# Segunda entrega

Diego Cubides,Larry castro, Tomas Mendez y Jersson avila

2023-04-21

## Contents



## 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.

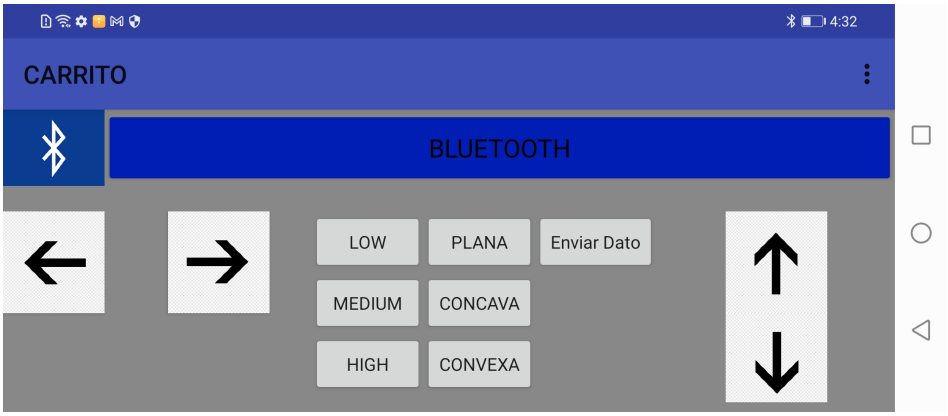



Figure 1: App

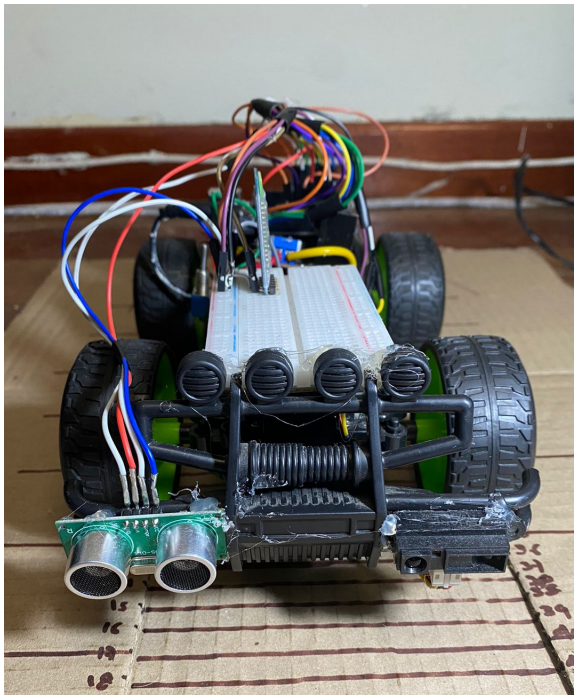


Figure 2: Carro

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

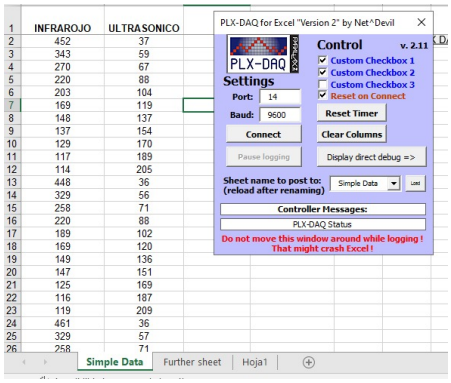



Figure 3: PLX-DAQ

## 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"))
```

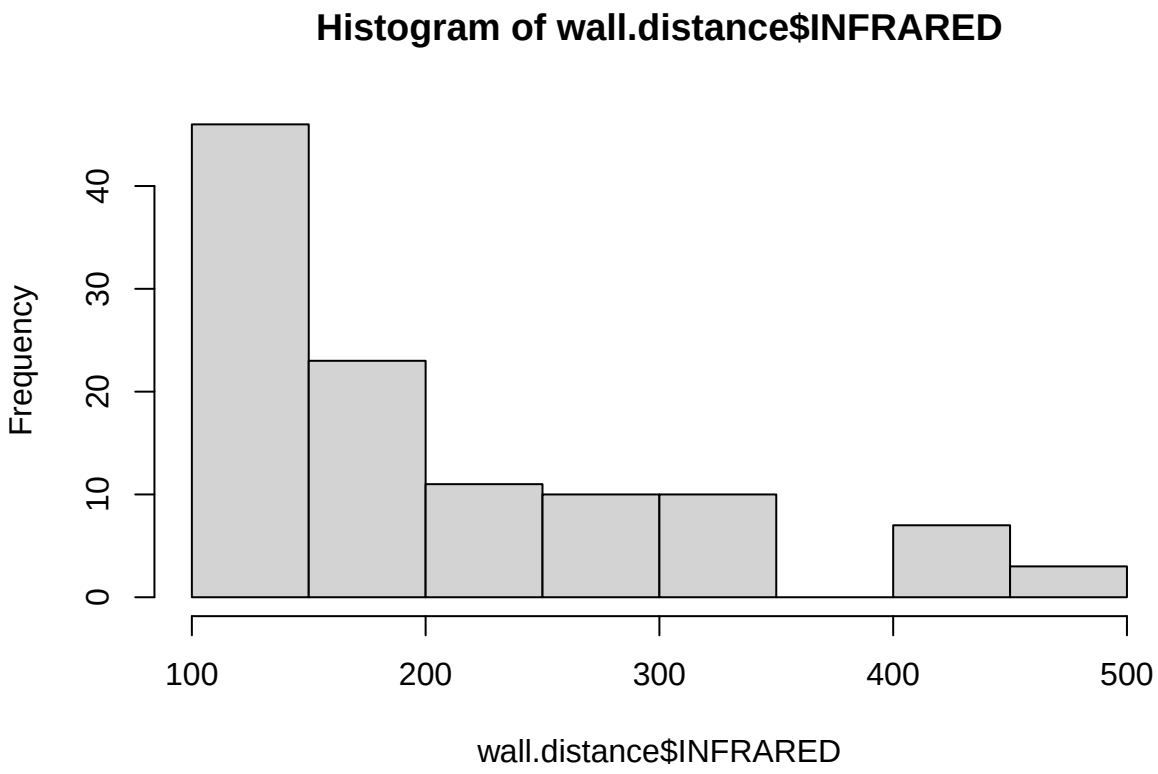
Se hace el analisis exploratorio de datos para el dataset

