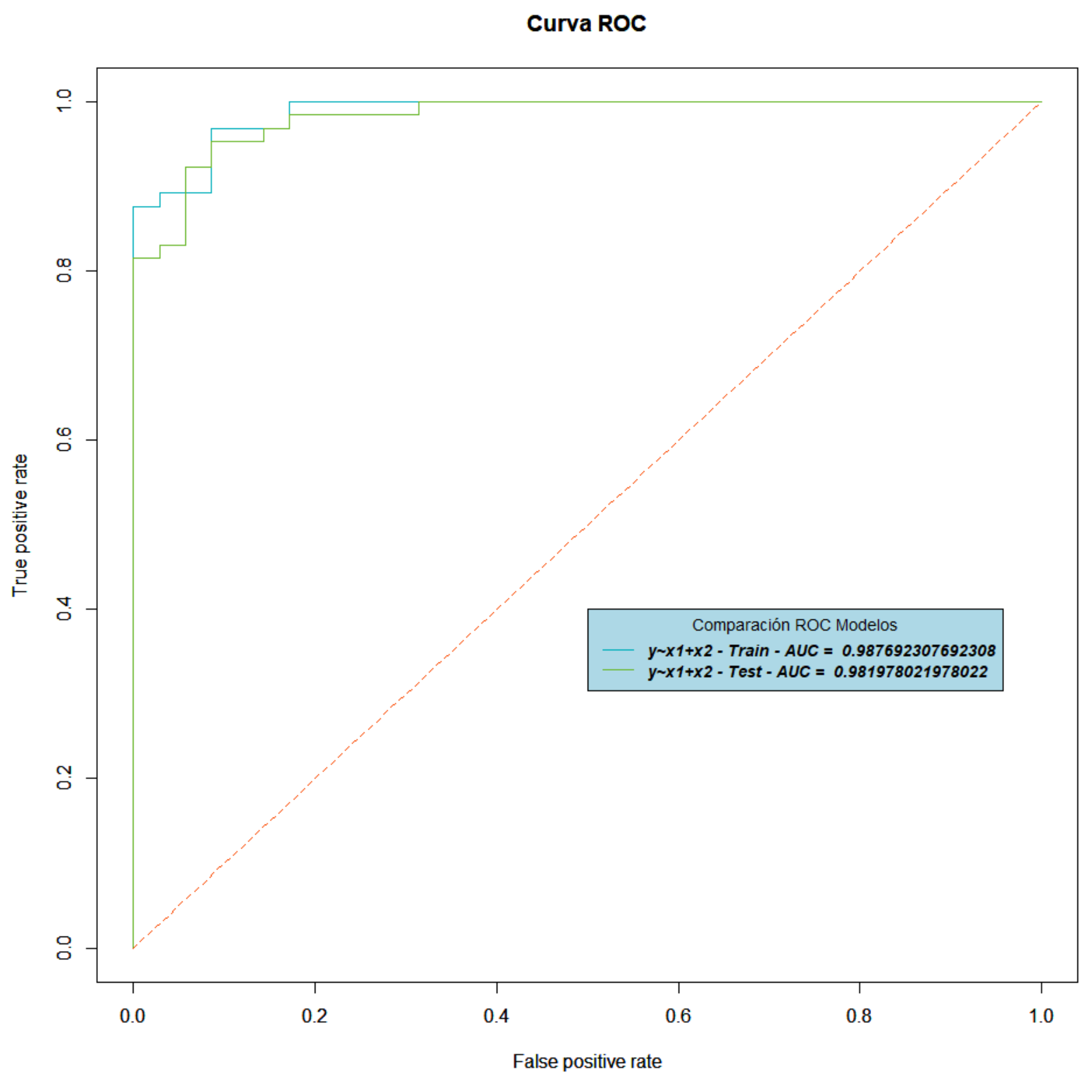
```
##      [,9] [,10]     [,11]     [,12]     [,13]     [,14]     [,15]
## [1,]  0.8   0.8 0.8285714 0.8285714 0.8285714 0.8285714 0.8285714
##          [,16]     [,17]     [,18]     [,19]     [,20]     [,21]     [,22]
## [1,] 0.8285714 0.8285714 0.8285714 0.8285714 0.8285714 0.8285714 0.8285714
##          [,23]     [,24]     [,25]     [,26]     [,27]     [,28]     [,29]
## [1,] 0.8285714 0.8285714 0.8285714 0.8285714 0.8285714 0.8285714 0.8285714
##          [,30]     [,31]     [,32]     [,33]     [,34]     [,35]     [,36]
## [1,] 0.8285714 0.8285714 0.8285714 0.8285714 0.8285714 0.8285714 0.8285714
##          [,37]     [,38]     [,39]     [,40]     [,41]     [,42]     [,43]
## [1,] 0.8285714 0.8285714 0.8285714 0.8285714 0.8285714 0.8285714 0.8285714
##          [,44]     [,45]     [,46]     [,47]     [,48]     [,49]     [,50]
## [1,] 0.8285714 0.8285714 0.8285714 0.8285714 0.8285714 0.8571429 0.8571429
##          [,51]     [,52]     [,53]     [,54]     [,55]     [,56]     [,57]
## [1,] 0.8571429 0.8571429 0.8857143 0.8857143 0.8857143 0.9142857 0.9142857
##          [,58]     [,59]     [,60]     [,61]     [,62]     [,63]     [,64]
## [1,] 0.9142857 0.9142857 0.9142857 0.9142857 0.9142857 0.9142857 0.9142857
##          [,65]     [,66]     [,67]     [,68]     [,69]     [,70]     [,71]
## [1,] 0.9142857 0.9142857 0.9142857 0.9142857 0.9142857 0.9142857 0.9142857
##          [,72]     [,73]     [,74]     [,75]     [,76]     [,77]     [,78]
## [1,] 0.9142857 0.9142857 0.9142857 0.9142857 0.9142857 0.9428571 0.9428571
##          [,79]     [,80] [,81] [,82] [,83] [,84] [,85] [,86] [,87] [,88]
## [1,] 0.9428571 0.9714286     1     1     1     1     1     1     1     1
##      [,89] [,90] [,91] [,92] [,93] [,94] [,95] [,96] [,97] [,98] [,99]
## [1,]     1     1     1     1     1     1     1     1     1     1     1
##      [,100] [,101]
## [1,]      1      1
```

```r
#Curva ROC

library(ROCR)
```

```
## Loading required package: gplots

##
## Attaching package: 'gplots'

## The following object is masked from 'package:stats':
##
##     lowess
```

```r
yhat=predict(glm.com,data.frame(x1=x1,x2=x2),type='response')

rocplot =function (pred , truth , C,...){
  predob = prediction (pred , truth)
  perf = performance (predob , "tpr", "fpr")
  return(perf)}

AUC_ROC =function (pred , truth , ...){
  predob = prediction (pred , truth)
  Area = performance (predob , "auc")
  return(Area@y.values)}
```

```
plot(rocplot(predict(glm.com,data.frame(x1=x1,x2=x2),type='response'),y),col=
"#00AFBB",main="Curva ROC")
par(new=TRUE)
plot(rocplot(predict(glm.com,data.frame(x1=x1_new,x2=x2_new),type='response')
,y_new),col="#6BB82E",main="Curva ROC")
par(new=TRUE)
lines(c(seq(0,1,0.01)), c(seq(0,1,0.01)), col = "#FC4E07", type="l", lty=2)

AUC_ROC_tra =
AUC_ROC(predict(glm.com,data.frame(x1=x1,x2=x2),type='response'),y)
AUC_ROC_tes =
AUC_ROC(predict(glm.com,data.frame(x1=x1_new,x2=x2_new),type='response'),y_ne
w)

legend(0.5, 0.4, legend=c(paste("y~x1+x2 - Train - AUC = ", AUC_ROC_tra) ,
paste("y~x1+x2 - Test - AUC = ", AUC_ROC_tes)),
       col=c("#00AFBB", "#6BB82E"), lty=c(1,1), cex=0.9,
       title="Comparación ROC Modelos", text.font=4, bg='lightblue')
```

