TFM_Modelado

2024-02-27

```
# Carga de datos.
getwd()
```

```
## [1] "C:/Users/34682/Desktop/TFM/Posibles_datasets/vehicles"
```

```
setwd("C:/Users/34682/Desktop/TFM/Posibles_datasets/vehicles/")
v<-read.csv('Australian vehicle Prices Clean.csv')


# Validación cruzada 4-fold.
# Separación conjuntos de entrenamiento, validación y test.
set.seed(9)
train_ind <- sample(seq_len(nrow(v)), size =floor(0.9*nrow(v)))
v_train <- v[train_ind, ]
v_test <- v[-train_ind, ]
valid_cros_ind_1 <- sample(seq_len(nrow(v_train)),
                           size =floor(0.5*nrow(v_train)))
v_train_aux1 <- v_train[valid_cros_ind_1, ]
v_train_aux2 <- v_train[-valid_cros_ind_1, ]
valid_cros_ind_2 <- sample(seq_len(nrow(v_train_aux1)),
                           size =floor(0.5*nrow(v_train_aux1)))
valid_cros_ind_3 <- sample(seq_len(nrow(v_train_aux2)),
                           size =floor(0.5*nrow(v_train_aux2)))
v_train_1 <- v_train_aux1[valid_cros_ind_2, ]
v_train_2 <- v_train_aux1[-valid_cros_ind_2, ]
v_train_3 <- v_train_aux2[valid_cros_ind_3, ]
v_train_4 <- v_train_aux2[-valid_cros_ind_3, ]
v_train_i1 <- rbind(v_train_2, v_train_3, v_train_4)
v_train_i2 <- rbind(v_train_1, v_train_3, v_train_4)
v_train_i3 <- rbind(v_train_1, v_train_2, v_train_4)
v_train_i4 <- rbind(v_train_1, v_train_2, v_train_3)


# Normal con función de enlace la identidad.
mg1_1 <- glm(Price~ Year + UsedOrNew + Transmission + Engine + DriveType
             + FuelType  + FuelConsumption + Kilometres + Cylinders +
               BodyType + Doors + Seats + DollarAustralian + PriceIndex,
             family= gaussian(link= "identity"),v_train_i1)
pred_val <-predict(mg1_1, newdata = v_train_1, ty="response")
mg1_pr1_val <- mean(abs(v_train_1$Price - pred_val))
mg1_pr1_tr <- mean(abs(v_train_i1$Price - mg1_1$fitted.values))
rC1_1 <- 1- (mg1_1$deviance/mg1_1$null.deviance)

mg1_2 <- glm(Price~ Year + UsedOrNew + Transmission + Engine + DriveType
```

```
            + FuelType  + FuelConsumption + Kilometres + Cylinders +
              BodyType + Doors + Seats + DollarAustralian + PriceIndex,
            family= gaussian(link= "identity"),v_train_i2)
pred_val <-predict(mg1_2, newdata = v_train_2, ty="response")
mg1_pr2_val <- mean(abs(v_train_2$Price - pred_val))
mg1_pr2_tr <- mean(abs(v_train_i2$Price - mg1_2$fitted.values))
rC1_2 <- 1- (mg1_2$deviance/mg1_2$null.deviance)

mg1_3 <- glm(Price~ Year + UsedOrNew + Transmission + Engine + DriveType
            + FuelType  + FuelConsumption + Kilometres + Cylinders +
              BodyType + Doors + Seats + DollarAustralian + PriceIndex,
            family= gaussian(link= "identity"),v_train_i3)
pred_val <-predict(mg1_3, newdata = v_train_3, ty="response")
mg1_pr3_val <- mean(abs(v_train_3$Price - pred_val))
mg1_pr3_tr <- mean(abs(v_train_i3$Price - mg1_3$fitted.values))
rC1_3 <- 1- (mg1_3$deviance/mg1_3$null.deviance)

mg1_4 <- glm(Price~ Year + UsedOrNew + Transmission + Engine + DriveType
            + FuelType  + FuelConsumption + Kilometres + Cylinders +
              BodyType + Doors + Seats + DollarAustralian + PriceIndex,
            family= gaussian(link= "identity"),v_train_i4)
pred_val <-predict(mg1_4, newdata = v_train_4, ty="response")
mg1_pr4_val <- mean(abs(v_train_4$Price - pred_val))
mg1_pr4_tr <- mean(abs(v_train_i4$Price - mg1_4$fitted.values))
rC1_4 <- 1- (mg1_4$deviance/mg1_4$null.deviance)

#MAE_v
mg1_pr_val <- mean(c(mg1_pr1_val, mg1_pr2_val, mg1_pr3_val, mg1_pr4_val))
#MAE_train
mg1_pr_tr <- mean(c(mg1_pr1_tr, mg1_pr2_tr, mg1_pr3_tr, mg1_pr4_tr))
#mean(pseudo R^2)
rC1 <- mean(c(rC1_1, rC1_2, rC1_3, rC1_4))
c(mg1_pr_val, mg1_pr_tr, rC1)
```

```
## [1] 8598.3121663 8586.7025402    0.6511772
```

```
#Normal con función de enlace log
mg2_1 <- glm(Price~ Year + UsedOrNew + Transmission + Engine + DriveType
            + FuelType  + FuelConsumption + Kilometres + Cylinders +
              BodyType + Doors + Seats + DollarAustralian + PriceIndex,
            family= gaussian(link= "log"),v_train_i1)
pred_val <-predict(mg2_1, newdata = v_train_1, ty="response")
mg2_pr1_val <- mean(abs(v_train_1$Price - pred_val))
mg2_pr1_tr <- mean(abs(v_train_i1$Price - mg2_1$fitted.values))
rC2_1 <- 1- (mg2_1$deviance/mg2_1$null.deviance)

