Checkpoint 2 - Grupo 33

Introduccion

Para empezar hicimos los mismos cambios que realizamos en el checkpoint anterior: Cambiamos los NaNs por las modas, excepto por los nans de *Country* que decidimos reemplazarlo por una variable nueva “Otro” en algunas instancias y “PRT” en otras**.** Luego, para refinar cambiamos los valores Undefined de *Market Segment* y *Distribution Channel* por la moda. Valores extraños como *ADR* negativo o en cero los reemplazamos por su promedio, lo mismo con *Requiered Car Parking Spaces*. O los casos donde los adultos eran 55 o cero vimos de cambiarlos por la moda, los mismo hicimos con *Children* y *Babies*.

También experimentamos con eliminar duplicados en ciertos casos. Por ejemplo cuando Previous Cancellations superaba el valor 19 ya que el desviación estándar era de 18.6. Otro caso fue en Days In Waiting List que notamos que el caso de 391 parecía ser siempre el mismo cliente que hacía una misma reserva y cancelaba, esto unas 33 veces.

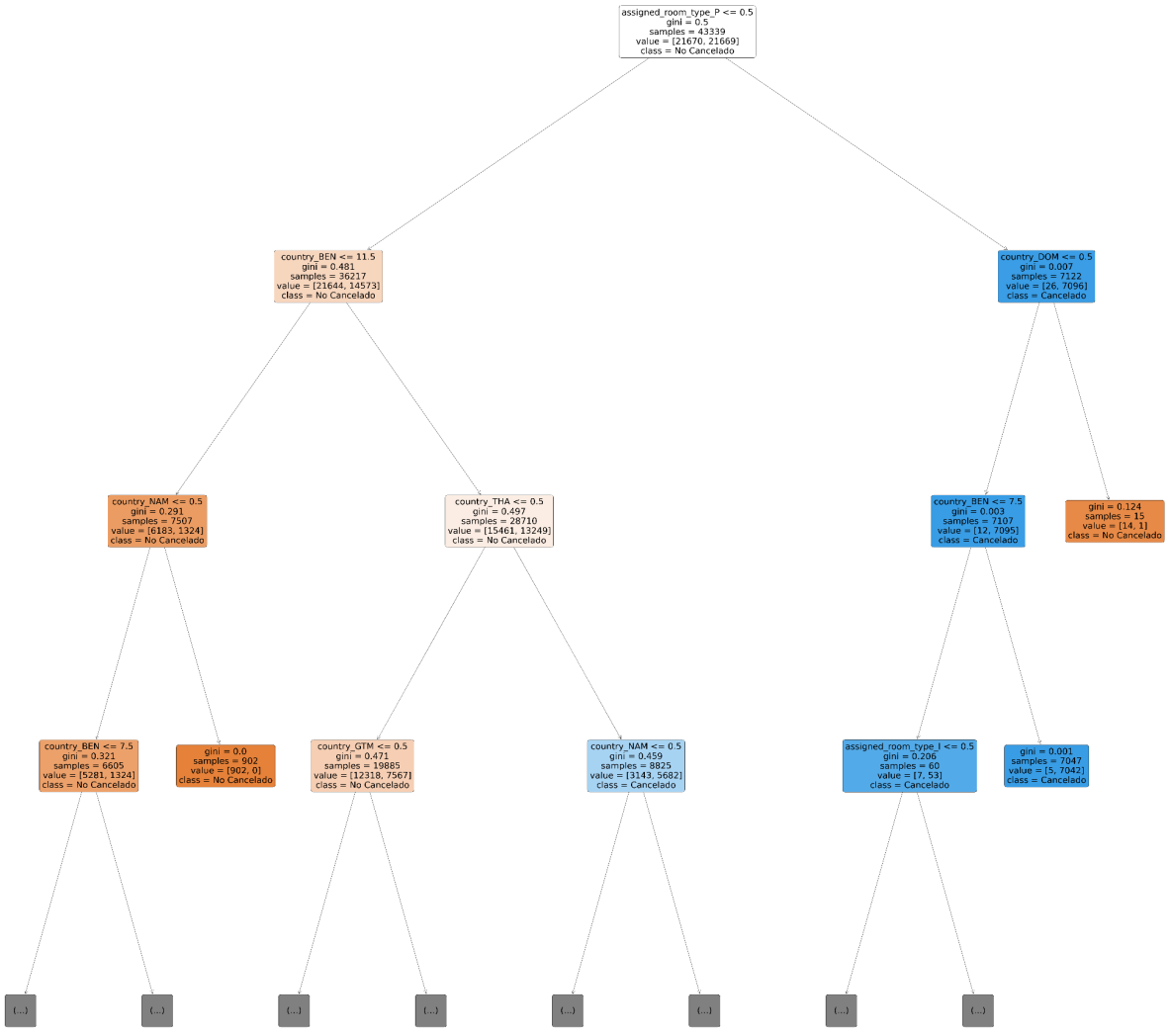
Otra técnica que también exploramos, aunque al final no fue utilizada, fue tratar de predecir los valores NaN de Agent con otro árbol. El proceso fue muy parecido al que hicimos para predecir Is Canceled, pero se hacía antes de realizar todo el proceso de filtrado.

