Segunda entrega

Diego Cubides,Larry castro, Tomas Mendez y Jersson avila

2023-04-21

Contents

Link del repositorio: https://github.com/DiegoCubides/SegundaEntrega

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.

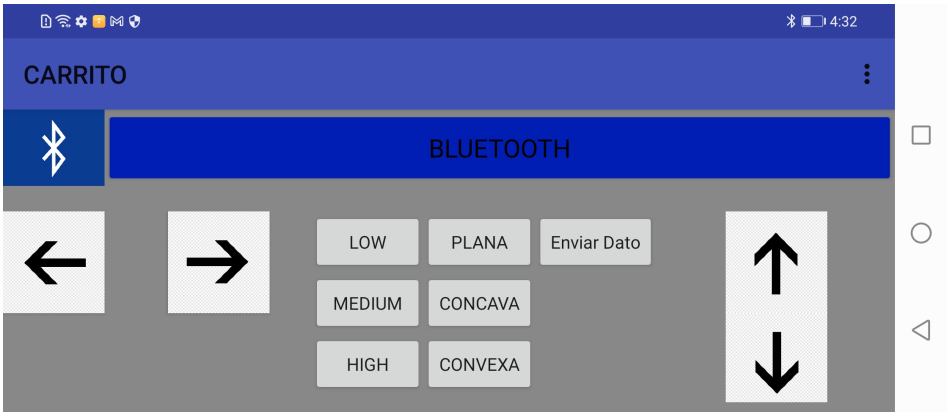


Figure 1: App

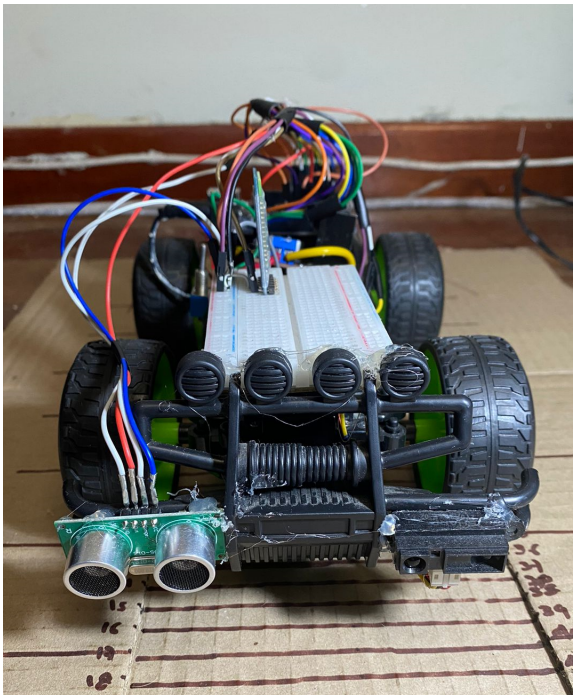


Figure 2: Carro

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

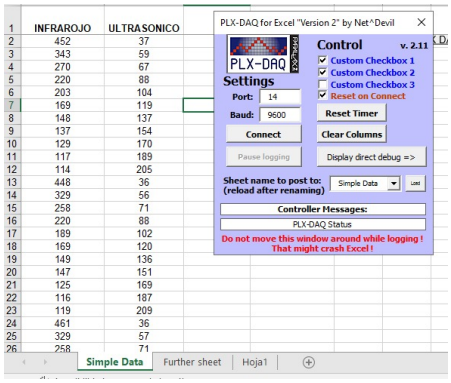


Figure 3: PLX-DAQ

- 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"))
