Checkpoint 2 - Grupo 12

Introducción

Se utilizaron las técnicas de random forest y k-folds cross validation

Partimos del dataset depurado en el anterior checkpoint, y efectuamos las siguientes modificaciones que consideramos necesarias antes de armar los modelos:

- La variable *"country"* fue reemplazada por los respectivos continentes de cada país en la columna llamada *"nombre_continente" para aplicarle one-hot encoding*
- *"arrival_date"* fue convertida desde una fecha a un número entero representando los milisegundos desde el 01/01/1970
- A las demás variables cualitativas se les aplicó one-hot encoding

Para la búsqueda de mejores hiperparámetros para encontrar el mejor árbol de decisión, utilizamos la técnica de RandomizedSearchCV.

Breve comentario de técnicas exploradas, pruebas realizadas y si efectuaron nuevas modificaciones sobre el dataset. Cualquier implementación realizada por el equipo se debe detallar en esta sección.

Construcción del modelo

- ¿Optimizaron hiperparámetros? ¿Cuáles?
  - Buscamos optimizar los siguientes hiperparámetros:
    -*'min_samples_split'*: Establece el número mínimo de muestras requeridas para dividir un nodo en dos nodos hijas en un árbol de decisión.
    -*'min_samples_leaf'*: Establece el número mínimo de muestras requeridas para que un nodo sea una hoja en el árbol de decisión.
    -*'max_features'*: Este parámetro determina la cantidad máxima de características (variables) que se considerarán al buscar la mejor división en cada nodo del árbol.
    -*'max_depth'*: Este parámetro limita la profundidad máxima del árbol de decisión.
    -*"criterion'*: Este parámetro define la función de calidad que se utilizará para medir la calidad de una división en un árbol de decisión. Los dos criterios más comunes son "gini" y "entropy".
    -*"ccp_alpha'*: Este parámetro controla la complejidad del árbol de decisión mediante el costo-complejidad de poda.

Donde aplicamos RandomSearchCV, y obtuvimos los siguientes hiperparámetros:

- 'min_samples_split': 18,
- 'min_samples_leaf': 5,
- 'max_features': 'sqrt',
- 'max_depth': 110,
- ''criterion': 'entropy',
- ''ccp_alpha': 0.0

- ¿Utilizaron K-fold Cross Validation?¿Cuántos folds utilizaron?

  Para el RandomizedSearchCV se utilizaron 35 folds.

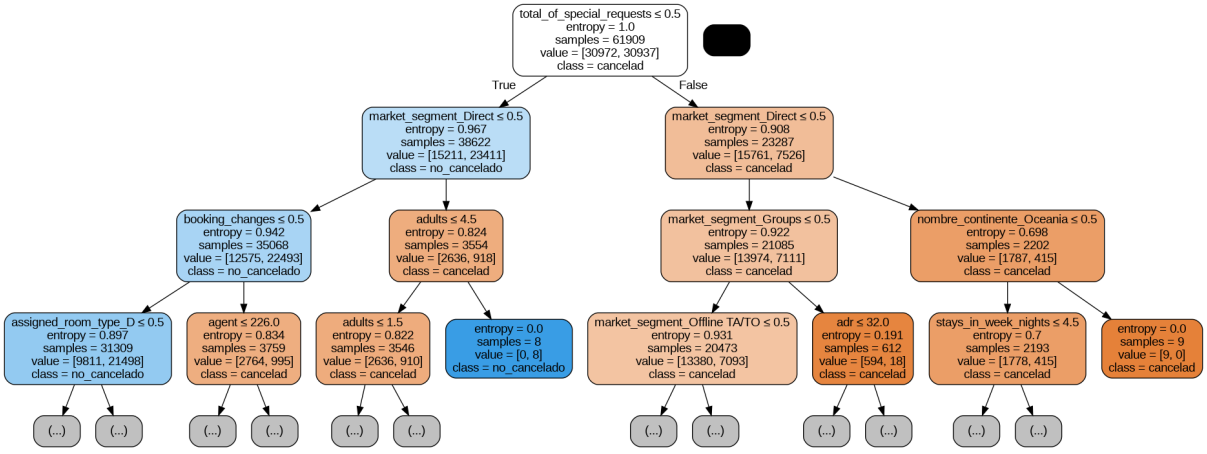
- ¿Qué métrica consideran adecuada para buscar los hiperparámetros?

