Universidad del Desarrollo
Facultad de Ingeniería

Taller 3

Taller de Inteligencia de Negocios

Docente:

Mauricio Herrera

Integrantes:

Jan Frese

Jorge Ramírez

Marcelo Cáceres

11 de noviembre de 2020

Problema 1

Para la resolución de este problema, se necesitaron instalar las librerías “dplyr” y “ggplot2”, junto con la descarga de la base de datos a utilizar “CO2_Mauna Loa Observatory.csv”.

Como paso previo a realizar los ejercicios propuestos para este problema, se eliminaron todas las variables y bases de datos ya registrados en la consola, junto con la limpieza de gráficos.

```
### Problema 1 ###
rm(list=ls()) # Para limpiar todas las variables
graphics.off() # Borra todos los gráficos
datos <- read.csv("SEM 6/TIN/Taller 3/CO2_Mauna Loa Observatory.csv")
install.packages("dplyr")
install.packages("ggplot2")
```

Pregunta 1.1:

Para encontrar la concentración anual promedio para cada año de “Carbon Dioxide” (ppm), se agruparon todos los datos por año y se calculó el promedio de la variable pedida, redondeando esta al segundo decimal por efecto de la base de datos, se omiten todos los datos que contengan NA y se realiza un diagrama de dispersión de la concentración de CO2 promedio con los valores obtenidos (**Fig 1**).

```
# Ejercicio 1
length(unique(datos$Year))
library(dplyr)
datos <- arrange(datos, desc(Year))
años <- group_by(datos,Year)
porAño <- summarise(años,Promedio=round(mean(Carbon.Dioxide..ppm.,na.rm=T),2))
porAño <- na.omit(porAño)
porAño

library(ggplot2)
ggplot(porAño) + geom_point(aes(x=Year, y=Promedio,)) +
```

ggtitle("Diagrama de concentración de CO2 promedio por año")