```

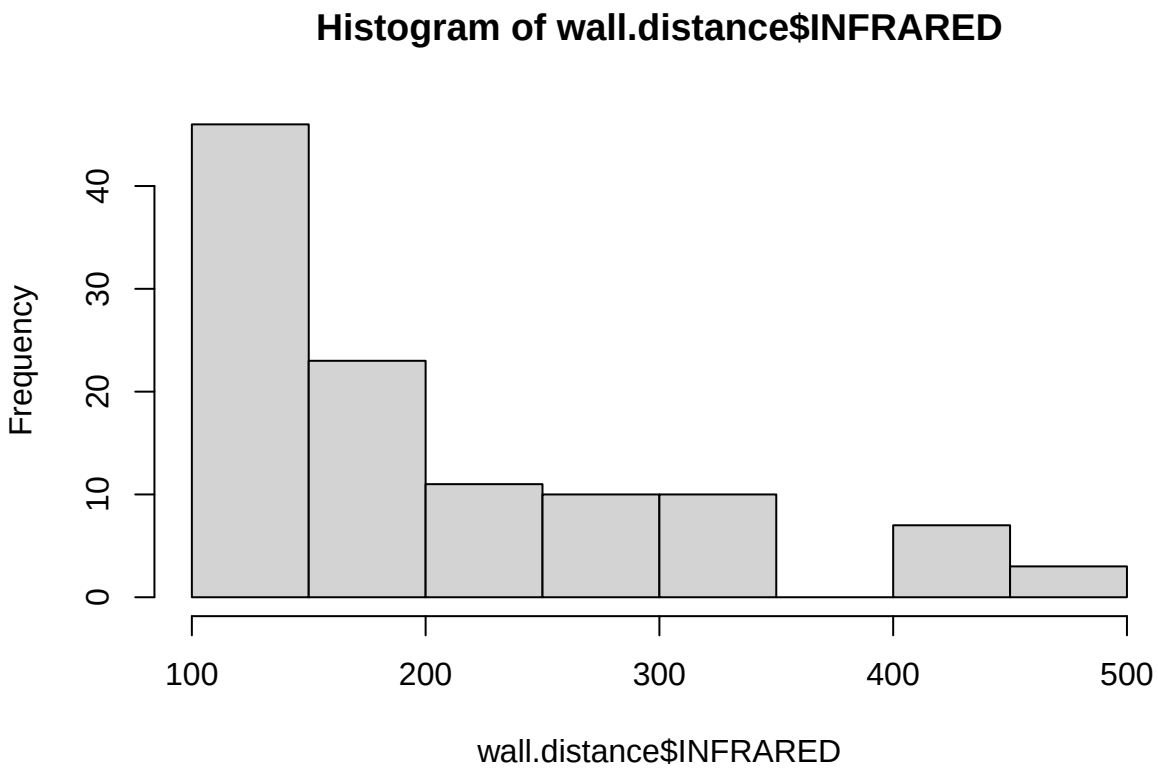
Se hace el analisis exploratorio de datos para el dataset

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

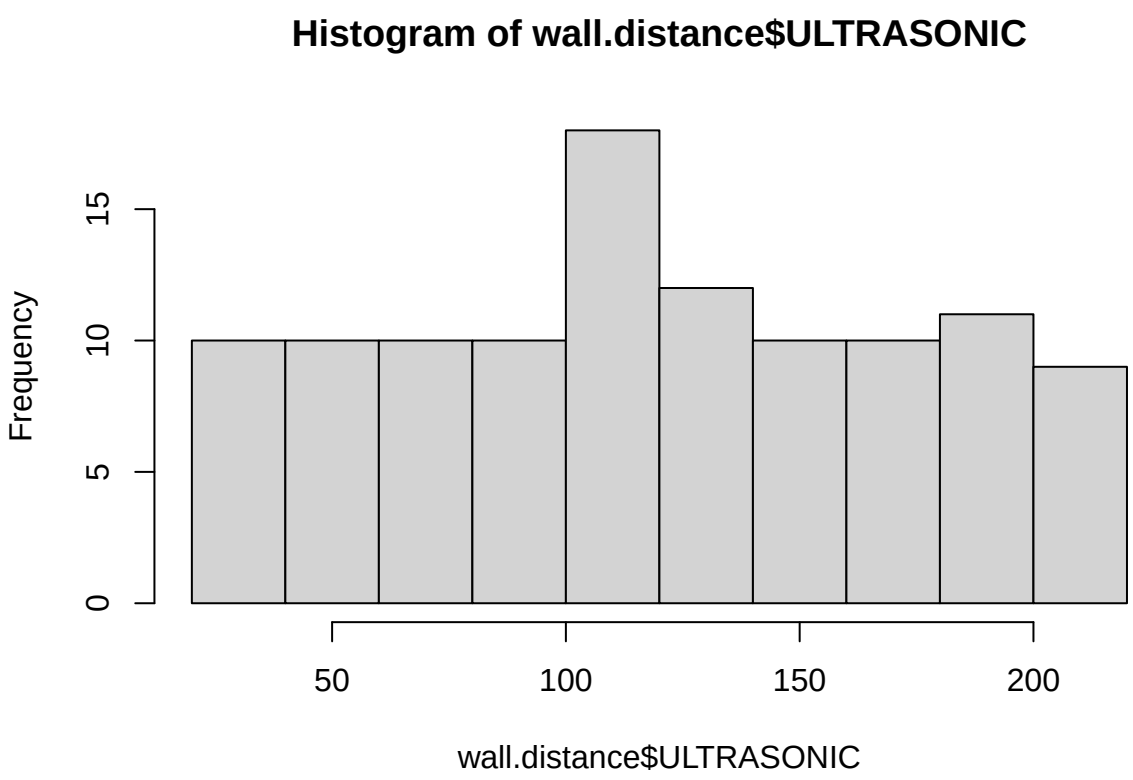
INFRARED	ULTRASONIC	DISTANCE(cm)
Min. :113.0	Min. : 35.0	Min. :10
1st Qu.:133.8	1st Qu.: 71.0	1st Qu.:20
Median :169.0	Median :120.0	Median :35
Mean :207.5	Mean :119.9	Mean :35
3rd Qu.:258.8	3rd Qu.:168.5	3rd Qu.:50
Max. :461.0	Max. :209.0	Max. :60

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)
```

