Actividad - Metodo Supervisados

Javier Eduardo Jaimes Velasquez[[1]](#footnote-20)

Brandon Valencia Murillo[[2]](#footnote-22)

2024-07-11

Table of Contents

# 1 Actividad Metodos Supervisados

A continuacion se detalla a manera de informer las actividades solicitadas como parte del modulo de metodos supervisados. En el siguiento documento se busca dar una respuesta a las siguientes preguntas:

* Manipulacion y carga del conjunto de datos *attrition*, parte de la libreria *modeldata*.
* Definir un modelo de Random Forest. (RF) a partir de la columna \*attrition.
* Definir un modelo GLM, a partir del modelo y conjuntos de datos anterior mencionado.
* Definir un modelo Supervised Vector Machine (SVM) a partir del modulo y conjuntos de datos anterior mencionado.
* A partir de lo anterior: Cual modelo seria el mejor?
* Obtener las matriz de confusion de cada modelo.

## 1.1 Requerimientos iniciales

Se cargan las librerias necesarias: *magrittr, factoextra, readr, tidyverse, caret, class, modeldata, glmnet, ranger y naivebayes*.

library(caret)

## Loading required package: ggplot2

## Loading required package: lattice

library(class)  
library(modeldata)  
library(glmnet)

## Loading required package: Matrix

## Loaded glmnet 4.1-8

library(ranger)  
library(tidyverse)

## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──  
## ✔ dplyr 1.1.4 ✔ readr 2.1.5  
## ✔ forcats 1.0.0 ✔ stringr 1.5.1  
## ✔ lubridate 1.9.3 ✔ tibble 3.2.1  
## ✔ purrr 1.0.2 ✔ tidyr 1.3.1

## ── Conflicts ────────────────────────────────────────── tidyverse\_conflicts() ──  
## ✖ tidyr::expand() masks Matrix::expand()  
## ✖ dplyr::filter() masks stats::filter()  
## ✖ dplyr::lag() masks stats::lag()  
## ✖ purrr::lift() masks caret::lift()  
## ✖ tidyr::pack() masks Matrix::pack()  
## ✖ tidyr::unpack() masks Matrix::unpack()  
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors

## 1.2 Conjunto de Datos

A continuacion se describe el conjunto de datos, **attrition**:

data("attrition")

Al resumir los datos, validaremos que la columna *Attrition* sea de tipo factor:

str(attrition)