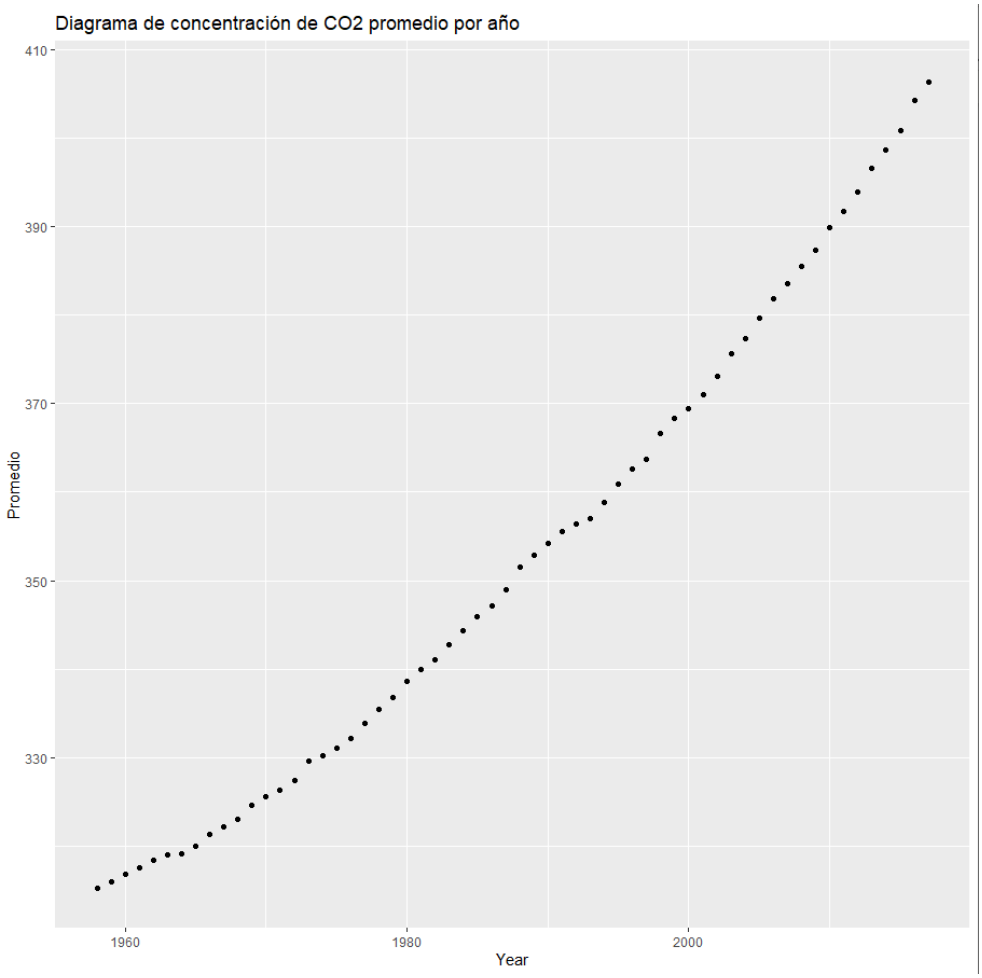

Fig 1.

Como se puede apreciar en el diagrama de la **Fig 1,** la concentración de CO2 desde el año 1990 hasta el año 2017 (rango de tiempo trabajado durante el ejercicio) fue en aumento año tras año, con una subida casi constante desde el año 2000 en adelante.

Pregunta 1.2:

Se pide realizar tres particiones de tiempo: 1958 - 1970, 1971 - 2000 y 2001 – 2017, se graficaron las tres particiones para finalmente realizar una regresión lineal de cada partición con el fin de determinar las tasas de variación de estas.

```
# Ejercicio 2

particion1 <- porAño[1:13,]

particion1


particion2 <- porAño[14:43,]

particion2


particion3 <- porAño[44:60,]

particion3


plot(particion1)

plot(particion2)

plot(particion3)


modelo1 <- lm(Promedio~Year,data = particion1)

plot(particion1) + abline(modelo1)


modelo2 <- lm(Promedio~Year,data = particion2)

plot(particion2) + abline(modelo2)


modelo3 <- lm(Promedio~Year,data = particion3)

plot(particion3) + abline(modelo3)


summary(modelo1)

summary(modelo2)

summary(modelo3)
```

Se obtuvieron los tres gráficos de dispersión correspondientes de la concentración de CO2 en estos años (**Fig 2, Fig 3, Fig 4**).