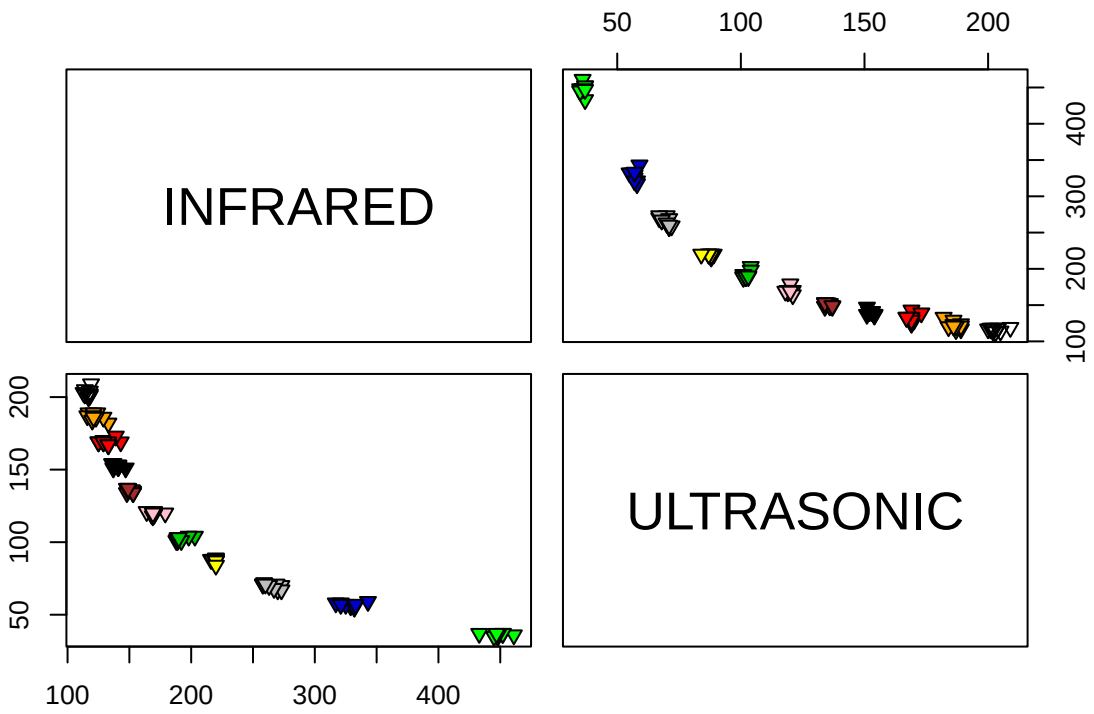



```
hist(wall.distance$ULTRASONIC,breaks = 10)
```



