Algoritmos machine learning

Andres Hernandez Moncada cod: 95663 Andres Mendoza cod: 86204 David Martinez cod: 81639 Jaime Gonzalez cod:33236 Juan Sebastian Rodriguez cod: 92699

2023-05-29

Ambientes

Diseñamos tres ambientes con techos a diferentes alturas y una pared de diferente color

• Hábitat 1 color verde.

• Hábitat 2 color azul.

• Hábitat 3 color amarillo.

Adquisición de datos

Para la adquisición de datos utilizamos los siguientes sensores ultrasonido HC-SR04, sensor de color TCS3200, para la captura de datos utilizamos un arduino uno el cual comunicamos por bluetooth al computador y realizamos la captura por medio de PLX-DAQ en excel una vez obtenidas las lecturas procedemos a realizar el postprocesado y guardamos el archivo en formato .csv.

Datos adquiridos por arduino

Datos para entrenar los modelos de machine learning con 216 observaciones.

```
##             hora ultrasonido rojo verde azul   habitat
## 1 12:10:57 a. m.           0   12   104  104 habitat 1
## 2 12:10:57 a. m.         514   12   104  104 habitat 1
## 3 12:10:57 a. m.         519   63   102  102 habitat 1
## 4 12:10:57 a. m.         514   12   104   98 habitat 1
## 5 12:10:57 a. m.         520   40   104  104 habitat 1
## 6 12:10:57 a. m.         520   66   102  102 habitat 1
```

Datos para predicción con 6 observaciones dos por ambiente, las dos primeras observaciones corresponden al hábitat 1, las dos siguientes son del hábitat 3 y las últimas dos corresponden al hábitat 2.

```
##   ultrasonido rojo verde azul
## 1         496   82    51   48
## 2         984   85    47   50
## 3        1515   16    22   35
## 4        1515   16    22   35
## 5        1083  235   213  120
## 6        1065  221   187  104
```

Modelo KNN

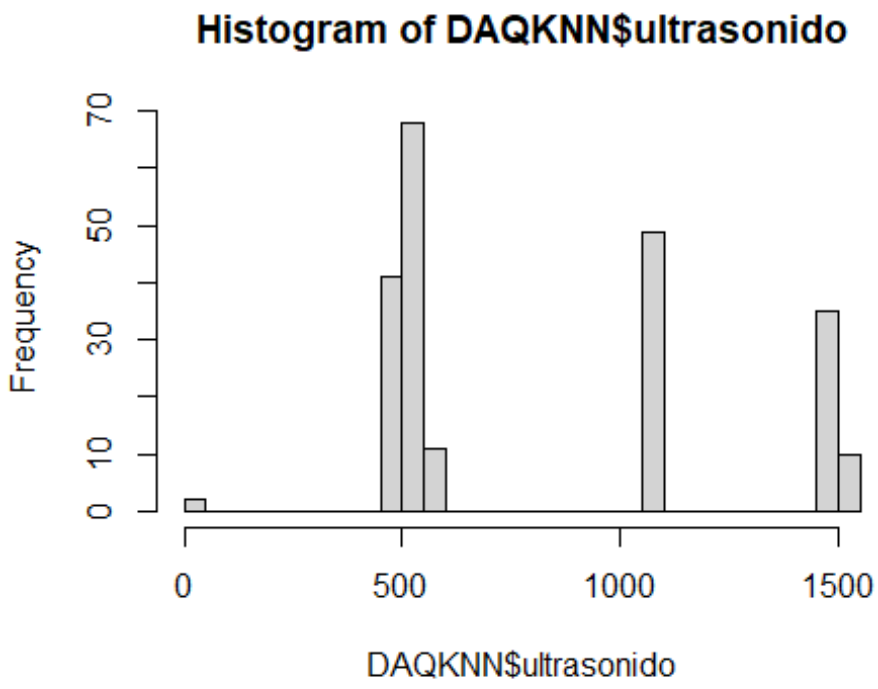
Código de predicción para las variables de los sensores de ultrasonido.

```
library(tidyverse)
library(dplyr)
Prediccion <- read.csv("C:/Users/USER/Desktop/machine
3corte/Prediccion.csv", sep=";")
DAQEntrenamiento <- read.csv("C:/Users/USER/Desktop/machine
3corte/DAQEntrenamiento.csv", sep=";")

X<-Prediccion$ultrasonido
Y<-Prediccion$real
B<-cov(X,Y)/var(X)
A<-mean(Y)-B*mean(X)

normalise <-function(x){
  return((x-min(x))/(max(x)-min(x)))}

DAQKNN=DAQEntrenamiento
DAQKNN<-mutate(DAQKNN,regresion_ultrasonido=A+ultrasonido*B)
```

