Untitled

Jhonatan Smith Garcia

10/11/2021

a) Cree un conjunto de datos de entrenamiento del 75% y el restante 25 % tratelo como datos de test o de prueba

```
bank <- read.csv(file.choose(), sep = ",")
```

Se procede a realizar la division de la base de datos.

```
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 4.0.5
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
set.seed(1039705595) # Se selecciona una semilla para la extraccion de la muestra aleatoria
datab <- sort(sample(nrow(bank), nrow(bank)*.75))
# Datos de entrenamiento
train<-bank[datab,]
# Datos de prueba
test<-bank[-datab,]
ytrain<-bank[datab,8]
ytest<-bank[-datab,8]
newdata1<- train %>% dplyr::select(-8)
newdata2<- test %>% dplyr::select(-8)
```

b) Se ajusta Naive Bayes con los datos de entrenamiento. En R...

```
library(naivebayes)
```

```
## Warning: package 'naivebayes' was built under R version 4.0.5
```

```
## naivebayes 0.9.7 loaded
```

```
modelo_NB <- naive_bayes(loan ~ ., data = train)
```

C) Para implementar Knn se debe utilizar variables indicadoras para las categoricas y normalizar las continuas. Tenga presente que de las 17 variables de la base de datos 9 (8 sin contar la supervisora) son categoricas.

Primero, se implementan las variables indicadoras (Dummies) y luego se normalizan las continuas.

Se construye un df con todas las variable para luego sacar de alli los datos de prueba y entrenamiento dada las indicadoras y las varuables normalizadas.

Dado el siguiente codigo, se normalizan las variables continuas, pues se requieren bajo la misma escala para proceder.

```
normalize <- function(x) {
norm <- ((x - min(x))/(max(x) - min(x)))
return (norm)
}
train2<-(newdata[newdatasort,])
train2n<-normalize(train2[-50])
test2<-(newdata[-newdatasort,])
test2n<-normalize(test2[-50])
# Datos de entrenamiento para Knn
ytrain2<-newdata[newdatasort,50]
# Datos de prueba para Knn
ytest2<-newdata[-newdatasort,50]
```

Se simulan knn con k de 1 a 10

```
for (i in 1:10) {
# Se modela con los datos de entrenamiento con un k=i
mod2train<- knn(train = train2n, test = train2n, cl = ytrain2, k=i, prob=TRUE)
# Se modela con los datos de prueba con un k=i
mod2test<- knn(train = train2n, test = test2n, cl = ytrain2, k=i,prob=TRUE)
tk<-table(mod2train,ytrain2)
t1n<-table(mod2test,ytest2)
# Training error
Training_error_KNN<-(tk[1,2]+tk[2,1])/(sum(tk))
print(table(Training_error_KNN,i))
}
```

```
##                   i
## Training_error_KNN 1
##                  0 1
##                    i
## Training_error_KNN  2
##   0.113009198423127 1
##                    i
## Training_error_KNN  3
##   0.105841595986143 1
##                    i
## Training_error_KNN  4
##   0.122088161509975 1
##                    i
## Training_error_KNN  5
##   0.121132481185044 1
##                    i
## Training_error_KNN  6
##   0.124955202484769 1
##                    i
## Training_error_KNN  7
##   0.126388722972166 1
##                    i
## Training_error_KNN  8
```

```
##   0.127344403297097 1
##                     i
## Training_error_KNN  9
##   0.128300083622028 1
##                     i
## Training_error_KNN  10
##   0.129016843865727  1
```

Bajo estos criterios se escoge k = 3 puesto que tiene un error de entrenamiento bajo y un k mas pequeño puede estar sobre ajustando el modelo (como posiblemente sucede en k=1)

Se ajusta el modelo con el K seleccionado.

```
modelo_knn<-knn(train = train2n, test = train2n, cl = ytrain2, k=3, prob=TRUE)
```

D) La Regresion logistica será tal que:

```
modelo_logistico <- glm(as.factor(train2$loan) ~ . ,data=train2n, family = "binomial")
```

E) Se implementa LDA con R:

```
# ytrain: variable loan con 1,0
newdata1<- train %>% dplyr::select(-8)
modelo_LDA=lda(ytrain~., data = newdata1)
```