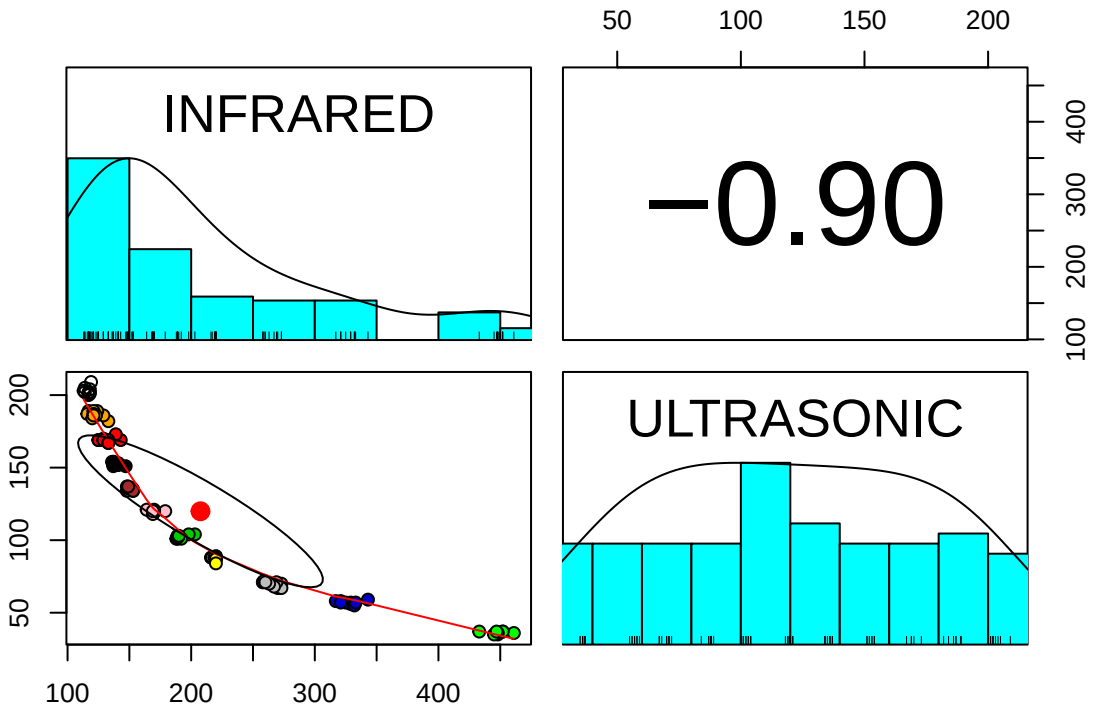
Debido a que la variable predictora es la distancia se tiene que pasar como factor,Graficando cada dato del sensor con la función hist se puede ver que los datos están distribuidos y que son viables para hacer aprendizaje de maquina

```
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
## -1.95862 -0.36128 -0.01493  0.44893  1.19582
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.845533   0.687613  -1.230    0.223
## INFRARED    -0.001395   0.001618  -0.862    0.391
## ULTRASONIC   0.301293   0.003037  99.215   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6105 on 74 degrees of freedom
## Multiple R-squared:  0.9986, Adjusted R-squared:  0.9986
## F-statistic: 2.712e+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
##  9.671645 34.772509 45.362404 50.194251 55.935556 39.922391 35.059849 45.055530
##         9        10        11        12        13        14        15        16
## 25.366856 29.923920 34.771114 39.321201 51.084176  9.375933 25.362670 29.317148
##        17        18        19        20        21        22        23        24
## 49.892958 55.929975 25.362670 34.773904 44.754238 54.424907 59.850967 25.059982
##        25        26        27        28        29        30        31        32
## 39.922391 60.159236  9.078826 15.880279 29.622627 49.284791 29.922524 40.223684
##        33
```

```
## 59.852362
```

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
9.671645	10	0.3283552
34.772509	35	0.2274910
45.362404	45	0.3624042
50.194251	50	0.1942506
55.935556	55	0.9355563
39.922391	40	0.0776088

```
promedio_error
```

```
## [1] 0.3812353
```