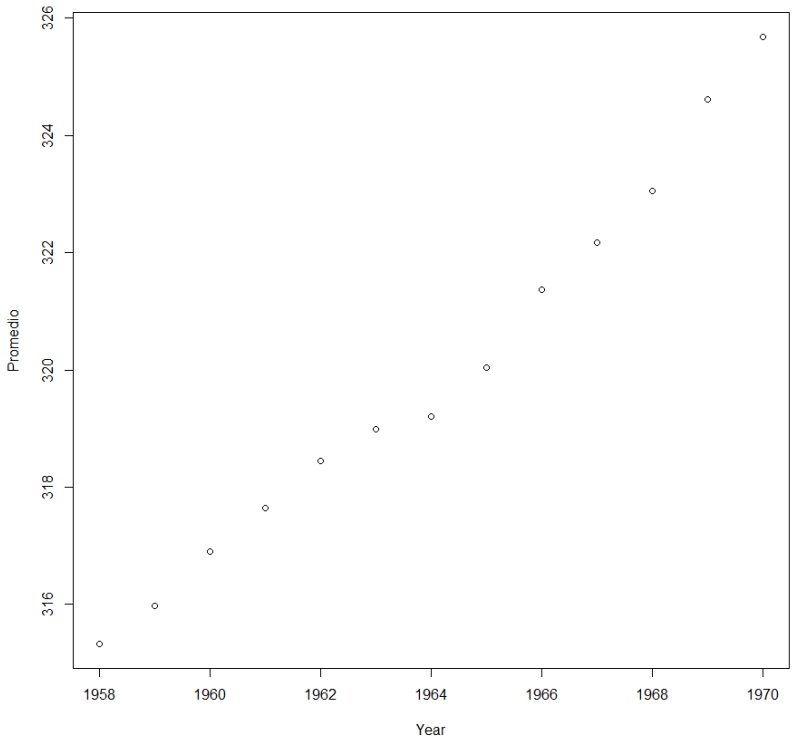

Fig 2

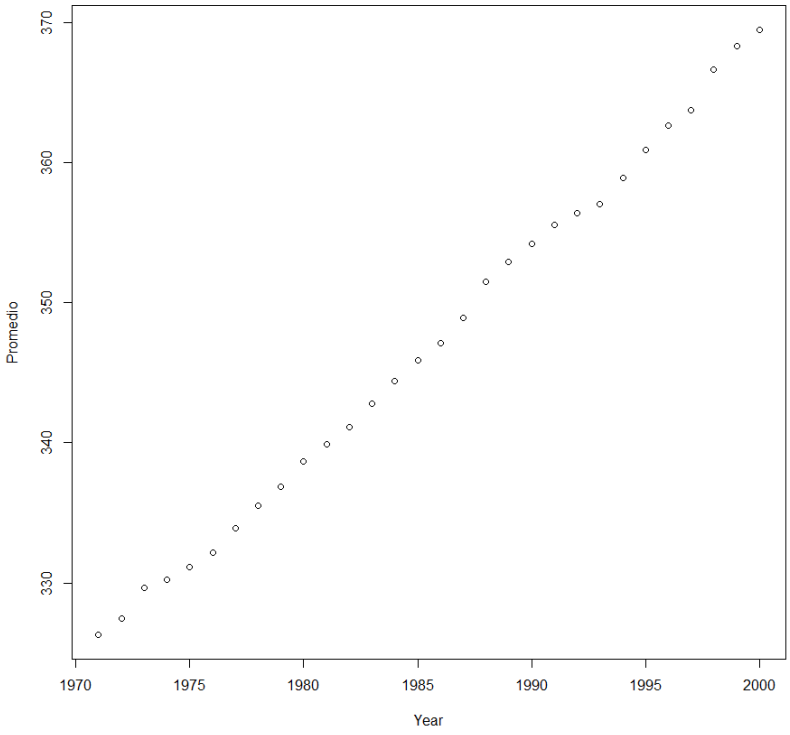

Fig 3

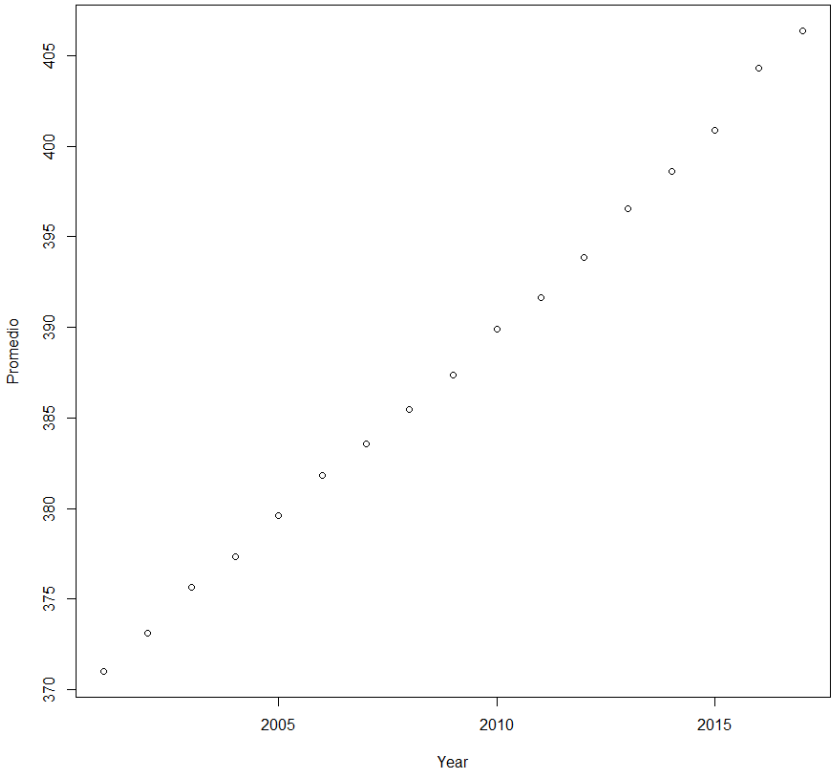

Fig 4

Tras los "summary" requeridos al final del desarrollo de este ejercicio podemos obtener los valores de la tasa de variación correspondientes a cada partición, siendo la tasa de variación la suma entre el valor "Intercept" y "Year" en los coeficientes, arrojando los siguientes valores:

- Partición 1: CO2(ppm) = -1303 + 0,826*Año
- Partición 2: CO2(ppm) = -2618 +1,493*Año
- Partición 3: CO2(ppm) =-3961 + 2,165*Año

Pregunta 1.3:

Para estimar la concentración de CO2 en el año 2030, calculamos el valor promedio de la recta la cual arroja la tasa de variación y la multiplicamos por el año pedido, de la siguiente manera:

```
# la ecuacion de la recta es Promedio = -3.961 + 2.165*Year


# Ejercicio 3

# Para el año 2030 es

b <- -3961 + 2.165*2030

b

# para 1960

c <- -3961 + 2.165*1960

c
```

Obteniendo que, en el año 2030, la concentración de CO2 será de 433.95 (ppm) aprox. Mientras que en el año 1960 la concentración fue de 282.2 (ppm) aprox. Para ambas predicciones, se decidió utilizar la fórmula de la tercera partición, ya que es la más cercana al 2030. Por otra parte, se ocupó esta misma ecuación para el año 1960, ya que no se especificaba cuál recta se debía utilizar.

Problema 2

Para el desarrollo de este problema, se necesitó instalar las librerías “klaR” y se utilizará la base de satos “winequality-red.csv”. Se solicita generar una clasificación para la calidad del vino.

En la base de datos se encontraron 12 variables, de las cuales las primeras 11 son de carácter independiente, mientras que la última correspondiente a “quality” será la que se evaluará.

