Práctica 2: Big Data y minería de datos

Manuel Torres Acosta

Índice

Resumen / Abstract

Español: los árboles de decisión son potentes modelos cuya lógica interna se puede visualizar mediante una representación intuitiva. Aplicamos uno de éstos modelos sobre un conjunto de datos que estudia la prevalencia de la diabetes entre los indígenas del pueblo pima para evaluar su efectividad a la hora de predecir la presencia o ausencia de enfermedad.

English: decision trees are powerful models whose inner logic can be represented with an intuitive visualization. We train one of these models on a dataset about the prevalence of diabetes among the pima indians to test it's ability to predict the presence or absence of the disease.

Objetivos

La práctica actual consiste en aplicar un árbol de decisión sobre una de las problemáticas propuestas: predecir la probabilidad de fallecimiento por coronavirus o la probabilidad de tener diabetes. Cada tema tiene su correspondiente conjunto de datos para entrenar el modelo, y en mi caso he escogido la temática de la diabetes ya que tras examinar los datos del coronavirus encontré algunas peculiaridades que me llevaron a pensar que quizás el modelo generado no sería demasiado rico. La inmensa mayoría de los fallecidos pertenecen a la población de Wuhan en los datos, de modo que es muy posible que el árbol resultante esté basado únicamente en dicha variable. Por ello seleccioné los datos de la diabetes, la mayor diversidad de datos puede llevar a que el modelo tenga más variables en cuenta, además de que la calidad de los datos era mayor (menos valores perdidos).

En cualquier caso, debemos ajustar el modelo y comprobar su rendimiento sobre el conjunto de datos seleccionado.

Método

Muestra

```
#Cargamos los datos
originalData <- read.csv("diabetes.csv")
```

El fichero original contiene 768 observaciones y 9 variables. Seleccionamos únicamente las variables relevantes para el modelo y nos quedamos con las observaciones que tengan todos los datos completos.

```
df <- originalData

#Quitamos las variables que no vamos a usar
df$Outcome <- NULL

#Transformamos valores extraños en NA
df$Pregnancies[df$Pregnancies == 0] = 999
df[df == 0] = NA

#Restablecemos los ceros en la variable Embarazos
df$Pregnancies[df$Pregnancies == 999] = 0

#Limpiamos los perdidos
df <- df[complete.cases(df), ]

#Calculamos la variable a predecir
df$Diabetes <- ifelse(df$Glucose < 100,
                      "Sin diabetes",
                      "Con diabetes")

#Eliminamos la variable glucosa
df$Glucose <- NULL

#Guardamos los nombres de las variables predictoras
VariablesUsed <- colnames(df)[1:7]

#Restablecemos los identificadores de fila
rownames(df) <- 1:nrow(df)
```

Tras los filtros, el fichero contiene 392 observaciones y 8 variables. Hemos descartado 376 observaciones y la variable Glucosa, ya que la hemos usado para construir la variable a predecir. Si la conservamos, el árbol de decisión se basaría únicamente en ella para hacer las clasificaciones. También convertimos los valores 0 en perdidos, excepto para la variable embarazos (Hayashi & Yukita, 2016).

