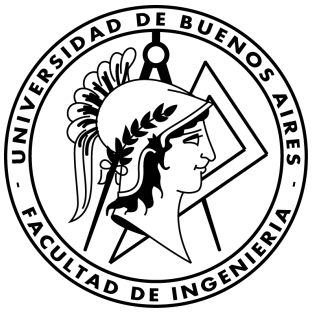

Trabajo Práctico I — Checkpoint II

[75.06/95.58] Organizacion de Datos
Primer cuatrimestre de 2023

Alumno	Número de padrón	Email
Masri, Noah	108814	noahmasri19@gmail.com
Ayala, Camila	107440	cayala@fi.uba.ar
Loscalzo, Melina	106571	mloscalzo@fi.uba.ar

En este segundo checkpoint, comenzamos el entrenamiento de nuestro modelo para poder luego realizar predicciones. El objetivo era predecir si la reserva sera cancelada, en base al resto de los datos que contamos sobre la reserva. Para esto trabajamos en entrenar un arbol de decision con nuestro dataset.

En primer lugar, separamos el conjunto de train y test y generamos un arbol con parametros completamente arbitrarios, de una maxima profundidad de 20. Entrenamos con este un modelo, realizamos una prediccion, y con esta prediccion calculamos su matriz de confusion para darnos una idea de que tan bien encaminadas estabamos. Al ser todos los datos de nuestro dataset ya trabajado, comenzamos a dudar un poco de los resultados, ademas de que una profundidad de 20 resultaba un poco excesivo.

Una vez hecho esto, buscamos optimizar los hiperparametros con la ayuda de kfolds. Buscamos distintos sets de los mejores hiperparametros, variando la cantidad de folds. Por default sklearn usa 5 folds, por lo cual buscamos adicionalmente con 4 y 6 folds. La metrica que tomamos en cuenta para buscar los mejores parametros fue el F1 score, ya que esta realiza un balance entre recall y precision, por lo que es la mas general y representativa.

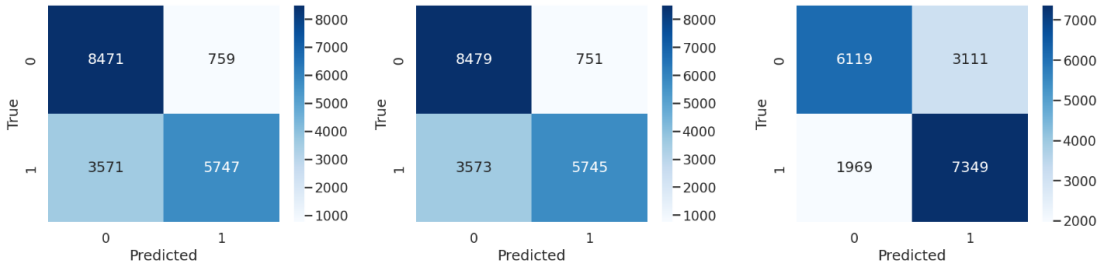

Figura 1: Matriz de confusion con 4, 5 y 6 folds respectivamente

Asi pudimos ver que las metricas obtenidas realizando predicciones con los hiperparametros obtenidos por los distintos CV con 4 y 5 folds eran casi iguales, mientras que los mejores parametros hallados con 6 folds hacian que se incrementen muchisimo los falsos positivos, y a su vez los verdaderos positivos. Optamos por quedarnos con el caso de 5 folds, ya que este arbol tenia mucho menos profundidad que el generado por el caso de 4 folds, y daban casi las mismas metricas.

Las metricas obtenidas con la prediccion sobre los conjuntos de datos de train y test son similares. Esto no nos asegura que no haya overfitting, pero si las metricas en el conjunto de entrenamiento fuesen mucho mayores si seria un claro indicador de que estamos haciendo overfitting en nuestro modelo.

Finalmente, probamos buscar hiperparametros con el metodo grid search, que sabemos es mas lento, pero deseabamos ver si los resultados lo valian. Obtuvimos con este un mejor F1 score, pero no por mucho, y si tardo mucho el algoritmo en conseguir nuestro mejor conjunto. Sin embargo, decidimos quedarnos con los hiperparametros otorgados por este ya que

Pudimos observar que en la parte mas cercana a la raiz del arbol se encontraban variables cuyo peso ya intuiamos, como las variables lead time, deposit type non refund, previous cancelations, pero tambien aparecieron otras como required parking spaces y agent que nos dejaron dudando un poco de si se trataba de causalidad o casualidad, las cuales continuaremos observando en las siguientes etapas.