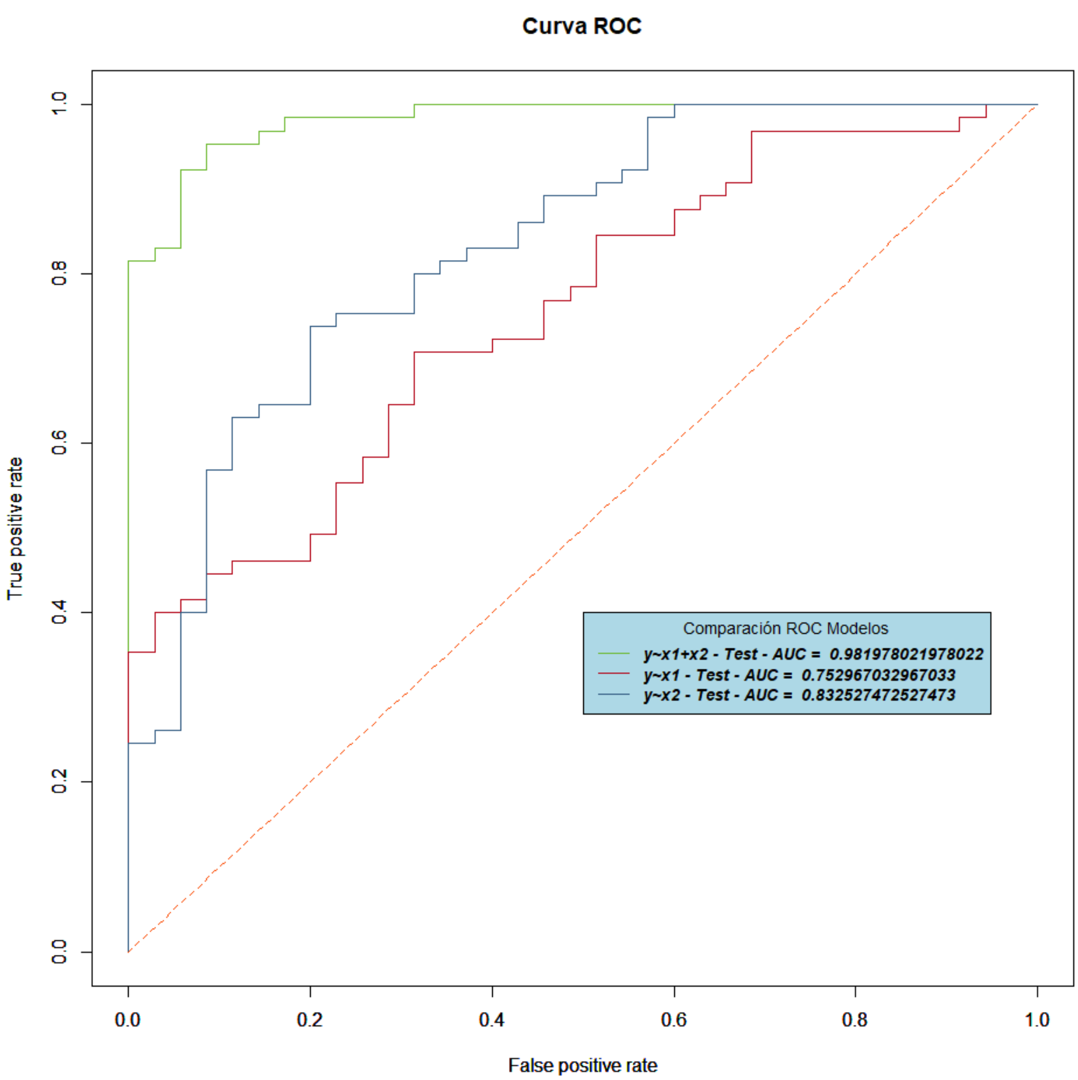


1.f) Comparar con modelo con una sola variable explicativa

Hacer lo mismo usando una sola variable explicativa en el modelo logístico. Superponer ambas curvas ROC y elegir el mejor modelo.

```
par(new=FALSE)
plot(rocplot(predict(glm.com,data.frame(x1=x1_new,x2=x2_new),type='response')
,y_new),col="#6BB82E",main="Curva ROC")
par(new=TRUE)
plot(rocplot(predict(glm.res1,data.frame(x1=x1_new),type='response'),y_new),c
ol="#B30417")
par(new=TRUE)
plot(rocplot(predict(glm.res2,data.frame(x2=x2_new),type='response'),y_new),c
ol="#325b82")
par(new=TRUE)
lines(c(seq(0,1,0.01)), c(seq(0,1,0.01)), col = "#FC4E07", type="l", lty=2)
```

```r
AUC_ROC_res1 =
AUC_ROC(predict(glm.res1,data.frame(x1=x1_new),type='response'),y_new)
AUC_ROC_res2 =
AUC_ROC(predict(glm.res2,data.frame(x2=x2_new),type='response'),y_new)


legend(0.5, 0.4, legend=c(paste("y~x1+x2 - Test - AUC = ", AUC_ROC_tes),
paste("y~x1 - Test - AUC = ", AUC_ROC_res1), paste("y~x2 - Test - AUC = ",
AUC_ROC_res2)),
       col=c("#6BB82E", "#B30417", "#325b82"), lty=c(1,1,1), cex=0.9,
       title="Comparación ROC Modelos", text.font=4, bg='lightblue')
```



Parte 2 - Modelo de predicción

En la página del UCI https://archive.ics.uci.edu/ml/datasets.php bajar los datos de Cancer (Breast Cancer Wisconsin). El objetivo consiste en predecir si el tumor es benigno o maligno a partir de varias variables explicativas. Dividir aleatoriamente el conjunto de datos en train/test.

Estructura de la base utilizada:

A	ttribute Domain
id.	Sample code number id number
x1.	Clump Thickness 1 - 10
x2.	Uniformity of Cell Size 1 - 10
x3.	Uniformity of Cell Shape 1 - 10
x4.	Marginal Adhesion 1 - 10
x5.	Single Epithelial Cell Size 1 - 10
x6.	Bare Nuclei 1 - 10
x7.	Bland Chromatin 1 - 10
x8.	Normal Nucleoli 1 - 10
x9.	Mitoses 1 - 10
y.	Class: (2 for benign, 4 for malignant)

```
library(readr)

db <- read_csv("breast-cancer-wisconsin.data",col_names = FALSE,
               col_types = cols(X7 = col_double()))

db[db == '?'] = as.numeric(NA)
db <- na.omit(db)

colnames(db) = c('Id','x1','x2','x3','x4','x5','x6','x7','x8','x9','y')

#Se modifica para que Y='Maligno' sea igual a 1, mientras que Y='Benigno' sea
igual a 0.
db$y = ifelse(db$y==4,1,0)

ind = sample(2,nrow(db),replace=TRUE,prob = c(0.8,0.2))

train = db[ind==1,]

test = db[ind==2,]

X_train = train[2:10]