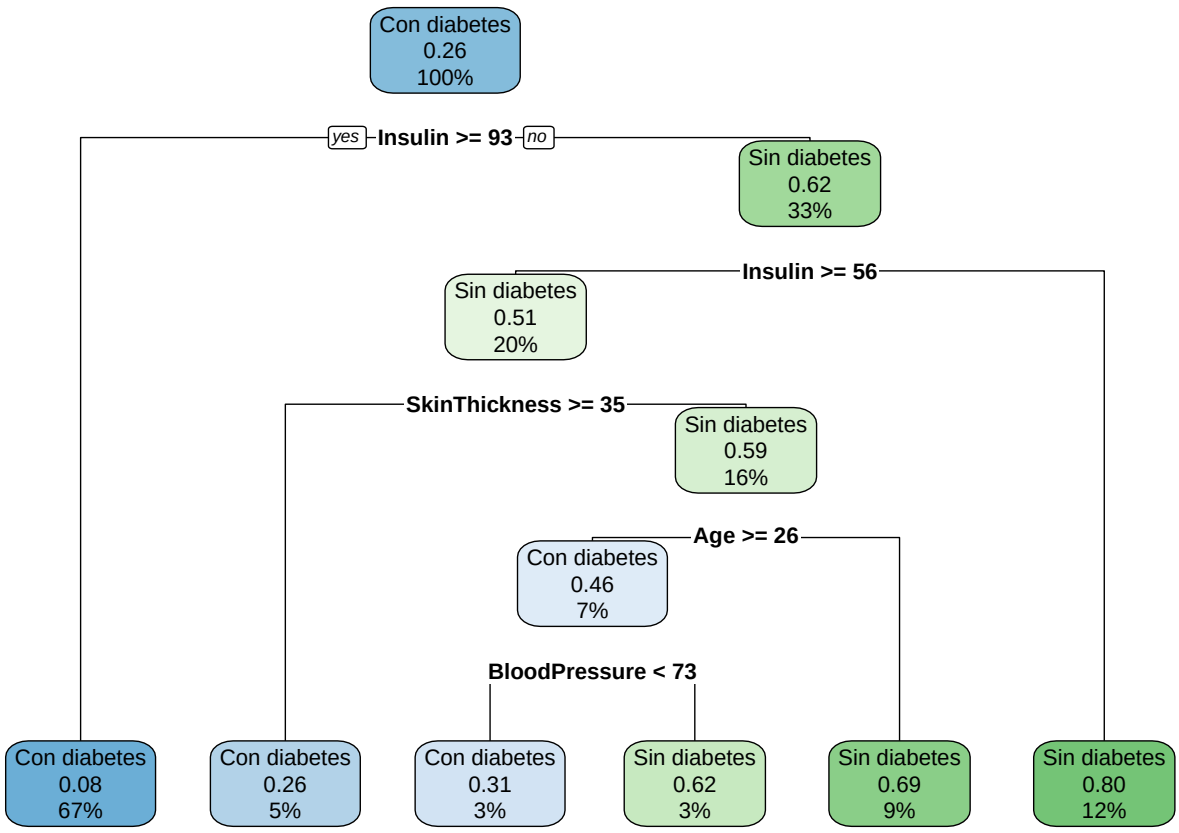
Construcción del árbol

Para construir el árbol de decisión utilizamos la función `rpart()`. El parámetro `formula` indica la variable a predecir y las que usaremos como predictoras. En nuestro caso utilizamos "." para indicarle que use todas las del conjunto de datos. El parámetro `method` indica que queremos que utilice un clasificador, pues la misma función se puede utilizar para resolver problemas de regresión (variable a predecir numérica).

```
library(rpart)
library(rpart.plot)

#Creamos el arbol utilizando todas las variables
Arbol <- rpart(formula = Diabetes ~ .,
               data = df,
               method = "class")

# plot(Arbol) ; text(Arbol) #Sin paquetes adicionales
rpart.plot(Arbol,
           cex = 1.5)
```



Resultados

Rendimiento del modelo

```
#Utilizamos el modelo para generar los pronósticos
pred <- predict(Arbol,
                data = df[VariablesUsed],
                type = "prob")

#Nos devuelve la probabilidad de que tenga diabetes
#Dicotomizamos la clasificacion
df$Pronostico <- ifelse(pred[, "Con diabetes"] > 0.5,
                "Con diabetes", "Sin diabetes")

caret::confusionMatrix(data = as.factor(df$Pronostico),
                reference = as.factor(df$Diabetes))
```

```
## Confusion Matrix and Statistics
##
##                Reference
## Prediction     Con diabetes Sin diabetes
##   Con diabetes          264           31
##   Sin diabetes           26           71
##
##                Accuracy : 0'8546
##                  95% CI : (0'8157, 0'888)
##     No Information Rate : 0'7398
##     P-Value [Acc > NIR] : 0'00000002844
##
##                   Kappa : 0'6162
##
##  Mcnemar's Test P-Value : 0'5962
##
##             Sensitivity : 0'9103
##             Specificity : 0'6961
##          Pos Pred Value : 0'8949
##          Neg Pred Value : 0'7320
##              Prevalence : 0'7398
##          Detection Rate : 0'6735
##    Detection Prevalence : 0'7526
##       Balanced Accuracy : 0'8032
##
##        'Positive' Class : Con diabetes
##
```

Conclusiones

Podemos comprobar que el modelo funciona razonablemente bien sobre el conjunto de datos, pues ha logrado conseguir una precisión (*accuracy*) de 0'855. En las instrucciones de la práctica se especificaba que debíamos calcular dicha métrica de rendimiento, pero la salida del modelo no es compatible con ella y tuvo que ser dicotomizada. Por defecto el modelo devuelve la probabilidad de que una observación pertenezca a cada categoría, lo cual aporta más información.

En las instrucciones también se entiende que debemos estimar el rendimiento del modelo sobre los mismos datos sobre los que fue entrenado, pero normalmente se suele dividir la muestra entre entrenamiento y prueba.

En lo que a la interpretación del árbol de decisiones se refiere, para cada nodo muestra el diagnóstico más probable hasta el momento, así como la probabilidad de que no tenga diabetes. El porcentaje representa la cantidad de la muestra que se incluye en ése punto.

Anexos[1]

Variables (1):

Pregnancies	BloodPressure	SkinThickness	Insulin	BMI
1	66	23	94	28'1
0	40	35	168	43'1
3	50	32	88	31'0
2	70	45	543	30'5
1	60	23	846	30'1

Variables (2):

DiabetesPedigreeFunction	Age	Diabetes	Pronostico
0'167	21	Sin diabetes	Con diabetes
2'288	33	Con diabetes	Con diabetes
0'248	26	Sin diabetes	Con diabetes
0'158	53	Con diabetes	Con diabetes
0'398	59	Con diabetes	Con diabetes

[1]No se incluyen todos los datos en el informe por el tamaño de la matriz. Solo las primeras filas para todas las variables

Referencias

Datos obtenidos de kaggle

Hayashi, Y., & Yukita, S. (2016). Rule extraction using recursive-rule extraction algorithm with j48graft combined with sampling selection techniques for the diagnosis of type 2 diabetes mellitus in the pima indian dataset.