Una observación importante es que nuestros resultados variaron dependiendo de la semilla elegida en el train_test_split().

Construcción del modelo

El modelo utilizado fue un árbol de decisión, pero antes de entrenarlo utilizamos el objeto RandomizedSearchCV para buscar los mejores hiperparametros que se adecuen a nuestro dataset con sus respectivas modificaciones, los hiperparametros probados fueron el “criterion”, “max_depth”, “min_samples_leaf”, “min_samples_split” y el “ccp_alpha”. Primero empezamos poniendo rangos amplios de valores hasta quedarnos con los que siempre elegía el algoritmo, hasta darnos cuenta que tanto min_samples_leaf como min_samples_split siempre elegía los valores por defecto asi

como con la poda- Los hiperparametros que mas funcionaron fue max_depth en 15 con un criterio de entropy, aunque en algunas ejecuciones la profundidad era de 14 y con criterio de gini. Para optimizar los hiperparametros utilizamos 5 folds ya que al probar con números más grandes no había grandes diferencias en la mejora de las métricas.
Nuestra métrica a mejorar que siempre fue f1_score siempre se mantuvo en 0,84 y 0,85 nunca pudiendo alcanzar el umbral de los 0,86.
El árbol de nuestra mejor predicción fue:



Cuadro de Resultados

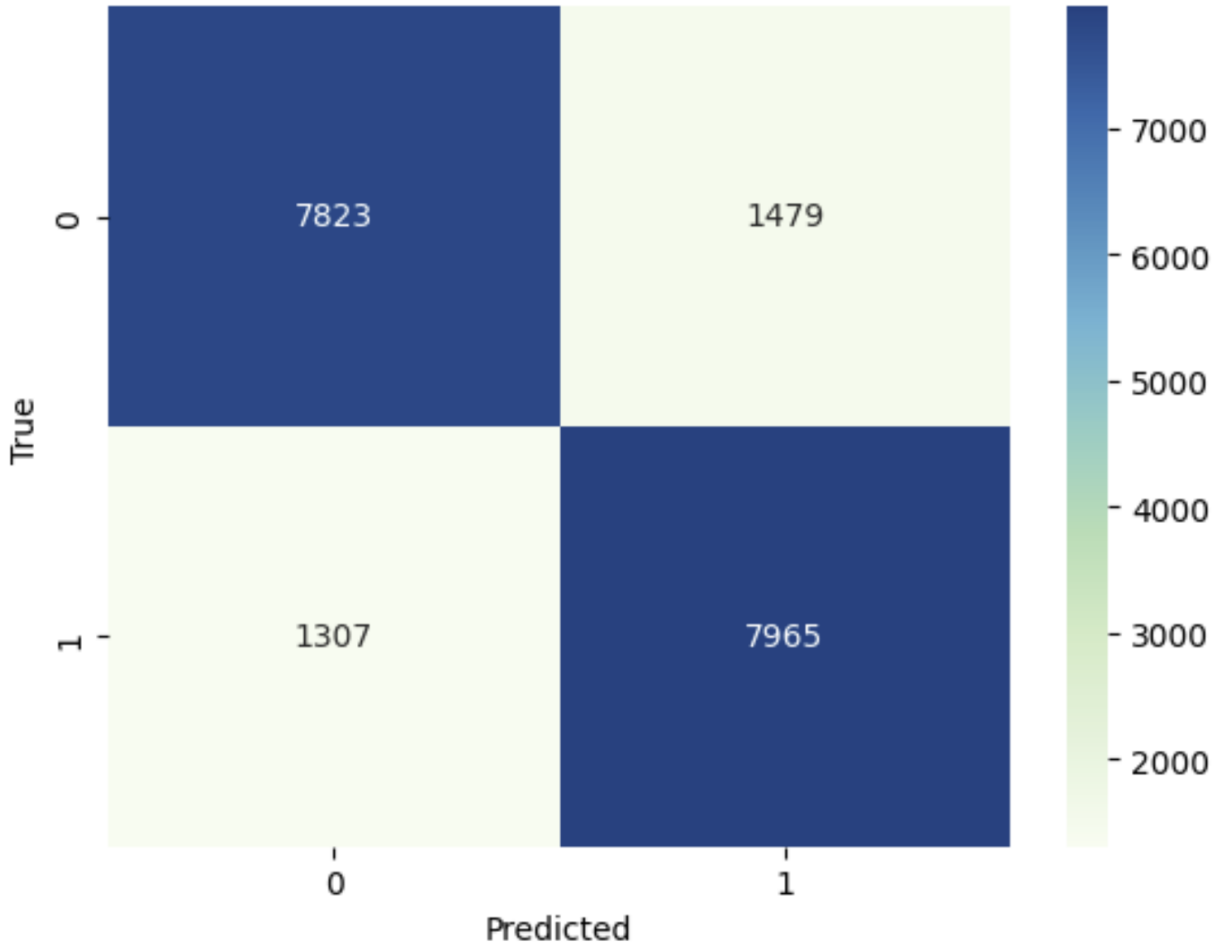