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

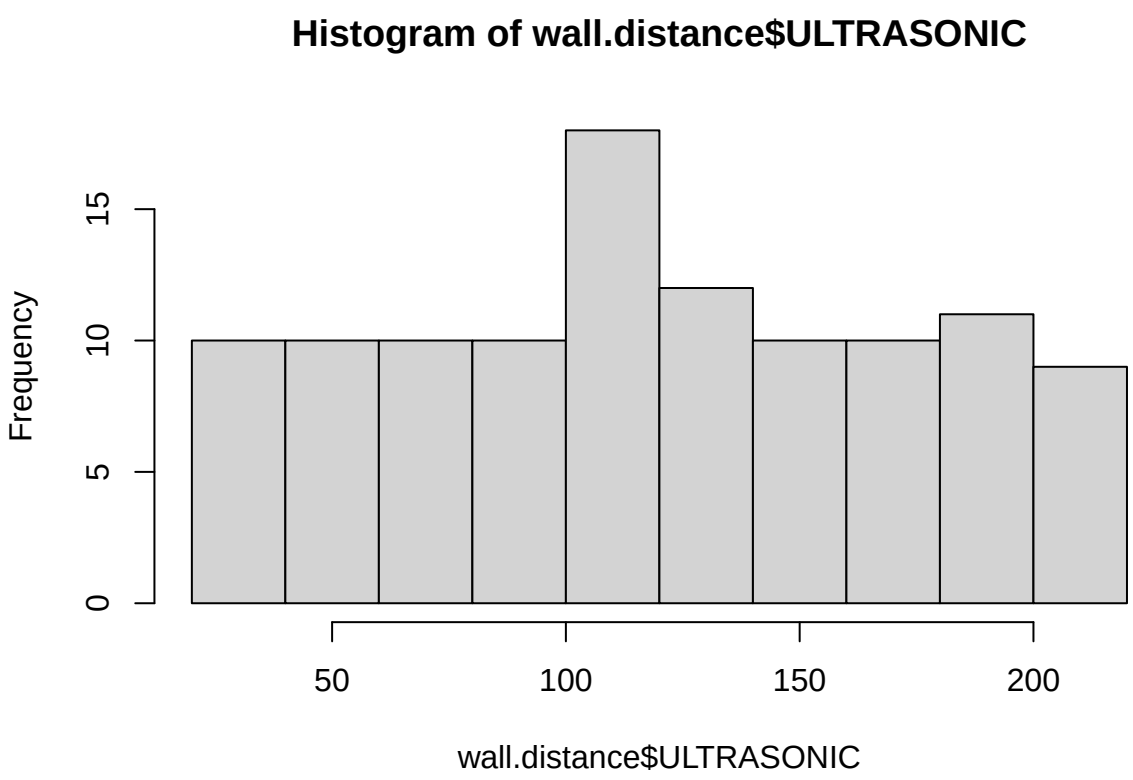
| INFRARED | ULTRASONIC | DISTANCE(cm) |
|---|---|---|
| Min. :113.0 | Min. : 35.0 | Min. :10 |
| 1st Qu.:133.8 | 1st Qu.: 71.0 | 1st Qu.:20 |
| Median :169.0 | Median :120.0 | Median :35 |
| Mean :207.5 | Mean :119.9 | Mean :35 |
| 3rd Qu.:258.8 | 3rd Qu.:168.5 | 3rd Qu.:50 |
| Max. :461.0 | Max. :209.0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

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

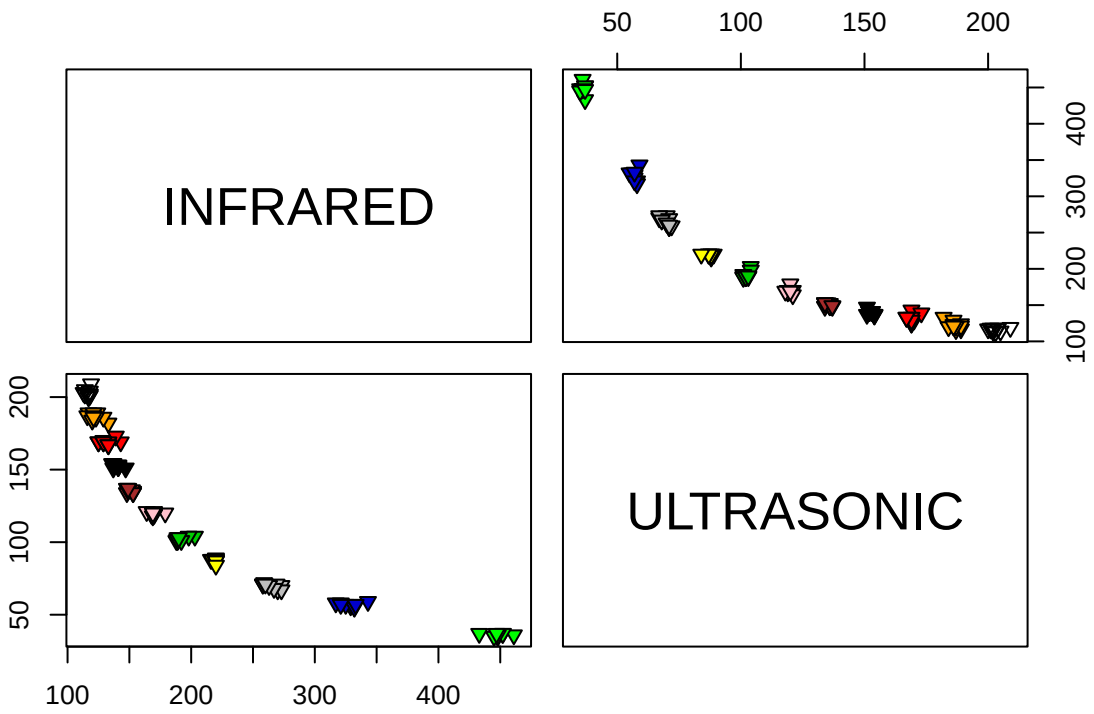