mg2_2 <- glm(Price~ Year + UsedOrNew + Transmission + Engine + DriveType
            + FuelType + FuelConsumption + Kilometres + Cylinders
            + BodyType + Doors + Seats + DollarAustralian + PriceIndex,
            family= gaussian(link= "log"),v_train_i2)
pred_val <-predict(mg2_2, newdata = v_train_2, ty="response")
mg2_pr2_val <- mean(abs(v_train_2$Price - pred_val))
mg2_pr2_tr <- mean(abs(v_train_i2$Price - mg2_2$fitted.values))
```

```r
rC2_2 <- 1- (mg2_2$deviance/mg2_2$null.deviance)

mg2_3 <- glm(Price~ Year + UsedOrNew + Transmission + Engine + DriveType
             + FuelType  + FuelConsumption + Kilometres + Cylinders +
               BodyType + Doors + Seats + DollarAustralian + PriceIndex,
             family= gaussian(link= "log"),v_train_i3)
pred_val <-predict(mg2_3, newdata = v_train_3, ty="response")
mg2_pr3_val <- mean(abs(v_train_3$Price - pred_val))
mg2_pr3_tr <- mean(abs(v_train_i3$Price - mg2_3$fitted.values))
rC2_3 <- 1- (mg2_3$deviance/mg2_3$null.deviance)

mg2_4 <- glm(Price~ Year + UsedOrNew + Transmission + Engine + DriveType
             + FuelType  + FuelConsumption + Kilometres + Cylinders +
               BodyType + Doors + Seats + DollarAustralian + PriceIndex,
             family= gaussian(link= "log"),v_train_i4)
pred_val <-predict(mg2_4, newdata = v_train_4, ty="response")
mg2_pr4_val <- mean(abs(v_train_4$Price - pred_val))
mg2_pr4_tr <- mean(abs(v_train_i4$Price - mg2_4$fitted.values))
rC2_4 <- 1- (mg2_4$deviance/mg2_4$null.deviance)

#MAE_v
mg2_pr_val <- mean(c(mg2_pr1_val, mg2_pr2_val, mg2_pr3_val, mg2_pr4_val))
#MAE_train
mg2_pr_tr <- mean(c(mg2_pr1_tr, mg2_pr2_tr, mg2_pr3_tr, mg2_pr4_tr))
#mean(pseudo R^2)
rC2 <- mean(c(rC2_1, rC2_2, rC2_3, rC2_4))
c(mg2_pr_val, mg2_pr_tr, rC2)
```

```
## [1] 7307.4239934 7287.8358525    0.7209465
```

```r
#Normal con función de enlace inversa
mg3_1 <- glm(Price~ Year + UsedOrNew + Transmission + Engine + DriveType
             + FuelType  + FuelConsumption + Kilometres + Cylinders +
               BodyType + Doors + Seats + DollarAustralian + PriceIndex,
             family= gaussian(link= "inverse"),v_train_i1)
pred_val <-predict(mg3_1, newdata = v_train_1, ty="response")
mg3_pr1_val <- mean(abs(v_train_1$Price - pred_val))
mg3_pr1_tr <- mean(abs(v_train_i1$Price - mg3_1$fitted.values))
rC3_1 <- 1- (mg3_1$deviance/mg3_1$null.deviance)

mg3_2 <- glm(Price~ Year + UsedOrNew + Transmission + Engine + DriveType
             + FuelType  + FuelConsumption + Kilometres + Cylinders +
               BodyType + Doors + Seats + DollarAustralian + PriceIndex,
             family= gaussian(link= "inverse"),v_train_i2)
pred_val <-predict(mg3_2, newdata = v_train_2, ty="response")
mg3_pr2_val <- mean(abs(v_train_2$Price - pred_val))
mg3_pr2_tr <- mean(abs(v_train_i2$Price - mg3_2$fitted.values))
rC3_2 <- 1- (mg3_2$deviance/mg3_2$null.deviance)

mg3_3 <- glm(Price~ Year + UsedOrNew + Transmission + Engine + DriveType
             + FuelType  + FuelConsumption + Kilometres + Cylinders +
               BodyType + Doors + Seats + DollarAustralian + PriceIndex,
             family= gaussian(link= "inverse"),v_train_i3)
```

```r
pred_val <-predict(mg3_3, newdata = v_train_3, ty="response")
mg3_pr3_val <- mean(abs(v_train_3$Price - pred_val))
mg3_pr3_tr <- mean(abs(v_train_i3$Price - mg3_3$fitted.values))
rC3_3 <- 1- (mg3_3$deviance/mg3_3$null.deviance)

mg3_4 <- glm(Price~ Year + UsedOrNew + Transmission + Engine + DriveType
             + FuelType  + FuelConsumption + Kilometres + Cylinders +
               BodyType + Doors + Seats + DollarAustralian + PriceIndex,
             family= gaussian(link= "inverse"),v_train_i4)
pred_val <-predict(mg3_4, newdata = v_train_4, ty="response")
mg3_pr4_val <- mean(abs(v_train_4$Price - pred_val))
mg3_pr4_tr <- mean(abs(v_train_i4$Price - mg3_4$fitted.values))
rC3_4 <- 1- (mg3_4$deviance/mg3_4$null.deviance)
#MAE_v
mg3_pr_val <- mean(c(mg3_pr1_val, mg3_pr2_val, mg3_pr3_val, mg3_pr4_val))
#MAE_train
mg3_pr_tr <- mean(c(mg3_pr1_tr, mg3_pr2_tr, mg3_pr3_tr, mg3_pr4_tr))
#mean(pseudo R^2)
rC3 <- mean(c(rC3_1, rC3_2, rC3_3, rC3_4))
c(mg3_pr_val, mg3_pr_tr, rC3)
```

```
## [1] 8161.7934057 8125.1032348    0.6825537
```

