Car Evaluation 2018 summary

Juan Carlos Bailón Elvira, Gerardo González-Cordero,
José López-Jiménez, Luis Alberto Segura Delgado
{bailone,segura2010}@correo.ugr.es, g2cordr@gmail.com, joselj@ugr.es

5 de diciembre de 2017

Car Evaluation 2018

Introducción

Trabajo para el curso de Ciencia Abierta del programa de Doctorado de la UGR. El dataset utilizado para este trabajo es "Car Evaluation Data Set", de Bohanec M. y Zupan B. UCI Machine Learning Repository. Este dataset está formado por 6 variables, además de la variable de clase que indica si el coche es aceptable o no.

Las variables son las siguientes:

- **buying**: Precio del coche. Valores: vhigh, high, med, low.
- **maint**: Precio de mantenimiento. Valores: vhigh, high, med, low.
- **doors**: Número de puertas. Valores: 2, 3, 4, 5more.
- **persons**: Número de pasajeros. Valores: 2, 4, more.
- **lug_boot**: Tamaño del maletero. Valores: small, med, big.
- **safety**: Seguridad del coche. Valores: high, med, low.

Reglas de Asociación

Se han aplicado reglas de asociación con el objetivo de detectar asociaciones entre las diferentes variables del dataset, de forma que podamos comprender mejor dicho dataset y qué variables son mas interesantes de cara a clasificar y evaluar los coches.

En primer lugar aplicamos el algoritmo Apriori, con un soporte minimo de 0.09 y una confianza de 0.8.

```
####################APLICAMOS REGLAS DE ASOCIACION#############################
# Aplica Apriori
library('arules')
library('arulesViz')
library('caret')

rules <- apriori(car_data, parameter = list(minlen=2, supp=0.09, conf=0.8), control = list(verbose=F))
length(rules)
```

```
## [1] 16
```

```
rules.sorted <- sort(rules, by="support")
inspect(rules.sorted)
```

```
##      lhs                          rhs           support    confidence
## [1]  {safety=low}              => {class=unacc} 0.33333333 1.0000000
## [2]  {persons=2}               => {class=unacc} 0.33333333 1.0000000
## [3]  {buying=vhigh}            => {class=unacc} 0.20833333 0.8333333
```

```
## [4]  {maint=vhigh}               => {class=unacc} 0.20833333 0.8333333
## [5]  {lug_boot=big,safety=low}   => {class=unacc} 0.11111111 1.0000000
## [6]  {persons=2,lug_boot=big}    => {class=unacc} 0.11111111 1.0000000
## [7]  {persons=4,safety=low}      => {class=unacc} 0.11111111 1.0000000
## [8]  {persons=more,safety=low}   => {class=unacc} 0.11111111 1.0000000
## [9]  {lug_boot=med,safety=low}   => {class=unacc} 0.11111111 1.0000000
## [10] {persons=2,lug_boot=med}    => {class=unacc} 0.11111111 1.0000000
## [11] {persons=2,safety=high}     => {class=unacc} 0.11111111 1.0000000
## [12] {persons=2,safety=med}      => {class=unacc} 0.11111111 1.0000000
## [13] {lug_boot=small,safety=low} => {class=unacc} 0.11111111 1.0000000
## [14] {persons=2,safety=low}      => {class=unacc} 0.11111111 1.0000000
## [15] {persons=2,lug_boot=small}  => {class=unacc} 0.11111111 1.0000000
## [16] {lug_boot=small,safety=med} => {class=unacc} 0.09085648 0.8177083
##      lift     count
## [1]  1.428099 576
## [2]  1.428099 576
## [3]  1.190083 360
## [4]  1.190083 360
## [5]  1.428099 192
## [6]  1.428099 192
## [7]  1.428099 192
## [8]  1.428099 192
## [9]  1.428099 192
## [10] 1.428099 192
## [11] 1.428099 192
## [12] 1.428099 192
## [13] 1.428099 192
## [14] 1.428099 192
## [15] 1.428099 192
## [16] 1.167769 157
```

Como podemos ver, obtenemos un conjunto de 16 reglas. Ordenadas por la medida de soporte, las dos primeras reglas nos proporcionan gran cantidad de información, puesto que nos indican que en un 33% (soporte=0.33) de los casos de nuestro dataset la compra de dicho coche será inaceptable en caso de que la seguridad proporcionada por este sea baja o el número de pasajeros sea 2 (es el mínimo). Esto nos indica que, en general, segun nuestro dataset aquellos coches que cumplan alguna o ambas de estas reglas seran, probablemente, inaceptables.