```
hist(wall.distance$ULTRASONIC,breaks = 10)
```



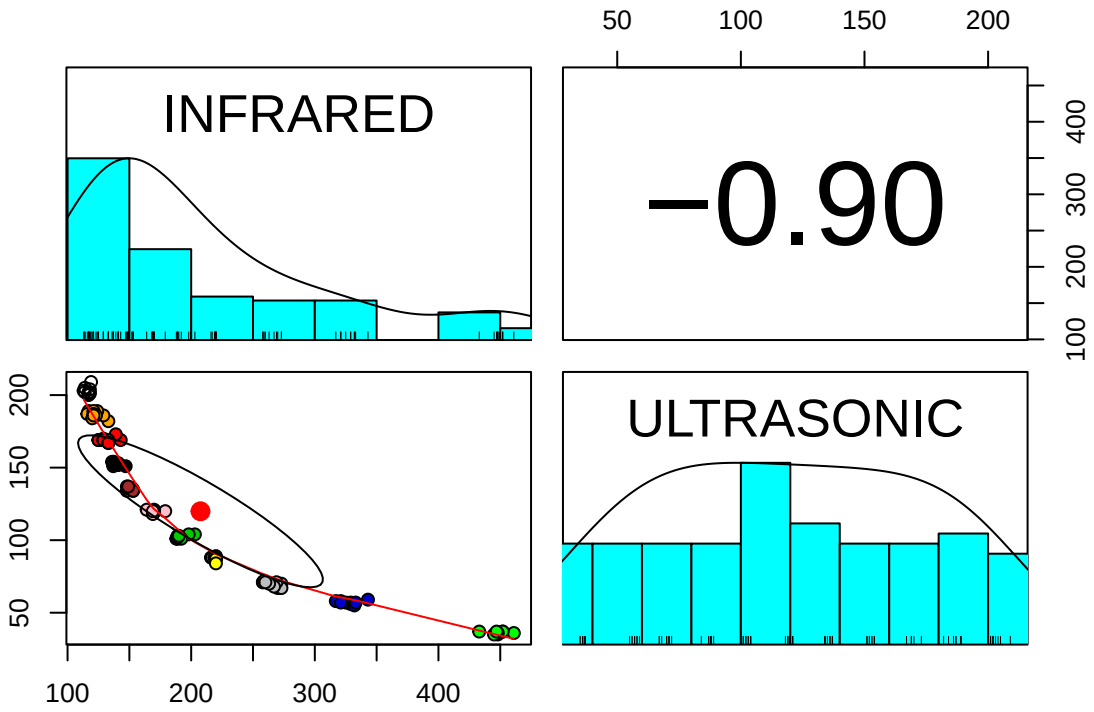

Debido a que la variable predictora es la distancia se tiene que pasar como factor,Graficando cada dato del sensor con la función hist se puede ver que los datos están distribuidos y que son viables para hacer aprendizaje de maquina