Modelo	F1-Test	Presicion Test	Recall Test	Metrica X/Accuracy	Kaggle
modelo_1	0.8511434067108358	0.8433926302414231	0.8590379637618637	0.8500053838699257	0.85087
modelo_2	0.8514243354866038	0.8355851174392018	0.8678756476683938	0.8487086860408691	0.84948
modelo_3	0.8575052854122622	0.8419304618578101	0.8736672051696285	0.8548508668803058	0.84948

Matriz de Confusion

Los cuatro valores de la matriz de confusión podemos observar los siguientes escenarios:

- Verdaderos positivos (TP): datos reales de la clase “cancelada” que fueron correctamente clasificadas por el modelo.
- Falsos positivos (FP): datos reales de la clase “no cancelada” que fueron incorrectamente clasificadas como positivas por el modelo.
- Verdaderos negativos (TN): datos reales de la clase “no cancelada” que fueron correctamente clasificadas como negativas por el modelo.
- Falsos negativos (FN): datos reales de la clase “cancelada” que fueron incorrectamente clasificadas como negativas por el modelo.

A partir de estos datos se puede conseguir algunas métricas de rendimiento.



Tareas Realizadas

Integrante	Tarea
Dolores Levi	Análisis de nuevos outliers. Predicción con árboles.
Lucas Nahuel Raimondi	Análisis de nuevos outliers. Predicción con árboles.
Manule Jesus Davila Sanchez	Análisis de nuevos outliers. Predicción con árboles.