Primero se transforma en una variable factor para que no influya en el modelo y luego se planta una semilla para entrenar el modelo con el 70% de los datos de la semilla y testear con el 30% restante.

```
### Problema 2 ###
data <- read.csv("SEM 6/TIN/Taller 3/winequality-red.csv", sep = ";")
#install.packages("klaR")
library(klaR)
head(data)
data$quality <- as.factor(data$quality)
str(data)
head(data)
set.seed(123)
train <- sample(1:nrow(data),0.7*nrow(data),replace=F)
Train.data<- data[train,]
Test.data<-data[-train,]
modelo <- lda(quality~.,data=Train.data)
modelo
head(Train.data)
p=predict(modelo, Test.data[,-12])
p
names(p)
p.class=p$class
p.class
tabla <- table(p.class,Test.data$quality)
tabla
tabla
confusionMatrix(tabla)
```

Tras ejecutar el modelo con los datos ya entrenados según el método “LDA”, se ocupa la variable “predict” para predecir lo que podría arrojar el clasificador con la excepción de la última variable dependiente, la cual se elimina, para predecir se ocuparon los datos del otro 30% restante no entrenados para realizar la prueba.

Al realizar la tabla de confusión, la sensibilidad de la clase 3 arrojó un 0%, mientras que la clase 4 un 7%, siendo la clase 5 la con mayor resultado de un 77%. Por otra parte, en la Especificidad, la clase 3,4,7 y 8 arrojaron un valor de 99%, mientras que la clase 5 un 67%.

A modo de mención, al observar los valores de Predicción Positivos, se tiene que los mas relevantes son los de la clase 5,6 y 7 rodeando el 60%, mientras que la clase 3 y 8 arrojaron un 0%.

Por otra parte, los valores de Predicción Negativos en su mayoría arrojaron valores mayores a 90%, siendo la clase 5 la única bajo esta cifra con un 78%.