```r
#Gamma con función de enlace la identidad
# mg4_1 <- glm(Price~ Year + UsedOrNew + Transmission + Engine +
#DriveType
# + FuelType  + FuelConsumption + Kilometres + Cylinders + BodyType
# + Doors + Seats + DollarAustralian + PriceIndex ,
# family= Gamma(link= "identity"),v_train_i1)
# El algoritmo de estimación de parámetros no converge y
# no se puede considerar dicho modelo.

#Gamma con función de enlace log
mg5_1 <- glm(Price~ Year + UsedOrNew + Transmission + Engine + DriveType +
               FuelType  + FuelConsumption + Kilometres + Cylinders +
               BodyType + Doors + Seats + DollarAustralian + PriceIndex ,
             family= Gamma(link= "log"),v_train_i1)
pred_val <-predict(mg5_1, newdata = v_train_1, ty="response")
mg5_pr1_val <- mean(abs(v_train_1$Price - pred_val))
mg5_pr1_tr <- mean(abs(v_train_i1$Price - mg5_1$fitted.values))
rC5_1 <- 1- (mg5_1$deviance/mg5_1$null.deviance)

mg5_2 <- glm(Price~ Year + UsedOrNew + Transmission + Engine + DriveType +
               FuelType  + FuelConsumption + Kilometres + Cylinders +
               BodyType + Doors + Seats + DollarAustralian + PriceIndex ,
             family= Gamma(link= "log"),v_train_i2)
pred_val <-predict(mg5_2, newdata = v_train_2, ty="response")
mg5_pr2_val <- mean(abs(v_train_2$Price - pred_val))
mg5_pr2_tr <- mean(abs(v_train_i2$Price - mg5_2$fitted.values))
rC5_2 <- 1- (mg5_2$deviance/mg5_2$null.deviance)

mg5_3 <- glm(Price~ Year + UsedOrNew + Transmission + Engine + DriveType +
               FuelType  + FuelConsumption + Kilometres + Cylinders +
```

```
              BodyType + Doors + Seats + DollarAustralian + PriceIndex ,
            family= Gamma(link= "log"),v_train_i3)
pred_val <-predict(mg5_3, newdata = v_train_3, ty="response")
mg5_pr3_val <- mean(abs(v_train_3$Price - pred_val))
mg5_pr3_tr <- mean(abs(v_train_i3$Price - mg5_3$fitted.values))
rC5_3 <- 1- (mg5_3$deviance/mg5_3$null.deviance)

mg5_4 <- glm(Price~ Year + UsedOrNew + Transmission + Engine + DriveType
             + FuelType  + FuelConsumption + Kilometres + Cylinders +
               BodyType + Doors + Seats + DollarAustralian + PriceIndex ,
             family= Gamma(link= "log"),v_train_i4)
pred_val <-predict(mg5_4, newdata = v_train_4, ty="response")
mg5_pr4_val <- mean(abs(v_train_4$Price - pred_val))
mg5_pr4_tr <- mean(abs(v_train_i4$Price - mg5_4$fitted.values))
rC5_4 <- 1- (mg5_4$deviance/mg5_4$null.deviance)

#MAE_v
mg5_pr_val <- mean(c(mg5_pr1_val, mg5_pr2_val, mg5_pr3_val, mg5_pr4_val))
#MAE_train
mg5_pr_tr <- mean(c(mg5_pr1_tr, mg5_pr2_tr, mg5_pr3_tr, mg5_pr4_tr))
#mean(pseudo R^2)
rC5 <- mean(c(rC5_1, rC5_2, rC5_3, rC5_4))
c(mg5_pr_val, mg5_pr_tr, rC5)
```

```
## [1] 7306.0531022 7294.2724383    0.7671627
```

```
#Gamma con función de enlace inversa.
# mg6_1 <- glm(Price~ Year + UsedOrNew + Transmission + Engine +
# DriveType + FuelType  + FuelConsumption + Kilometres + Cylinders +
# BodyType + Doors + Seats + DollarAustralian + PriceIndex ,
# family= Gamma(link= "inverse"),v_train_i1)
# El algoritmo de estimación de parámetros no converge y
# no se puede considerar dicho modelo.

#Inversa gaussiana con función de enlace identidad
# mg7_1 <- glm(Price~ Year + UsedOrNew + Transmission + Engine
# + DriveType
# + FuelType  + FuelConsumption + Kilometres + Cylinders + BodyType
# + Doors +
#   Seats + DollarAustralian + PriceIndex ,
# family= inverse.gaussian(link= "identity"),v_train_i1)
# El algoritmo de estimación de parámetros no converge y
# no se puede considerar dicho modelo.

#Inversa gaussiana con función de enlace log
# mg8_1 <- glm(Price~ Year + UsedOrNew + Transmission + Engine +
# DriveType +
# FuelType  + FuelConsumption + Kilometres + Cylinders + BodyType +
# Doors +
#   Seats + DollarAustralian + PriceIndex ,
# family= inverse.gaussian(link= "log"),v_train_i1)
# pred_val <-predict(mg8_1, newdata = v_train_1, ty="response")
# mg8_pr1_val <- mean(abs(v_train_1$Price - pred_val))
```