Ademas, también hemos obtenido las siguientes reglas:

- [11] {persons=2,safety=high} => {class=unacc} 0.11111111 1.0000000 1.428099 192
- [12] {persons=2,safety=med} => {class=unacc} 0.11111111 1.0000000 1.428099 192
- [14] {persons=2,safety=low} => {class=unacc} 0.11111111 1.0000000 1.428099 192

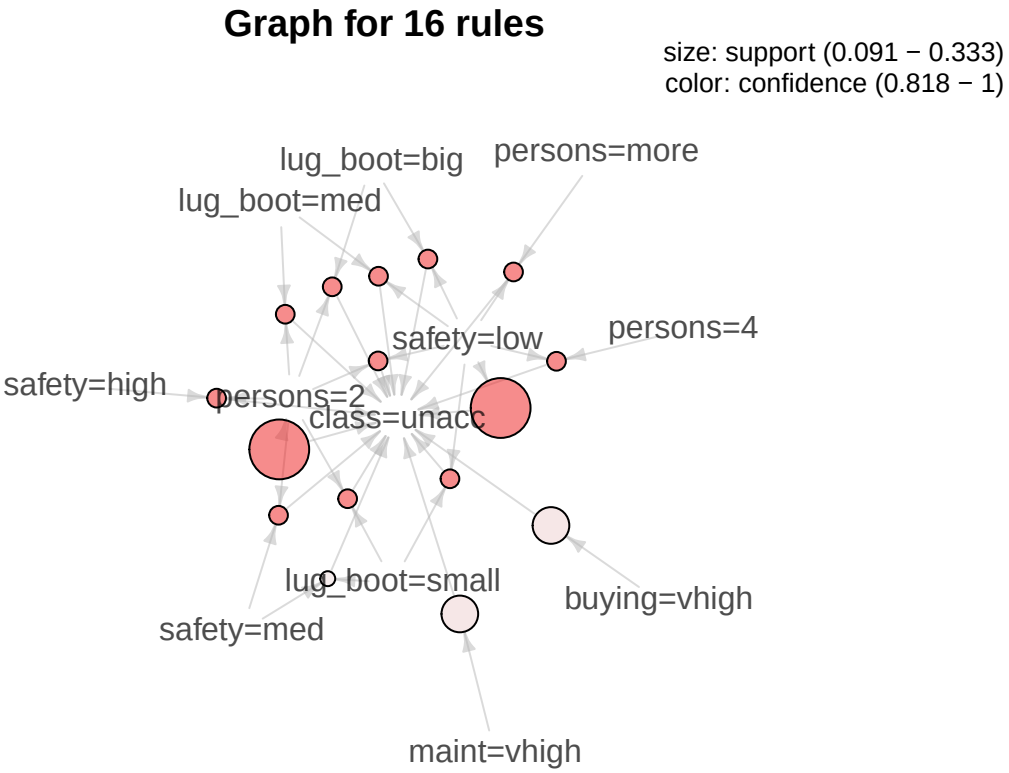
Que nos indican que, si el número de pasajeros es 2, no importa la seguridad del coche pues este será igualmente inaceptable.

```
plot(rules, method="graph", control=list(type="items"), measure='support', shading='confidence')
```

```
## Warning: Unknown control parameters: type
```

```
## Available control parameters (with default values):
## main  =  Graph for 16 rules
## nodeColors    =  c("#66CC6680", "#9999CC80")
## nodeCol   =  c("#EE0000FF", "#EE0303FF", "#EE0606FF", "#EE0909FF", "#EE0C0CFF", "#EE0F0FFF", "#EE121
## edgeCol   =  c("#474747FF", "#494949FF", "#4B4B4BFF", "#4D4D4DFF", "#4F4F4FFF", "#515151FF", "#53535
```

```
## alpha     =  0.5
## cex   =  1
## itemLabels    =  TRUE
## labelCol  =  #000000B3
## measureLabels     =  FALSE
## precision     =  3
## layout    =  NULL
## layoutParams  =  list()
## arrowSize     =  0.5
## engine    =  igraph
## plot  =  TRUE
## plot_options  =  list()
## max   =  100
## verbose   =  FALSE
```



También podemos ver la reglas de forma visual a partir del gráfico anterior. Podemos ver las relaciones entre items (pares atributo-valor) siguiendo las flechas que los unen. Los puntos intermedios nos indican el valor de las medidas, en este caso el tamaño nos indica el soporte y el color la confianza. En el gráfico podemos ver cómo los items 'safety=low' y 'persons=2' están unidos a puntos de un tamaño mayor y finalmente se unen a 'class=unacc'. Esto nos indica, como ya comentábamos antes, que si la seguridad es baja o el número de pasajeros 2, entonces, el coche es inaceptable.