```
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.dista
```




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 dividiendlo 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)")
                                  ,drop=F]
test.data  <-  wall.distance[-sample.index
                                 ,c(predictors,"DISTANCE(cm)")
                                 ,drop=F]
```

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
## -1.2642 -0.3575  0.1001  0.4204  1.1991
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.2028725  0.6444104  -1.867   0.0659 .
## INFRARED    -0.0002242  0.0014838  -0.151   0.8803
## ULTRASONIC   0.3023824  0.0029057 104.066   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5987 on 74 degrees of freedom
## Multiple R-squared:  0.9987, Adjusted R-squared:  0.9987
## F-statistic: 2.822e+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
## 16.560795 34.742745 40.190336 15.656787 25.357458 29.597761 35.045127 61.968368
##         9        10        11        12        13        14        15        16
## 20.208439 35.042886 45.030023 49.867693 59.549533 19.902694 49.871728 55.011332
##        17        18        19        20        21        22        23        24
## 59.852813 35.044903 49.870831  9.883947 39.887729 44.727641 19.904936 29.597761
##        25        26        27        28        29        30        31        32
## 39.887729 45.333526 49.870831 39.282068 49.265170 55.012677 20.207991 35.045127
##        33
```

```
## 55.013125
```

Para el caso del modelo multilineal se logra identificar que la variable que mas tiene relevancia en el entrenamiento es la de ULTRASOINIC, de los 33 datos que se tenian para hacer el test de prueba el modelo predijo correctamente las distancias de todos, para ver el error cuadratico 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 |
|---|---|---|
| 16.56080 | 15 | 1.5607952 |
| 34.74274 | 35 | 0.2572550 |
| 40.19034 | 40 | 0.1903356 |
| 15.65679 | 15 | 0.6567865 |
| 25.35746 | 25 | 0.3574581 |
| 29.59776 | 30 | 0.4022390 |

