FACULTAD DE INGENIERIA
Universidad de Buenos Aires

Trabajo Práctico 1 — Informe final

[75.06/95.58] Organización de Datos
Segundo cuatrimestre de 2023

Grupo 15

Ayala, Tomás Gabriel - tayala@fi.uba.ar - 105336
Giacobbe, Juan Ignacio - jgiacobbe@fi.uba.ar - 109866
Olaran, Sebastian - solaran@fi.uba.ar - 109410

Docente corrector:
Pereira, Francisco

Índice

1. Introducción

Durante este primer trabajo práctico hemos explorado diversas técnicas de exploración de datos para poder crear modelos predictivos que den la mejor performance posible.

En la primera etapa nos dedicamos a hacer un preprocesamiento de los datos: hemos analizado variable a variable los valores que tomaban, su distribución de datos, sus valores atípicos(teniendo en cuenta el contexto de los datos), y diversas relaciones que tenían con otras variables. Mediante visualizaciones gráficas hemos obtenido esta información sobre cada una de las variables. Para los valores atípicos, hemos usado técnicas para encontrar outliers univariados(por ejemplo, mediante el gráfico de box plots), y multivariados(los detectamos usando distancia de Mahalanobis y mediante la creación de isolation forests).

Luego del preprocesamiento hemos realizado ingeniería de features: en un principio no descartamos ninguna variable de los datasets, pero hemos modificado algunos valores mal cargados de las mismas, y hemos hecho un tratamiento para los valores atípicos(muchos los hemos eliminado). También, hemos transformado ciertas variables usando técnicas como One Hot Encoding y Ordinal Encoding, para no tener problemas al crear los modelos predictivos. En un principio usamos todas las variables del dataset como features para los modelos, pero estos últimos han dado muy malas predicciones, debido a que el número de features era muy alto, y los modelos se terminaban complejizando mucho. Entonces, hemos decidido acotar el número de features que usamos para los mismos, a continuación los enumeramos y damos una idea de la información que nos proporciona cada una:

- **hotel**: indica el tipo de hotel en el que se realizó la reserva.
- **lead time**: indica el tiempo desde que se reservó hasta que la persona llegó al hotel.
- **arrival date year/month/week number/day of month**: indican la fecha de llegada.
- **stays in weekend/week nights**: indican las noches que están reservadas.
- **adults, children, babies**: indican la cantidad de adultos, niños y bebés respectivamente.
- **meal**: indica el tipo de comida que incluye la reserva.
- **country**: el país destino de la reserva.
- **market segment**: Segmento del mercado al cual está destinada la reserva.
- **distribution chanel**: Canal de distribución de reservas
- **is repeated guest**: Indica si el cliente ya ha tenido experiencia reservando.
- **previous cancellations/ previous bookings not canceled**: Indican la cantidad de reservas canceladas y no canceladas de un cliente.
- **reserved room type/ assigned room type**: Tipo de habitación reservada y asignada respectivamente.
- **booking changes**: Cambios que hubieron en el booking.
- **deposit type**: Tipo de depósito que tuvo la reserva.
- **agent**: Agente de viajes al que está asignada la reserva.
- **company**: Empresa a cargo del booking de la reserva.
- **days in waiting list**: Días en que la reserva estuvo en lista de espera.
- **customer type**: Tipo de cliente que tuvo la reserva.

- **adr**: Precio promedio de alquiler por día.
- **required car parking spaces**: Cantidad de espacios de estacionamiento.
- **required car parking spaces**: Espacios de estacionamiento requeridos
- **total of special requests**: Cantidad de invitados.

Una vez que obtuvimos el dataset modificado y las features correspondientes, comenzamos con la segunda etapa del trabajo, que consistió en crear modelos predictivos. Para la creación de modelos, hemos utilizado el dataset de train para entrenarlos, y además, a este mismo dataset lo hemos separado en dos datasets(uno para train y otro para test), de esta forma podíamos realizar un entrenamiento supervisado de los mismos, porque al tener el subconjunto de test(que provino del dataset de train), el modelo podía ajustarse dependiendo del valor que "debería"predecir, ya que el dataset de train justamente contaba con el valor final de la variable target.

Para los modelos también hemos usado K-Fold Cross Validation, y usamos para la mayoría de modelos el algoritmo de Random Search para optimizar los hiperparámetros de los modelos. En algunos modelos también usamos Grid Search, como por ejemplo para algunas redes neuronales.

Los modelos creados fueron: árboles de decisión, ensambles de modelos híbridos(del tipo Voting y Stacking), XGBoost, KNN, SVM, Random Forests y Redes Neuronales. Para todos los modelos usamos Cross Validation para entrenarlos, y una vez entrenados usamos la matriz de confusión y las métricas de precisión, recall, F1-score y accuracy para medir sus performances en el conjunto de validación. Luego predecimos el dataset de test y subimos la predicción de cada modelo a la competencia de Kaggle. Utilizamos tanto Random Search como Grid Search para la búsqueda de hiperparámetros óptimos, y podemos concluir que, en términos de tiempo, Random Search resultó ser menos costoso, y en cuanto al rendimiento, no encontramos diferencias significativas.

