Big Data, Machine Learning & Business Intelligence

Lab: Implementar Arboles de Decisión en Lenguaje R

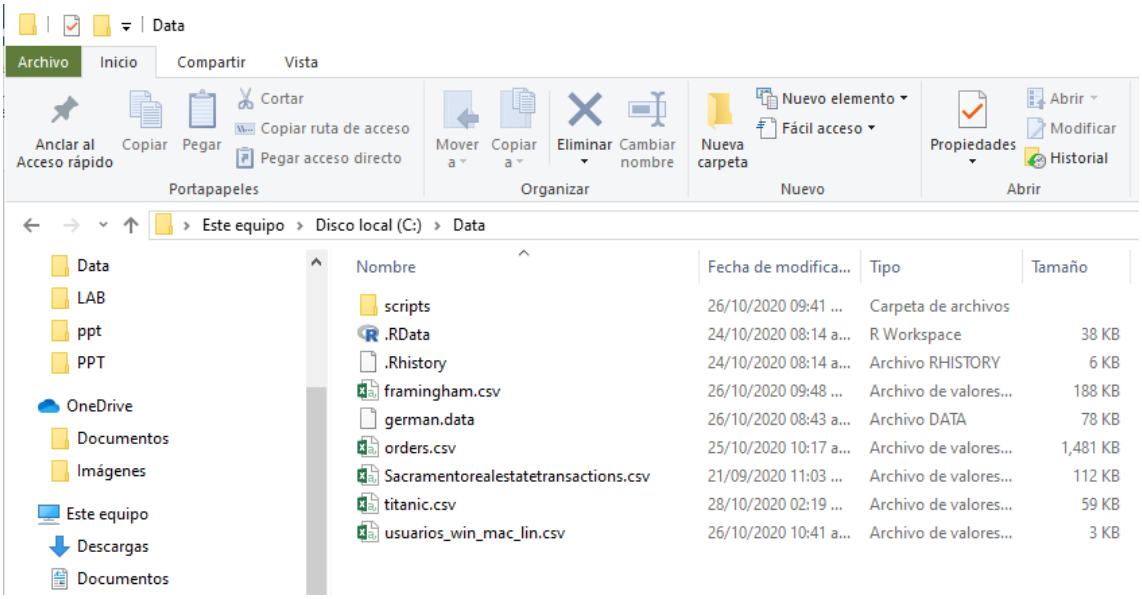
Objetivos

- Implementar el algoritmo de machine learning supervisado arboles de decisión en el lenguaje R

Procedimiento

Importación de Datos

1. Descargar el archivo titanic.csv, del repositorio GitHub del curso y copiarlo en C:/Data



2. Importar los datos y crear el data frame.

```
setwd("C:/Data")
titanic <-read.csv("titanic.csv")
str(titanic)
head(titanic)
```

3. Crear el cojunto de datos de entrenamiento y el de pruebas.

   Los datos no están ordenados aleatoriamente sino secuencialmente de acuerdo a la variable categórica de interés. Esto es un problema importante y se debe corregir

antes de dividir los datos en entrenamiento y test. Para desordenar la lista de observaciones, se puede usar la función sample().

```
shuffle_index <- sample(1:nrow(titanic))
head(shuffle_index)
```

Ahora se usa estos índices para generar un ordenamiento aleatorio del conjunto de datos.

```
titanic <- titanic[shuffle_index, ]
head(titanic)
```

Limpiar los Datos

4. Limpia los datos para eliminar los NA y realiza las transformaciones covenientes
   El conjunto de datos tiene algunas inconvenientes:
   - Existen valores NA's, por lo tanto deben ser eliminados.
   - Prescindir de variables innecesarias
   - Crear-convertir variables a tipo factor de ser necesario (e.g., pclass y survived)

```
#Limpiar datos
library(dplyr)
# Elimina Variables
clean_titanic <- titanic %>%
 select(-c( Cabin, Name, Ticket)) %>%
 #Convierte a factor
 mutate(Pclass = factor(Pclass, levels = c(1, 2, 3), labels = c('Upper', 'Middle', 'Lower')),
        Survived = factor(Survived, levels = c(0, 1), labels = c('No', 'Yes'))) %>%
 na.omit()
glimpse(clean_titanic)
```

Dividir el conjunto de Datos

5. Divide el conjunto de datos, 80% para entrenamiento y 20% para test.

```
library(dplyr)
data_train <- clean_titanic %>% dplyr::sample_frac(.8)
data_test  <- dplyr::anti_join(clean_titanic, data_train, by = 'PassengerId') # se debe
tener un id
data_train <- dplyr::select(data_train, -PassengerId )
data_test <- dplyr::select(data_test, -PassengerId )
head(data_train)
dim(data_train)
```

```
dim(data_test)
```

