

# Segunda entrega

Diego Cubides, Larry castro, Tomas Mendez y Jersson avila

2023-04-20

## Contents

Introducción . . . . .	1
Materiales . . . . .	1
Software . . . . .	1
Procedimiento . . . . .	2
<b>1.1 adquisicion de datos</b>	<b>2</b>
<b>1.2 Modelo lineal y regresion multilineal</b>	<b>3</b>
<b>2.1 Adquisición de datos</b>	<b>9</b>
Validacion de modelos . . . . .	15

## Introducción

El presente reporte está basado en la implementación de un aprendizaje de máquina para poder predecir tres tipos de obstáculos mediante el algoritmo de Knn y la distancia bajo un modelo lineal y otro multilíneal de dos sensores incorporados en un carro a control remoto. Este robot fue implementado en Arduino y una app móvil con el fin de obtener datos de un sensor infrarrojo y un ultrasónico a distancias y obstáculos diferentes.

## Materiales

- Sensor ultrasónico US-016
- Sensor Infrarrojo 2Y0A21 F
- Modulo puente h l298n
- Carro a control remoto
- bluetooth hc-05
- Arduino UNO

## Software

- RStudio
- Arduino
- App inventor
- Excel
- PLX-DAQ

## Procedimiento

### 1.1 adquisision de datos

Se programo un carro a control remoto vía bluetooth capaz de moverse a diferentes velocidades y en cualquier dirección, este fue controlado con una app creada en app inventor la cual permitio controlar el movimiento del carro y enviar la acción para que envié el dato censado en ese momento.

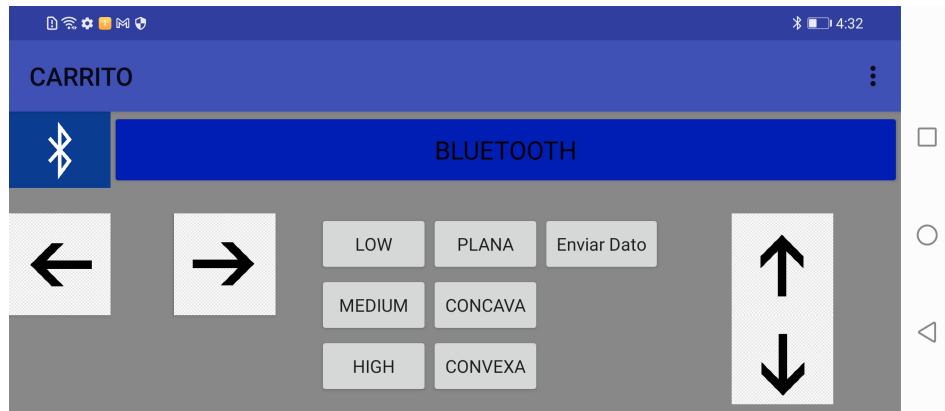


Figure 1: App

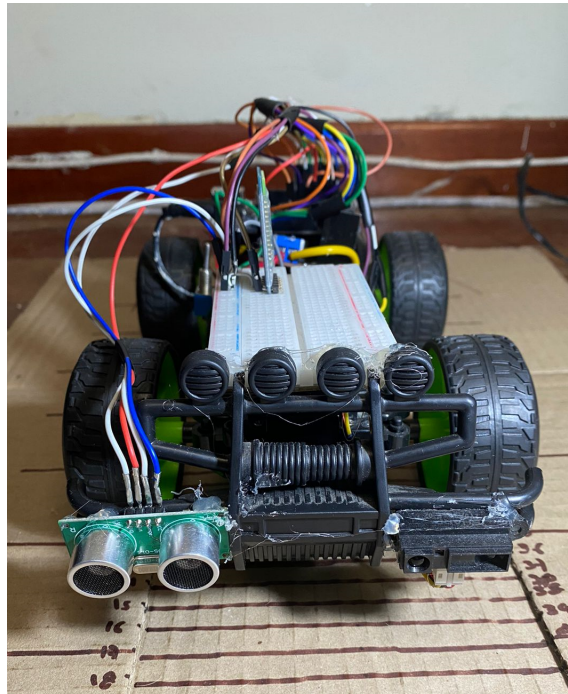


Figure 2: Carro

Usando la comunicación Serial entre el Arduino y el módulo bluetooth se pudieron captar los datos de los sensores en Excel, para poder comunicar Excel con el Arduino se usó el software PLZ-DAQ