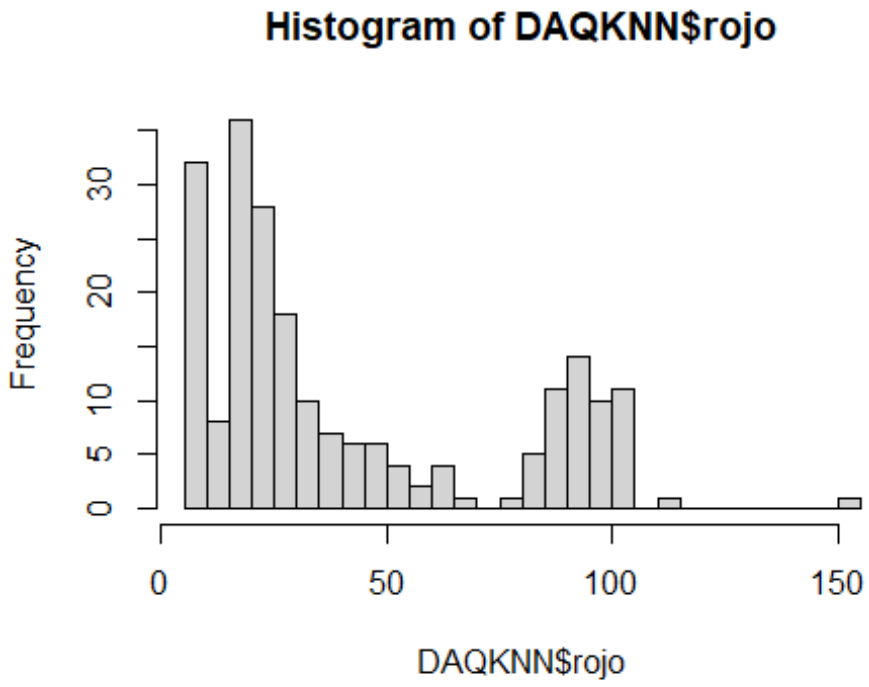
Analisis de datos

Como podemos observar en las graficas de histogramas para cada variable es posible diferencias cada habitat en el histograma del sensor de ultra sonido y en el histograma del valor del sensor rgvb que corresponde al color rojo.

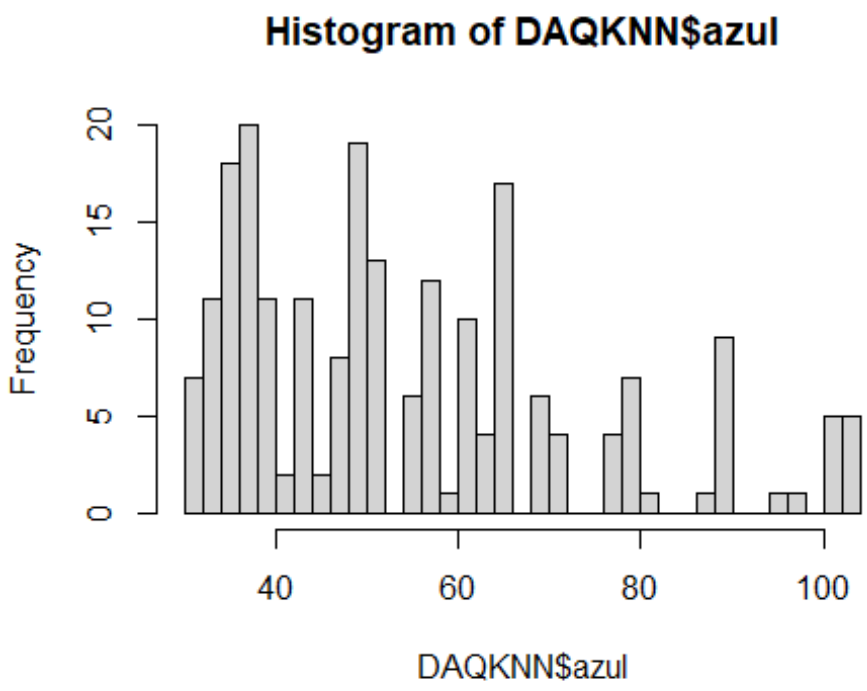
```
hist(DAQKNN$ultrasonido,breaks=50)
```