y_train = train$y
```

```r
X_test = test[2:10]

y_test = test$y
```

2.a) Estimar modelo

Estimar el modelo completo. Analizar el aporte de cada variable y dar el valor del AIC.

```r
glm.modelo=glm(y_train~.,family=binomial, data=X_train)

summary(glm.modelo)

##
## Call:
## glm(formula = y_train ~ ., family = binomial, data = X_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.2255  -0.1135  -0.0596   0.0328   2.7433
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -10.252033   1.373875  -7.462 8.51e-14 ***
## x1            0.570478   0.179619   3.176 0.001493 **
## x2            0.002269   0.220381   0.010 0.991785
## x3            0.345145   0.255919   1.349 0.177450
## x4            0.045260   0.157226   0.288 0.773452
## x5            0.239809   0.186277   1.287 0.197962
## x6            0.381811   0.103478   3.690 0.000224 ***
## x7            0.494826   0.188409   2.626 0.008631 **
## x8            0.159187   0.115741   1.375 0.169017
## x9            0.449819   0.362432   1.241 0.214564
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 709.440  on 551  degrees of freedom
## Residual deviance:  81.317  on 542  degrees of freedom
## AIC: 101.32
##
## Number of Fisher Scoring iterations: 8

anova(glm.modelo,test='Chisq')

## Analysis of Deviance Table
##
## Model: binomial, link: logit
##
```

```
## Response: y_train
##
## Terms added sequentially (first to last)
##
##
##      Df Deviance Resid. Df Resid. Dev  Pr(>Chi)
## NULL                   551     709.44
## x1    1   356.58       550     352.86 < 2.2e-16 ***
## x2    1   203.02       549     149.84 < 2.2e-16 ***
## x3    1    14.18       548     135.66 0.0001662 ***
## x4    1     9.79       547     125.87 0.0017558 **
## x5    1     7.31       546     118.56 0.0068584 **
## x6    1    24.16       545      94.40 8.848e-07 ***
## x7    1     8.84       544      85.56 0.0029527 **
## x8    1     2.13       543      83.43 0.1440477
## x9    1     2.11       542      81.32 0.1460076
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
AIC_modelo = glm.modelo$aic
AIC_modelo
```

```
## [1] 101.3167
```

Conclusión 2.a)

Las variables explicativas mantienen siempre un β positivo (salvo el intercept o β_0 que es negativo), lo cual implica que, en la medida que aumenta el valor de cada variable, mayor es la probabilidad de que el tumor sea maligno.