F) Se calcula la matriz de confusion, el error de training MSE y se grafica:

```
train_NB<-predict(modelo_NB,newdata1,type="class")
predicted<-predict(modelo_logistico,newdata=train2n,type = "response")
```

```
## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from a rank-deficient fit may be misleading
```

```
optCutOff <- optimalCutoff(train2$loan, predicted)[1]
glm.pred <- ifelse(predicted > 0.48, 1, 0)
table(glm.pred,ytrain)
```

```
##         ytrain
## glm.pred   no  yes
##        0 7270 1075
##        1   22    4
```

```
lda.class<-predict(modelo_LDA,newdata1,type=c("class"))$class
train_lda<-ifelse(ytrain==lda.class,0,1)
```

```
# Modelo Naive Bayes
tabla_NB_E<-table(train_NB,ytrain)
tabla_NB_E
```

```
##         ytrain
## train_NB   no  yes
##      no  6753  892
##      yes  539  187
```

```
# Knn
tabla_knn_E<-table(modelo_knn,ytrain)
tabla_knn_E
```

```
##           ytrain
## modelo_knn   no  yes
##        no  6906  981
```

```
##        yes  386   98
```

```
# Modelo logistico
tabla_lo_E<-table(glm.pred,ytrain)
tabla_lo_E
```

```
##         ytrain
## glm.pred   no  yes
##        0 7270 1075
##        1   22    4
```

```
# Modelo LDA
tabla_LDA_E<-table(train_lda,ytrain)
tabla_LDA_E
```

```
##          ytrain
## train_lda   no  yes
##         0 7248   40
##         1   44 1039
```

Las tablas anteriores son las respectivas matrices de confusion asociada a cada modelo. Note que los elementos de la diagonal principal son los aciertos de prediccion y cualquier elemento fuera de ella representa un error a la hora de predecir.

LDA es el modelo que mejor acierta a la hora de predecir, luego, seguiria Naive Bayes, Logistico y Knn en ultimo.

Se calcula el trainning MSE para cada modelo:

```
lista<-list(tabla_NB_E,tabla_knn_E,tabla_lo_E,tabla_LDA_E)
training_MSE<-NULL
for (i in lista) {
training_MSE<-c(training_MSE,(i[1,2]+i[2,1])/sum(i))
}
training_MSE<-data.frame(matrix(training_MSE,nrow = 1))
names(training_MSE)<-c("Naive Bayes","Knn","Logistico","LDA")
round(training_MSE,5)
```

```
##   Naive Bayes    Knn Logistico     LDA
## 1     0.17095 0.1633   0.13105 0.01003
```

Y se observa que el modelo con menor MSE es LDA.

Graficas ROC:

```
NB_E_ROC<-roc(response = ytrain, predictor = ifelse(train_NB=="yes",1,0))
```

```
## Setting levels: control = no, case = yes
```

```
## Setting direction: controls < cases
```

```
knn_E_ROC<-roc(response = ytrain, predictor = as.numeric(modelo_knn))
```

```
## Setting levels: control = no, case = yes
## Setting direction: controls < cases
```

