Checkpoint 3 - Grupo 25

Introducción

Para el desarrollo de este checkpoint desarrollamos 4 clasificadores y 2 ensambles con el fin de mejorar nuestra predicción, evaluamos su resultado de métricas y su costo computacional con el objetivo de encontrar el mejor. Evaluamos los siguientes clasificadores: KNN, SVM, RF, XGBoost y de los ensambles híbridos de tipo Voting y Stacking (introduciendo Regresión Logística), mejorando los hiper parámetros para maximizar su rendimiento.

Construcción del modelo

Creamos los modelos, evaluamos cómo resultan sin mejorar los hiper parámetros, luego los optimizamos y estudiamos su progreso.

KNN: 'weights', 'n_neighbors', 'metric', 'algorithm'

SVN:
- Kernel Lineal: ‘C’
- Kernel Radial: ‘C’, ‘gamma’

RF: 'n_estimators’, 'max_features’, 'min_samples_split’, 'min_samples_leaf’, 'bootstrap'

XGBoost: 'n_estimators’, 'learning_rate', 'max_depth', 'min_child_weight’, 'subsample’, 'colsample_bytree', 'reg_alpha', 'reg_lambda'

Voting: los modelos usados fueron
- XGBoost (con mejores hiperparámetros)
- Random Forest (con mejores hiperparámetros)
- KNN (con mejores hiperparámetros)

Stacking: los modelos entrenados fueron
- Random Forest con 50 árboles de decisión.
- SVM
- KNN

Meta-modelo: Regresión Logística (de probar con varios el que mejor rindió).

Cuadro de Resultados

Modelo	F1-Test	Presicion Test	Recall Test	Kaggle	Tiempo ejecución
KNN	0.78	0.75	0.82	0.76	14 min
SVM	0.80	0.82	0.77	0.34	108 min
Random Forest	0.83	0.87	0.80	0.832	48 min
XGBoost	0.85	0.86	0.84	0.84	27 min
Voting	0.85	0.86	0.83	0.84	2 min
Stacking	0.84	0.86	0.82	0.84	36 min
Regresión Log.	0.76	0.78	0.73	0.77	2 min

KNN: busca a los k-vecinos más cercanos en un punto dado para encontrar similitudes. Dada una “votación” determina a qué grupo pertenecen; los datos similares estarán más cerca en el espacio.

SVM: trata de agregar dimensiones a las observaciones para poder identificar una separación; construye un hiperplano o conjunto de hiperplanos dependiendo si los conjuntos son linealmente separables. Una vez que se encuentra el hiperplano óptimo, se utiliza para clasificar nuevos datos. La clasificación para nuevos puntos consiste en qué lado del hiperplano se ubican.

Random Forest: se realiza una partición del dataset a nivel de columnas y de observaciones, y se entrena un árbol sin podar con cada una, al finalizar me quedo con el mejor de ellos y repito el proceso. Entonces tendremos n mejores modelos clasificadores. Cuando llega una nueva observación, cada modelo la clasificará. La clasificación queda determinada por la mayoría.

XG Boost: está basado en boosting y construye un conjunto de árboles de forma secuencial, y cada uno se enfoca en corregir los errores de los anteriores. Incluye términos de regularización en la función de pérdida para controlar el sobreajuste y mejorar la generalización del modelo.