2.b) Analiza significancia del modelo

Averiguar si el modelo es significativo al 5 %.

```
chi2=glm.modelo$null.deviance - glm.modelo$deviance

ddl=glm.modelo$df.null-glm.modelo$df.residual

pvalor=pchisq(chi2,ddl,lower.tail=F)
pvalor
```

```
## [1] 1.920788e-129
```

Conclusión 2.b)

El modelo se ajusta bien a los datos, siendo significativo al 5%, ya que el p-valor obtenido es muy inferior a 0.05.

2.c) Modelo reducido

Estimar un modelo donde estén presentes las variables significativas al 5% del apartado anterior.

```
res <- NULL
for(var in row.names(summary(glm.modelo)$coefficients)) {
  if(var != '(Intercept)'){
    if(summary(glm.modelo)$coefficients[var,4] < .05){
      res <- rbind(res,var)
    }
  }
}

formula <- as.formula(paste("y_train~", paste(res, collapse="+")))
formula

## y_train ~ x1 + x6 + x7

reduced.model <- glm(formula,family=binomial, data=X_train, maxit=100)
summary(reduced.model)

##
## Call:
## glm(formula = formula, family = binomial, data = X_train, maxit = 100)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.5713  -0.1513  -0.0613   0.0348   2.3906
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -10.09912    1.11211  -9.081  < 2e-16 ***
## x1            0.86911    0.14501   5.994 2.05e-09 ***
## x6            0.56275    0.08993   6.258 3.90e-10 ***
## x7            0.79747    0.14917   5.346 8.98e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 709.44  on 551  degrees of freedom
## Residual deviance: 106.65  on 548  degrees of freedom
## AIC: 114.65
##
## Number of Fisher Scoring iterations: 8

#Se verificó el cálculo de la desvianza
dev = -2*logLik(reduced.model)
dev
```

```
## 'log Lik.' 106.6525 (df=4)
```

```r
deviance(reduced.model)
```

```
## [1] 106.6525
```

```r
anova(reduced.model,test='Chisq')
```

```
## Analysis of Deviance Table
##
## Model: binomial, link: logit
##
## Response: y_train
##
## Terms added sequentially (first to last)
##
##
##      Df Deviance Resid. Df Resid. Dev  Pr(>Chi)
## NULL                   551     709.44
## x1    1   356.58       550     352.86 < 2.2e-16 ***
## x6    1   200.05       549     152.81 < 2.2e-16 ***
## x7    1    46.15       548     106.65 1.094e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
AIC_modelores = reduced.model$aic
AIC_modelores
```

```
## [1] 114.6525
```

2.d) Estimar Modelo Forward

Estimar un modelo simplificado con el método forward.

```r
library("MASS")
nothing = glm(y_train ~ 1,family=binomial, data=X_train)

step.fwd =
stepAIC(nothing,scope=list(lower=formula(nothing),upper=formula(glm.modelo)),
direction="forward")
```

```
## Start:  AIC=711.44
## y_train ~ 1
##
##        Df Deviance    AIC
## + x2    1   197.82 201.82
## + x3    1   206.34 210.34
## + x6    1   271.09 275.09
## + x7    1   297.47 301.47
## + x5    1   325.01 329.01
## + x1    1   352.86 356.86
## + x8    1   369.62 373.62
```

```
## + x4    1   378.37 382.37
## + x9    1   576.25 580.25
## <none>      709.44 711.44
##
## Step:  AIC=201.82
## y_train ~ x2
##
##        Df Deviance    AIC
## + x6    1   128.58 134.58
## + x1    1   149.84 155.84
## + x7    1   158.06 164.06
## + x3    1   171.43 177.43
## + x8    1   175.33 181.33
## + x5    1   180.94 186.94
## + x4    1   181.74 187.74
## + x9    1   188.49 194.49
## <none>      197.82 201.82
##
## Step:  AIC=134.58
## y_train ~ x2 + x6
##
##        Df Deviance    AIC
## + x1    1   102.91 110.91
## + x7    1   114.53 122.53
## + x8    1   114.69 122.69
## + x3    1   115.09 123.09
## + x5    1   123.45 131.45
## + x9    1   125.61 133.61
## <none>      128.58 134.58
## + x4    1   126.84 134.84
##
## Step:  AIC=110.91
## y_train ~ x2 + x6 + x1
##
##        Df Deviance    AIC
## + x7    1   91.848 101.85
## + x8    1   95.354 105.35
## + x3    1   97.926 107.93
## + x5    1   98.959 108.96
## <none>     102.912 110.91
## + x4    1  100.955 110.95
## + x9    1  101.597 111.60
##
## Step:  AIC=101.85
## y_train ~ x2 + x6 + x1 + x7
##
##        Df Deviance     AIC
## + x8    1   87.498  99.498
## + x5    1   88.279 100.279
## + x3    1   88.319 100.319
```

```
## + x9    1   89.704 101.704
## <none>       91.848 101.848
## + x4    1   90.868 102.868
##
## Step:  AIC=99.5
## y_train ~ x2 + x6 + x1 + x7 + x8
##
##        Df Deviance     AIC
## + x5    1   85.082  99.082
## + x9    1   85.478  99.478
## <none>       87.498  99.498
## + x3    1   85.563  99.563
## + x4    1   86.846 100.846
##
## Step:  AIC=99.08
## y_train ~ x2 + x6 + x1 + x7 + x8 + x5
##
##        Df Deviance     AIC
## <none>       85.082  99.082
## + x9    1   83.117  99.117
## + x3    1   83.694  99.694
## + x4    1   84.704 100.704
```

```
anova(step.fwd,test='Chisq')
```

```
## Analysis of Deviance Table
##
## Model: binomial, link: logit
##
## Response: y_train
##
## Terms added sequentially (first to last)
##
##
##      Df Deviance Resid. Df Resid. Dev  Pr(>Chi)
## NULL                   551     709.44
## x2    1   511.62       550     197.82 < 2.2e-16 ***
## x6    1    69.24       549     128.58 < 2.2e-16 ***
## x1    1    25.67       548     102.91 4.056e-07 ***
## x7    1    11.06       547      91.85 0.0008802 ***
## x8    1     4.35       546      87.50 0.0370059 *
## x5    1     2.42       545      85.08 0.1200973
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
AIC_modelofwd = step.fwd$aic
AIC_modelofwd
```

```
## [1] 99.0816
```