A partir de las reglas de asociación nos queda claro que las variables mas importantes para evaluar un coche son el numero de pasajeros y la seguridad. Si alguna de estas variables es "mala", entonces el coche no sera aceptable.

Random Forest

```
#######PASAMOS A RANDOM FOREST PARA PREDECIR LA SEGURIDAD DEL COCHE SEGUN ATRIBUTOS

## 75%  para entrenar
smp_size <- floor(0.75 * nrow(car_data))

set.seed(123)
train_ind <- sample(seq_len(nrow(car_data)), size = smp_size)

train <- car_data[train_ind, ]
test <- car_data[-train_ind, ]

#como control haremos un CV con los siguientes parámertos
control <- trainControl(method="repeatedcv", number=10, repeats=3)
metric <- "Accuracy"
set.seed(141)
mtry <- sqrt(ncol(train))
tunegrid <- expand.grid(.mtry=mtry)
rf_default <- train(class~., data=train, method="rf", metric=metric, tuneGrid=tunegrid, trControl=contr

#visualizamos el acc del RF
print(rf_default)
```

```
## Random Forest
##
## 1296 samples
##    6 predictors
##    4 classes: 'acc', 'good', 'unacc', 'vgood'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 3 times)
## Summary of sample sizes: 1168, 1165, 1168, 1164, 1167, 1166, ...
## Resampling results:
##
##   Accuracy   Kappa
##   0.8647597  0.6755509
##
## Tuning parameter 'mtry' was held constant at a value of 2.645751
```

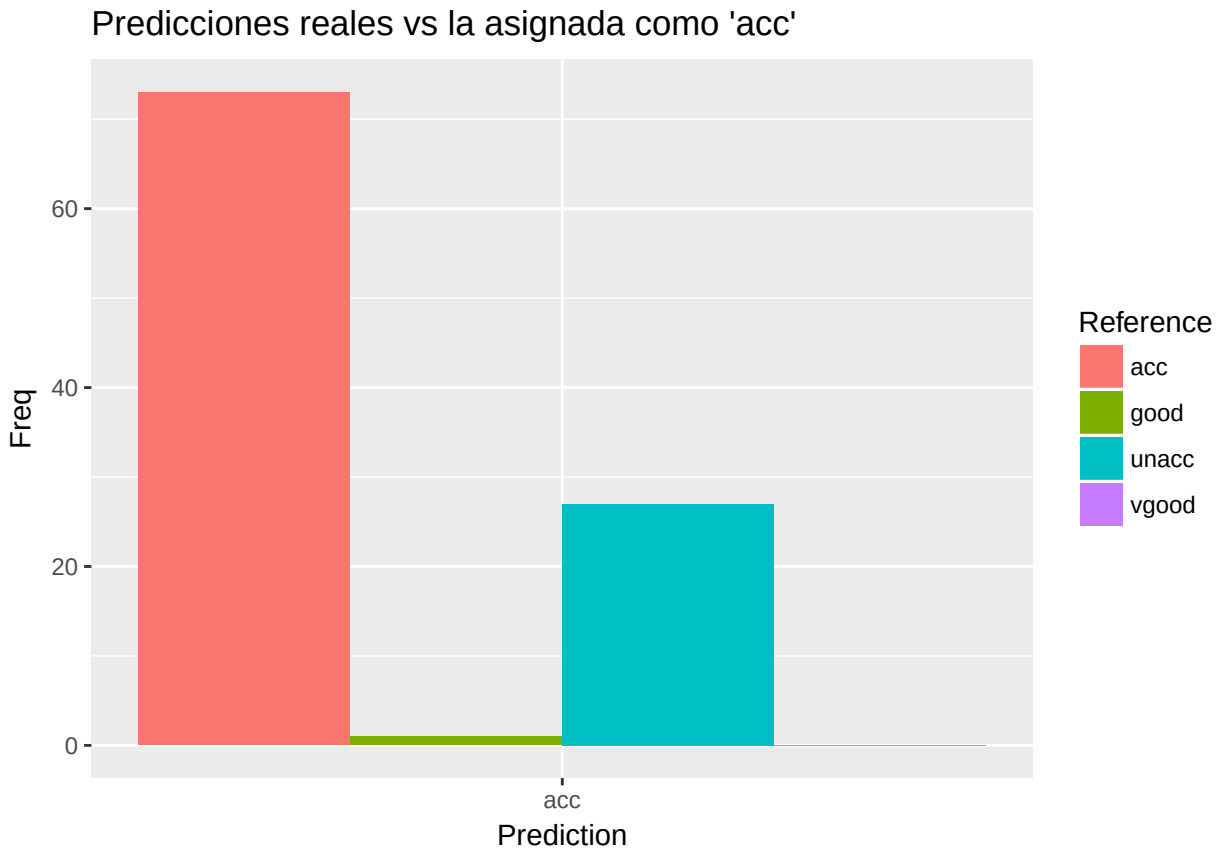
```
#
# Random Forest
#
# 1296 samples
# 6 predictor
# 4 classes: 'acc', 'good', 'unacc', 'vgood'
#
# No pre-processing
# Resampling: Cross-Validated (10 fold, repeated 3 times)
# Summary of sample sizes: 1165, 1166, 1166, 1167, 1168, 1167, ...
# Resampling results:
#
#   Accuracy   Kappa
# 0.8608195  0.6643527
```

```
#
# Tuning paramter 'mtry' was held constant at a value of 2.645751
#

# Con los datos de test validamos el modelo y vemos la matriz de confusión
test_predict <- predict(rf_default,test)
cfMatrix <- confusionMatrix(data = test$class, test_predict)

cfMatrix
```

