Organización de Datos (75.06)
Trabajo Práctico 1: Reservas de Hotel. Checkpoint 3
Cuatrimestre, año y grupo: 2ºC 2023
Grupo: 11 - Los Pandas
Corrector: Mateo Suster
Fecha de entrega: 18 de octubre de 2023
Datos de los integrantes:

Labollita, Francisco	103456	flabollita@fi.uba.ar
Mundani Vegega, Ezequiel	102312	emundani@fi.uba.ar
Otegui, Matías Iñaki	97263	motegui@fi.uba.ar

Introducción

En este checkpoint se implementaron los clasificadores K-Nearest Neighbors, Support Vector Machine, XGBoost y los ensambles Random Forest, ensamble Voting y ensamble Stacking.

Para todos los clasificadores se creó una columna que dice si la habitación pedida fue la asignada, se eliminaron las filas con datos faltantes en *country*, *market_segment* y *distribution_channel*. Se eliminaron las columnas consideradas irrelevantes: *arrival_date_year*, *arrival_date_day_of_month*, *stays_in_weeknd_nights*, *stays_in_week_days*, *children*, *babies* , *reserved_room_type*, *deposit_type*, *company*, *adr* e *id*. También se eliminaron los valores considerados outliers de la columna *adults*. Por último se hizo one-hot encoding de las columnas categóricas.

Para KNN y SVM se normalizaron las columnas numéricas.

Para RF y XGBoost se transformaron en booleanas las columnas *required_car_parking_spaces*, *days_in_waiting_list*, *babies*, *previous_cancelations*, *special_requests previous_bookings_not_canceled*, *booking_changes*.

Para los ensambles tipo voting y stacking se decidió que todos los modelos utilicen un mismo dataset. En este se decidió por transformar en booleanas las columnas nombradas anteriormente, en normalizar los valores numéricos cuantitativos (algo útil para KNN y SVM y que no afecta al funcionamiento de RF y XGBoost) y realizar las modificaciones comunes a todos los clasificadores.

Construcción de los modelos

K-Nearest Neighbors

Los hiperparámetros óptimos para el modelo K-nearest neighbors (KNN) fueron encontrados optimizando parámetro por parámetro y dieron como resultado: metric=canberra, n_neighbors=77 y weights=distance.

Support Vector Machine

Dado a que entrenar modelos tipo SVM es mucho más costoso que con otros, se decidió por utilizar un dataset de entrenamiento reducido. Se optimizaron los hiperparámetros para este dataset y finalmente se entrenó al modelo con el dataset completo (sin considerar la parte utilizada para pruebas), utilizando los mejores hiperparámetros obtenidos anteriormente. Estos fueron kernel=rbf, gamma=scale y C=9.

Random Forest

Empezando con hiperparámetros aleatorios y luego mejorando a partir de eso mediante un Grid Search, los mejores hiperparámetros fueron criterion=gini, min_samples_leaf=1, min_samples_split=4, n_estimators=50 y oob_score=True.

XGBoost

Utilizando un Grid Search, los mejores hiperparámetros fueron subsample=0,7; colsample_bytree=0,6; n_estimators=400, max_depth=10 y learning_rate=0,1.

Ensamble Voting y Stacking

Para estos ensambles se utilizaron los mejores clasificadores obtenidos anteriormente (el mejor árbol de decisión, KNN, SVM, RF y XGBoost). Los hiperparámetros fueron de estos clasificadores, no de los ensambles en sí.

El metamodelo utilizado para el ensamble stacking fue el de regresión logística.

Cuadro de resultados para el conjunto test

Modelo	F1 Score	Precision	Recall	Accuracy	Kaggle
KNN	0,839	0,799	0,883	0,829	0,801
SVM	0,842	0,831	0,853	0,840	0,830
Random Forest	**0,858**	**0,868**	**0,848**	**0,859**	**0,861**
XGBoost	0,864	0,859	0,868	0,863	0,859
Voting	0,868	0,872	0,865	0,868	0,850
Stacking	0,864	0,863	0,864	0,862	0,848

Cuadro 1: Tabla de resultados

En la tabla se pueden comparar los resultados de todos los clasificadores y ensambles utilizados. El KNN es un clasificador donde las predicciones se basan en la clase de los 'k' puntos mas cercanos en el espacio de características. El SVM en cambio busca encontrar el hiper-plano óptimo que mejor separa las clases en un espacio multidimensional, maximizando el margen entre los puntos mas cercanos de las clases. El clasificador Random Forest construye múltiples arboles de decisión durante el entrenamiento y luego combina sus decisiones para alcanzar un solo resultado. El ultimo clasificador utilizado es el XGBoost. Utiliza el método de boosting para construir un modelo predictivo a partir de varios modelos débiles (generalmente árboles de decisión) de manera secuencial, enfocándose en corregir los errores de predicción en cada etapa y logrando una alta precisión en la clasificación y regresión de datos. El ensamble tipo Voting combina las predicciones de varios modelos base, en nuestro caso todos los clasificadores mencionados anteriormente, para obtener una predicción final. Esta predicción final se determina por votación o promediando las predicciones individuales de los modelos base. Finalmente, el ensamble tipo Stacking busca entrenar diferentes modelos (modelos base) y un modelo más, que decide, dada una instancia nueva, qué modelo usar. Se utiliza el concepto de meta-aprendizaje para reemplazar el mecanismo de voto.

Matriz de Confusión

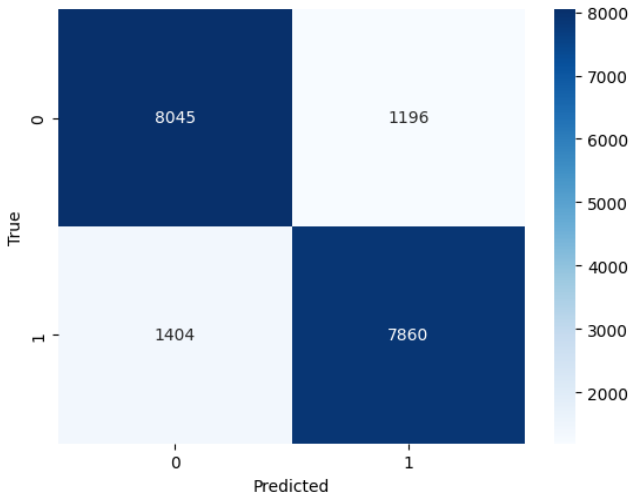

Figura 1: Matriz de confusión del mejor modelo obtenido (Random Forest).

Comparando al mejor modelo del checkpoint anterior, se puede ver que bajaron los falsos negativos considerablemente a 1196 e incrementaron los falsos negativos a 1404.

Distribución de tareas

Integrante	Tareas
Labollita, Francisco	SVM, Informe
Mundani Vegega, Ezequiel	KNN, SVM, Voting, Stacking, Informe
Otegui, Matías Iñaki	RF, XGBoost, Stacking, Informe