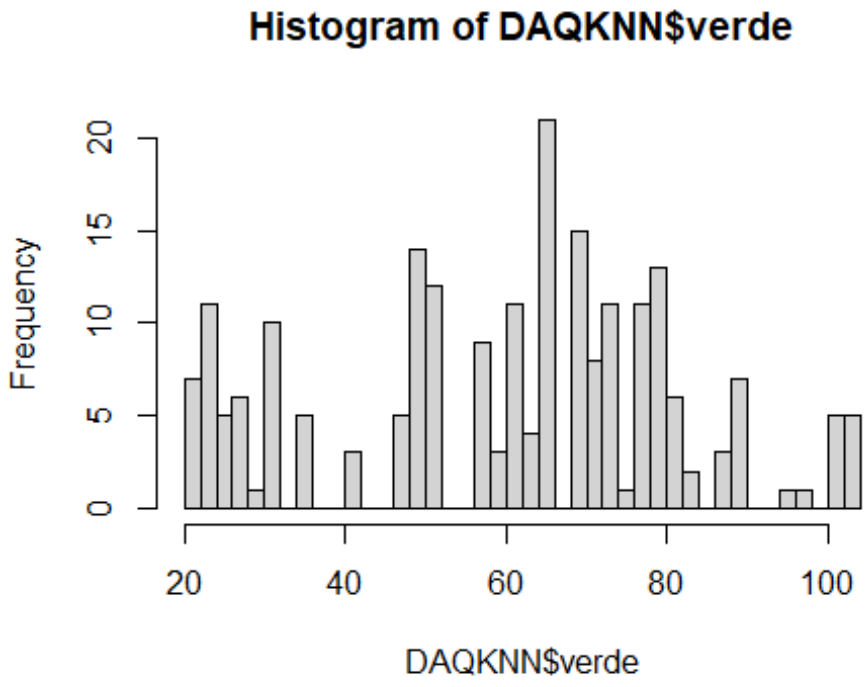

```
hist(DAQKNN$rojo,breaks=50)
```




```
hist(DAQKNN$azul,breaks=50)
```

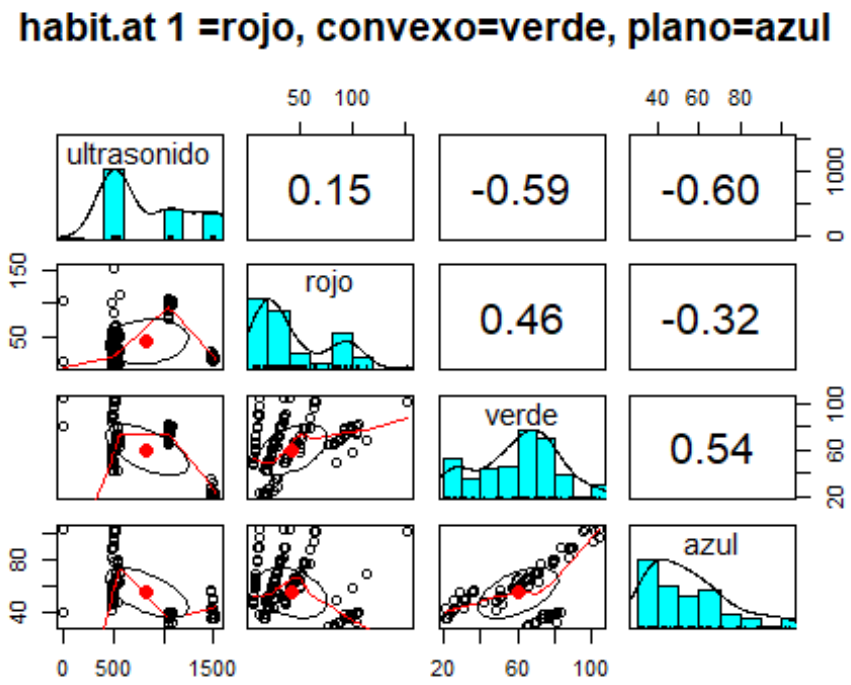



```
hist(DAQKNN$verde,breaks=50)
```




```
library(psych)
pairs.panels(DAQKNN[2:5], pch=21, main=("habit.at 1 =rojo, convexo=verde,
```

```
plano=azul")
             , bg=c("red","green2","blue")[unclass(DAQKNN$habitad)])
```