## 'data.frame': 1470 obs. of 31 variables:  
## $ Age : int 41 49 37 33 27 32 59 30 38 36 ...  
## $ Attrition : Factor w/ 2 levels "No","Yes": 2 1 2 1 1 1 1 1 1 1 ...  
## $ BusinessTravel : Factor w/ 3 levels "Non-Travel","Travel\_Frequently",..: 3 2 3 2 3 2 3 3 2 3 ...  
## $ DailyRate : int 1102 279 1373 1392 591 1005 1324 1358 216 1299 ...  
## $ Department : Factor w/ 3 levels "Human\_Resources",..: 3 2 2 2 2 2 2 2 2 2 ...  
## $ DistanceFromHome : int 1 8 2 3 2 2 3 24 23 27 ...  
## $ Education : Ord.factor w/ 5 levels "Below\_College"<..: 2 1 2 4 1 2 3 1 3 3 ...  
## $ EducationField : Factor w/ 6 levels "Human\_Resources",..: 2 2 5 2 4 2 4 2 2 4 ...  
## $ EnvironmentSatisfaction : Ord.factor w/ 4 levels "Low"<"Medium"<..: 2 3 4 4 1 4 3 4 4 3 ...  
## $ Gender : Factor w/ 2 levels "Female","Male": 1 2 2 1 2 2 1 2 2 2 ...  
## $ HourlyRate : int 94 61 92 56 40 79 81 67 44 94 ...  
## $ JobInvolvement : Ord.factor w/ 4 levels "Low"<"Medium"<..: 3 2 2 3 3 3 4 3 2 3 ...  
## $ JobLevel : int 2 2 1 1 1 1 1 1 3 2 ...  
## $ JobRole : Factor w/ 9 levels "Healthcare\_Representative",..: 8 7 3 7 3 3 3 3 5 1 ...  
## $ JobSatisfaction : Ord.factor w/ 4 levels "Low"<"Medium"<..: 4 2 3 3 2 4 1 3 3 3 ...  
## $ MaritalStatus : Factor w/ 3 levels "Divorced","Married",..: 3 2 3 2 2 3 2 1 3 2 ...  
## $ MonthlyIncome : int 5993 5130 2090 2909 3468 3068 2670 2693 9526 5237 ...  
## $ MonthlyRate : int 19479 24907 2396 23159 16632 11864 9964 13335 8787 16577 ...  
## $ NumCompaniesWorked : int 8 1 6 1 9 0 4 1 0 6 ...  
## $ OverTime : Factor w/ 2 levels "No","Yes": 2 1 2 2 1 1 2 1 1 1 ...  
## $ PercentSalaryHike : int 11 23 15 11 12 13 20 22 21 13 ...  
## $ PerformanceRating : Ord.factor w/ 4 levels "Low"<"Good"<"Excellent"<..: 3 4 3 3 3 3 4 4 4 3 ...  
## $ RelationshipSatisfaction: Ord.factor w/ 4 levels "Low"<"Medium"<..: 1 4 2 3 4 3 1 2 2 2 ...  
## $ StockOptionLevel : int 0 1 0 0 1 0 3 1 0 2 ...  
## $ TotalWorkingYears : int 8 10 7 8 6 8 12 1 10 17 ...  
## $ TrainingTimesLastYear : int 0 3 3 3 3 2 3 2 2 3 ...  
## $ WorkLifeBalance : Ord.factor w/ 4 levels "Bad"<"Good"<"Better"<..: 1 3 3 3 3 2 2 3 3 2 ...  
## $ YearsAtCompany : int 6 10 0 8 2 7 1 1 9 7 ...  
## $ YearsInCurrentRole : int 4 7 0 7 2 7 0 0 7 7 ...  
## $ YearsSinceLastPromotion : int 0 1 0 3 2 3 0 0 1 7 ...  
## $ YearsWithCurrManager : int 5 7 0 0 2 6 0 0 8 7 ...

## 1.3 Definicion de Modelos

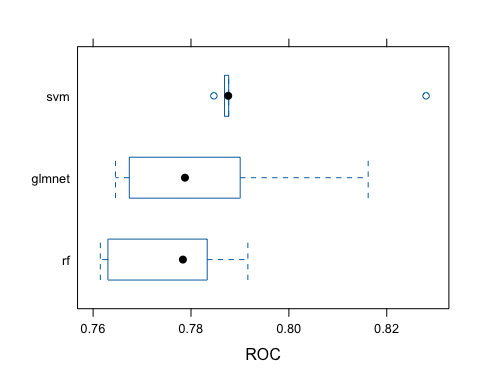
A continuacion se definen los siguientes modelos: RF, GLM y SVM.

set.seed(100)  
  
trabajadores <- attrition %>% select(-c()) %>% mutate(Attrition = as.factor(Attrition))  
  
# Se establece una particion de los datos, para definir los dos conjuntos el de prueba y el de entrenamiento  
index\_trabajadores <- createDataPartition(y = trabajadores$Attrition, p = 0.7, list = F)  
  
# Se definen los conjuntos de prueba y de entrenamiento  
trabajadores\_train <- trabajadores[index\_trabajadores, ]  
trabajadores\_test <- trabajadores[-index\_trabajadores, ]  
  
 # Se establecen los grupos que mantienen el balance de los mismos durante la definicion y prueba de los modelos.  
trabajadores\_fold <- createFolds(trabajadores\_train$Attrition, k = 5)  
  
# Se crea el control de entrenamiento, definiendo el parametro index igual a los folds previamenmte creados  
trabajadores\_ctrl <- trainControl(savePredictions = T, classProbs = T, summaryFunction = twoClassSummary, index = trabajadores\_fold)  
  
#### Modelo GML ####  
trabajadores\_glm\_model <- train(Attrition ~ ., trabajadores\_train, metric = "ROC", method = "glmnet", trControl = trabajadores\_ctrl)  
  
  
#### RF ####  
trabajadores\_rf\_model <- train(Attrition ~ ., trabajadores\_train, metric = "ROC", method = "ranger", trControl = trabajadores\_ctrl)  
  