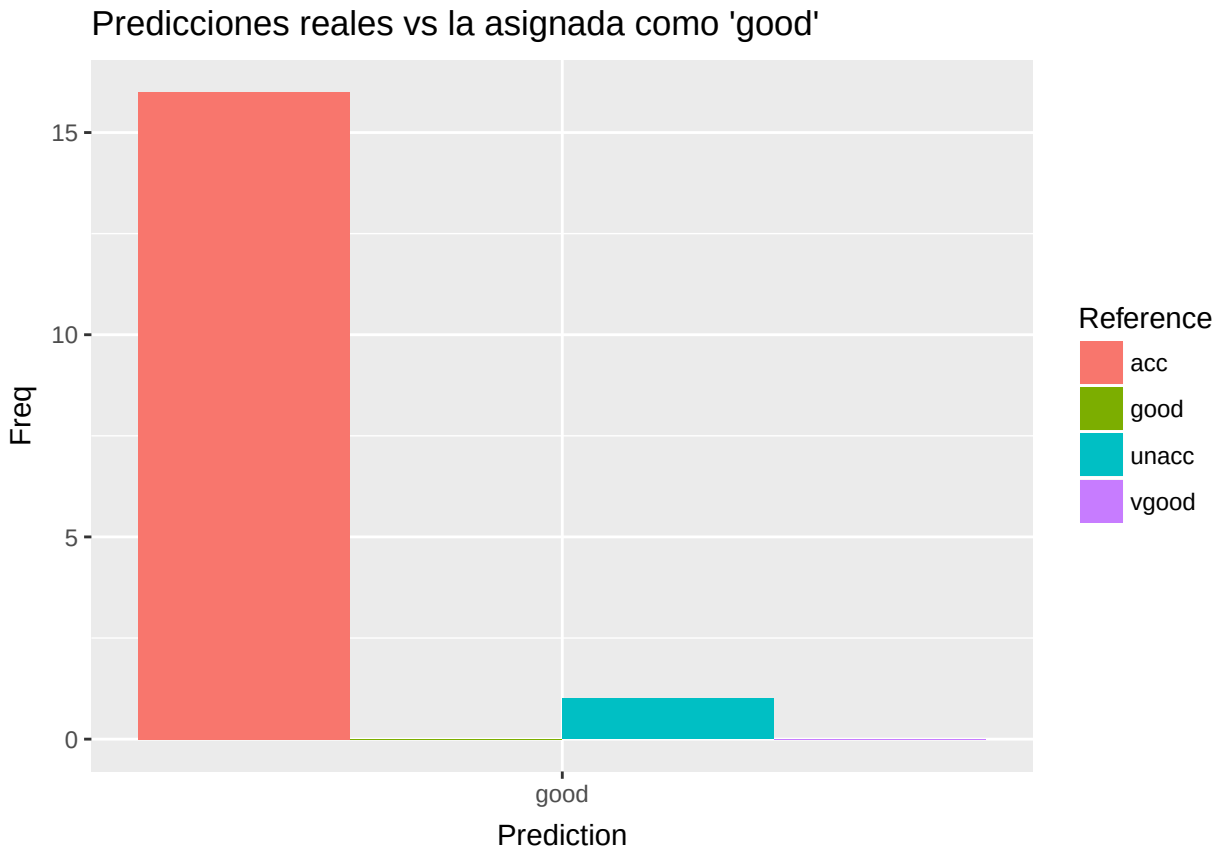
```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction acc good unacc vgood
##      acc    73    1    27     0
##      good   16    0     1     0
##      unacc   1    0   294     0
##      vgood  11    0     6     2
##
## Overall Statistics
##
##                Accuracy : 0.8542
##                  95% CI : (0.8173, 0.8861)
##     No Information Rate : 0.7593
##     P-Value [Acc > NIR] : 7.922e-07
##
##                   Kappa : 0.6581
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: acc Class: good Class: unacc Class: vgood
## Sensitivity              0.7228    0.000000       0.8963      1.00000
## Specificity              0.9154    0.960557       0.9904      0.96047
## Pos Pred Value           0.7228    0.000000       0.9966      0.10526
## Neg Pred Value           0.9154    0.997590       0.7518      1.00000
## Prevalence               0.2338    0.002315       0.7593      0.00463
## Detection Rate           0.1690    0.000000       0.6806      0.00463
## Detection Prevalence     0.2338    0.039352       0.6829      0.04398
## Balanced Accuracy        0.8191    0.480278       0.9434      0.98023
```

```
#
#
# Confusion Matrix and Statistics
#
# Reference
# Prediction acc good unacc vgood
# acc    72    1    28     0
# good   15    0     2     0
# unacc   1    0   294     0
# vgood  14    0     3     2
#
# Overall Statistics
#
```