Proporción de dataset

```
muestra <-DAQKNN[1:5,]
prop.table((table(muestra$ultrasonido)))

##
##   0 514 519 520
## 0.2 0.4 0.2 0.2

prop.table((table(muestra$rojo)))

##
##  12  40  63
## 0.6 0.2 0.2

prop.table((table(muestra$azul)))

##
##  98 102 104
## 0.2 0.2 0.6

prop.table((table(muestra$verde)))

##
## 102 104
## 0.2 0.8
```

podemos observar las proporciones para cada variable predictora, al ser números menores a cero nos indica que los datos adquiridos son aptos para técnicas de machine learning.

ingenieria de caracteristica

Normalizamos los datos de los sensores por el metodo de z score.

```
normData<-DAQKNN
standarData<-DAQKNN
normData$ultrasonido<-normalise(normData$ultrasonido)

normData$rojo<-normalise(normData$rojo)
normData$azul<-normalise(normData$azul)
normData$verde<-normalise(normData$verde)



standarData$ultrasonido<-scale(normData$ultrasonido)
standarData$rojo<-scale(normData$rojo)
standarData$azul<-scale(normData$azul)
standarData$verde<-scale(normData$verde)

colnames(standarData)[1:5]<-
c("ultrasonido[,1]","laser[,1]","rojo[,1]","azul[,1]","verde[,1]")
```

KNN

Para entrenar el modelo de KNN utilizamos el 70% del dataset DAQEntrenamiento el cual tiene un total de 216 observaciones, para los datos de prueba utilizamos el 30% de los datos.

Utilizamos la librería class para entrenar el modelo y la librería gmodels para verificar el rendimiento del modelo KNN.

```
sample.index<-sample(3:nrow(DAQKNN)
                     , nrow(DAQKNN)*0.7
                     , replace=FALSE)
k<-3

predictors<-c("ultrasonido","rojo",
              "azul","verde")
train.data<-DAQKNN[sample.index
                   ,c(predictors,"habitat")
                   , drop=F]
test.data<-DAQKNN[-sample.index
                  ,c(predictors,"habitat")
                  , drop=F]
library(class)
predictors<- knn(train=train.data[predictors]
                   ,test=test.data[predictors]
```

```
                  ,cl=train.data$habitat
                  ,k=k)
```