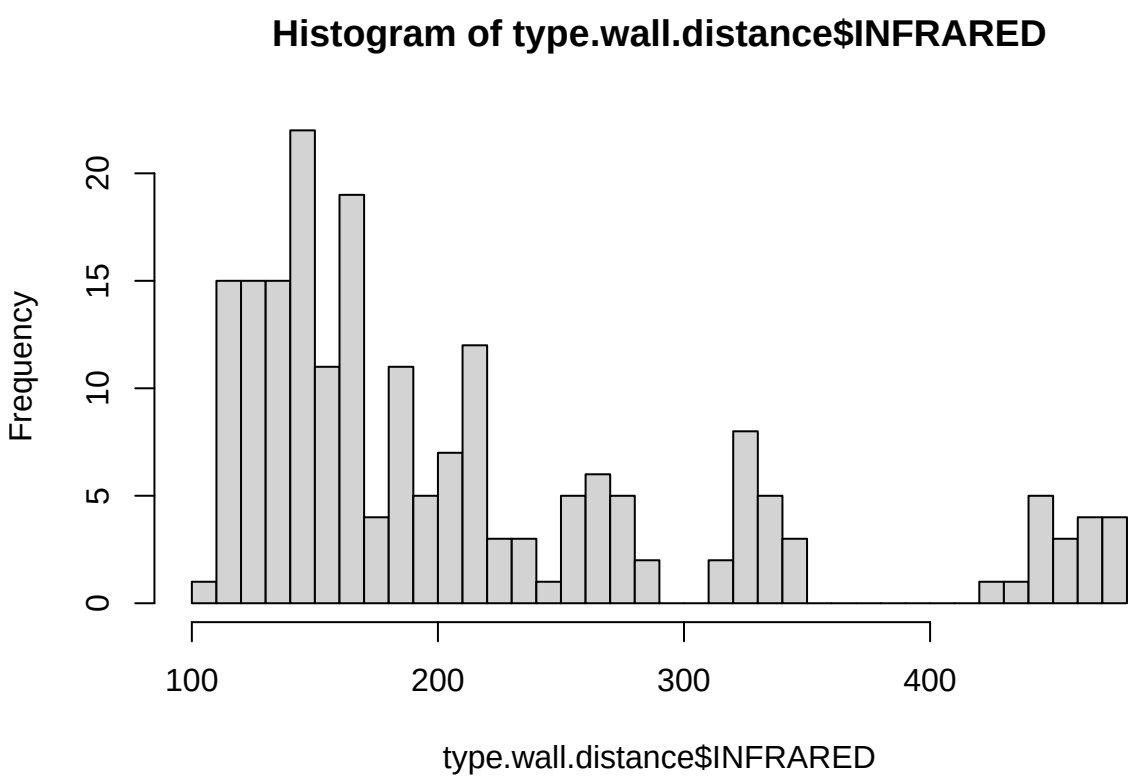
- 2.1 Adquisición de datos

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.