2.e) Estimar Modelo Stepwise

Estimar un modelo simplificado con el método stepwise.

```
step.bot =
stepAIC(glm.modelo,scope=list(lower=formula(nothing),upper=formula(glm.modelo
)), direction="both")
```

```
## Start:  AIC=101.32
## y_train ~ x1 + x2 + x3 + x4 + x5 + x6 + x7 + x8 + x9
##
##        Df Deviance     AIC
## - x2    1   81.317  99.317
## - x4    1   81.399  99.399
## - x5    1   82.934 100.934
## - x3    1   82.950 100.950
## - x8    1   83.299 101.299
## <none>      81.317 101.317
## - x9    1   83.430 101.430
## - x7    1   89.001 107.001
## - x1    1   94.616 112.616
## - x6    1   96.737 114.737
##
## Step:  AIC=99.32
## y_train ~ x1 + x3 + x4 + x5 + x6 + x7 + x8 + x9
##
##        Df Deviance     AIC
## - x4    1   81.401  97.401
## - x5    1   82.995  98.995
## - x8    1   83.310  99.310
## <none>      81.317  99.317
## - x9    1   83.517  99.517
## - x3    1   84.667 100.667
## + x2    1   81.317 101.317
## - x7    1   89.534 105.534
## - x1    1   94.943 110.943
## - x6    1   96.738 112.738
##
## Step:  AIC=97.4
## y_train ~ x1 + x3 + x5 + x6 + x7 + x8 + x9
##
##        Df Deviance     AIC
## - x5    1   83.146  97.146
## - x8    1   83.399  97.399
## <none>      81.401  97.401
## - x9    1   83.847  97.847
## - x3    1   85.136  99.136
## + x4    1   81.317  99.317
## + x2    1   81.399  99.399
## - x7    1   89.970 103.970
```

```
## - x1     1   94.946 108.946
## - x6     1  100.306 114.306
##
## Step:  AIC=97.15
## y_train ~ x1 + x3 + x6 + x7 + x8 + x9
##
##          Df Deviance     AIC
## <none>        83.146  97.146
## + x5     1   81.401  97.401
## - x8     1   85.831  97.831
## - x9     1   86.447  98.447
## + x4     1   82.995  98.995
## + x2     1   83.057  99.057
## - x3     1   89.999 101.999
## - x7     1   91.968 103.968
## - x1     1   96.314 108.314
## - x6     1  104.941 116.941
```

```
step.bot2 =
stepAIC(nothing,scope=list(lower=formula(nothing),upper=formula(glm.modelo)),
direction="both")
```

```
## Start:  AIC=711.44
## y_train ~ 1
##
##          Df Deviance    AIC
## + x2     1   197.82 201.82
## + x3     1   206.34 210.34
## + x6     1   271.09 275.09
## + x7     1   297.47 301.47
## + x5     1   325.01 329.01
## + x1     1   352.86 356.86
## + x8     1   369.62 373.62
## + x4     1   378.37 382.37
## + x9     1   576.25 580.25
## <none>       709.44 711.44
##
## Step:  AIC=201.82
## y_train ~ x2
##
##          Df Deviance    AIC
## + x6     1   128.58 134.58
## + x1     1   149.84 155.84
## + x7     1   158.06 164.06
## + x3     1   171.43 177.43
## + x8     1   175.33 181.33
## + x5     1   180.94 186.94
## + x4     1   181.74 187.74
## + x9     1   188.49 194.49
## <none>       197.82 201.82
```

```
## - x2    1   709.44 711.44
##
## Step:  AIC=134.58
## y_train ~ x2 + x6
##
##        Df Deviance    AIC
## + x1    1   102.91 110.91
## + x7    1   114.53 122.53
## + x8    1   114.69 122.69
## + x3    1   115.09 123.09
## + x5    1   123.45 131.45
## + x9    1   125.61 133.61
## <none>      128.58 134.58
## + x4    1   126.84 134.84
## - x6    1   197.82 201.82
## - x2    1   271.09 275.09
##
## Step:  AIC=110.91
## y_train ~ x2 + x6 + x1
##
##        Df Deviance    AIC
## + x7    1   91.848 101.85
## + x8    1   95.354 105.35
## + x3    1   97.926 107.93
## + x5    1   98.959 108.96
## <none>     102.912 110.91
## + x4    1  100.955 110.95
## + x9    1  101.597 111.60
## - x1    1  128.579 134.58
## - x6    1  149.842 155.84
## - x2    1  152.805 158.81
##
## Step:  AIC=101.85
## y_train ~ x2 + x6 + x1 + x7
##
##        Df Deviance     AIC
## + x8    1   87.498  99.498
## + x5    1   88.279 100.279
## + x3    1   88.319 100.319
## + x9    1   89.704 101.704
## <none>      91.848 101.848
## + x4    1   90.868 102.868
## - x7    1  102.912 110.912
## - x2    1  106.653 114.653
## - x1    1  114.526 122.526
## - x6    1  119.826 127.826
##
## Step:  AIC=99.5
## y_train ~ x2 + x6 + x1 + x7 + x8
##
```

```
##         Df Deviance     AIC
## + x5    1   85.082  99.082
## + x9    1   85.478  99.478
## <none>      87.498  99.498
## + x3    1   85.563  99.563
## + x4    1   86.846 100.846
## - x8    1   91.848 101.848
## - x2    1   93.983 103.983
## - x7    1   95.354 105.354
## - x1    1  106.626 116.626
## - x6    1  113.380 123.380
##
## Step:  AIC=99.08
## y_train ~ x2 + x6 + x1 + x7 + x8 + x5
##
##         Df Deviance     AIC
## <none>      85.082  99.082
## + x9    1   83.117  99.117
## - x5    1   87.498  99.498
## - x2    1   87.552  99.552
## + x3    1   83.694  99.694
## - x8    1   88.279 100.279
## + x4    1   84.704 100.704
## - x7    1   93.184 105.184
## - x1    1  104.711 116.711
## - x6    1  107.405 119.405
```

```
anova(step.bot,test='Chisq')
```

```
## Analysis of Deviance Table
##
## Model: binomial, link: logit
##
## Response: y_train
##
## Terms added sequentially (first to last)
##
##
##      Df Deviance Resid. Df Resid. Dev  Pr(>Chi)
## NULL                   551     709.44
## x1    1   356.58       550     352.86 < 2.2e-16 ***
## x3    1   198.70       549     154.16 < 2.2e-16 ***
## x6    1    49.31       548     104.85 2.183e-12 ***
## x7    1    14.69       547      90.16 0.0001267 ***
## x8    1     3.71       546      86.45 0.0540352 .
## x9    1     3.30       545      83.15 0.0692357 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
AIC_modelobot = step.bot$aic
AIC_modelobot

## [1] 97.14575
```