```
# mg8_pr1_tr <- mean(abs(v_train_i1$Price - mg8_1$fitted.values))
# rC8_1 <- 1- (mg8_1$deviance/mg8_1$null.deviance)
# mg8_2 <- glm(Price~ Year + UsedOrNew + Transmission + Engine +
# DriveType +
# FuelType  + FuelConsumption + Kilometres + Cylinders + BodyType
# + Doors
# + Seats + DollarAustralian + PriceIndex ,
# family= inverse.gaussian(link= "log"),v_train_i2)
# El algoritmo de estimación de parámetros no converge y
# no se puede considerar dicho modelo.

#Inversa gaussiana con función de enlace inversa
# mg9_1 <- glm(Price~ Year + UsedOrNew + Transmission + Engine +
# DriveType +
# FuelType  + FuelConsumption + Kilometres + Cylinders + BodyType +
# Doors +
#   Seats + DollarAustralian + PriceIndex ,
# family= inverse.gaussian(link= "inverse"),v_train_i1)
# El algoritmo de estimación de parámetros no converge y
# no se puede considerar dicho modelo.

#Inversa gaussiana con función de enlace inversa cuadrática
# mg10_1 <- glm(Price~ Year + UsedOrNew + Transmission + Engine +
# DriveType + FuelType  + FuelConsumption + Kilometres + Cylinders
# + BodyType
# + Doors + Seats + DollarAustralian + PriceIndex ,
# family= inverse.gaussian(link= "1/mu^2"),v_train_i1)
# El algoritmo de estimación de parámetros no converge y
# no se puede considerar dicho modelo.

#Poisson con función de enlace identidad
# mg11_1 <- glm(as.integer(Price)~ Year + UsedOrNew + Transmission
# + Engine
# + DriveType + FuelType  + FuelConsumption + Kilometres + Cylinders +
#   BodyType + Doors + Seats + DollarAustralian + PriceIndex ,
# family= poisson(link= "identity"),v_train_i1)
# El algoritmo de estimación de parámetros no converge y no
# se puede considerar dicho modelo.

#Poisson con función de enlace log
mg12_1 <- glm(as.integer(Price)~ Year + UsedOrNew + Transmission +
                Engine +
                DriveType + FuelType  + FuelConsumption + Kilometres +
                Cylinders + BodyType + Doors + Seats + DollarAustralian +
                PriceIndex ,family= poisson(link= "log"),v_train_i1)
pred_val <-predict(mg12_1, newdata = v_train_1, ty="response")
mg12_pr1_val <- mean(abs(as.integer(v_train_1$Price) - round(pred_val,0)))
mg12_pr1_tr <- mean(abs(as.integer(v_train_i1$Price) -
                          round(mg12_1$fitted.values,0)))
rC12_1 <- 1- (mg12_1$deviance/mg12_1$null.deviance)

mg12_2 <- glm(Price~ Year + UsedOrNew + Transmission + Engine +
                DriveType +
```

```
                FuelType  + FuelConsumption + Kilometres + Cylinders +
                BodyType + Doors + Seats + DollarAustralian + PriceIndex,
              family= poisson(link= "log"),v_train_i2)
pred_val <-predict(mg12_2, newdata = v_train_2, ty="response")
mg12_pr2_val <- mean(abs(as.integer(v_train_2$Price) - round(pred_val,0)))
mg12_pr2_tr <- mean(abs(as.integer(v_train_i2$Price) -
                          round(mg12_2$fitted.values,0)))
rC12_2 <- 1- (mg12_2$deviance/mg12_2$null.deviance)

mg12_3 <- glm(Price~ Year + UsedOrNew + Transmission + Engine +
                DriveType +
                FuelType  + FuelConsumption + Kilometres + Cylinders +
                BodyType + Doors + Seats + DollarAustralian + PriceIndex,
              family= poisson(link= "log"),v_train_i3)
pred_val <-predict(mg12_3, newdata = v_train_3, ty="response")
mg12_pr3_val <- mean(abs(as.integer(v_train_3$Price) - round(pred_val,0)))
mg12_pr3_tr <- mean(abs(as.integer(v_train_i3$Price) -
                          round(mg12_3$fitted.values,0)))
rC12_3 <- 1- (mg12_3$deviance/mg12_3$null.deviance)

mg12_4 <- glm(Price~ Year + UsedOrNew + Transmission + Engine +
                DriveType +
                FuelType  + FuelConsumption + Kilometres + Cylinders +
                BodyType + Doors + Seats + DollarAustralian + PriceIndex,
              family= poisson(link= "log"),v_train_i4)
pred_val <-predict(mg12_4, newdata = v_train_4, ty="response")