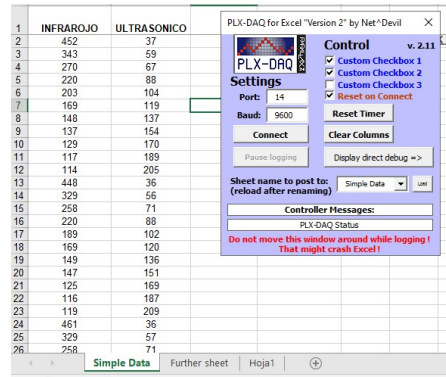


Figure 3: PLX-DAQ

## 1.2 Modelo lineal y regresion multilineal

Luego de tener el dataset de los datos tomados se carga la libreria tidyverse y el dataset

```
library(tidyverse)
folder <- dirname(rstudioapi::getSourceEditorContext()$path )
wall.distance <-read_csv(paste0(folder,"/dataset_wall_distance.csv"))
```

Se hace el analisis exploratorio de datos para el dataset

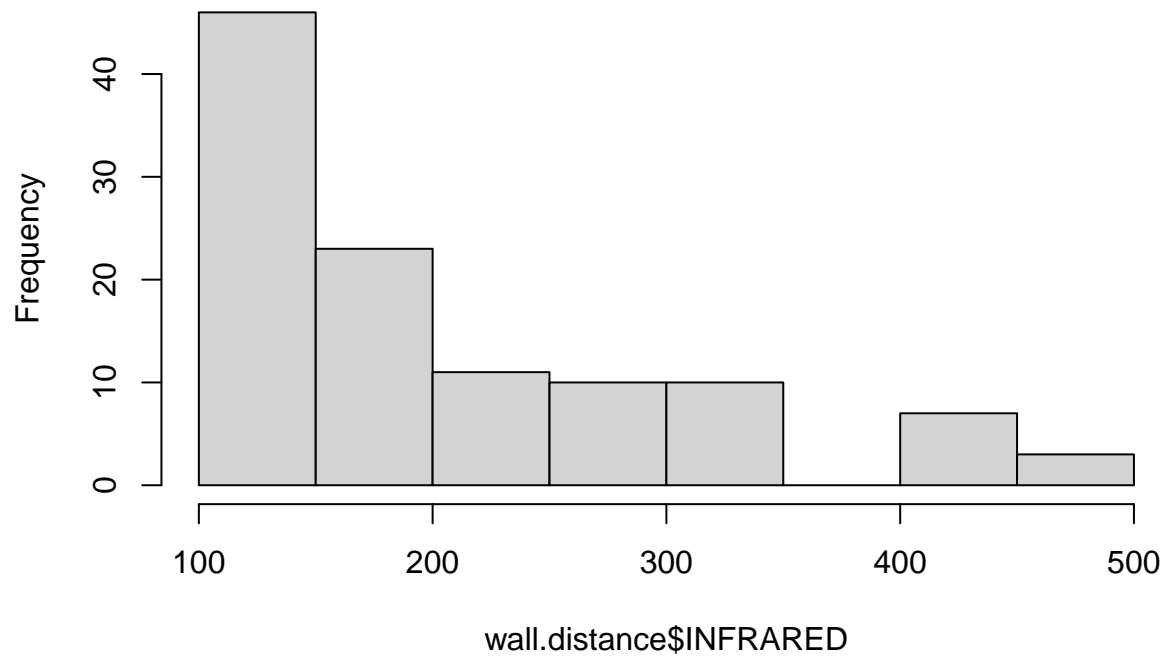
```
kable(summary(wall.distance))
```

INFRARED	ULTRASONIC	DISTANCE(cm)
Min. :113.0	Min. : 35.0	Min. :10
1st Qu.:133.8	1st Qu.: 71.0	1st Qu.:20
Median :169.0	Median :120.0	Median :35
Mean :207.5	Mean :119.9	Mean :35
3rd Qu.:258.8	3rd Qu.:168.5	3rd Qu.:50
Max. :461.0	Max. :209.0	Max. :60

De la tabla se puede identificar los valores maximos y minimos que se captaron con los sensores, de igual manera los valores por el primer cuartil y el tercer cuartil