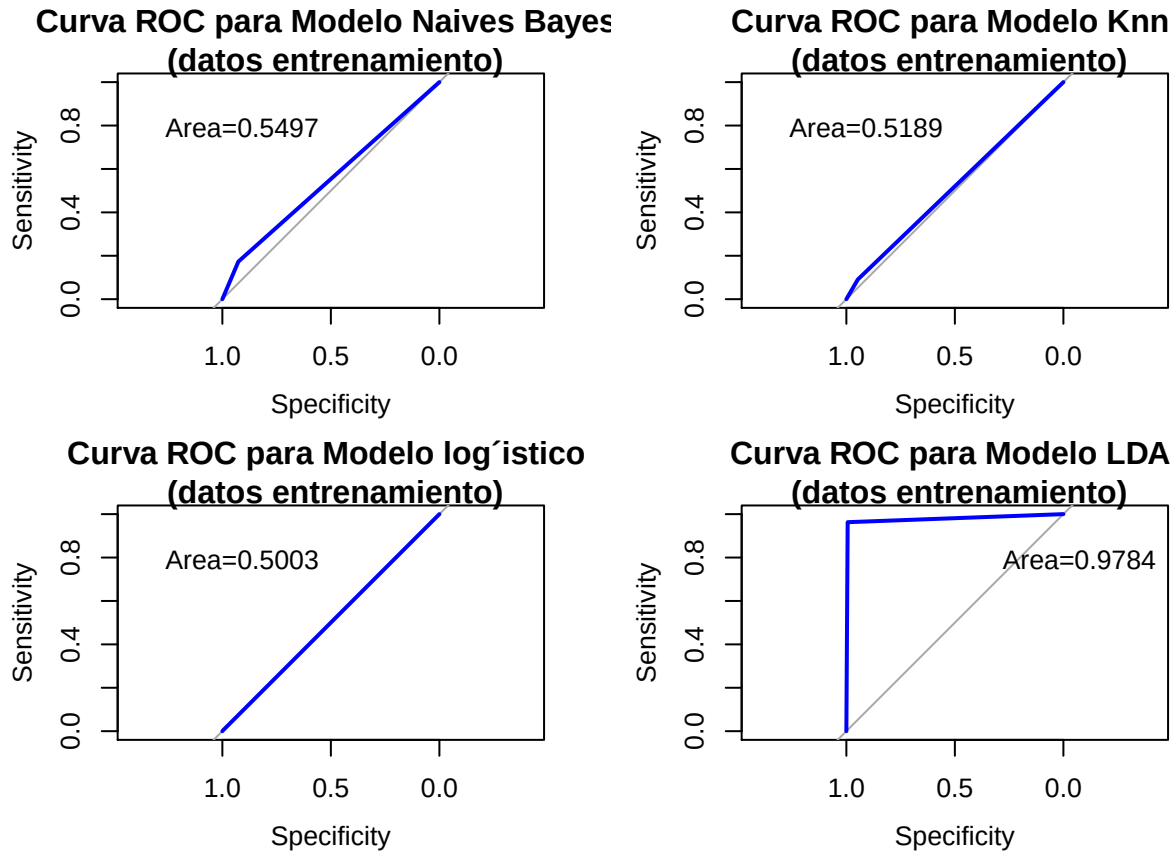
```
logis_E_ROC<-roc(response = ytrain, predictor = glm.pred)
```

```
## Setting levels: control = no, case = yes
## Setting direction: controls < cases
```

```
LDA_E_ROC<-roc(response = ytrain, predictor = train_lda)
```

```
## Setting levels: control = no, case = yes
## Setting direction: controls < cases
```

```
par(mfrow=c(2,2))
plot(NB_E_ROC,main="Curva ROC para Modelo Naives Bayes \n (datos entrenamiento)",
col="blue")
legend("topleft",paste("Area",as.character(round(NB_E_ROC$auc,4)),sep = "="),
bty = "n")
plot(knn_E_ROC,main="Curva ROC para Modelo Knn \n (datos entrenamiento)",
col="blue")
legend("topleft",paste("Area",as.character(round(knn_E_ROC$auc,4)),sep = "="),
bty = "n")
plot(logis_E_ROC,main="Curva ROC para Modelo log´ıstico \n (datos entrenamiento)",
col="blue")
legend("topleft",paste("Area",as.character(round(logis_E_ROC$auc,4)),sep = "="),
bty = "n")
plot(LDA_E_ROC,main="Curva ROC para Modelo LDA \n (datos entrenamiento)",
col="blue")
legend("topright",paste("Area",as.character(round(LDA_E_ROC$auc,4)),sep = "="),
bty = "n")
```



El modelo que, mejor resultados ofrece a la hora de predecir segun la corva ROC es el modelo LDA. Basta ver la curva y su valor AUC de casi 100%.

G) La matriz de consfusion para los 4 modelos de prueba es:

```
test_NB<-predict(modelo_NB,newdata=newdata2,type="class")
test_knn<-knn(train = train2n, test = test2n, cl = ytrain2, k=2, prob=F)
pred<- predict(modelo_logistico,newdata = test2n,type = "response")
```

```
## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from a rank-deficient fit may be misleading
```

```
test_logs<-ifelse(pred >= 0.5, 1, 0)
lda.class_t<-predict(modelo_LDA,newdata2,type=c("class"))$class
test_LDA<-ifelse(ytest==lda.class_t,0,1)
tabla_NB_T<-table(test_NB,ytest)
tabla_knn_T<-table(test_knn,ytest)
tabla_lo_T<-table(test_logs,ytest)
tabla_LDA_T<-table(test_LDA,ytest)
```

```
tabla_NB_T
```

```
##        ytest
## test_NB   no  yes
##     no  2211  308
##     yes  199   73
```

```
tabla_knn_T
```

```
##         ytest
## test_knn   no  yes
##      no  2218  352
##      yes  192   29
```

```
tabla_lo_T
```

```
##          ytest
## test_logs   no  yes
##         1 2410  381
```

En este resultado, el valor asignado a la segunda fila (1) es cero. Implicando que la matriz es de la 2x1, 2 columnas y una fila

```
tabla_LDA_T
```

```
##         ytest
## test_LDA   no  yes
##        0 2389    9
##        1   21  372
```

Calculo del MSE para cada modelo:

Para Naiev Bayes

```
MSE_NB_T = (tabla_NB_T[1,2]+tabla_NB_T[2,1])/sum(tabla_NB_T[,])
MSE_NB_T
```

```
## [1] 0.1816553
```