mg12_pr4_val <- mean(abs(as.integer(v_train_4$Price) - round(pred_val,0)))
mg12_pr4_tr <- mean(abs(as.integer(v_train_i4$Price) -
                          round(mg12_4$fitted.values,0)))
rC12_4 <- 1- (mg12_4$deviance/mg12_4$null.deviance)

#MAE_v
mg12_pr_val <- mean(c(mg12_pr1_val, mg12_pr2_val, mg12_pr3_val,
                      mg12_pr4_val))
#MAE_train
mg12_pr_tr <- mean(c(mg12_pr1_tr, mg12_pr2_tr, mg12_pr3_tr, mg12_pr4_tr))
#mean(pseudo R^2)
rC12 <- mean(c(rC12_1, rC12_2, rC12_3, rC12_4))
c(mg12_pr_val, mg12_pr_tr, rC12)
```

```
## [1] 7245.3824468 7230.8703060    0.7595578
```

```
#Poisson con función de enlace sqrt
mg13_1 <- glm(as.integer(Price)~ Year + UsedOrNew + Transmission +
                Engine + DriveType + FuelType  + FuelConsumption
              + Kilometres
              + Cylinders + BodyType + Doors + Seats + DollarAustralian +
                PriceIndex ,family= poisson(link= "sqrt"),v_train_i1)
pred_val <-predict(mg13_1, newdata = v_train_1, ty="response")
mg13_pr1_val <- mean(abs(as.integer(v_train_1$Price) - round(pred_val,0)))
mg13_pr1_tr <- mean(abs(as.integer(v_train_i1$Price) -
                          round(mg13_1$fitted.values,0)))
rC13_1 <- 1- (mg13_1$deviance/mg13_1$null.deviance)
```

```
mg13_2 <- glm(Price~ Year + UsedOrNew + Transmission + Engine +
                DriveType +
                FuelType  + FuelConsumption + Kilometres + Cylinders +
                BodyType + Doors + Seats + DollarAustralian + PriceIndex,
              family= poisson(link= "sqrt"),v_train_i2)
pred_val <-predict(mg13_2, newdata = v_train_2, ty="response")
mg13_pr2_val <- mean(abs(as.integer(v_train_2$Price) - round(pred_val,0)))
mg13_pr2_tr <- mean(abs(as.integer(v_train_i2$Price) -
                          round(mg13_2$fitted.values,0)))
rC13_2 <- 1- (mg13_2$deviance/mg13_2$null.deviance)

mg13_3 <- glm(Price~ Year + UsedOrNew + Transmission + Engine +
                DriveType +
                FuelType  + FuelConsumption + Kilometres + Cylinders +
                BodyType + Doors + Seats + DollarAustralian + PriceIndex,
              family= poisson(link= "sqrt"),v_train_i3)
pred_val <-predict(mg13_3, newdata = v_train_3, ty="response")
mg13_pr3_val <- mean(abs(as.integer(v_train_3$Price) - round(pred_val,0)))
mg13_pr3_tr <- mean(abs(as.integer(v_train_i3$Price) -
                          round(mg13_3$fitted.values,0)))
rC13_3 <- 1- (mg13_3$deviance/mg13_3$null.deviance)

mg13_4 <- glm(Price~ Year + UsedOrNew + Transmission + Engine +
                DriveType +
                FuelType  + FuelConsumption + Kilometres + Cylinders +
                BodyType + Doors + Seats + DollarAustralian + PriceIndex,
              family= poisson(link= "sqrt"),v_train_i4)
pred_val <-predict(mg13_4, newdata = v_train_4, ty="response")
mg13_pr4_val <- mean(abs(as.integer(v_train_4$Price) - round(pred_val,0)))
mg13_pr4_tr <- mean(abs(as.integer(v_train_i4$Price) -
                          round(mg13_4$fitted.values,0)))
rC13_4 <- 1- (mg13_4$deviance/mg13_4$null.deviance)

#MAE_v
mg13_pr_val <- mean(c(mg13_pr1_val, mg13_pr2_val, mg13_pr3_val,
                      mg13_pr4_val))
#MAE_train
mg13_pr_tr <- mean(c(mg13_pr1_tr, mg13_pr2_tr, mg13_pr3_tr,
                     mg13_pr4_tr))
#mean(pseudo R^2)
rC13 <- mean(c(rC13_1, rC13_2, rC13_3, rC13_4))
c(mg13_pr_val, mg13_pr_tr, rC13)
```

```
## [1] 7627.6240238 7613.9599924    0.7367348
```

```
#Modelo lineal final obtenido: poisson con familia log
mg_full <-glm(as.integer(Price)~ Year + UsedOrNew + Transmission +
                Engine + DriveType + FuelType  + FuelConsumption
              + Kilometres
              + Cylinders + BodyType + Doors + Seats +
                DollarAustralian +
                PriceIndex ,family= poisson(link= "log"),v_train)
```

```r
# Optimización del modelo y validación