Luego calculamos la exactitud, al ser una tabla 6x6 arrojó una exactitud de 62% aprox. Por lo que es un bajo porcentaje y bastante mejorable al no ser el ideal (sobre 90%).

Problema 3

Para este problema relacionado con una campaña de marketing directo realizado por una institución bancaria utilizaremos la base de datos “bank-3.csv” con tal de comparar entre varios clasificadores para encontrar el de mayor desempeño. Entre los clasificadores se eligieron tres: Creación de la regresión logística, LDA y SVM.

Al ejecutar la base de datos del banco, primero se convirtió a factor la variable dependiente ‘y’, luego se planta una semilla y entrenamos el modelo según la semilla 123 con el 70% de los datos, mientras que el 30% restante de los datos se utilizará para efectos de los datos test (prueba).

Inciso a.-

```
### Problema 3 ###

# inciso a

datos1 <- read.csv("SEM 6/TIN/Taller 3/bank-3.csv", sep=";")

datos1$y <- as.factor(datos1$y)

str(datos1)

head(datos1)
```

```
set.seed(123)
train1 <- sample(1:nrow(datos1),0.7*nrow(datos1),replace=F)
Train.data1<- datos1[train1,]
Test.data1<-datos1[-train1,]
#install.packages("caret")
library(caret)
library(klaR)
```

- Primer clasificador: Modelo de Regresión Logística

```
# primer clasificador
modeloRL <- glm(y~.,data=Train.data1,family=binomial)
summary(modeloRL)
coef(modeloRL)
summary(modeloRL)$coef

glm.probs=predict(modeloRL,type="response")
glm.probs[1:10]
contrasts(datos1$y)
glm.pred=rep("No",3164)

glm.pred[glm.probs>.5]="Si"

tabla <- table(glm.pred,Train.data1$y)
tabla

TP=tabla[1,1]
TN=tabla[2,2]
FP=tabla[1,2]
FN=tabla[2,1]

(exactitud=(TP+TN)/(TP+TN+FN+FP)) #91%
```

(precision=TP/(TP+FP)) #92%

(sensibilidad=TP/(TP+FN)) #97%

(Especificidad=TN/(TN+FP)) #39%

Al observar los datos arrojados tras realizar el modelo utilizando la variable 'y' en función del resto de las variables independientes según los datos entrenados (70% del set.seed(123)), se obtienen los siguientes valores a destacar:

jobblue-collar, maritalmarried, educationunknown, housingyes, loanyes, contactunknown, monthjan, monthjul, monthmar, monthmay, monthnov, monthoct, duration, poutcomesuccess.

Tras esto, se procedió a generar la Tabla de Confusión y tuvo verdaderos positivos 2735 y verdaderos negativos 130. De la tabla cabe destacar los siguientes valores:

1. Exactitud: 91%
2. Precisión: 92%
3. Sensibilidad: 97%
4. Especificidad: 39%

- Segundo Clasificador: LDA

```
# segundo clasificador
modelo1.1 <- lda(y~.,data=Test.data1)
modelo1.1
head(Test.data1)
p1.1=predict(modelo1.1, Test.data1[,-17])
p1.1
names(p1.1)
p.class1.1=p1.1$class
p.class1.1
tabla1.1 <- table(p.class1.1,Test.data1$y)
tabla1.1

TP1.1=tabla1.1[1,1]
TN1.1=tabla1.1[2,2]
```

```
FP1.1=tabla1.1[1,2]
FN1.1=tabla1.1[2,1]

(exactitud1.1=(TP1.1+TN1.1)/(TP1.1+TN1.1+FN1.1+FP1.1)) #91%
(precision1.1=TP1.1/(TP1.1+FP1.1)) #93%
(sensibilidad1.1=TP1.1/(TP1.1+FN1.1)) #97%
(Especificidad1.1=TN1.1/(TN1.1+FP1.1)) #42%
```

Al igual que en el modelo anterior, para este clasificador ocupamos el 70% de los datos entrenados. Se ocupo la variable 'predict' para predecir el comportamiento considerando solamente las variables independientes de los datos entrenados, eliminando así la variable dependiente 17 que no es de interés para el predict del modelo.