Para Knn

```
MSE_Knn_t = (tabla_knn_T[1,2]+tabla_knn_T[2,1])/sum(tabla_knn_T[,])
MSE_Knn_t
```

```
## [1] 0.1949122
```

Para Regresion Losgistica:

```
MSE_RL =  (tabla_lo_T[1,2])/sum(tabla_NB_T[,])
MSE_RL
```

```
## [1] 0.1365102
```

Para LDA:

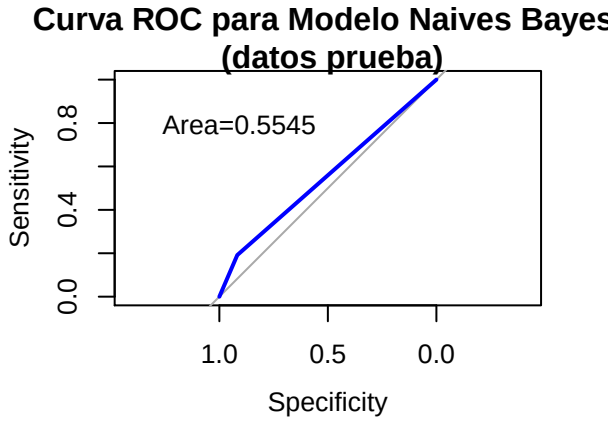
```
MSE_LDA = (tabla_LDA_T[1,2]+tabla_LDA_T[2,1])/sum(tabla_LDA_T[,])
MSE_LDA
```

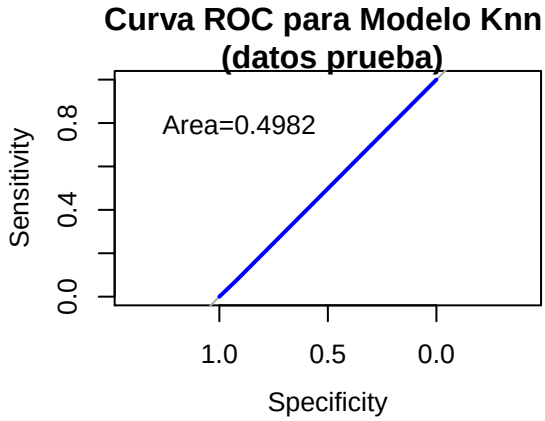
```
## [1] 0.01074884
```

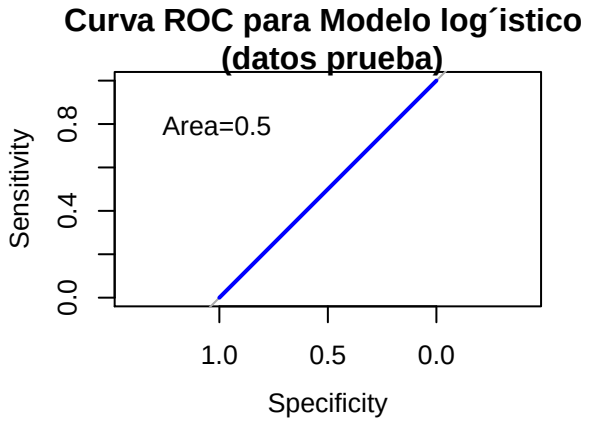
El MSE mas pequeño es de LDA.

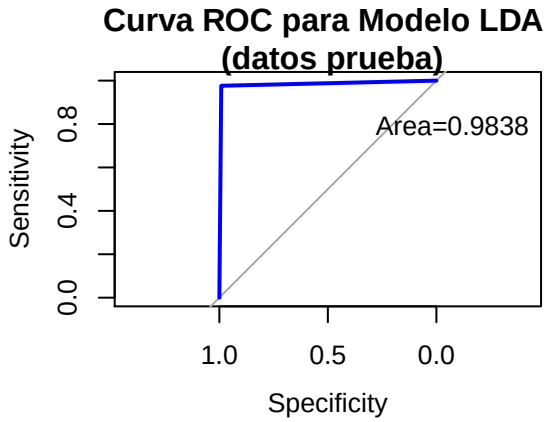
Finalmente, graficas ROC:

```
## Setting levels: control = no, case = yes
## Setting direction: controls < cases
## Setting levels: control = no, case = yes
## Setting direction: controls < cases
## Setting levels: control = no, case = yes
## Setting direction: controls < cases
## Setting levels: control = no, case = yes
## Setting direction: controls < cases
```









H) El modelo con mejor desempeño fue el del modelo de LDA dada las estimaciones de sus parametros, teniendo en cuenta que fue el que mejor MSE ha tenido a lo largo de todas las pruebas.