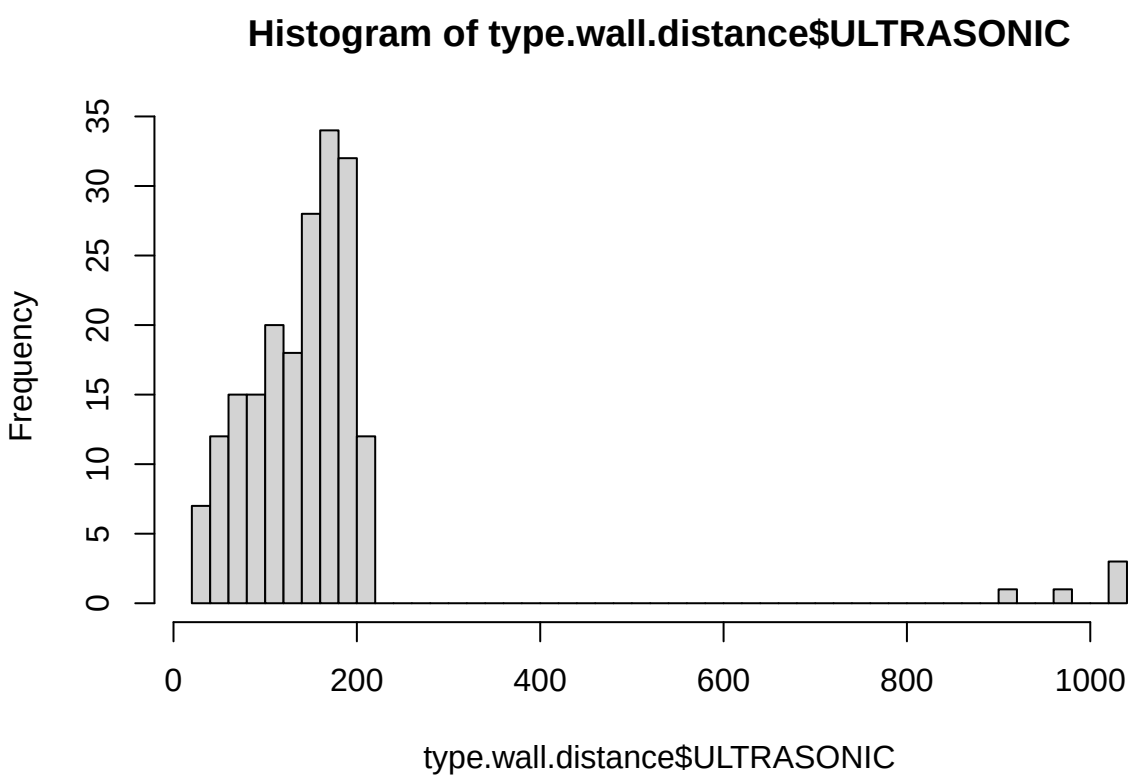
```
kable(head(type.wall.distance))
```

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))
```

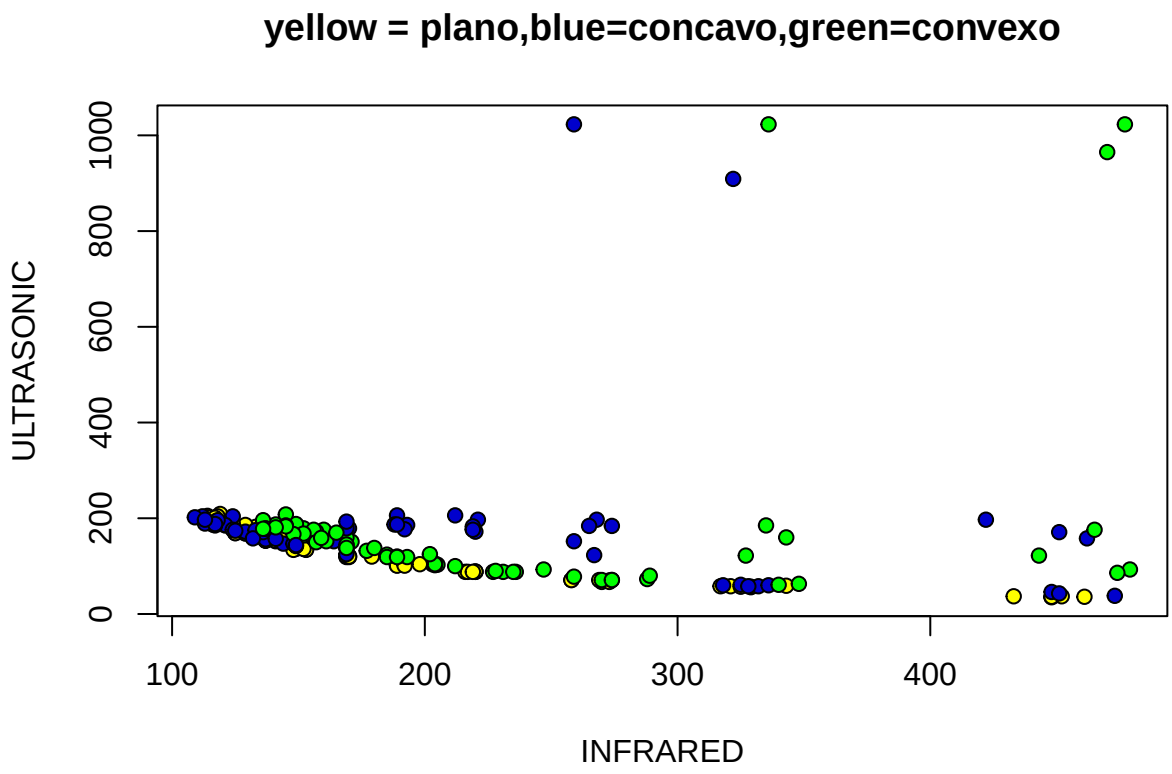
INFRARED	ULTRASONIC	DISTANCE(cm)	TYPE
Min. :109.0	Min. : 35.0	Min. :10	Length:198
1st Qu.:141.0	1st Qu.: 101.0	1st Qu.:20	Class :character
Median :174.0	Median : 152.0	Median :35	Mode :character
Mean :213.5	Mean : 158.8	Mean :35	NA
3rd Qu.:259.0	3rd Qu.: 179.0	3rd Qu.:50	NA
Max. :479.0	Max. :1023.0	Max.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

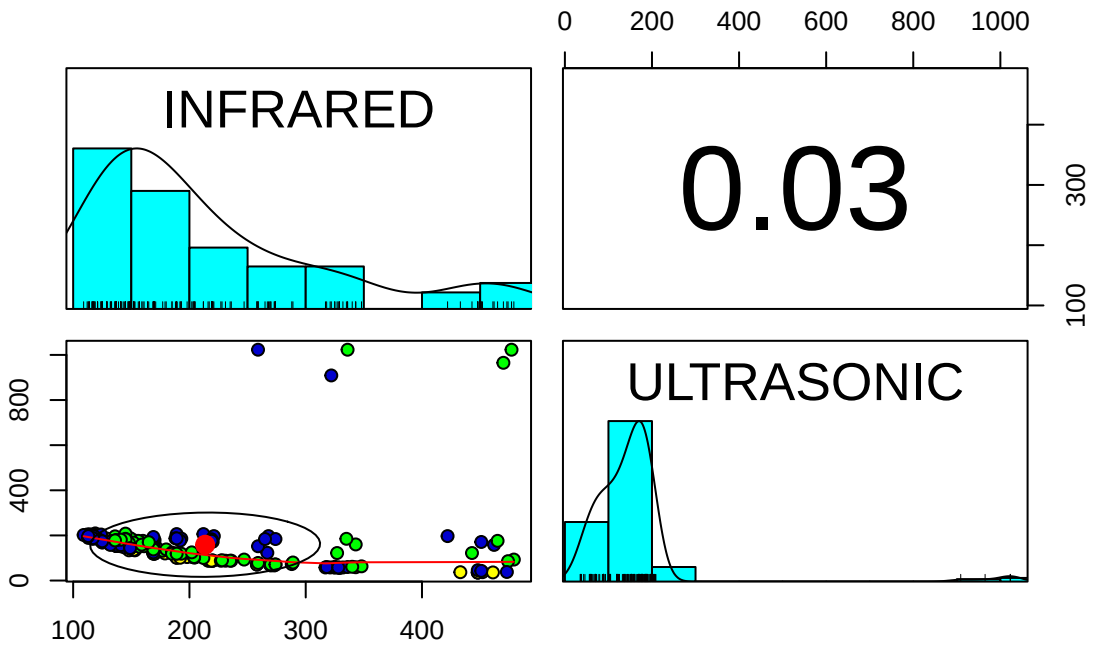
```
type.wall.distance$TYPE <-as.factor(type.wall.distance$TYPE)
```

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      23       0     0
##    CONVEXA       0      21     0
##    PLANA         0       0    16
##
## Overall Statistics
##
##                Accuracy : 1
##                  95% CI : (0.9404, 1)
##     No Information Rate : 0.3833
##     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                  1.0000           1.00       1.0000
## Specificity                  1.0000           1.00       1.0000
## Pos Pred Value               1.0000           1.00       1.0000
## Neg Pred Value               1.0000           1.00       1.0000
## Prevalence                   0.3833           0.35       0.2667
## Detection Rate               0.3833           0.35       0.2667
## Detection Prevalence         0.3833           0.35       0.2667
## Balanced Accuracy            1.0000           1.00       1.0000
```