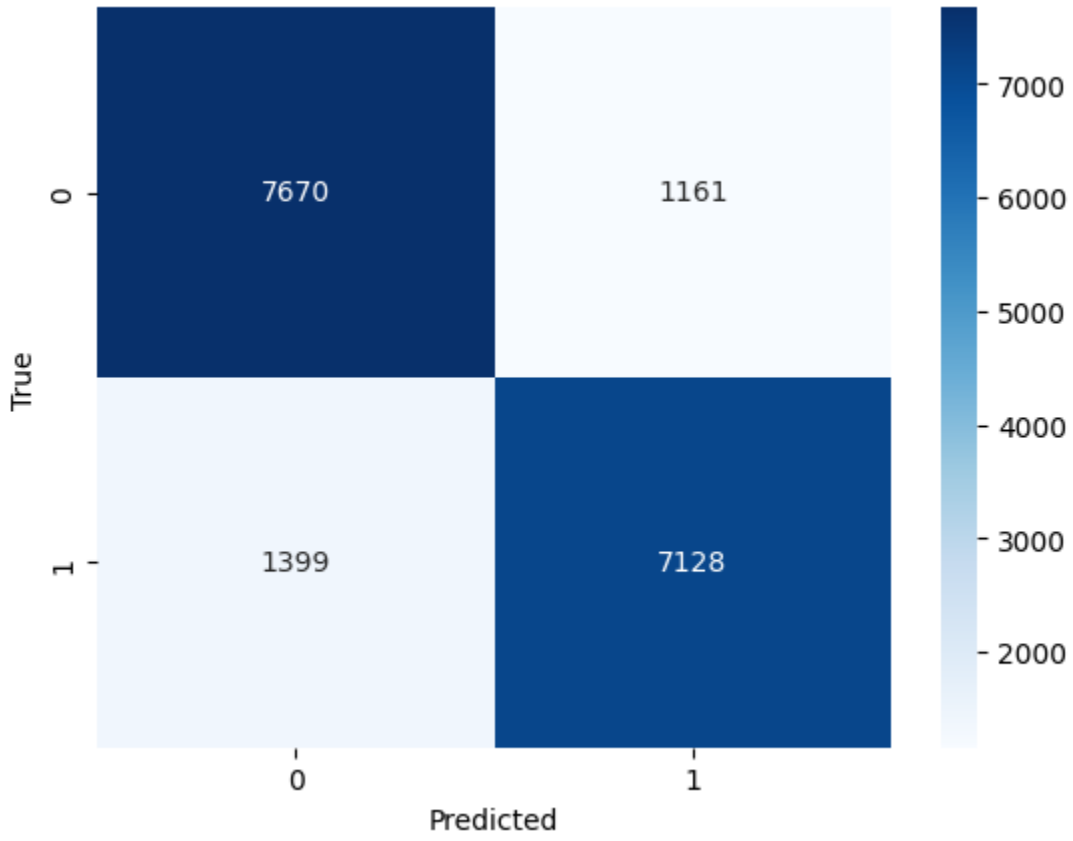
Voting: entrenar varios modelos distintos basados en el mismo dataset. La clasificación queda determinada por la mayoría.

Stacking: la idea principal es entrenar varios modelos dado un conjunto de datos y a continuación entrenar uno nuevo que, cuando reciba una nueva instancia de datos, determinará qué modelo es el más adecuado para dicha instancia.

Regresión Logística: método de análisis que predice una variable categórica binaria.

Matriz de Confusion

Analizamos la matriz de confusión del clasificador XGBoost.



- Verdaderos negativos hay 7670 casos. Las predicciones acerca de las reservas no canceladas hay un total de 7670 predicciones acertadas.
- Verdaderos positivos hay 7128 casos. Estos son los casos en los que el modelo predice correctamente que una reserva es cancelada.
- Falsos positivos hay 1161 casos. Estos son los casos en los que el modelo predice incorrectamente que una reserva se cancelará cuando, en realidad, no se cancela.
- Falsos negativos hay 1399 casos. Estos son los casos en los que el modelo predice incorrectamente que una reserva no se cancelará cuando, de hecho, se cancela.

Entre todos los modelos que construimos y testeamos, el XGBoost superó a todos en las métricas de prueba una vez mejorados sus hiper parámetros, dándonos nuestro más alto score.
Observamos una cantidad pareja de verdaderos positivos y negativos superando de la manera más alta que lo hicieron nuestros modelos a los falsos positivos y negativos.
Otra observación fue que en cuanto nuestros archivos de train y test, el XGBoost fue el mejor, pero al pasar el testeo de kaggle, lo fue con el clasificador Stacking. Este último también mostró un gran desempeño, pero en cuanto a tiempos de ejecución es muy costoso.

Tareas Realizadas

Integrante	Tarea
Miranda Marenzi	Armado de reporte Construcción de los modelos de predicción Optimización de parámetros
Mariana Noelia Juarez Goldemberg	Clasificador SVM Armado de reporte Evaluación de los modelos Correción
Lisandro Roman	Armado de reporte Optimización del dataset