```
promedio_error
```

```
## [1] 0.3080391
```

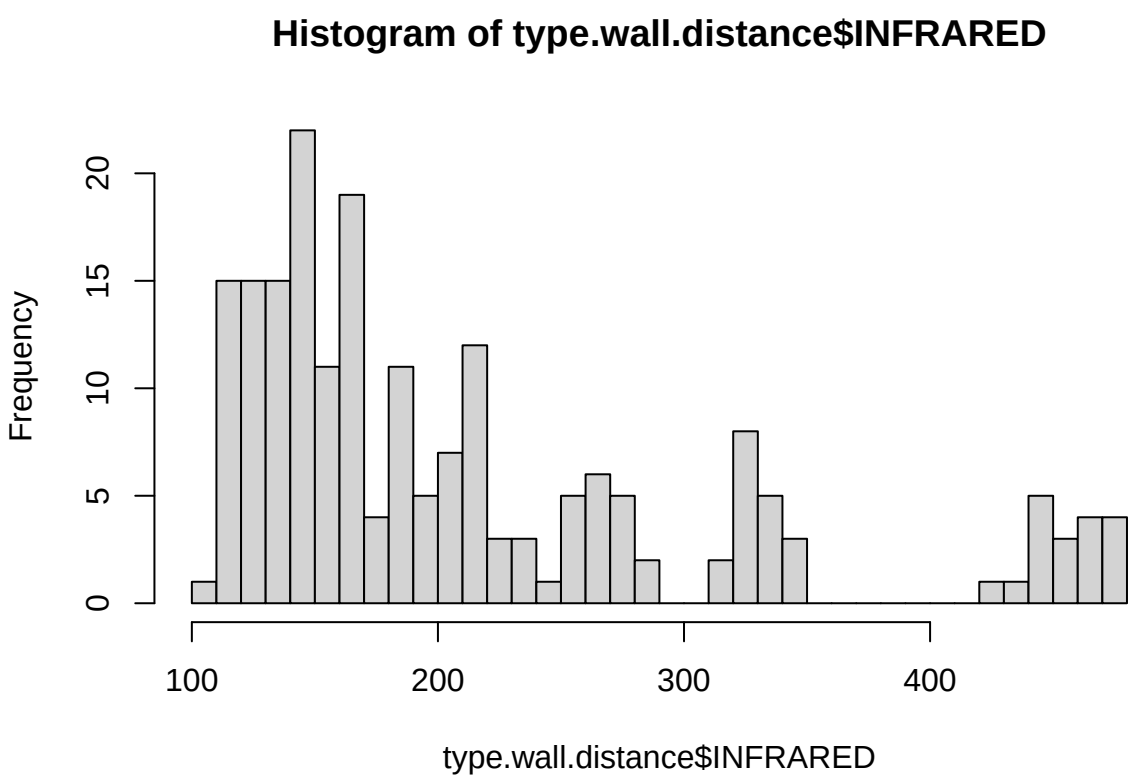
## 2.1 Adquisición de datos

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

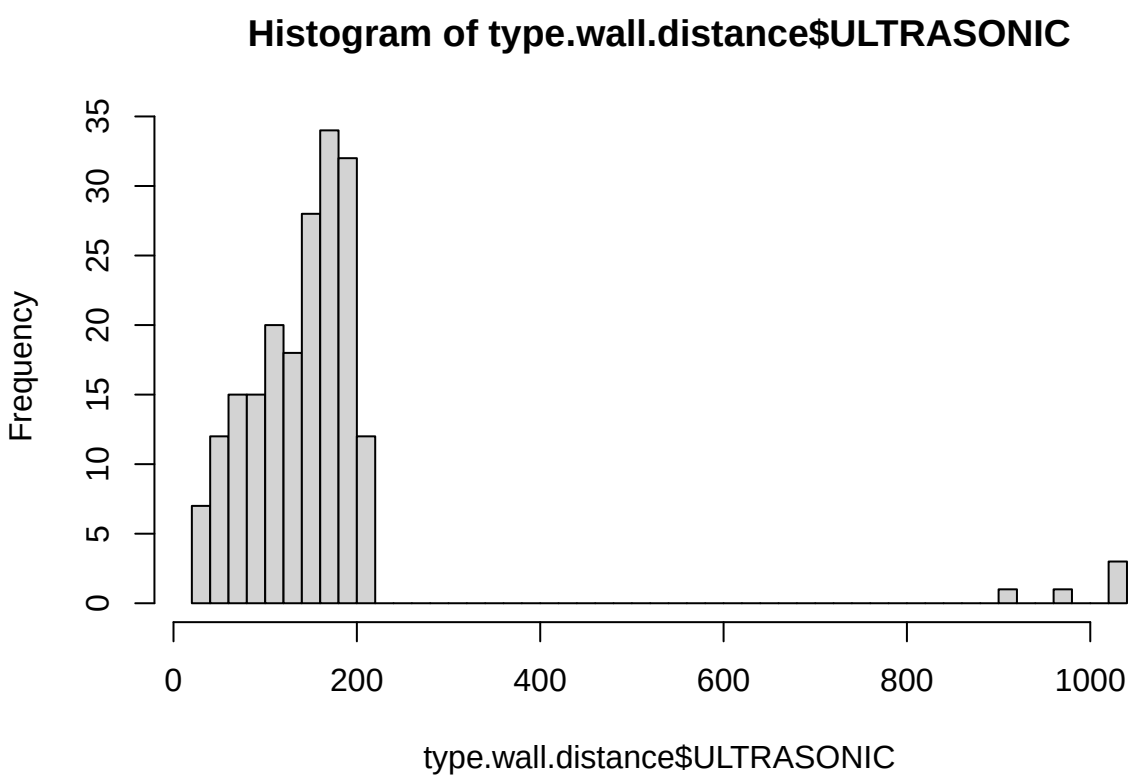
```
kable(head(type.wall.distance))
```

| 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)
```




```
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))
```