Se consideró la mejora en la métrica F1-score para la búsqueda de parámetros.

- ¿Cuánto mejoró la métrica desde el modelo inicial al final?

  - En nuestro primer modelo utilizamos un árbol de decisión básico, obtuvimos un F1-score de 0.751 en kaggle.
  - Para un segundo modelo, buscamos optimizar los hiperparámetros con RandomizedSearchCV y pocas iteraciones, y nos dió 0.756.
  - Para el tercer modelo, se probaron varias iteraciones para el RandomizedSearchCV, de manera tal que se ajustaron los hiperparámetros para darnos un mejor f1-score, que nos dió hacia dónde obtenemos los mejores resultados, logramos obtener F1-score de 0.776.

- Árbol obtenido:

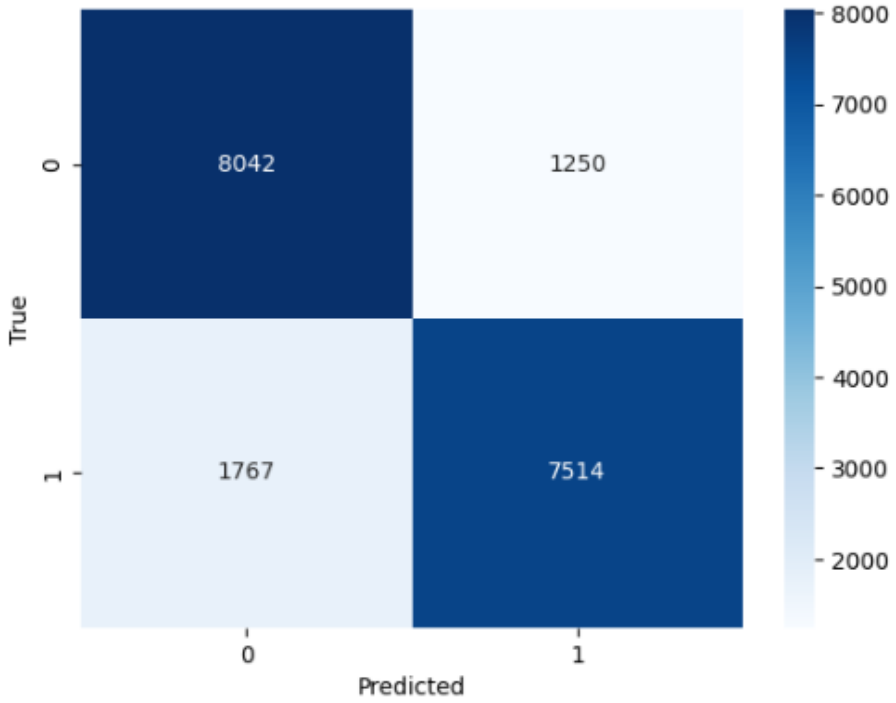


Cuadro de Resultados

Modelo	F1-Test	Presicion Test	Recall Test	Accuracy	Macro Average	Weighted Average	Kaggle
modelo_1	0.80	0.81 / 0.80	0.80 / 0.81	0.80	0.80	0.80	0.752

modelo_2	0.80	0.78/ 0.80	0.81/ 0.78	0.79	0.79	0.79	0.756
modelo_3	0.84/ 0.83	0.82/ 0.86	0.87/ 0.81	0.84	0.84	0.84	0.77

Matriz de Confusión



Conclusión: Se puede observar que hay más verdaderos negativos y positivos que falsos negativos y positivos, y por lo tanto se puede concluir que el mejor modelo tiene un buen nivel de predicción.

Tareas Realizadas

Indicar brevemente en qué tarea trabajo cada integrante del equipo, si trabajaron en las mismas tareas lo detallan en cada caso (como en el ejemplo el armado de reporte).

Integrante	Tarea
Gaston Sabaj	Armado y evaluación de modelos, transformaciones sobre el dataset de test y train,
Juan Yago Pimenta	Tratamiento de variables Generación del árbol default Armado del informe