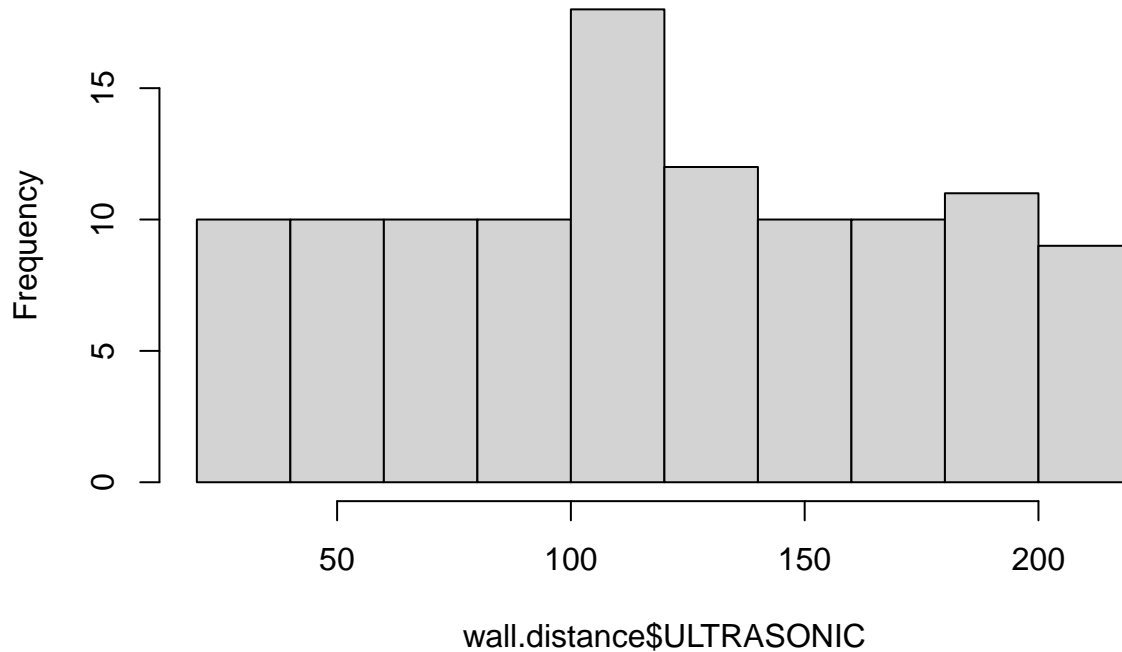
```
hist(wall.distance$INFRARED,breaks = 10)
```