2. Cuadro de Resultados

A continuacion podemos observar un cuadro de las metricas de cada mejor predictor de cada modelo:

Modelo	CHPN	F1-Test	Precision Test	Recall Test	Accuracy	Kaggle
Árbol de Decisión	2	0.84871	0.84884	0.84881	0.84915	0.80227
Voting	3	0.853	0.865	0.841	0.854	0.820
Random Forest **(Mejor modelo)**	3	0.841	0.858	0.825	0.843	0.829
Red Neuronal	4	0.8164	0.9309	0.7271	0.8350	0.80193

El modelo de **Árbol de Decisión (Base)** tiene como objetivo predecir resultados a partir de observaciones en un conjunto de datos, utilizando criterios definidos por las características de las observaciones. Cuando una nueva observación se introduce en el modelo, se desplaza a través de los nodos del árbol hasta llegar a un nodo hoja, donde se le asigna una clase específica. Este enfoque se basa en la estructura de un árbol de decisión que se construye a partir de los datos disponibles.

El modelo de **Random Forest** combina varios modelos (en este caso arboles de decisión) para mejorar el rendimiento general y reducir el sobreajuste. Esto se hace mediante el proceso de promedio o votación de los resultados de los árboles individuales.

En el ensamble híbrido **Voting** la idea consiste en combinar múltiples modelos: es más probable que se obtenga una predicción precisa, ya que los errores individuales de cada modelo tienden a compensarse entre sí. El método de votación es especialmente útil cuando se utilizan diversos

tipos de modelos o algoritmos, ya que cada uno puede capturar diferentes aspectos de los datos o tener fortalezas en diferentes áreas. Al combinar sus predicciones, se puede lograr un rendimiento general mejorado.

El modelo de **Red Neuronal Multicapa (MLP)** se utiliza para predecir resultados a partir de observaciones. Consiste en capas de neuronas interconectadas que procesan la información y generan predicciones al pasar una observación a través de ellas. Es especialmente adecuada para problemas complejos y no lineales debido a su capacidad para aprender y representar patrones sofistficados en los datos.

El modelo que hemos identificado como el predictor más eficaz en nuestro trabajo práctico es el **Random Forest**. Esta elección se basa en una serie de factores que respaldan su eficiencia y su capacidad para proporcionar resultados sólidos.

En primer lugar, las métricas de rendimiento que evaluamos, como la precisión, la recall, accuracy y F1-score, revelaron que el modelo logró resultados sobresalientes en comparación con otros modelos que consideramos. Esto significa que fue capaz de predecir con precisión y consistencia, lo que es fundamental en tareas de predicción.

Además, observamos el rendimiento del modelo en Kaggle,donde nuestro modelo demostró su eficacia con un conjunto de datos nuevo desconocido. Esto confirma que su rendimiento es generalizable y no está sobreajustado a nuestros datos de entrenamiento específicos.

Un aspecto destacado es que, a pesar de que en un primer análisis no parecía estar demasiado ajustado a los datos de entrenamiento, su capacidad para adaptarse al conjunto de prueba fue destacable. Esto sugiere que nuestro modelo pudo generalizar patrones relevantes y no se limitó a memorizar los datos de entrenamiento, lo que es esencial para hacer predicciones precisas en situaciones del mundo real.

3. Conclusiones

Las conclusiones generales de nuestro trabajo son las siguientes:

- **Análisis Exploratorio de Datos (EDA)**: El análisis exploratorio de los datos resultó ser una etapa fundamental en nuestro trabajo. Nos permitió comprender mejor la naturaleza de los datos, identificar los features necesarios para crear modelos, para identificar valores atipicos,y seleccionar las características más relevantes para nuestros modelos. Esto nos ayudó a tomar decisiones más informadas a lo largo del proceso y poder obtener modelos optimos.

- **Preprocesamiento de Datos**: Las tareas de preprocesamiento, como la limpieza de datos, la transformacion de variables categóricas y la normalización, contribuyeron significativamente a mejorar la performance de nuestros modelos. Eliminaron ruido en los datos y permitieron a los modelos trabajar de manera más efectiva.

- **Rendimiento de los Modelos**: Después de entrenar y evaluar varios modelos, encontramos que el modelo XGBoost tuvo el mejor desempeño en el conjunto de prueba (TEST). Este modelo superó a otros al lograr un F1-score más alto.

- **Rendimiento en Kaggle**: Nuestro mejor modelo,Random Forest,fue el que mejor rendimiento competitivo en la plataforma Kaggle, lo que indica que es efectivo en un entorno del mundo real,y un modelo muy confiable al momento de predecir.

- **Simplicidad vs. Desempeño**: El modelo más sencillo de entrenar y más rápido en nuestro caso fue el Árbol de Decisión. Si bien no obtuvo el mejor rendimiento en términos de métricas, su simplicidad y velocidad pueden ser útiles en situaciones donde la eficiencia en el tiempo de entrenamiento es crítica y un rendimiento ligeramente inferior es aceptable.

- **Uso Productivo del Modelo**: Creemos que nuestro mejor modelo, Random Forest, puede ser utilizado de manera productiva en aplicaciones del mundo real, como la detección de fraudes, o probelas de dominio similares. Sin embargo, es importante realizar pruebas adicionales y ajustes antes de su implementación final.
- **Mejoras Potenciales**: Para mejorar nuestros resultados, podríamos considerar lo siguiente:

1. Explorar más a fondo las características y las interacciones entre ellas.
2. Aumentar el tamaño del conjunto de datos de entrenamiento si es posible.
3. Explotar de forma más exhaustiva el procesamiento de datos, y realizando una ingeniería de features más potente.
4. Afinar los hiperparámetros de los modelos para lograr un mejor rendimiento. Si se dispondría de mayor tiempo para entrenar los modelos posiblemente se mejorará la performance de los mismos.

4. Horas dedicadas

Integrante	Promedio semanal(Hs)
Tomas Gabriel Ayala	7
Sebastian Olaran	7
Juan Ignacio Giacobbe	6