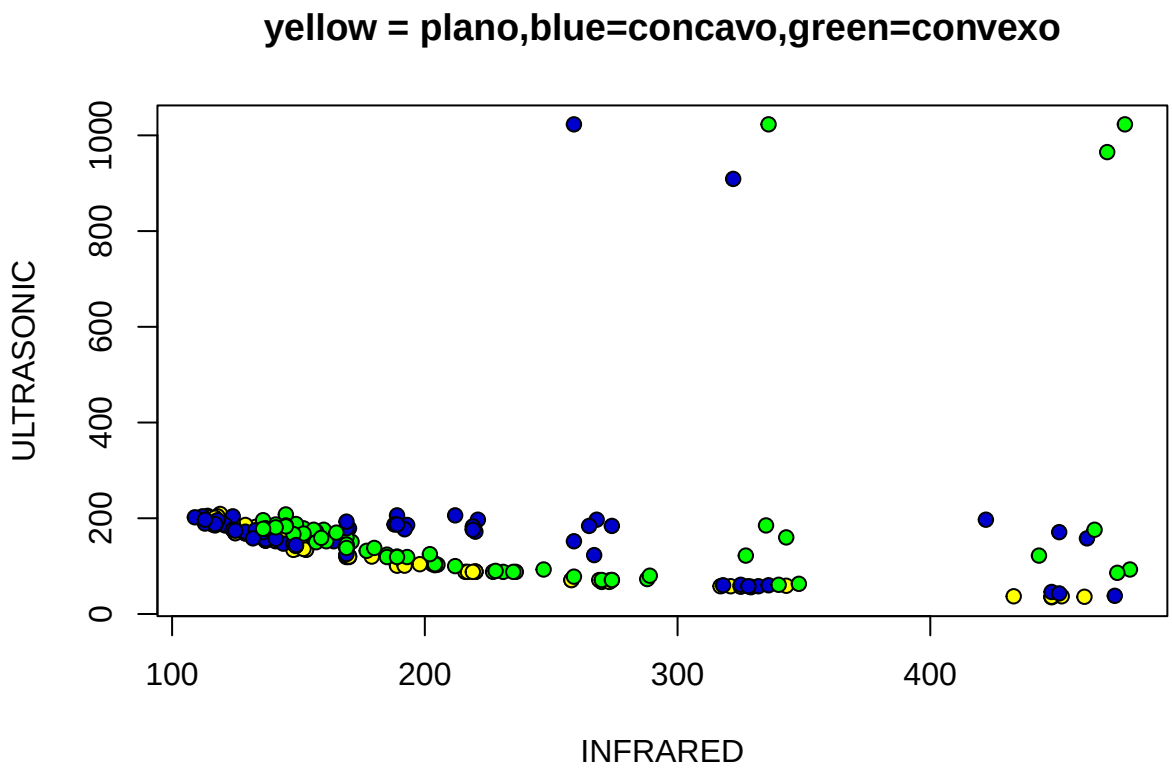
| INFRARED | ULTRASONIC | DISTANCE(cm) | TYPE |
|---|---|---|---|
| Min. :109.0 | Min. : 35.0 | Min. :10 | Length:198 |
| 1st Qu.:141.0 | 1st Qu.: 101.0 | 1st Qu.:20 | Class :character |
| Median :174.0 | Median : 152.0 | Median :35 | Mode :character |
| Mean :213.5 | Mean : 158.8 | Mean :35 | NA |
| 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.

