Reporte de resultados

Gradient Boosting VS Logit

Erick Fajardo

```
# Libraries
library(tidyverse)
library(caret)
library(xgboost)
```

Descripción de los datos

```
# Base de datos
egresos <- read_csv(here::here("./data/3_final/egresos19_mun_clean.csv")) %>%
  mutate(corrup = factor(ifelse(prop_corrup > 0, "corrupto", "no_corrupto"))) %>%
  select(starts_with("partida"), corrup)

# Train y test sets
training_samples <- egresos$corrup %>%
  createDataPartition(p = 0.7, list = FALSE)
train_set  <- egresos[training_samples, ]
test_set <- egresos[-training_samples, ]
```

En este primer intento se compara el desempeño del algoritmo Gradient Boosting contra una Regresión Logística.

Ambos algoritmos son utilizados para clasificar a los municipios de acuerdo con su estructura de egresos. En total se cuenta con información de 233 municipios y 234 variables de egresos, las cuales están al nivel de partidas.

Los municipios fueron clasificados en corruptos y no corruptos con base en la proporción de la población que contestó que la corrupción es un fenómeno muy presente en su entidad de residencia (ECIG, 2019).Se asignó el valor de 1 (corrupto) si la proporción es mayor a 1 y 0 (no corrupto) en caso contrario:

$$Corrup = \begin{cases} 1 & \text{si proporción que contestó muy frecuente} > 0 \\ 0 & \text{si proporción que contestó muy frecuente} = 0 \end{cases}$$

partida_generica_dependencias_diversas	corrup
610217545	corrupto
128169228	corrupto
499712370	corrupto
1059671687	corrupto
673669503	corrupto
315714567	corrupto

Esta clasificación se hizo con la finalidad de evitar un problema de sesgo por prevalencia, el cual se presenta cuando la prevalencia de la viariable de resultados (en este caso corrupción) es muy cercana ya sea a 0 o 1. Con esta regla de clasificación se obtiene una prevalencia de:

```
# Prevalencia de municipios corruptos
round(mean(egresos$corrup == "corrupto"), 4)
```

```
## [1] 0.5365
```

Gradient Boosting

```
# Train Gradient boosting
model_xgbTree <- train(
  corrup ~ ., data = train_set, method = "xgbTree",
  trControl = trainControl("cv", number = 5)
)

# Predictions
y_hat_xgbTree <- model_xgbTree %>% predict(test_set)
```

Evaluación

Este algoritmo presenta una precisión de 0.72, la cual es buena, además de contar con valores balanceados de sensitividad (habilidad del algoritmos para predecir valores positivos que sí son positivos) y especificidad (habilidad del algoritmo para predecir valores negativos que sí son negativos) que, a su vez, son buenos. El intervalo de confianza cuenta con un bajo valor-p, por lo que es posible confiar en los resultados del algoritmo.

```
## Confusion Matrix and Statistics
##
##              Reference
## Prediction    corrupto no_corrupto
```

```
##   corrupto          23           5
##   no_corrupto       14          27
##
##                Accuracy : 0.7246
##                  95% CI : (0.6038, 0.8254)
##     No Information Rate : 0.5362
##     P-Value [Acc > NIR] : 0.001053
##
##                   Kappa : 0.4567
##
##  Mcnemar's Test P-Value : 0.066457
##
##             Sensitivity : 0.6216
##             Specificity : 0.8438
##          Pos Pred Value : 0.8214
##          Neg Pred Value : 0.6585
##              Prevalence : 0.5362
##          Detection Rate : 0.3333
##    Detection Prevalence : 0.4058
##       Balanced Accuracy : 0.7327
##
##        'Positive' Class : corrupto
##
```

Regresión Logística

```
# Train
model_Logit <- glm(corrup ~ ., data = train_set, family = "binomial")

# Predictions
y_hat_logit <- model_Logit %>% predict(test_set, type = "response") # type = "response"

# To factor
y_hat_logit <- ifelse(y_hat_logit > 0.5, "corrupto", "no_corrupto") %>%
  factor()

# Binary predictions
y_hat_logit_bin <- ifelse(y_hat_logit == "corrupto", 1, 0)
```