**Histogram of wall.distance\$INFRARED**



```
hist(wall.distance$ULTRASONIC,breaks = 10)
```

## Histogram of wall.distance\$ULTRASONIC

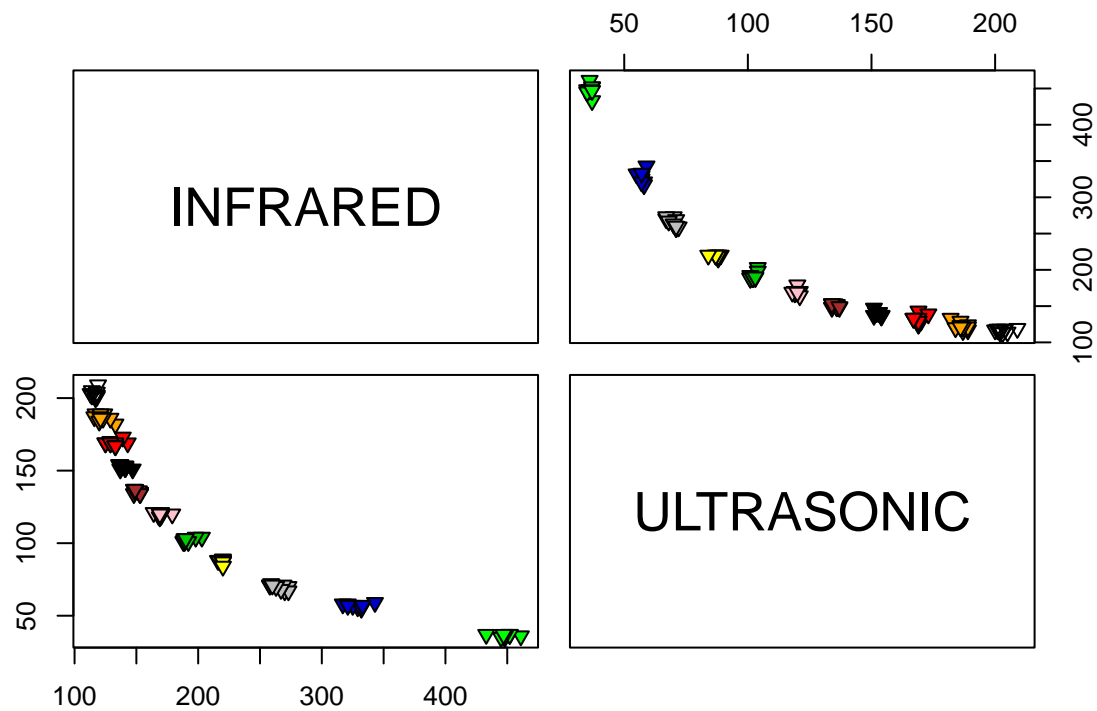


Debido a que la variable predictora es la distancia se tiene que pasar como factor, Graficando cada dato del sensor con la función hist se puede ver que los datos están distribuidos y que son viables para hacer aprendizaje de maquina

```

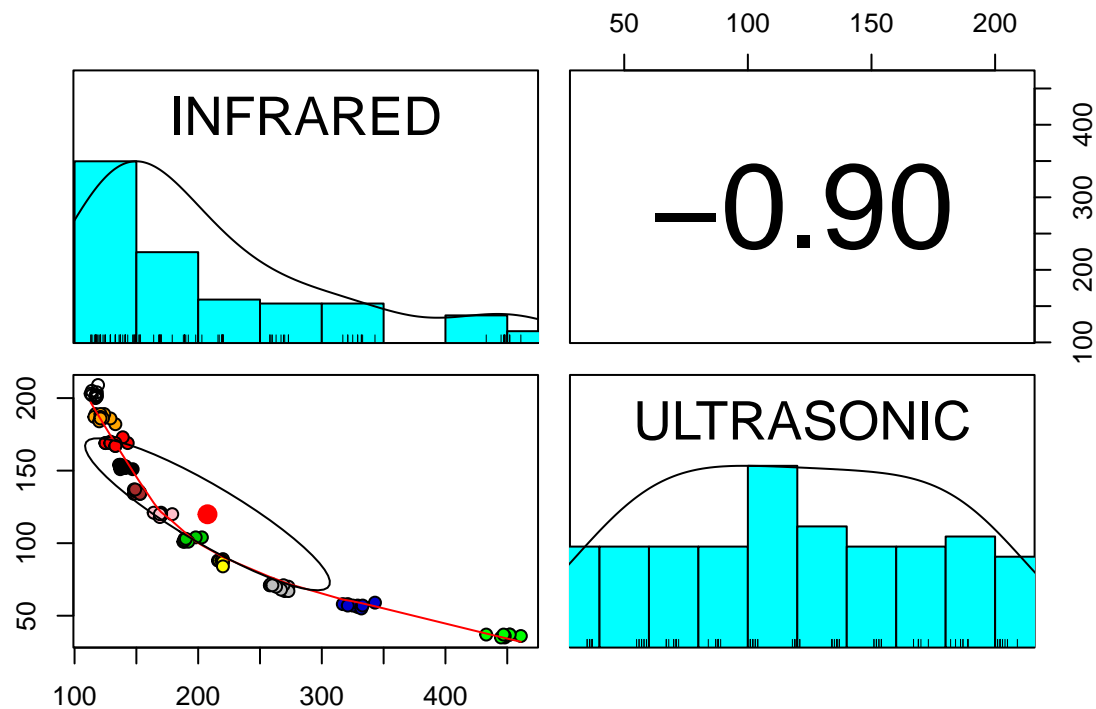
wall.distance2$`DISTANCE(cm)` <-as.factor(wall.distance2$`DISTANCE(cm)`)
pairs(wall.distance[c("INFRARED", "ULTRASONIC")],
      ,pch=25
      ,bg=c("green", "blue3", "gray", "yellow", "green3", "pink", "brown", "black", "red", "orange")[unclass(wall.dist

```



Cuando se captaron los datos se identificó que un sensor crece inversamente al otro, graficando todos los datos en función a la distancia se notó que tienen una buena distribución y que se logran identificar grupos en todos los datos

```
pairs.panels(wall.distance2[c("INFRARED", "ULTRASONIC")],
             ,pch=21,bg=c("green","blue3","gray","yellow","green3","pink","brown","black","red","orange"))
```



En esta grafica se ve mejor la relacion de las varibales y se ve que tiene una relacion negativa de -0.9, siendo esto un factor muy importante para poder hacer el aprendizaje de maquina

```
kable(prop.table(table(wall.distance$`DISTANCE(cm)`)))
```

Var1	Freq
10	0.0909091
15	0.0909091
20	0.0909091
25	0.0909091
30	0.0909091
35	0.0909091
40	0.0909091
45	0.0909091
50	0.0909091
55	0.0909091
60	0.0909091

de los 110 datos que se obtuvieron hay la misma cantidad en cada distancia identificada -regresion lineal para sensor infrarojo

```
x <- dataset$INFRARED
y <- dataset$`DISTANCE(cm)`
b=cov(x,y)/var(x)
a=mean(y)-b*mean(x)
a+b*121
```

```
## [1] 47.46267
```

- regresion lineal para sensor infrarojo
- Regresion Multilineal Para la regresion multilineal se van a tener dos predictores los cuales son los datos del infrarojo y del ultrasonico, se hace cross validation con el dataset dividiendolo en un 70% para entramiento y un 30% para prueba

```
predictors <- c( "INFRARED", "ULTRASONIC")
sample.index <- sample(1:nrow(wall.distance)
                      ,nrow(wall.distance)*0.7
                      ,replace = F)
train.data <- wall.distance[sample.index
                           ,c(predictors,"DISTANCE(cm)")]
test.data <- wall.distance[-sample.index
                          ,c(predictors,"DISTANCE(cm)")]
```

