Tarea 9

COA-501 Herramientas de cómputo para investigadores

Iván F. Quiroz Ibáñez

03 de noviembre de 2022

Instrucciones: Con el conjunto de datos de Boston, ejecute un modelo lineal (lm)con un conjunto de datos de entrenamiento (train) con 70 % de las observaciones y 30 % para el conjunto de datos de prueba (test). Defina cinco funciones para calcular los cinco criterios de desempeño del modelo con base en los valores observado y predichos. Grafique los valores predichos de los tres modelos versus los valores observados, en los conjuntos de prueba. Comente sus resultados.

Entrada de datos

```
#lectura de base de detos
library(MASS)
Boston <- Boston
Boston <- Boston[,-1]
#instalar gt
#devtools::install_github("rstudio/gt")
library(gt)
head(Boston,10)%>%gt()
```

zn	indus	chas	nox	rm	age	dis	rad	tax	ptratio	black	lstat	medv
18.0	2.31	0	0.538	6.575	65.2	4.0900	1	296	15.3	396.90	4.98	24.0
0.0	7.07	0	0.469	6.421	78.9	4.9671	2	242	17.8	396.90	9.14	21.6
0.0	7.07	0	0.469	7.185	61.1	4.9671	2	242	17.8	392.83	4.03	34.7
0.0	2.18	0	0.458	6.998	45.8	6.0622	3	222	18.7	394.63	2.94	33.4
0.0	2.18	0	0.458	7.147	54.2	6.0622	3	222	18.7	396.90	5.33	36.2
0.0	2.18	0	0.458	6.430	58.7	6.0622	3	222	18.7	394.12	5.21	28.7
12.5	7.87	0	0.524	6.012	66.6	5.5605	5	311	15.2	395.60	12.43	22.9
12.5	7.87	0	0.524	6.172	96.1	5.9505	5	311	15.2	396.90	19.15	27.1
12.5	7.87	0	0.524	5.631	100.0	6.0821	5	311	15.2	386.63	29.93	16.5
12.5	7.87	0	0.524	6.004	85.9	6.5921	5	311	15.2	386.71	17.10	18.9

Regresion lineal múltiple

```
# Split the data into training and testing set
set.seed(123)
# Normalize the data
maxs <- apply(Boston, 2, max)
```

```r
mins <- apply(Boston, 2, min)
scaled <- as.data.frame(scale(Boston, center = mins,
                              scale = maxs - mins))
index<- sample(1:nrow(Boston), round(0.70 * nrow(Boston)))
train_ <- Boston[index,]
test_ <- Boston[-index,]

#otra forma
#split <- sample.split(data, SplitRatio = 0.8)#muestra del 80%
#train <- subset(data, split == "TRUE") #80%
#test <- subset(data, split == "FALSE") #20%

ml <- lm(medv~.,data = train_)
summary(ml)
```

```
##
## Call:
## lm(formula = medv ~ ., data = train_)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -11.646  -2.655  -0.671   1.821  25.320
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.753e+01  6.150e+00   6.102 2.84e-09 ***
## zn           4.871e-02  1.683e-02   2.894  0.00404 **
## indus       -4.658e-02  7.972e-02  -0.584  0.55943
## chas         4.176e+00  1.036e+00   4.029 6.92e-05 ***
## nox         -1.367e+01  4.722e+00  -2.894  0.00404 **
## rm           3.204e+00  5.054e-01   6.339 7.35e-10 ***
## age         -8.553e-04  1.638e-02  -0.052  0.95840
## dis         -1.471e+00  2.424e-01  -6.068 3.45e-09 ***
## rad          2.370e-01  7.876e-02   3.009  0.00281 **
## tax         -1.027e-02  4.715e-03  -2.177  0.03013 *
## ptratio     -8.464e-01  1.618e-01  -5.231 2.96e-07 ***
## black        7.938e-03  3.468e-03   2.289  0.02267 *
## lstat       -6.123e-01  5.915e-02 -10.352  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.846 on 341 degrees of freedom
## Multiple R-squared:  0.7256, Adjusted R-squared:  0.7159
## F-statistic: 75.12 on 12 and 341 DF,  p-value: < 2.2e-16
```

```r
#R^2
y_pred <- predict(ml, test_)
SSE = sum((y_pred-test_$medv)^2)
SST = sum((y_pred-mean(test_$medv))^2)
r2_test = 1 - SSE/SST
r2_test
```

```
## [1] 0.6268143
```

```r
#CME y RCME
MSE.lm <- sum((test_$medv-y_pred)^2)/nrow(test_)
MSE.lm
```

```
## [1] 23.2211
```

```r
RMSE.ml <- sqrt(MSE.lm)
RMSE.ml
```

```
## [1] 4.818828
```

```r
#funciones mse,rms,rae,mae,mape
cme<- function(yobs, ypre)
{
  n <- length(yobs)
  suma.error <- sum ((yobs - ypre) ^ 2 ) / n
  return(suma.error)
}
cme.lm.test <- cme(test_$medv, y_pred)
cme.lm.test
```

```
## [1] 23.2211
```

```r
rms <- function(yobs, ypre)
{
  n <- length(yobs)
  error_ <- yobs - ypre
  suma.error <- sum (error_ ^ 2 )
  return(sqrt(suma.error / n))
}
rms.lm.test <- rms(test_$medv, y_pred)
rms.lm.test
```

```
## [1] 4.818828
```

```r
rae<- function(yobs, ypre)
{
  error_ <- yobs - ypre
  suma.error <- sum (error_ ^ 2 )
  suma.yi <- sum(yobs ^ 2)
  return(sqrt(suma.error / suma.yi ))
}
rae.lm.test <- rae(test_$medv, y_pred)
rae.lm.test
```

```
## [1] 0.1933804
```

```r
mae<- function(yobs, ypre)
{
  n <- length(yobs)
```

```
  error_ <- yobs - ypre
  suma.error <- sum (abs(error_))
  return(suma.error / n )
}
mae.lm.test <- mae(test_$medv, y_pred)
mae.lm.test
```

```
## [1] 3.520509
```

```
mape<- function(yobs, ypre)
{
  n <- length(yobs)
  error_ <- (yobs - ypre) / yobs
  suma.error <- sum (abs(error_))
  return((suma.error / n ) * 100 )
}
mape.lm.test <- mape(test_$medv, y_pred)
mape.lm.test
```