Verificación del rendimiento

La verificación la realizamos con 65 datos y obtenemos un bajo porcentaje de error.

```
library(gmodels)
CrossTable(x=test.data$habitat,y=predictors)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## | Chi-square contribution |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  65
##
##
##                   | predictors
## test.data$habitat | habitat 1 | habitat 2 | habitat 3 | Row Total |
## ------------------|-----------|-----------|-----------|-----------|
##         habitat 1 |        36 |         0 |         0 |        36 |
##                   |     8.654 |     6.092 |     7.754 |           |
##                   |     1.000 |     0.000 |     0.000 |     0.554 |
##                   |     0.900 |     0.000 |     0.000 |           |
##                   |     0.554 |     0.000 |     0.000 |           |
## ------------------|-----------|-----------|-----------|-----------|
##         habitat 2 |         1 |        11 |         0 |        12 |
##                   |     5.520 |    39.614 |     2.585 |           |
##                   |     0.083 |     0.917 |     0.000 |     0.185 |
##                   |     0.025 |     1.000 |     0.000 |           |
##                   |     0.015 |     0.169 |     0.000 |           |
## ------------------|-----------|-----------|-----------|-----------|
##         habitat 3 |         3 |         0 |        14 |        17 |
##                   |     5.322 |     2.877 |    29.191 |           |
##                   |     0.176 |     0.000 |     0.824 |     0.262 |
##                   |     0.075 |     0.000 |     1.000 |           |
##                   |     0.046 |     0.000 |     0.215 |           |
## ------------------|-----------|-----------|-----------|-----------|
##      Column Total |        40 |        11 |        14 |        65 |
##                   |     0.615 |     0.169 |     0.215 |           |
## ------------------|-----------|-----------|-----------|-----------|
##
##
```

regresión logística Para el modelo de regresión logística utilizamos la librería (caret), cargamos los datos de entrenamiento y los datos reales para predecir.

Aplicamos cross validation de 5 divisiones y 10 repeticiones para mejorar el desempeño.

```
library(caret)
DAQEntrenamiento <- read.csv("C:/Users/USER/Desktop/machine
3corte/DAQEntrenamiento.csv", sep=";")
DAQReal <- read.csv("C:/Users/USER/Desktop/machine 3corte/DAQReal.csv",
sep=";")
fit.control <- trainControl(method = "repeatedcv"
                            , number = 5, repeats = 10)
```

entrenamos el modelo fit estableciendo la variable habitat como clasificatoria y las variables de los sensores como predictores, el método del modelo es multinom debido a que tenemos 3 ambientes para clasificar, al parámetro trControl le cargamos el modelo de cross validation previamente configurado.

```
fit <- train(habitat ~ ultrasonido + rojo + azul+verde,data =
DAQEntrenamiento
             , method = "multinom"
             , trControl = fit.control
             , trace = FALSE)
```

El modelo utiliza el mayor valor de exactitud en este caso es decay = 0.1.

```
fit

## Penalized Multinomial Regression
##
## 216 samples
##   4 predictor
##   3 classes: 'habitat 1', 'habitat 2', 'habitat 3'
##
## No pre-processing
## Resampling: Cross-Validated (5 fold, repeated 10 times)
## Summary of sample sizes: 173, 172, 173, 173, 173, 173, ...
## Resampling results across tuning parameters:
##
##   decay  Accuracy   Kappa
##   0e+00  0.9671247  0.9446729
##   1e-04  0.9666702  0.9439426
##   1e-01  0.9768288  0.9606639
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was decay = 0.1.
```

Predicción con datos fuera del entrenamiento

Al revisar los porcentajes de predicción nos podemos dar cuenta que nos genera un error en una predicción ya que en la segunda predicción el algoritmo predijo que se encontraba en el hábitat 3 con el 66% de certeza, pero se equivocó ya que los datos de la segunda predicción corresponden al habitat 1. las demás predicciones si son correctas.

```
predict(fit, DAQReal, type="prob")
```

```
##      habitat 1    habitat 2    habitat 3
## 1 9.313432e-01 6.391311e-03 6.226551e-02
## 2 3.326255e-01 2.161328e-03 6.652132e-01
## 3 4.668481e-03 4.424667e-06 9.953271e-01
## 4 4.668481e-03 4.424667e-06 9.953271e-01
## 5 7.638357e-05 9.999236e-01 5.810470e-16
## 6 2.245754e-05 9.999775e-01 4.088986e-15
```

decision tree Utilizamos la librería (rpart) para entrenar el modelo de predicción decision tree incluimos los dataset de entrenamiento y el real para la predicción.