#### SVM ####  
trabajadores\_svm\_model <- train(Attrition ~ ., trabajadores\_train, metric = "ROC", method = "svmRadial", tuneLength = 10, trControl = trabajadores\_ctrl)

## 1.4 Comparacion de Modelos

Ahora es momento de graficar los modelos para su debida comparacion:

list\_ml <- list(  
 glmnet = trabajadores\_glm\_model,  
 rf = trabajadores\_rf\_model,  
 svm = trabajadores\_svm\_model  
)  
  
  
comparacion\_modelos <- resamples(list\_ml)  
bwplot(comparacion\_modelos, metric = "ROC")



### 1.4.1 Matrix de Confuision por Modelo

A continuacion se imprimen las matrices de confusion por cada uno de los modelos

#### 1.4.1.1 Random Forest

En este modelo vemos una precision de con una desecersion de clientes del .

confusionMatrix(predict(object = trabajadores\_rf\_model, newdata = trabajadores\_test), trabajadores\_test$Attrition)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction No Yes  
## No 363 57  
## Yes 6 14  
##   
## Accuracy : 0.8568   
## 95% CI : (0.8206, 0.8882)  
## No Information Rate : 0.8386   
## P-Value [Acc > NIR] : 0.1657   
##   
## Kappa : 0.2548   
##   
## Mcnemar's Test P-Value : 2.988e-10   
##   
## Sensitivity : 0.9837   
## Specificity : 0.1972   
## Pos Pred Value : 0.8643   
## Neg Pred Value : 0.7000   
## Prevalence : 0.8386   
## Detection Rate : 0.8250   
## Detection Prevalence : 0.9545   
## Balanced Accuracy : 0.5905   
##   
## 'Positive' Class : No   
##

#### 1.4.1.2 GLM

En este modelo vemos una precision de con una desecersion de clientes del .

confusionMatrix(predict(object = trabajadores\_glm\_model, newdata = trabajadores\_test), trabajadores\_test$Attrition)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction No Yes  
## No 362 50  
## Yes 7 21  
##   
## Accuracy : 0.8705   
## 95% CI : (0.8354, 0.9004)  
## No Information Rate : 0.8386   
## P-Value [Acc > NIR] : 0.03728   
##   
## Kappa : 0.3664   
##   
## Mcnemar's Test P-Value : 2.651e-08   
##   
## Sensitivity : 0.9810   
## Specificity : 0.2958   
## Pos Pred Value : 0.8786   
## Neg Pred Value : 0.7500   
## Prevalence : 0.8386   
## Detection Rate : 0.8227   
## Detection Prevalence : 0.9364   
## Balanced Accuracy : 0.6384   
##   
## 'Positive' Class : No   
##

#### 1.4.1.3 SVM

En este modelo vemos una precision de con una desecersion de clientes del .

confusionMatrix(predict(object = trabajadores\_svm\_model, newdata = trabajadores\_test), trabajadores\_test$Attrition)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction No Yes  
## No 361 51  
## Yes 8 20  
##   
## Accuracy : 0.8659   
## 95% CI : (0.8305, 0.8963)  
## No Information Rate : 0.8386   
## P-Value [Acc > NIR] : 0.0655   
##   
## Kappa : 0.3442   
##   
## Mcnemar's Test P-Value : 4.553e-08   
##   
## Sensitivity : 0.9783   
## Specificity : 0.2817   
## Pos Pred Value : 0.8762   
## Neg Pred Value : 0.7143   
## Prevalence : 0.8386   
## Detection Rate : 0.8205   
## Detection Prevalence : 0.9364   
## Balanced Accuracy : 0.6300   
##   
## 'Positive' Class : No   
##

## 1.5 Conclusion

Los modelos se comportan de forma muy similar, y su varianza en la precision es muy poco haciendo que cualquier de los modelos encajaria para la solucion y entendimiento del problema. Tal vez se deberian integrar otras variables para determinar la eleccion, varibles que varian de acuerdo al contexto en el cual se ejecutaran estos modelos.

1. Politécnico Grancolombiano, [jajaimes4@poligran.edu.co](mailto:jajaimes4@poligran.edu.co) [↑](#footnote-ref-20)
2. Politécnico Grancolombiano, [brvalencia6@poligran.edu.co](mailto:brvalencia6@poligran.edu.co) [↑](#footnote-ref-22)