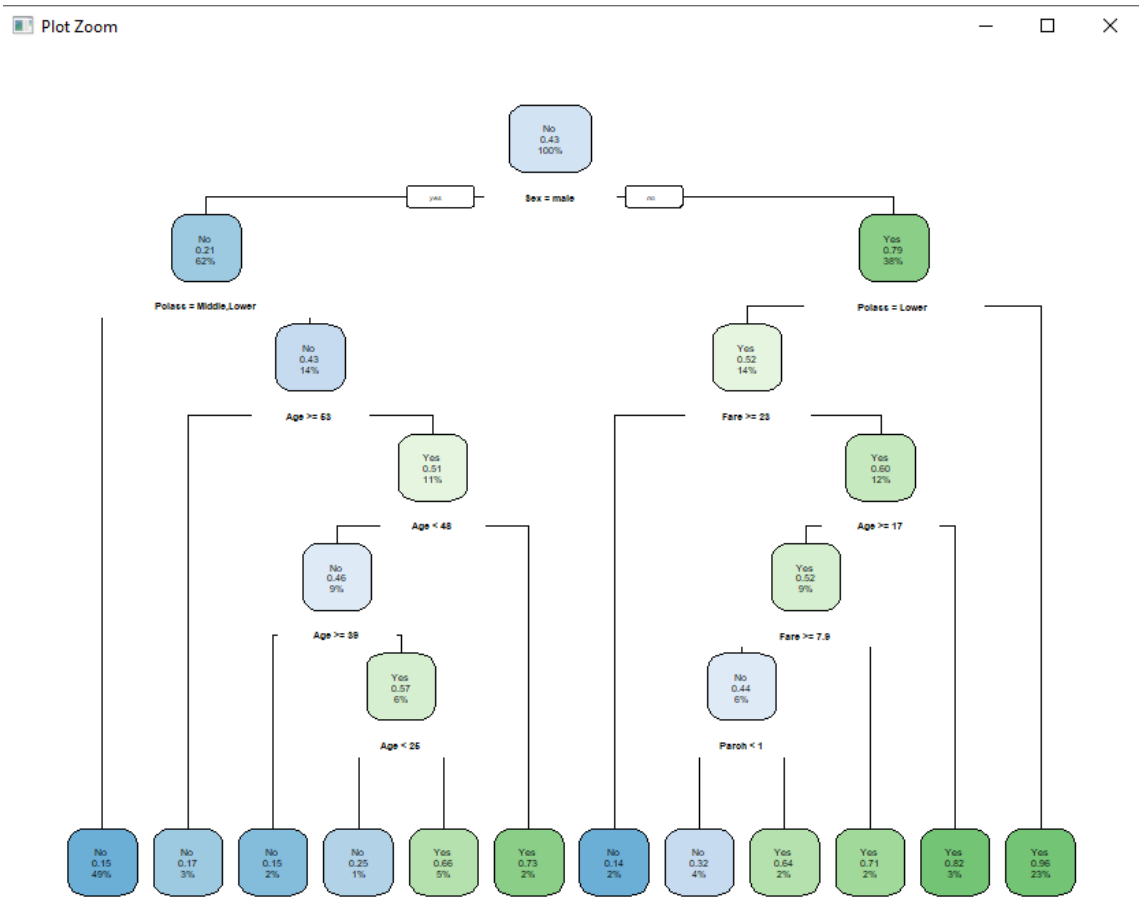
¿Cuantas filas tiene el conjunto de datos de entrenamiento?, ¿Cuantas el conjunto de datos de Pruebas?

Construcción del Modelo

6. Construir el modelo. Para tener graficos vistosos instala el paquete rpart.plot.

```
install.packages("rpart.plot")
library(rpart)
library(rpart.plot)

fit <- rpart(Survived~., data = data_train, method = 'class')
rpart.plot(fit, extra = 106)
```



Cada nodo muestra

- La clase predecida (died o survived),
- La probabilidad predecida de survival,
- El porcentaje de observaciones en el nodo.

¿Cuál es la variable más significativa para dividir el conjunto de datos?

- Los árboles de decisión requieren muy poca preparación de datos. Particularmente, no requieren escalamiento de atributos o centrado.
- Por defecto, rpart() use la medida de Gini para la división de los nodos.

Predicción

7. Realizar la predicción, comprobar con los datos de prueba y mostrar la matrix de confusión.

   Accuracy: Es un indicador del modelo que mide la precisión del modelo. Representa el número de las predicciones correctas respecto del total de observaciones.

```
# Prediccion
predict_unseen <-predict(fit, data_test, type = 'class')

# Matriz de Confusion y Accuracy
table_mat <- table(data_test$Survived, predict_unseen)
table_mat
accuracy_Test <- sum(diag(table_mat)) / sum(table_mat)
print(paste('Accuracy (Precision del Modelo):', accuracy_Test))
```

   ¿Cuál es el valor de Accuracy del modelo?

   ¿Cuál es el valor Accuracy si seleccionas 70% de las observaciones para entrenamiento y 30% para pruebas?

   _____