Se consideran las clases para la variable "p1" para crear la tabla de confusión con las variables dependientes de los datos entrenados, obteniendo de esta manera los siguientes valores:

1. Exactitud: 91%
2. Precisión: 93%
3. Sensibilidad: 97%
4. Especificidad: 42%

- Tercer Clasificador: SVM

```
# tercer clasificador
install.packages("e1071")
library(e1071)

svmfit4=svm(y~., data=Train.data1, kernel="linear")
p4=predict(svmfit4,Train.data1[,-17])

tabla2 <- table(predichos=p4,reales=Train.data1$y)
tabla2

TP2=tabla2[1,1]
TN2=tabla2[2,2]
```

```
FP2=tabla2[1,2]
FN2=tabla2[2,1]

(exactitud2=(TP2+TN2)/(TP2+TN2+FN2+FP2)) #89%
(precision2=TP2/(TP2+FP2)) #90%
(sensibilidad2=TP2/(TP2+FN2)) #98%
(Especificidad2=TN2/(TN2+FP2)) #24%
```

Para desarrollo de este clasificador se tuvo que recurrir a la librería "e1071" para realizar el clasificador SVM y así obtener los datos de la tabla de confusión que arrojó los siguientes resultados:

1. Exactitud: 89%
2. Precisión: 90%
3. Sensibilidad: 98%
4. Especificidad: 24%

Inciso b.-

```
# inciso b

# se va a seleccionar el clasificador 2 (lda), ya que hay una mayor cantidad
# de aciertos en más áreas

# segundo clasificador
modelo1.1 <- lda(y~.,data=Test.data1)
modelo1.1
head(Test.data1)
p1.1=predict(modelo1.1, Test.data1[,-17])
p1.1
names(p1.1)
p.class1.1=p1.1$class
```

```
p.class1.1

tabla1.1 <- table(p.class1.1,Test.data1$y)

tabla1.1


TP1.1=tabla1.1[1,1]

TN1.1=tabla1.1[2,2]

FP1.1=tabla1.1[1,2]

FN1.1=tabla1.1[2,1]


(exactitud1.1=(TP1.1+TN1.1)/(TP1.1+TN1.1+FN1.1+FP1.1)) #91%

(precision1.1=TP1.1/(TP1.1+FP1.1)) #93%

(sensibilidad1.1=TP1.1/(TP1.1+FN1.1)) #97%

(Especificidad1.1=TN1.1/(TN1.1+FP1.1)) #42%
```

Se escogió el clasificador 2 (LDA) debido a que era más acertado en diversas áreas, siendo casi igual que los otros dos, pero con una leve mejoría en Exactitud y Sensibilidad y un 20% mayor que el tercer clasificador (SVM) en Especificidad.

En base a esto se procede a realizar con el restante 30% de los datos el test, se define la nueva tabla de confusión en base a este clasificador y se obtienen los siguientes valores:

1. Exactitud: 91%
2. Precisión: 93%
3. Sensibilidad: 97%
4. Especificidad: 42%

Inciso c.-

Se solicita determinar que variable resulta ser la más importante en la clasificación, para esto se ocupó la variable dependiente de los datos testeados ‘y’ y se convirtieron en factor, luego se creó la matriz de confusión con “p.class1.1” y finalmente se entrenó el modelo con 3 repeticiones ocupando el método “repeatedcv”.