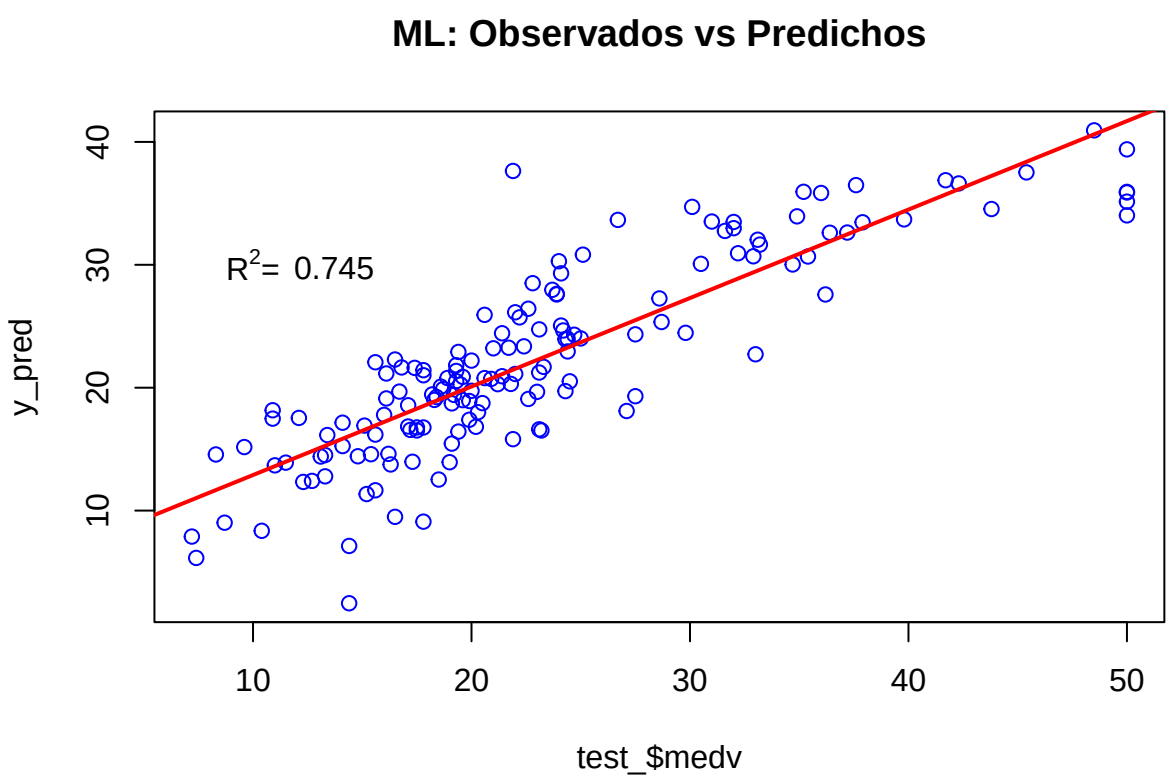
```
## [1] 16.12038
```

```
#Métricas
m <-data.frame(rbind(r2_test,cme.lm.test,rms.lm.test,
    rae.lm.test,mae.lm.test,mape.lm.test))
colnames(m) <- "Valor"

m
```

```
##                   Valor
## r2_test       0.6268143
## cme.lm.test  23.2211017
## rms.lm.test   4.8188278
## rae.lm.test   0.1933804
## mae.lm.test   3.5205086
## mape.lm.test 16.1203761
```

```
# Plot observados vs predichos
plot(test_$medv, y_pred, col = "blue",
     main = 'ML: Observados vs Predichos')
abline(lm(y_pred~test_$medv), lwd = 2, col="red")
text(10, 30, label=expression("R"^2* "="))
text(11,29.5, paste(round(cor(test_$medv,y_pred)^2, 3)), pos=4)
```



Interpretación de gráficas y resultados

De acuerdo con los resultados, se observa que el modelo lineal múltiple es bueno para predecir a la variable medv (mediana del precio de vivienda); aunque aún falta por predecir 25% de la variabilidad; sería interesante evaluar otros modelos no paramétricos como random forest o redes neuranales, además de un ACP para reducir la dimensionalidad de la base de datos. Las variables con menor capacidad de predicción son age e indus, todas las restantes, tienen un comportamiento aceptable.