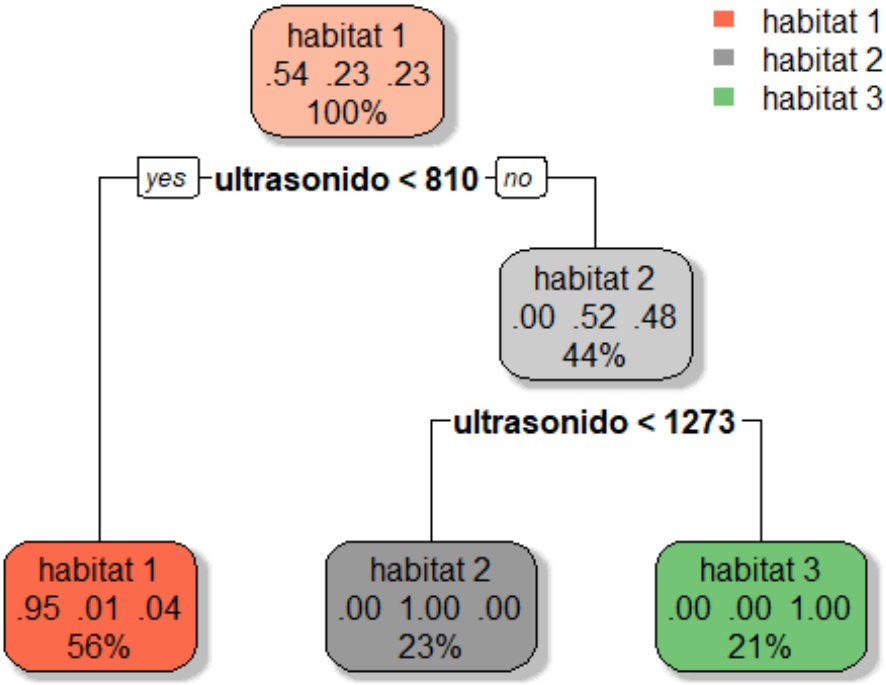
```
library(rpart)
DAQEntrenamiento <- read.csv("C:/Users/USER/Desktop/machine
3corte/DAQEntrenamiento.csv", sep=";")
DAQReal <- read.csv("C:/Users/USER/Desktop/machine 3corte/DAQReal.csv",
sep=";")
```

entrenamos el modelo con la variable habitat como clase y las variables de los sensores como predictoras, utilizamos el método class para entrenar el modelo como clasificación y le decimos que data es igual a los datos de entrenamiento.

```
library(rpart)
fit <-
  rpart(habitat ~ ultrasonido + rojo + azul+verde,
        method = "class",
        data= DAQEntrenamiento)
```

Para los datos de entrenamiento en el modelo de árbol de predicción podemos observar que tenemos un margen de error del 5% al predecir el hábitat 1, para predecir los habita 2 y 3 tenemos un 100% de predicción.

```
rpart.plot::rpart.plot(fit, shadow.col = "gray")
```