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

Validacion de modelos

Para la validacion de los tres modelos se tomaron muestras fuera del rango de las que se usaron para el entrenamiento del algoritmo,de esta manera se pudo comprobar el rendimiento y la presicion modelos creados. para los nuevos datos se usaron rangos entre los 65cm y 85cm con un intervalo de 5cm.

- Validacion modelo lineal por sensor Luego de tomar los datos se obtiene un nuevo dataset con laS medidas fuera del rango de entrenamiento.

```
kable(datase_prueba_lineal)
```

INFRARED	ULTRASONIC	DISTANCE(cm)
212	218	65
223	240	70
231	261	75
235	276	80
247	291	85
181	225	65
193	238	70
200	257	75
193	273	80
200	292	85

Se realizo la prueba del modelo con algunos valores y se obtuvo las siguientes respuestas:

- Modelo Lineal Infrarojo

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

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

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

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

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

Con los resultados se pudo concluir que el modelo del sensor infrarojo no tiene buena prediccion ya que estan los valores muy fuera de la respuesta esperada, esto se debe a que este sensor varia mucho los datos de las muestras y no tiene buena presicion.

- Modelo Lineal Ultrasonico

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

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

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

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

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

Para el modelo lineal ultrasonco se obtuvo una muy buena prediccion, el error entre el valor calculado y el valor real es muy bajo.