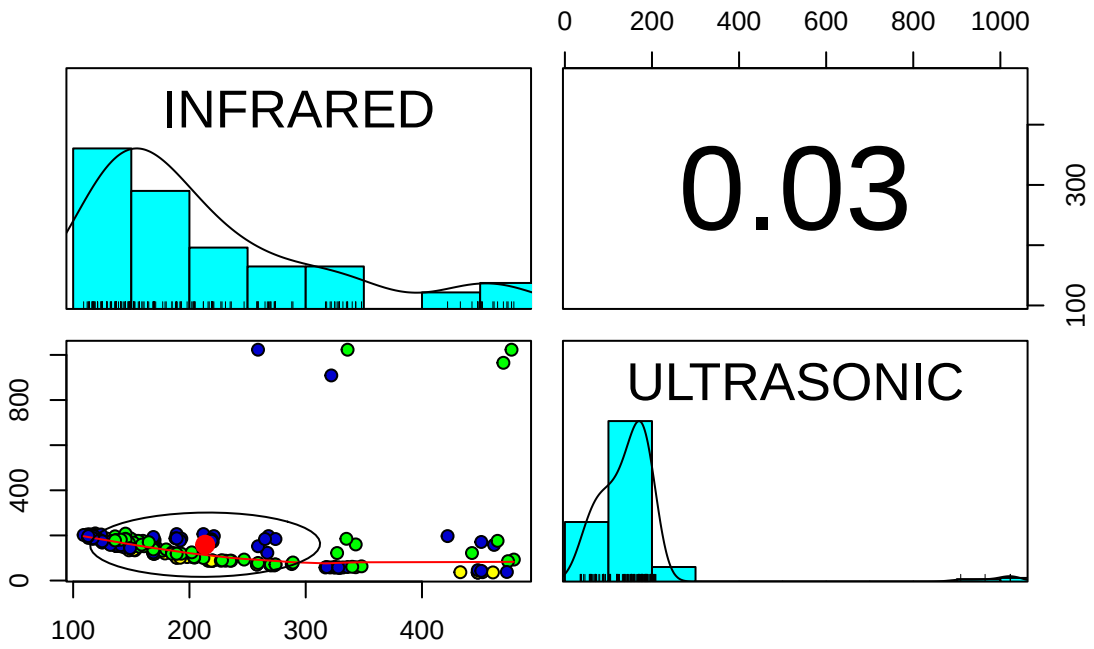
```
type.wall.distance$TYPE <-as.factor(type.wall.distance$TYPE)
```

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)])
```




```
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)])
```




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 identificiar los tres grupos visualemente.

```
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      24       0     0
##    CONVEXA       0      18     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                      0.4            0.3          0.3
## Detection Rate                  0.4            0.3          0.3
## Detection Prevalence            0.4            0.3          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 sensitividad y la especificidad son del 1.

```
Knnpredict
```