# Selección de parámetros. No se elimina ninguno.
library(car)
```

```
## Loading required package: carData
```

```r
library(MASS)
stepAIC(mg_full)
```

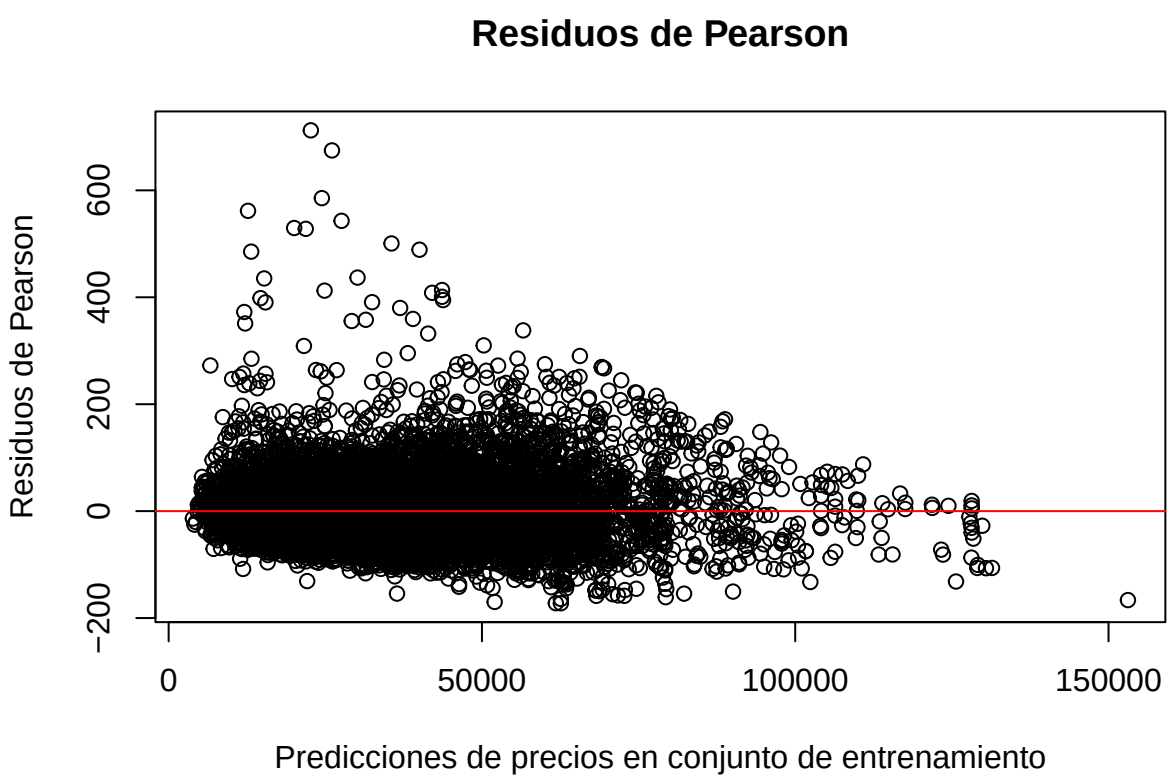
```
## Start:  AIC=39951242
## as.integer(Price) ~ Year + UsedOrNew + Transmission + Engine +
##     DriveType + FuelType + FuelConsumption + Kilometres + Cylinders +
##     BodyType + Doors + Seats + DollarAustralian + PriceIndex
##
##                     Df Deviance      AIC
## <none>                 39778481 39951242
## - UsedOrNew          1 39781812 39954572
## - Doors              1 39790904 39963664
## - Transmission       1 39793981 39966741
## - PriceIndex         1 39920238 40092998
## - DollarAustralian   1 39946437 40119197
## - Seats              1 39976552 40149312
## - FuelConsumption    1 40742429 40915189
## - BodyType           3 41187735 41360491
## - Engine             1 41216520 41389280
## - Cylinders          1 41390454 41563214
## - FuelType           2 43761688 43934446
## - Kilometres         1 44259910 44432670
## - Year               1 44887799 45060559
## - DriveType          3 45044310 45217066

##
## Call:  glm(formula = as.integer(Price) ~ Year + UsedOrNew + Transmission +
##     Engine + DriveType + FuelType + FuelConsumption + Kilometres +
##     Cylinders + BodyType + Doors + Seats + DollarAustralian +
##     PriceIndex, family = poisson(link = "log"), data = v_train)
##
## Coefficients:
##          (Intercept)                 Year         UsedOrNewUSED
##           -1.153e+02            6.183e-02           -1.068e-02
##   TransmissionManual               Engine        DriveTypeFront
##           -2.133e-02            1.415e-01           -2.930e-01
##       DriveTypeOther        DriveTypeRear         FuelTypeOther
##            5.729e-03           -1.030e-01            1.360e-02
##     FuelTypeUnleaded       FuelConsumption            Kilometres
##           -2.105e-01           -3.114e-02           -2.373e-06
##            Cylinders        BodyTypeOther         BodyTypeSedan
##            1.095e-01            2.358e-01            1.343e-01
## BodyTypeSUV/Ute/Tray                Doors                Seats
##            1.012e-01           -1.085e-02            2.400e-02
##     DollarAustralian            PriceIndex
```