- Validacion modelo regresion multilineal

Para la validación de este modelo se usaron los siguientes datos los cuales estan por fuera del rango en comaparación a los datos del entrenamiento.

```
kable(datase_prueba_lineal)
```

INFRARED	ULTRASONIC	DISTANCE(cm)
212	218	65
223	240	70
231	261	75
235	276	80
247	291	85
181	225	65
193	238	70
200	257	75
193	273	80
200	292	85

Ya que se tenia el modelo con el dataset de entrenamiento, ahora se hace la prediccion con los nuevos datos

```
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]

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
## -2.0129 -0.3998  0.0336  0.3956  1.1434
##
## Coefficients:
```

```
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.762073   0.668152  -1.141    0.258
## INFRARED    -0.001587   0.001613  -0.984    0.328
## ULTRASONIC   0.301263   0.002945 102.289   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6155 on 74 degrees of freedom
## Multiple R-squared:  0.9985, Adjusted R-squared:  0.9985
## F-statistic: 2.48e+04 on 2 and 74 DF,  p-value: < 2.2e-16
```

```
predictions <- predict(model,datase_prueba_lineal)
kable(predictions)
```

x
64.57668
71.18700
77.50082
82.01341
86.51330
66.73472
70.63209
76.34497
81.17628
86.88916

- Validacion modelo predictivo

Para la validacion del modelo predictivo se uso el siguiente dataset, este tiene valores fuera del rango en comparacion a los del entrenamiento

```
kable(datasetprueba)
```

INFRARED	ULTRASONIC	DISTANCE(cm)	TYPE
212	218	65	PLANA
223	240	70	PLANA
231	261	75	PLANA
235	276	80	PLANA
247	291	85	PLANA
212	221	65	CONCAVA
224	239	70	CONCAVA
239	254	75	CONCAVA
227	275	80	CONCAVA
224	293	85	CONCAVA
196	227	65	CONVEXA
211	244	70	CONVEXA
212	263	75	CONVEXA
216	1023	80	CONVEXA
224	1023	85	CONVEXA
204	226	65	CONVEXA
204	240	70	CONVEXA
216	256	75	CONVEXA
223	879	80	CONVEXA

INFRARED	ULTRASONIC	DISTANCE(cm)	TYPE
212	1023	85	CONVEXA
192	223	65	CONCAVA
200	238	70	CONCAVA
215	260	75	CONCAVA
232	288	80	CONCAVA
223	291	85	CONCAVA
181	225	65	PLANA
193	238	70	PLANA
200	257	75	PLANA
193	273	80	PLANA
200	292	85	PLANA

Teniendo ya el algoritmo Knn se procede a hacer la prueba con el nuevo dataset

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


dummy <- dummyVars(" ~ TYPE",data = type.wall.distance)
dummy2 <- dummyVars(" ~ TYPE",data = datasetprueba)


newdata <- data.frame(predict(dummy,newdata = type.wall.distance))
type.wall.distance <- cbind(type.wall.distance,newdata)
newdata2 <- data.frame(predict(dummy2,newdata = datasetprueba))
datasetprueba<- cbind(datasetprueba,newdata2)

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]
##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 = datasetprueba)

confusionMatrix(Knnpredict
              ,datasetprueba$TYPE)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction CONCAVA CONVEXA PLANA
##    CONCAVA      10       0     0
##    CONVEXA       0      10     0
##    PLANA         0       0    10
##
## Overall Statistics
##
##                Accuracy : 1
##                  95% CI : (0.8843, 1)
##     No Information Rate : 0.3333
##     P-Value [Acc > NIR] : 4.857e-15
##
##                   Kappa : 1
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: CONCAVA Class: CONVEXA Class: PLANA
## Sensitivity                  1.0000         1.0000       1.0000
## Specificity                  1.0000         1.0000       1.0000
## Pos Pred Value               1.0000         1.0000       1.0000
## Neg Pred Value               1.0000         1.0000       1.0000
## Prevalence                   0.3333         0.3333       0.3333
## Detection Rate               0.3333         0.3333       0.3333
## Detection Prevalence         0.3333         0.3333       0.3333
## Balanced Accuracy            1.0000         1.0000       1.0000
```

Con las respuestas de las predicciones del algoritmo se noto que si esta reconociendo correctamente los datos de validacion

Conlusiones

- Debido a que el modelo lineal del sensor infrarojo no fue el esperado se pudo concluir que este tipo de sensores tienen gran variacion de sus datos, esto hace que la el modelo no pueda ser entrenado.
- Para el algoritmo Knn que reconoce el tipo de obstaculo se determino que el sensor ultrasonico es el que mas tiene presicion en los datos y este es la variable con mas relevancia en el modelo, es decir que el sensor ultrasonico es el fundamental para el entrenamiento del algoritmo.
- Debido a que los sensores tienen una respuesta computacionalmente muy rapida en comparacion a la mecanica, se tuvo que hacer uso de banderas en la programacion y un tiempo de espera para que captara solo un dato en el instante desead. De esta manera se logra una buena captacion de los datos para realizar la practica.