Conclusión 2.e)

Se probó realizar el modelo stepwise, comenzando desde el modelo completo y desde el modelo sin variables explicativas (únicamente con β_0). Según los resultados obtenidos, el modelo que comienza con todas las variables, obtuvo mejor performance que el otro (con un AIC inferior).

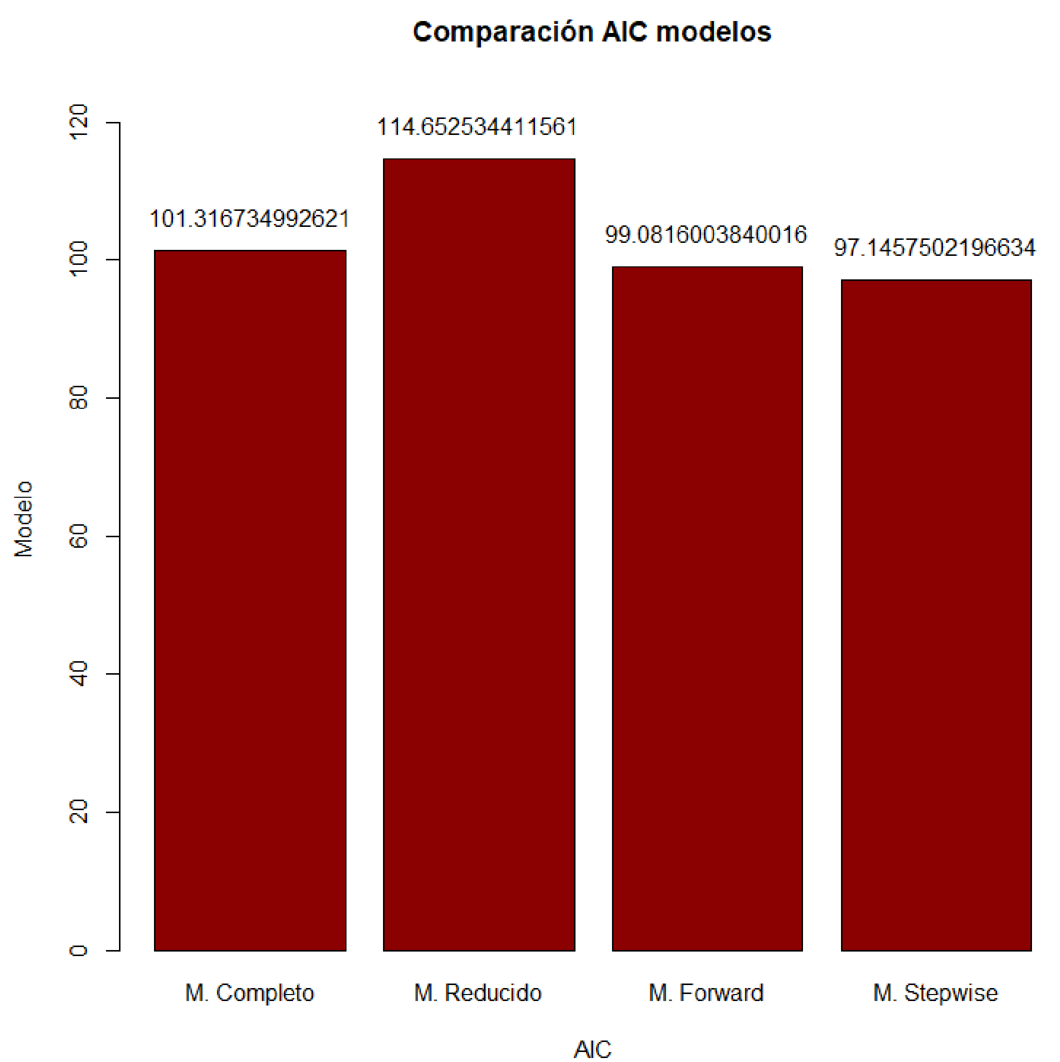
Cabe mencionar que, se ejecutó el método stepwise con dos orígenes ("desde el final" o "desde el principio"), ya que esta metodología es greedy y puede sacar o introducir variables (tomando la mejor decisión local), por lo que modelo puede ser distinto según el punto de partida. En este sentido, seleccionamos el que obtuvo mejor resultado para el caso planteado.