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

«««< HEAD

## Validacion de modelos

Para la validacion de modelos tomamos muestras desde distancias mas grandes de las que utilizamos para la creacion del modelo, en otras palabras utilizamos rango fuera de los rangos de entrenamiento, para de esta manera comprobar la validez de los modelos creados. En este nuevo data set de validacion utilizamos rangos de distancia de los 65 a los 85 centimetros.

```
x <- wall.distance$INFRARED
y <- wall.distance$`DISTANCE(cm)`
b=cov(x,y)/var(x)
a=mean(y)-b*mean(x)
a+b* 212
```

```
## [1] 34.35579
```

```
a+b* 223
```

```
## [1] 32.77144
```

```
a+b* 231
```

```
## [1] 31.61918
```

```
a+b* 235
```

```
## [1] 31.04306
```

```
a+b* 247
```

```
## [1] 29.31468
```

```
##Linear Regression For Ultrasonic Sensor
x <- wall.distance$ULTRASONIC
y <- wall.distance$`DISTANCE(cm)`
b=cov(x,y)/var(x)
a=mean(y)-b*mean(x)
a+b* 218
```

```
## [1] 64.71935
```

```
a+b* 240
```

```
## [1] 71.38548
```

```
a+b* 261
```

```
## [1] 77.7486
```

```
a+b* 276
```

```
## [1] 82.29368
```

```
a+b* 291
```

```
## [1] 86.83877
```

Validacion de modelos

Se evidencia en los resultados que en el sensor infrarojo fuera de rango pierde la exactitud y precision; y el sensor de ultasonido mantiene su exactitud y precision sin importar que esta por fuera del rango del modelo predictorio.

## Regresion lineal para el sensor Infrarojo, fuera del rango.

> a+b* 212 [1] 34.35579 a+b* 223 [1] 32.77144 a+b* 231 [1] 31.61918 a+b* 235 [1] 31.04306 a+b* 247 [1] 29.31468

## Regresion lineal para el sensor de Ultrasonido, fuera del rango.

a+b* 218 [1] 64.71935 > a+b* 240 [1] 71.38548 > a+b* 261 [1] 77.7486 > a+b* 276 [1] 82.29368 > a+b* 291 [1] 86.83877

## Reentrenamiento con un dataset de pruebas (data_pruebas)

Se evidencia en los resultados que en el sensor infrarojo fuera de rango pierde la exactitud y precision; y el sensor de ultasonido mantiene su exactitud y precision sin importar que esta por fuera del rango del modelo predictorio.

Regresion lineal para el sensor Infrarojo, fuera del rango.

> a+b* 212 [1] 34.35579 a+b* 223 [1] 32.77144 a+b* 231 [1] 31.61918 a+b* 235 [1] 31.04306 a+b* 247 [1] 29.31468

Regresion lineal para el sensor de Ultrasonido, fuera del rango.

a+b* 218 [1] 64.71935 > a+b* 240 [1] 71.38548 > a+b* 261 [1] 77.7486 > a+b* 276 [1] 82.29368 > a+b* 291 [1] 86.83877

```r
##reentrenamiento con un dataset de pruebas (data_pruebas)

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

```
## Rows: 110 Columns: 3
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## dbl (3): INFRARED, ULTRASONIC, DISTANCE(cm)
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
wall.distance2 <- read_csv(paste0(folder,"/dataset_wall_distance.csv"))
```

```
## Rows: 110 Columns: 3
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## dbl (3): INFRARED, ULTRASONIC, DISTANCE(cm)
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
data_prueba <- read_csv(paste0(folder,"/DATASET_PRUEBA_LINEAL.csv"))
```

```
## Rows: 10 Columns: 3
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## dbl (3): INFRARED, ULTRASONIC, DISTANCE(cm)
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

### Se hace un reentrenamiento

inplementando un nuevo dataset con los valores de distancia entre 65 y 85 para mirar el comportamiento de la regrecion multilineal con respecto a las funciones de prediccion el modelo. de esta forma los resultados odtenidos estuvieron semejantes entre el dataset y la prediccion

### Resultados de prediccion datos y del dataset del 65 a 85 1 2 3 4 5 6 7 8 9 64.28071 70.82564 77.07798 81.54730 86.00092 66.43089 70.28756 75.94490 80.73429

```
  10
```

## 86.39164

### se hace un rentrenamiento

inplementando un nuevo dataset con los valores de distancia entre 65 y 85 para mirar el comportamiento de la regrecion multilineal con respecto a las funciones de prediccion el modelo. de esta forma los resultados odtenidos estuvieron semejantes entre el dataset y la prediccion

##resultados de prediccion datos y del dataset del 65 a 85 1 2 3 4 5 6 7 8 9 64.28071 70.82564 77.07798 81.54730 86.00092 66.43089 70.28756 75.94490 80.73429

```
  10
```

86.39164 »»»> 480113853995acc4017db175efc4118a6aaf1886