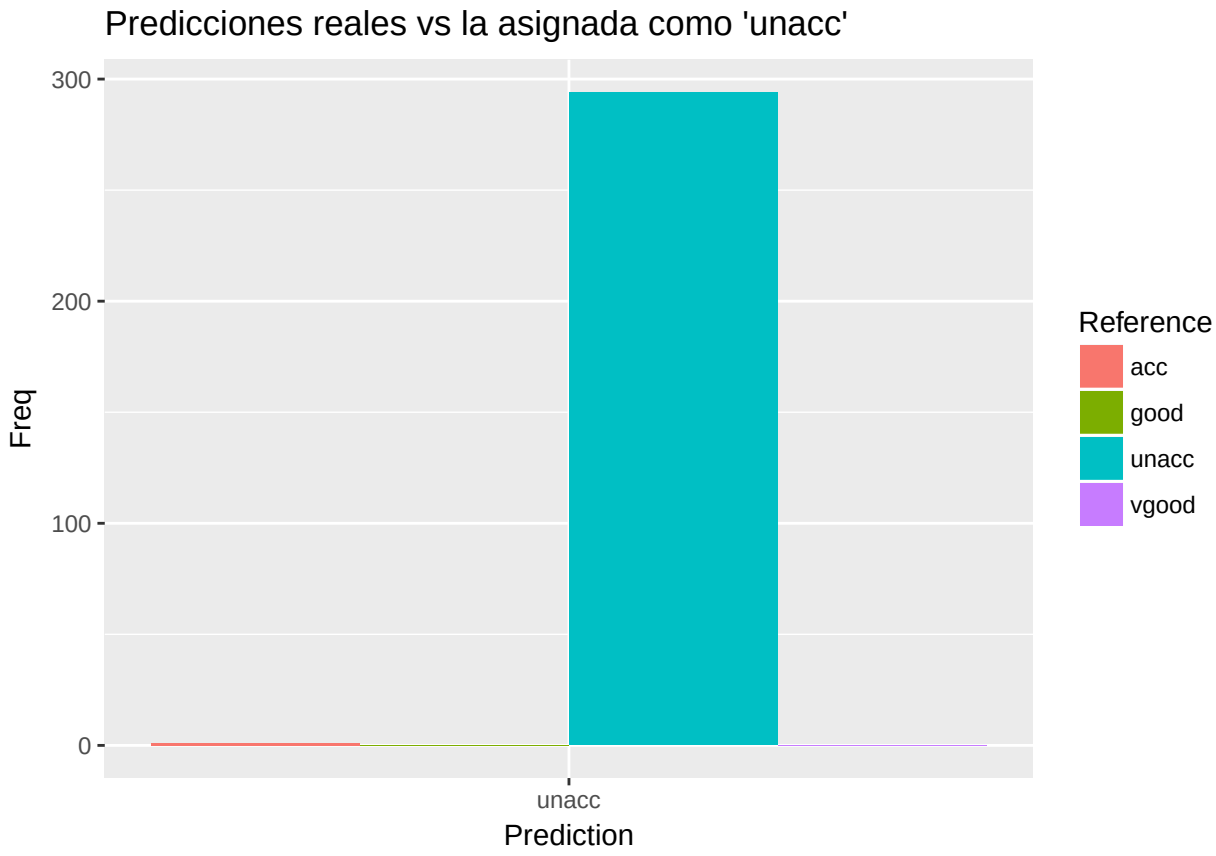
```
# Accuracy : 0.8519
# 95% CI : (0.8148, 0.884)
# No Information Rate : 0.7569
# P-Value [Acc > NIR] : 8.744e-07
#
# Kappa : 0.6535
# Mcnemar's Test P-Value : NA
#
# Statistics by Class:
#
#                      Class: acc Class: good Class: unacc Class: vgood
# Sensitivity              0.7059    0.000000       0.8991      1.00000
# Specificity              0.9121    0.960557       0.9905      0.96047
# Pos Pred Value           0.7129    0.000000       0.9966      0.10526
# Neg Pred Value           0.9094    0.997590       0.7591      1.00000
# Prevalence               0.2361    0.002315       0.7569      0.00463
# Detection Rate           0.1667    0.000000       0.6806      0.00463
# Detection Prevalence     0.2338    0.039352       0.6829      0.04398
# Balanced Accuracy        0.8090    0.480278       0.9448      0.98023

#Guardamos la tabla de la matriz de confusion para usarlo con gpplot
cm_table <- as.data.frame(cfMatrix$table)

#Para cada tipo de clase se guardan en diferentes DF y se usa ggplot para ver de manera mas grafica en
tab_acc <- cm_table[cm_table$Prediction=="acc",]

ggplot(tab_acc, aes(Prediction, Freq)) +
  ggtitle("Predicciones reales vs la asignada como 'acc'")+
  geom_bar(aes(fill = Reference), position = "dodge", stat="identity")
```