Con la funcion lm se hace el modelo en funcion a la distancia y los datos con los que se va a entrenar son los del train.data

```
model<- lm(`DISTANCE(cm)` ~ INFRARED + ULTRASONIC,train.data)
summary(model)
```

```
##
## Call:
## lm(formula = `DISTANCE(cm)` ~ INFRARED + ULTRASONIC, data = train.data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.77920 -0.31202  0.09355  0.33791  1.31708
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.393622   0.689909  -0.571   0.570
## INFRARED     -0.002192   0.001607  -1.364   0.177
## ULTRASONIC    0.298726   0.003088  96.751 <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6196 on 74 degrees of freedom
## Multiple R-squared:  0.9985, Adjusted R-squared:  0.9985
## F-statistic: 2.481e+04 on 2 and 74 DF,  p-value: < 2.2e-16

predictions <- predict(model,test.data)
predictions
```

```
##      1      2      3      4      5      6      7      8
## 30.228850 40.207372 50.106976  9.378408 15.613795 39.906454 44.391726 20.250328
##      9     10     11     12     13     14     15     16
## 30.239811 45.002331 59.391591 59.699085  9.710017 19.022541 25.418546 35.379531
##     17     18     19     20     21     22     23     24
## 44.703605 50.981232 35.080805 59.393783 15.607218 54.308867 59.690316  9.378408
##     25     26     27     28     29     30     31     32
## 19.940641 25.113244 29.662089 20.546861 34.485546 49.202030 20.245943 29.958623
##     33
```



```
## 54.904127
```

Para el caso del modelo multilíneal se logra identificar que la variable que mas tiene relevancia en el entrenamiento es la de ULTRASONIC, de los 33 datos que se tenían para hacer el test de prueba el modelo predijo correctamente las distancias de todos, para ver el error cuadrático medio se hizo lo siguiente

```
RMSE.df <- data.frame(predicted = predictions
                      ,reales=test.data$`DISTANCE(cm)`
                      ,RSE = sqrt((predictions - test.data$`DISTANCE(cm)`)^2))
promedio_error <- sum(RMSE.df$RSE)/nrow(RMSE.df )
kable(head(RMSE.df))
```

predicted	reales	RSE
30.228850	30	0.2288500
40.207372	40	0.2073724
50.106976	50	0.1069761
9.378408	10	0.6215918
15.613795	15	0.6137949
39.906454	40	0.0935456

```
promedio_error
```

```
## [1] 0.3907572
```

## 2.1 Adquisición de datos

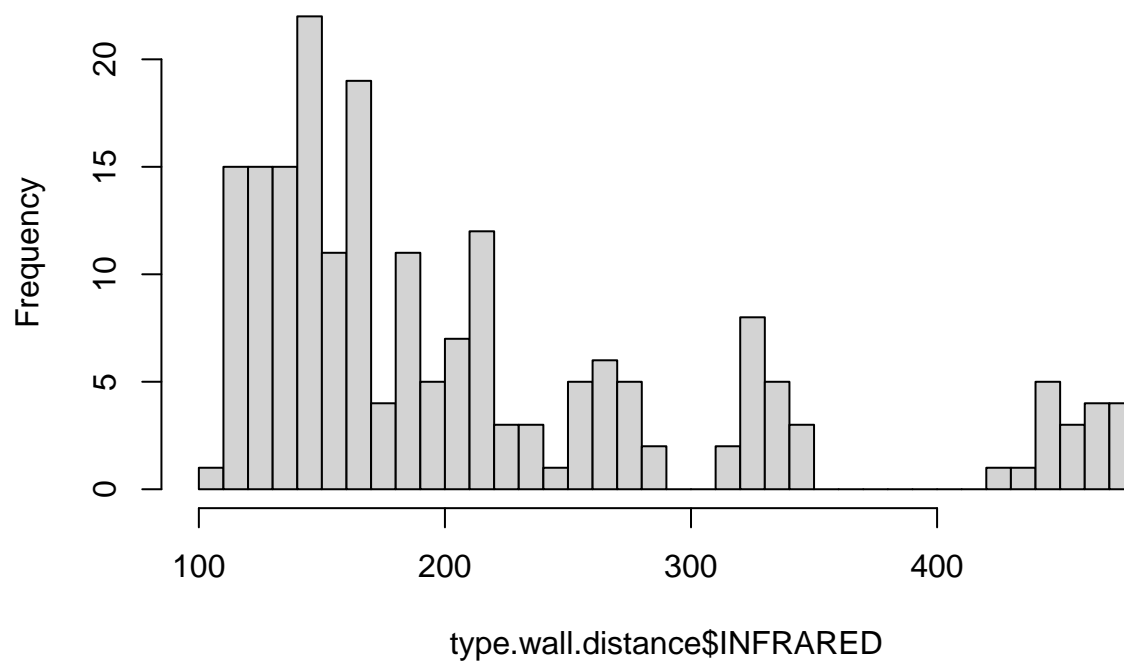
Para el modelo multilíneal se obtuvieron nuevas medidas y una nueva variable la cual es el tipo de obstáculo, de las 4 variables se obtuvieron 198 muestras.

```
kable(head(type.wall.distance))
```