Finalmente se genera y grafica la importancia para visualizar lo solicitado en el problema (**Fig 5.**)

```
# inciso c
library(caret)
library(klaR)
Test.data1$y=as.factor(Test.data1$y)
confusionMatrix(p.class1.1, Test.data1$y)

control <- trainControl(method="repeatedcv", number=10, repeats=3)
model <- train(y~., data=datos1, method="glm",
               preProcess="scale", trControl=control)
# estimamos la importancia de las variables en el modelo
importancia <- varImp(model, scale=FALSE)
# resumen de la importancia
print(importancia)
plot(importancia)
```

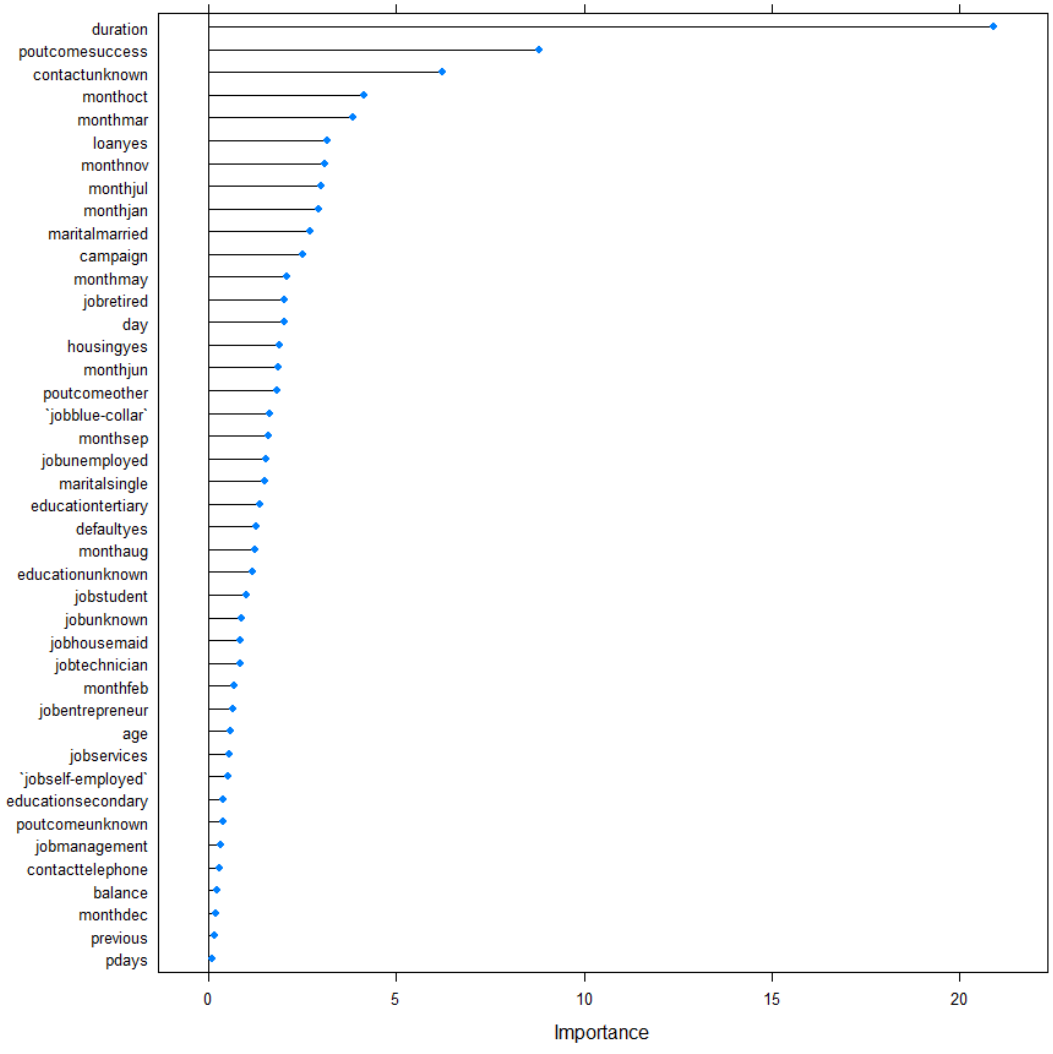

Fig 5.

A través del gráfico claramente la duración es la variable con mayor importancia con casi el doble que "poutcomesuccess" y "contactunknown" que vienen siendo las siguientes dos variables más relevantes para la clasificación.