Evaluación

La regresión logística cuenta con una precisión de 0.49, la cual no es buena, y a pesar de que sus valores de sensitividad y especificidad son balanceados, no destaca en ninguno de los dos. Es importante notar que el intervalo de confianza cuenta con un valor-p muy alto, por lo que en términos de inferencia no es posible confiar en este modelo.

```
## Confusion Matrix and Statistics
##
##              Reference
## Prediction    corrupto no_corrupto
##   corrupto           19          17
##   no_corrupto        18          15
##
##                Accuracy : 0.4928
##                  95% CI : (0.3702, 0.6159)
##     No Information Rate : 0.5362
##     P-Value [Acc > NIR] : 0.8011
##
##                   Kappa : -0.0177
##
##  Mcnemar's Test P-Value : 1.0000
##
##             Sensitivity : 0.5135
##             Specificity : 0.4688
##          Pos Pred Value : 0.5278
##          Neg Pred Value : 0.4545
##              Prevalence : 0.5362
##          Detection Rate : 0.2754
##    Detection Prevalence : 0.5217
##       Balanced Accuracy : 0.4911
##
##        'Positive' Class : corrupto
##
```

Comparación

En la siguiente tabla se observa que las métricas de evaluación son mejores en el algoritmo Gradient Boosting, dado que obtiene una mayor precisión y un mayor puntaje F1, mientras que cuenta con un RMSE menor.

Algoritmo	Accuracy	RMSE	F1_score
Gradient Boosting	0.7246	0.5247	0.7077
Logit	0.4928	0.7122	0.5205

Variables de importancia

Gradient Boosting

Las 10 partidas de egresos que más influencia tienen en la corrupción de acuerdo a Gradient Boosting:

	Overall
partida_generica_servicios_profesionales_cientificos_y_tecnicos_integrales	100.00000
partida_generica_otros_arrendamientos	59.69300
partida_generica_servicios_postales_y_telegraficos	42.69803
partida_generica_diversas_transferencias	37.19475
partida_generica_otros_vestuario_blancos_prendas_de_proteccion_y_articulos_deportivos	30.04784
partida_generica_otros_equipos	28.11252
partida_generica_division_de_terrenos_y_construccion_de_obras_de_urbanizacion	25.98777
partida_generica_refacciones_y_accesorios_menores_de_edificios	24.04565
partida_generica_dependencias_diversas	23.80556
partida_generica_equipo_de_defensa_y_seguridad	19.31917

Logit

Las 10 partidas de egresos que más influencia tienen en la corrupción de acuerdo a la Regresión Logística:

	Overall
partida_generica_servicios_funerarios_y_de_cementerios	0.0001013
partida_generica_vehiculos_y_equipo_terrestre	0.0000879
partida_generica_refacciones_y_accesorios_menores_de_equipo_de_computo_y_tecnologias_de_la_informacion	0.0000874
partida_generica_becas_y_otras_ayudas_para_programas_de_capacitacion	0.0000866
partida_generica_camaras_fotograficas_y_de_video	0.0000816
partida_generica_materiales_utiles_y_equipos_menores_de_tecnologias_de_la_informacion_y_comunicaciones	0.0000807
partida_generica_aportaciones_para_seguros	0.0000794
partida_generica_energia_electrica	0.0000758
partida_generica_otros_servicios_generales	0.0000750
partida_generica_subsidios_a_la_inversion	0.0000707

Notas de interpretación

Overall Accuracy

Es la media de las predicciones correctas que obtiene el algoritmo. Se calcula como el número de aciertos entre el número de observaciones:

$$OverallAccuracy = \frac{(\hat{Y} = 1,\ \text{cuando } Y = 1) + (\hat{Y} = 0\ ,\ \text{cuando } Y = 0)}{N}$$

Raíz del Error Cuadrado Medio (RMSE)

Es la desviación media de los valores predichos. Entre más cercano a 0 mejor.

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\hat{Y} - Y)^2}$$

F1-score

Es la media armónica de las métricas precision (proporción de valores predichos positivos que son reales positivos) y recall (proporción de valores reales positivos que son predecidos como positivos). Entre mayor sea su valor mejor.

$$F_1\text{-}score = 2 \times \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$