2.f) Mejor modelo según AIC

¿Cuál es el mejor modelo con el AIC?

```r
formula(glm.modelo)

## y_train ~ x1 + x2 + x3 + x4 + x5 + x6 + x7 + x8 + x9

AIC_modelo

## [1] 101.3167

formula(reduced.model)

## y_train ~ x1 + x6 + x7

AIC_modelores

## [1] 114.6525

formula(step.fwd)

## y_train ~ x2 + x6 + x1 + x7 + x8 + x5

AIC_modelofwd

## [1] 99.0816

formula(step.bot)

## y_train ~ x1 + x3 + x6 + x7 + x8 + x9

AIC_modelobot

## [1] 97.14575
```

```r
G_AIC = barplot(c(AIC_modelo,AIC_modelores,AIC_modelofwd,AIC_modelobot),
        main = "Comparación AIC modelos",
        xlab = "AIC",
        ylab = "Modelo",
        names.arg = c("M. Completo", "M. Reducido", "M. Forward", "M.
Stepwise"),
        col = "darkred",
        horiz = FALSE,
        ylim =
c(0,round(max(AIC_modelo,AIC_modelores,AIC_modelofwd,AIC_modelobot))+10))

text(G_AIC, c(AIC_modelo+5,AIC_modelores+5,AIC_modelofwd+5,AIC_modelobot+5),
labels=c(AIC_modelo,AIC_modelores,AIC_modelofwd,AIC_modelobot), xpd=TRUE)
```




```r
AIC_modelores - AIC_modelobot

## [1] 17.50678
```

```r
AIC_modelo - AIC_modelobot

## [1] 4.170985

AIC_modelofwd - AIC_modelobot

## [1] 1.93585
```

Conclusión 2.f)

En base al criterio de AIC, el mejor modelo fue el estimado con el método de stepwise ("modelobot"). Adicionalmente, es posible concluir que el modelo stepwise es "similar" al modelo forward (diferencias de AIC menor a 2), es "mejor" que el modelo completo (diferencia entre 4 y 7) y "mucho mejor" que el modelo reducido (diferencia mayor a 10).

2.g) Comparación sobre conjunto de Test

¿Cuál es el mejor modelo sobre la muestra de Test?

```r
yhat=predict(glm.modelo,X_test,type='response')
class_hat = ifelse(yhat<=0.5,0,1)
t=table(class_hat,y_test)
t

##          y_test
## class_hat  0  1
##         0 80  4
##         1  1 46

mean(class_hat==y_test) #accurancy

## [1] 0.9618321

sens_modelo = t[2,2]/(t[2,2]+t[1,2]) #sensibilidad
sens_modelo

## [1] 0.92

yhat=predict(reduced.model,X_test,type='response')
class_hat = ifelse(yhat<=0.5,0,1)
t=table(class_hat,y_test)
t

##          y_test
## class_hat  0  1
##         0 80  4
##         1  1 46

mean(class_hat==y_test) #accurancy

## [1] 0.9618321
```

```
sens_modelores = t[2,2]/(t[2,2]+t[1,2]) #sensibilidad
sens_modelores

## [1] 0.92

yhat=predict(step.fwd,X_test,type='response')
class_hat = ifelse(yhat<=0.5,0,1)
t=table(class_hat,y_test)
t

##          y_test
## class_hat  0  1
##         0 80  4
##         1  1 46

mean(class_hat==y_test) #accurancy

## [1] 0.9618321

sens_modelofwd = t[2,2]/(t[2,2]+t[1,2]) #sensibilidad
sens_modelofwd

## [1] 0.92

yhat=predict(step.bot,X_test,type='response')
class_hat = ifelse(yhat<=0.5,0,1)
t=table(class_hat,y_test)
t

##          y_test
## class_hat  0  1
##         0 80  4
##         1  1 46

mean(class_hat==y_test) #accurancy

## [1] 0.9618321

sens_modelobot = t[2,2]/(t[2,2]+t[1,2]) #sensibilidad
sens_modelobot

## [1] 0.92
```