```
##            2.180e-01             2.679e-03
##
## Degrees of Freedom: 14300 Total (i.e. Null);  14281 Residual
## Null Deviance:       165300000
## Residual Deviance: 39780000  AIC: 39950000
```

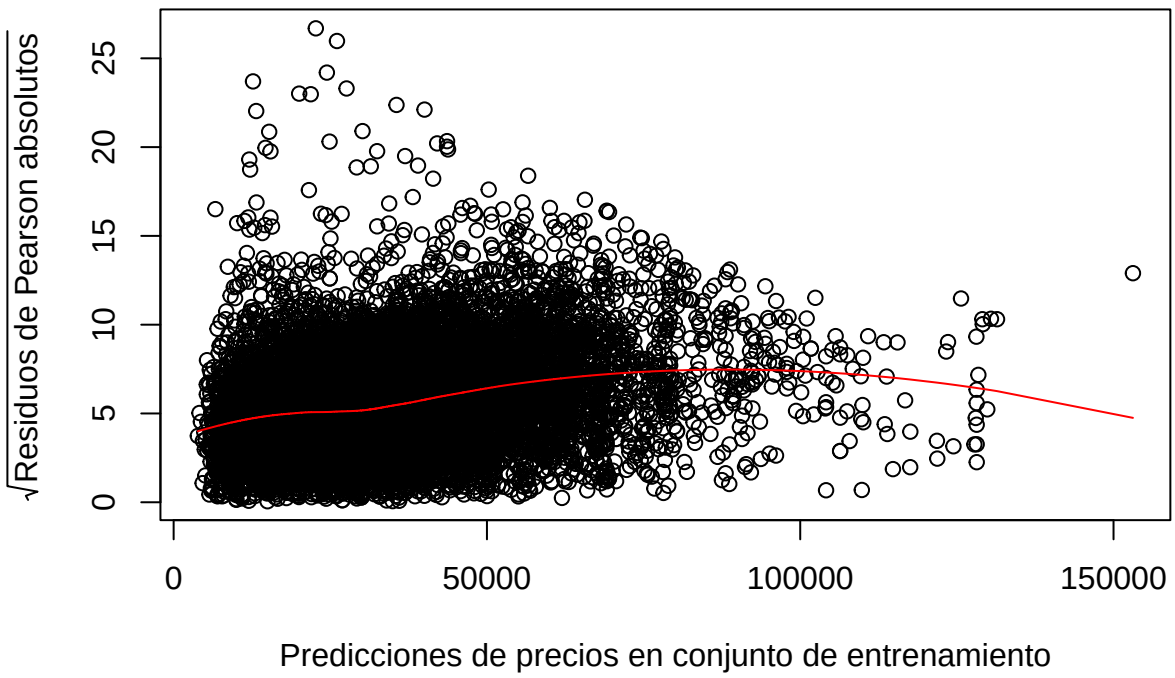
```r
summary(v)
```

```
##       Year       UsedOrNew          Transmission           Engine
##  Min.   :2000   Length:15891       Length:15891       Min.   :1.000
##  1st Qu.:2013   Class :character   Class :character   1st Qu.:2.000
##  Median :2017   Mode  :character   Mode  :character   Median :2.200
##  Mean   :2016                                         Mean   :2.341
##  3rd Qu.:2020                                         3rd Qu.:2.500
##  Max.   :2023                                         Max.   :6.000
##   DriveType           FuelType         FuelConsumption    Kilometres
##  Length:15891       Length:15891       Min.   : 0.000   Min.   :     1
##  Class :character   Class :character   1st Qu.: 6.700   1st Qu.: 43866
##  Mode  :character   Mode  :character   Median : 7.600   Median : 85952
##                                        Mean   : 7.653   Mean   : 98446
##                                        3rd Qu.: 8.500   3rd Qu.:143000
##                                        Max.   :15.000   Max.   :350000
##    Cylinders      BodyType             Doors            Seats
##  Min.   :3.00   Length:15891       Min.   :2.000   Min.   :2.000
##  1st Qu.:4.00   Class :character   1st Qu.:4.000   1st Qu.:5.000
##  Median :4.00   Mode  :character   Median :4.000   Median :5.000
##  Mean   :4.34                      Mean   :4.026   Mean   :5.099
##  3rd Qu.:4.00                      3rd Qu.:4.000   3rd Qu.:5.000
##  Max.   :8.00                      Max.   :5.000   Max.   :8.000
##      Price        DollarAustralian   PriceIndex
##  Min.   :    88   Min.   :1.241    Min.   : 65.79
##  1st Qu.: 18990   1st Qu.:1.473    1st Qu.:111.14
##  Median : 28990   Median :1.517    Median :120.42
##  Mean   : 33716   Mean   :1.534    Mean   :117.44
##  3rd Qu.: 42422   3rd Qu.:1.611    3rd Qu.:127.90
##  Max.   :139990   Max.   :1.773    Max.   :135.76
```