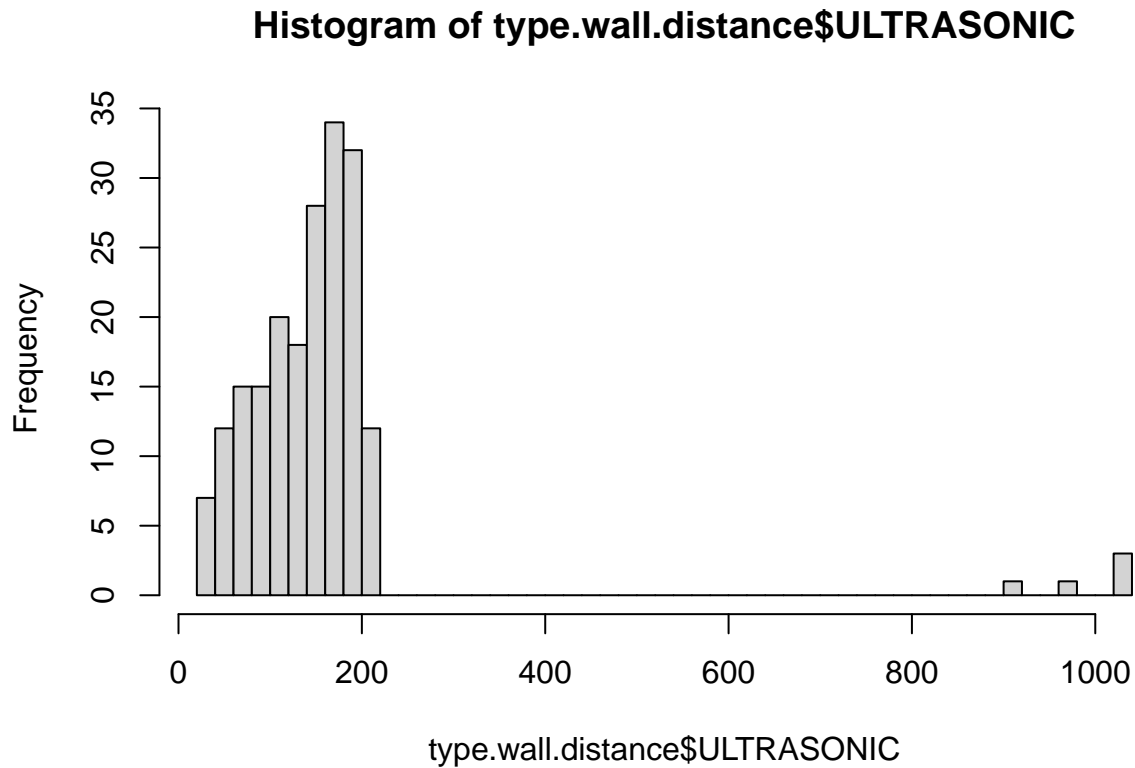
INFRARED	ULTRASONIC	DISTANCE(cm)	TYPE
452	37	10	PLANA
343	59	15	PLANA
270	67	20	PLANA
220	88	25	PLANA
203	104	30	PLANA
169	119	35	PLANA

```
hist(type.wall.distance$INFRARED,breaks = 50)
```

**Histogram of type.wall.distance\$INFRARED**



```
hist(type.wall.distance$ULTRASONIC,breaks = 50)
```



A diferencia del primer dataset, en este se obtuvieron valores por fuera del rango normal como lo muestra la grafica del sensor ultrasonico.

```
kable(summary(type.wall.distance))
```

INFRARED	ULTRASONIC	DISTANCE(cm)	TYPE
Min. :109.0	Min. : 35.0	Min. :10	Length:198
1st Qu.:141.0	1st Qu.: 101.0	1st Qu.:20	Class :character
Median :174.0	Median : 152.0	Median :35	Mode :character
Mean :213.5	Mean : 158.8	Mean :35	NA
3rd Qu.:259.0	3rd Qu.: 179.0	3rd Qu.:50	NA
Max. :479.0	Max. :1023.0	Max. :60	NA