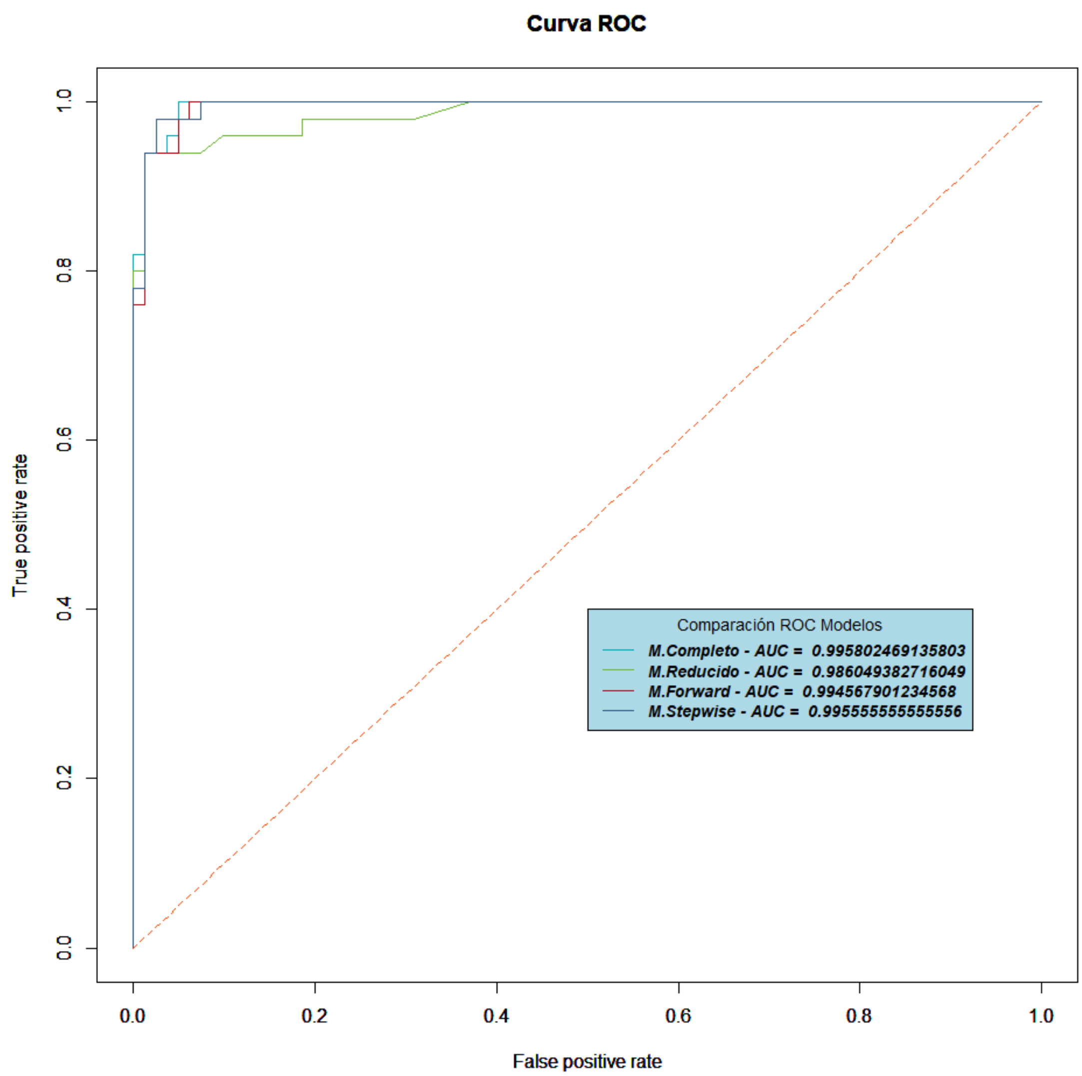
Conclusión 2.g)

Para los datos disponibles y utilizando el punto de corte de las clase con probabilidad igual 0.5, los resultados son iguales en el conjunto de Test (donde solo existen 130 casos), tanto en el accurancy como en la sensibilidad. No obstante, esto se produce por la semilla utilizada para la separación del conjunto Train y Test. Se probó que cambiando modificando la misma, el resultado si variaba para los diferentes modelos, siendo el que obtenía menor AIC (stepwise generalmente) arrojaba mejores resultados sobre el conjunto de Test.

2.h) Comparación curva ROC.

Trazar la curva ROC para cada uno de los modelos. ¿Cuál es el mejor?

```
library(ROCR)

yhat_modelo=predict(glm.modelo,X_test,type='response')
yhat_reducido=predict(reduced.model,X_test,type='response')
yhat_forw=predict(step.fwd,X_test,type='response')
yhat_both=predict(step.bot,X_test,type='response')

rocplot =function (pred , truth , C,...){
  predob = prediction (pred , truth)
  perf = performance (predob , "tpr", "fpr")
  return(perf)}

AUC_ROC =function (pred , truth , ...){
  predob = prediction (pred , truth)
  Area = performance (predob , "auc")
  return(Area@y.values)}

plot(rocplot(yhat_modelo,y_test),col="#00AFBB",main="Curva ROC")
par(new=TRUE)
plot(rocplot(yhat_reducido,y_test),col="#6BB82E",main="Curva ROC")
par(new=TRUE)
plot(rocplot(yhat_forw,y_test),col="#B30417",main="Curva ROC")
par(new=TRUE)
plot(rocplot(yhat_both,y_test),col="#325b82",main="Curva ROC")
par(new=TRUE)
lines(c(seq(0,1,0.01)), c(seq(0,1,0.01)), col = "#FC4E07", type="l", lty=2)

AUC_ROC_mod = AUC_ROC(yhat_modelo,y_test)
AUC_ROC_res = AUC_ROC(yhat_reducido,y_test)
AUC_ROC_fwd = AUC_ROC(yhat_forw,y_test)
AUC_ROC_bot = AUC_ROC(yhat_both,y_test)

legend(0.5, 0.4, legend=c(paste("M.Completo - AUC = ", AUC_ROC_mod) ,
paste("M.Reducido - AUC = ", AUC_ROC_res),
                          paste("M.Forward - AUC = ",
AUC_ROC_fwd),paste("M.Stepwise - AUC = ", AUC_ROC_bot)),
       col=c("#00AFBB", "#6BB82E", "#B30417", "#325b82"), lty=c(1,1),
cex=0.9,
       title="Comparación ROC Modelos", text.font=4, bg='lightblue')
```



Conclusión 2.h)

En este caso, el Modelo Completo es el que mantiene un área bajo la curva ROC mayor (realizando las pruebas sobre el conjunto de Test), por lo tanto es el que refleja mayor poder de clasificación en dicho conjunto. No osbtante, es preciso destacar que la diferencia con el Stepwise es muy reducida.

Para tener una mejor conclusión a partir de esta métrica, sería deseable contar con un conjunto de validación mayor.