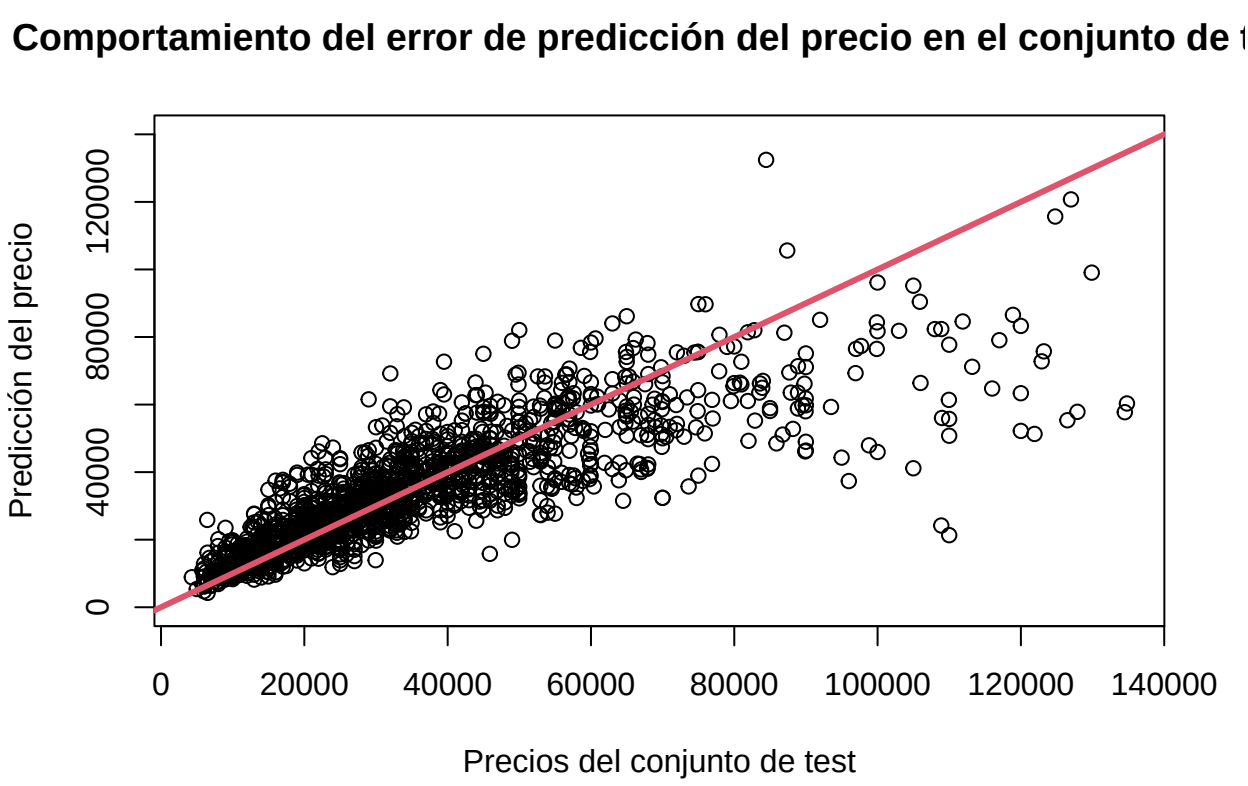
```r
# Gráficos de validación y coeficiente de determinación.
rC <- 1- (mg_full$deviance/mg_full$null.deviance)
par(mfrow = c(1, 1))
plot(mg_full$fitted.values, residuals(mg_full, type = "pearson"),
     xlab = "Predicciones de precios en conjunto de entrenamiento",
     ylab = "Residuos de Pearson",
     main = "Residuos de Pearson")
abline(h = 0, col = "red")  # Línea horizontal en y = 0
```




```r
d_aux <- data.frame(x = mg_full$fitted.values,
          y = predict(loess(sqrt(abs(residuals(mg_full,
                                               type = "pearson")))
                            ~ mg_full$fitted.values)))
d_aux <- d_aux[order(d_aux$x), ]
plot(mg_full$fitted.values, sqrt(abs(residuals(mg_full,
                                               type = "pearson"))),
     xlab = "Predicciones de precios en conjunto de entrenamiento",
     ylab = expression(sqrt("Residuos de Pearson absolutos")),
     main = "Raíz cuadrada de los Residuos de Pearson absolutos")
# Línea horizontal en y = 0
#lines(x, lm(y~poly(x,3))$fitted.values, col = "red")
lines(d_aux$x, d_aux$y, col = "red")
```




```
# Capacidad predictiva respecto al conjunto de test.
pred_test <-round(predict(mg_full, newdata = v_test, ty="response"),0)
err_test <- mean(abs(as.integer(v_test$Price) - round(pred_test,0)))


# Interpretación y extracción de conocimiento

# Distribución de error en conjunto de test
plot(v_test$Price, pred_test, xlab = 'Precios del conjunto de test',
ylab = 'Predicción del precio', main =
'Comportamiento del error de predicción del precio en el conjunto de test',
  ylim = c(0,140000))
abline(a=0, b=1, col=2, lw=3)
```




```
aux2 <- v_test[v_test$Price > 50000,]
pred_test_st <-round(predict(mg_full, newdata = aux2, ty="response"),0)
err_test_st <- mean(abs(as.integer(aux2$Price) - round(pred_test_st,0)))

# Efectos de variables explicativas
exp(mg_full$coefficients)
```

```
##          (Intercept)                 Year        UsedOrNewUSED
##         8.220319e-51         1.063782e+00         9.893776e-01
##   TransmissionManual               Engine       DriveTypeFront
##         9.788958e-01         1.151982e+00         7.460424e-01
##       DriveTypeOther        DriveTypeRear        FuelTypeOther
##         1.005745e+00         9.021227e-01         1.013692e+00
##     FuelTypeUnleaded      FuelConsumption           Kilometres
##         8.101853e-01         9.693422e-01         9.999976e-01
##            Cylinders        BodyTypeOther        BodyTypeSedan
##         1.115687e+00         1.265864e+00         1.143764e+00
## BodyTypeSUV/Ute/Tray                Doors                Seats
##         1.106485e+00         9.892131e-01         1.024289e+00
##     DollarAustralian           PriceIndex
##         1.243562e+00         1.002683e+00
```

```
# Ejemplos de coches más baratos y más caros posibles.
vb <-data.frame(Year = 2000, UsedOrNew = 'USED', Transmission = 'Manual',
                Engine = 1, FuelType = 'Unleaded',
```

```
                DriveType = 'Front', FuelConsumption = 15,
                Kilometres = 350000, Cylinders = 3,
                BodyType = 'SUV/Ute/Tray', Doors = 5, Seats = 2,
                DollarAustralian = 1.241, PriceIndex = 65.79)

vc <-data.frame(Year = 2023, UsedOrNew = 'NOT USED',
                Transmission = 'Automatic', Engine = 6,
                FuelType = 'Other',
                DriveType = 'Other', FuelConsumption = 0, Kilometres = 1,
                Cylinders = 8, BodyType = 'Other', Doors = 2, Seats = 8,
                DollarAustralian = 1.773, PriceIndex = 135.76)

# Precio medio de coche más caro e interval de confianza.
predict(mg_full, newdata = vc,ty="response")
```

```
##        1
## 314464.3
```

```
vc_l <- predict(mg_full, vc, ty="link")
exp(vc_l + ((predict(mg_full,newdata = vc, ty = "link",se.fit=T)$se.fit)*
              qnorm(0.975)))
```

```
##        1
## 314802.6
```

```
exp(vc_l + ((predict(mg_full,newdata = vc, ty = "link",se.fit=T)$se.fit)*
              qnorm(0.025)))
```

```
##        1
## 314126.3
```

```
# Precio medio de coche más barato e interval de confianza.
predict(mg_full, newdata = vb,ty="response")
```

```
##        1
## 1835.126
```

```
vb_l <- predict(mg_full, vb, ty="link")
exp(vb_l  + ((predict(mg_full,newdata = vb, ty = "link",se.fit=T)$se.fit)*
               qnorm(0.975)))
```

```
##        1
## 1836.825
```

```
exp(vb_l  + ((predict(mg_full,newdata = vb, ty = "link",se.fit=T)$se.fit)*
               qnorm(0.025)))
```

```
##        1
## 1833.429
```