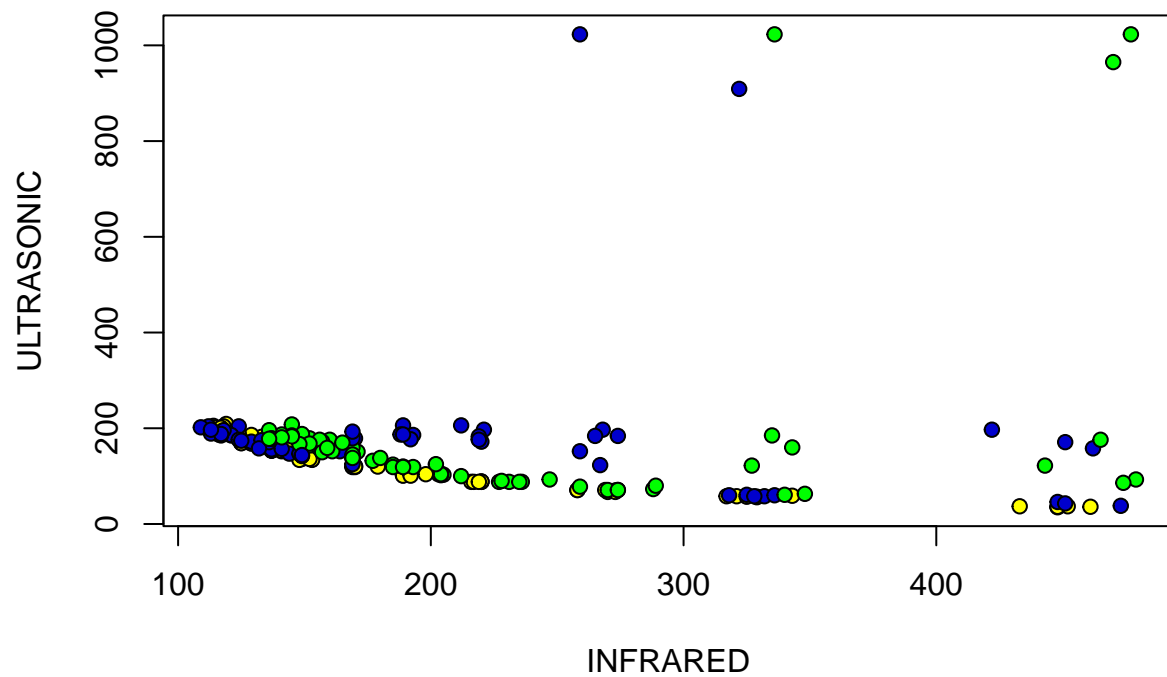
Con la funcion summary se puede ver una division de los datos tanto del infrarojo como el del ultrasonido y cuales fueron sus valores maximos y minimos.

```
type.wall.distance$TYPE <-as.factor(type.wall.distance$TYPE)
```

Debido a que la variable predictora es el tipo de obstaculo se tiene que pasar como factor, de cada

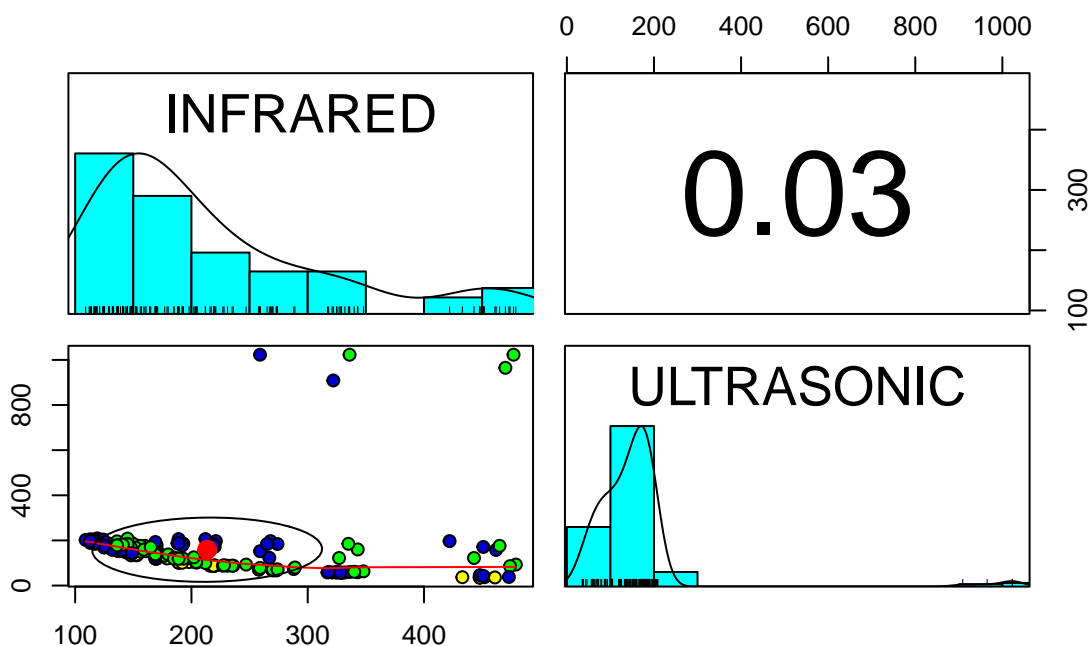
```
plot(type.wall.distance[1:2]
,main=c("yellow = plano,blue=concavo,green=convexo")
,pch=21,bg=c("green","blue3","yellow")[unclass(type.wall.distance$TYPE)])
```

**yellow = plano,blue=concavo,green=convexo**



```
library(psych)
pairs.panels(type.wall.distance[1:2]
,main=c("yellow = plano,blue=concavo,green=convexo")
,pch=21,bg=c("green","blue3","yellow")[unclass(type.wall.distance$TYPE)])
```