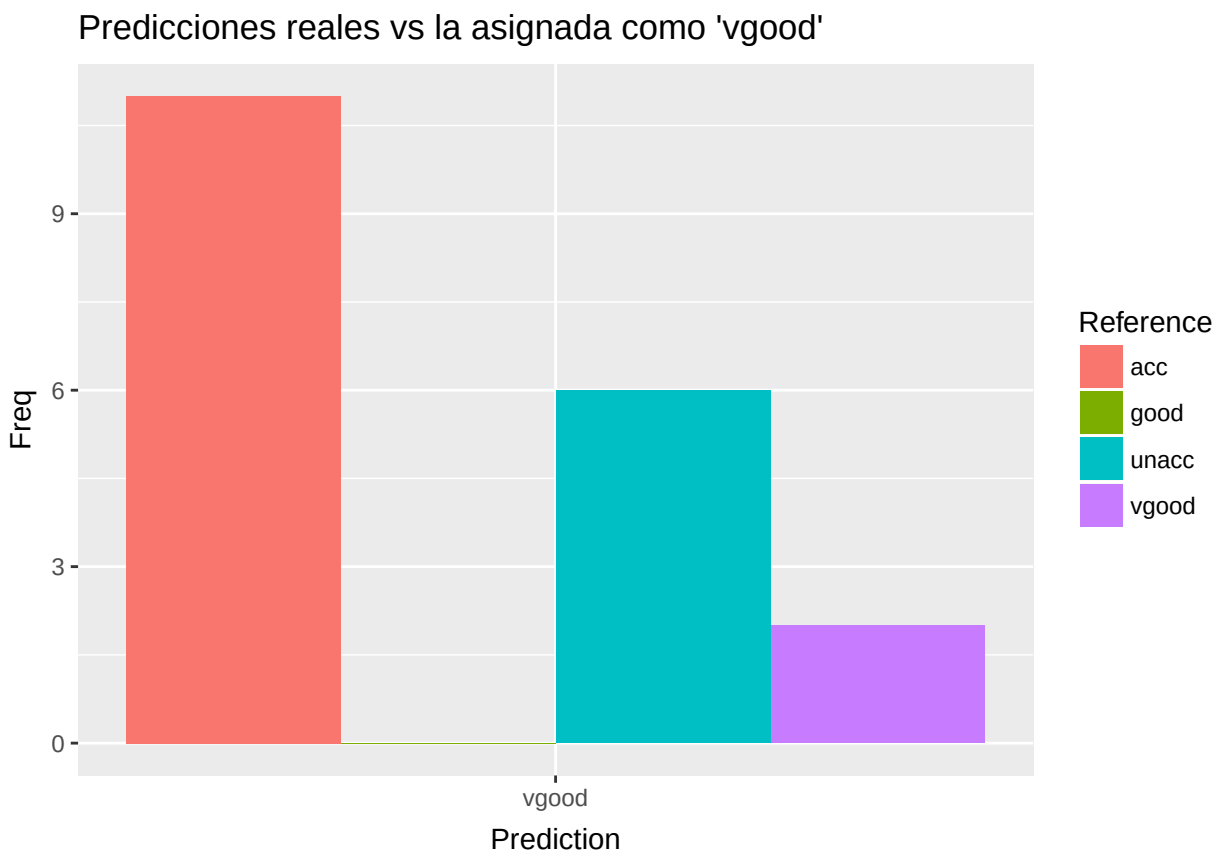

```
tab_good <- cm_table[cm_table$Prediction=="good",]

ggplot(tab_good, aes(Prediction, Freq)) +
  ggtitle("Predicciones reales vs la asignada como 'good'")+
  geom_bar(aes(fill = Reference), position = "dodge", stat="identity")
```




```
tab_unacc <- cm_table[cm_table$Prediction=="unacc",]

ggplot(tab_unacc, aes(Prediction, Freq)) +
  ggtitle("Predicciones reales vs la asignada como 'unacc'")+
  geom_bar(aes(fill = Reference), position = "dodge", stat="identity")
```




```
tab_vgood <- cm_table[cm_table$Prediction=="vgood",]

ggplot(tab_vgood, aes(Prediction,  Freq )) +
  ggtitle("Predicciones reales vs la asignada como 'vgood'")+
  geom_bar(aes(fill = Reference), position = "dodge", stat="identity")
```