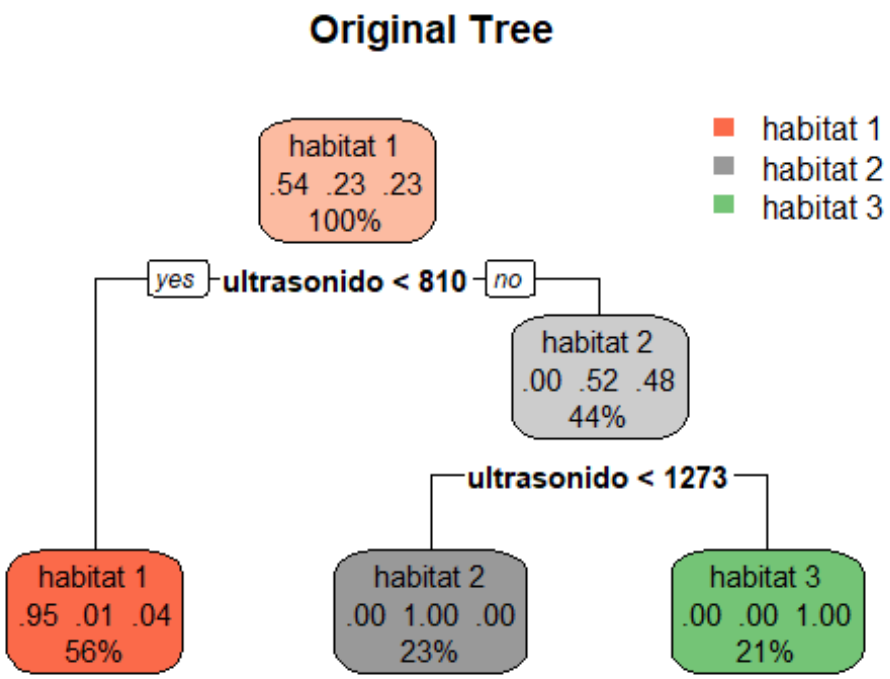


Podamos el árbol con el mínimo error para hacer que sea más eficiente el proceso de predicción.

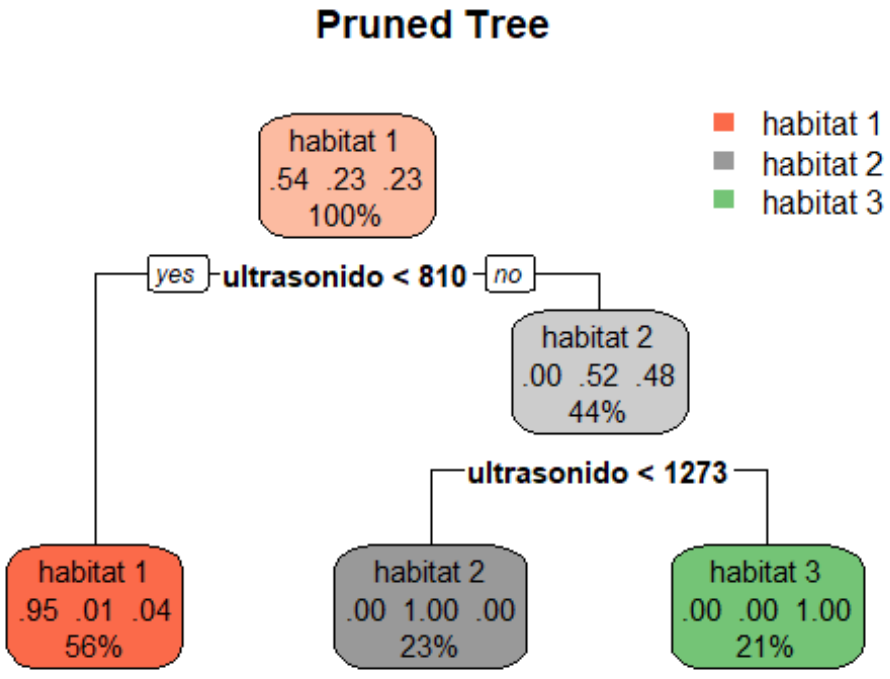
```
fit.pruned <-
  prune(fit, cp=fit$cptable[which.min(fit$cptable[,
                                                  "xerror"]),"CP"])

par(mfrow= c(1, 2))
```

Al comparar el árbol original con el árbol podado nos damos cuenta que el original ya se encuentra reducido al máximo para el dataset de entrenamiento.

```
rpart.plot::rpart.plot(fit,main= "Original Tree")
```




```
rpart.plot::rpart.plot(fit.pruned,main= "Pruned Tree")
```



Predicción con datos fuera del entrenamiento

Como podemos observar tenemos un error con la segunda predicción ya que esta es del hábitat 1 no del hábitat 2 las demás predicciones si son correctas ya que el robot capturo dos datos por cada hábitat en el siguiente orden hábitat 1, hábitat 3 y hábitat 2.

```
predict(fit, DAQReal, type="prob")
```

```
##   habitat 1   habitat 2  habitat 3
## 1 0.9508197 0.008196721 0.04098361
## 2 0.0000000 1.000000000 0.00000000
## 3 0.0000000 0.000000000 1.00000000
## 4 0.0000000 0.000000000 1.00000000
## 5 0.0000000 1.000000000 0.00000000
## 6 0.0000000 1.000000000 0.00000000
```