**yellow = plano,blue=concavo,green=convexo**



En este dataset se cambio mucho la relacion entre las variables, se tiene una relacion del 0.03 que no es un buen indicio para aprendizaje de maquina pero aun asi se logran identificar los tres grupos visualmente.

```
dummy <- dummyVars(" ~ TYPE",data = type.wall.distance)

newdata <- data.frame(predict(dummy,newdata = type.wall.distance))

type.wall.distance <- cbind(type.wall.distance,newdata)
kable(head(type.wall.distance))
```

INFRARED	ULTRASONIC	DISTANCE(cm)	TYPE	TYPE.CONCAVA	TYPE.CONVEXA	TYPE.PLANA
452	37	10	PLANA	0	0	1
343	59	15	PLANA	0	0	1
270	67	20	PLANA	0	0	1
220	88	25	PLANA	0	0	1
203	104	30	PLANA	0	0	1
169	119	35	PLANA	0	0	1

Se hace one hot encoding para la varibale TYPE, se tienen nuevas variables dummy y estas seran usadas como varibales predictoras para el aprendizaje de maquina.

```
sample.index <- sample(1:nrow(type.wall.distance)
                      ,nrow(type.wall.distance)*0.7
                      ,replace = F)

predictors <- c("INFRARED","ULTRASONIC","TYPE.CONCAVA","TYPE.CONVEXA","TYPE.PLANA")
```

```
train.data <- type.wall.distance[sample.index
                                ,c(predictors,"TYPE")
                                ,drop=F]
test.data <- type.wall.distance[-sample.index
                                ,c(predictors,"TYPE")
                                ,drop=F]
```

para hacer el entrenamiento se hizo cross validation dividiendo el dataset en el 70% para entrenamiento y el 30% para prueba

```
##KNN
ctrl <- trainControl(method = "cv",p=0.7) #variable de control
Knnfit <- train(TYPE ~ INFRARED+ULTRASONIC+TYPE.CONCAVA+TYPE.CONVEXA+TYPE.PLANA
                ,data = train.data
                ,method = "knn", trControl = ctrl
                ,preProcess= c("range")
                ,tuneLength=20)

Knnpredict <- predict(Knnfit,newdata = test.data)
confusionMatrix(Knnpredict
                ,test.data$TYPE)
```

```
## Confusion Matrix and Statistics
##
##              Reference
## Prediction CONCAVA CONVEXA PLANA
##   CONCAVA      18      0      0
##   CONVEXA       0     24      0
##   PLANA         0      0     18
##
## Overall Statistics
##
##              Accuracy : 1
##              95% CI : (0.9404, 1)
##   No Information Rate : 0.4
##   P-Value [Acc > NIR] : < 2.2e-16
##
##              Kappa : 1
##
##   Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##              Class: CONCAVA Class: CONVEXA Class: PLANA
## Sensitivity              1.0              1.0              1.0
## Specificity              1.0              1.0              1.0
## Pos Pred Value           1.0              1.0              1.0
## Neg Pred Value           1.0              1.0              1.0
## Prevalence               0.3              0.4              0.3
## Detection Rate           0.3              0.4              0.3
## Detection Prevalence     0.3              0.4              0.3
## Balanced Accuracy        1.0              1.0              1.0
```

Se entrena el algoritmo Knn en funcion de la variable TYPE y las variables predictoras son INFRARED,ULTRASONIC,TYPE.CONCAVA,TYPE.CONVEXA y TYPE.PLANA

con la matriz de confusion 3x3 se mira que de las muestras de test.data el algoritmo predice todas correctamente, tambien el accuary es de 1 y el p-value es menor a 2.2e-16 siendo este un valor muy importante estadisticamente. de igual manera se ve la sensibilidad y la especificidad son del 1.

```
Knnpredict
```

```
## [1] PLANA PLANA PLANA PLANA PLANA PLANA PLANA PLANA PLANA PLANA
## [10] PLANA PLANA PLANA PLANA PLANA PLANA PLANA PLANA PLANA PLANA
## [19] CONVEXA CONVEXA CONVEXA CONVEXA CONVEXA CONVEXA CONVEXA CONCAVA CONVEXA
## [28] CONVEXA CONVEXA CONCAVA CONCAVA CONCAVA CONCAVA CONCAVA CONCAVA CONVEXA
## [37] CONVEXA CONVEXA CONVEXA CONCAVA CONCAVA CONVEXA CONVEXA CONVEXA CONVEXA
## [46] CONCAVA CONCAVA CONCAVA CONCAVA CONVEXA CONVEXA CONVEXA CONCAVA CONCAVA
## [55] CONCAVA CONVEXA CONVEXA CONVEXA CONCAVA CONCAVA
## Levels: CONCAVA CONVEXA PLANA
```

## Validacion de modelos