El clasificador elaborado mediante la técnica Random Forests ha sido entrenado con el 75% de las muestras del dataset escogido. El 25% de las muestras restantes han sido empleadas para verificar el rendimiento del clasificador.

Como resultados más generales extraídos de la matriz de confusión, puede verse que la exactitud del clasificador se encuentra en 0.86, esto es, aproximadamente un 10% por encima de la Tasa de No Información (*NIR* por sus siglas en inglés).

Es destacable que la tasa de aciertos para las clases *unacc* y *acc* son notablemente más altas que para las otras dos clases *good* y *vgood*. Esta diferencia está directamente relacionada con la frecuencia con la que se dan estas clases en el *dataset* original. Cabe pensar que, de poder proporcionar mayor cantidad de muestras de estas últimas clases durante el entrenamiento, el rendimiento del clasificador mejoraría notablemente.

Conclusiones

A partir de la aplicacion de tecnicas de extraccion de reglas de asociacion hemos podido ver que las variables mas importantes para evaluar un coche son el numero de pasajeros y la seguridad. Si alguna de estas variables tiene un valor bajo, entonces el coche no sera aceptable.

El clasificador desarrollado obtiene una tasa de aciertos notablemente positiva para las clases más pobladas, especialmente a la hora de determinar si un coche es calificado como inaceptable o como alguna de las otras tres opciones.