(Mini) Informe

Comenzamos este segundo checkpoint adaptando el dataset de testeo al Dataset de train con el que trabajamos en la primera parte. Este trabajo consistió principalmente en eliminar columnas que no eran relevantes para el análisis, agregar otras que sí y modificar algunos registros que tenían Outliers. Luego aplicamos en ambos Datasets la técnica de One Hot Encoding para generar columnas numéricas correspondientes a las variables categóricas. Esto es necesario ya que los árboles de decisión que usamos en el tp sólo pueden trabajar con valores numéricos.

Una vez realizado el emparejamiento entre datasets de test y train, comenzamos a trabajar con árboles. Primero separamos el dataset de train hotels_train.csv en dos datasets: Uno lo utilizamos para entrenar nuestro modelo y el otro para testear el modelo obtenido. Elegimos una proporción de 80/20. Inicialmente, probamos con un árbol con una profundidad máxima de 20 y creamos un árbol utilizando el criterio Gini. Este modelo fue generado directamente tomando en cuenta todos los valores y sin generar ningún tipo de poda, para observar cómo se comporta el modelo sin hiperparametros. Al graficarlo obtenemos, por supuesto, un árbol muy grande y extremadamente complejo. Es en parte, debido a esto último que al hacer nuestra primera predicción exportando el csv obtuvimos un score de 0,79 en la competencia de Kaggle (nada mal para no haber UTILIZADO HIPERPARAMETROS). No debemos olvidar la gran cantidad de bifuraciones y reglas que tiene este árbol.

Para mejorar nuestra predicación y encontrar un árbol más performante, procedemos a crear un nuevo árbol con el que trabajamos para mejorar sus hiperparametros. En éste, usamos 10 folds y probamos 15 combinaciones posibles entre los parámetros para buscar la mejor entre todas ellas. Para ellos se buscó la optimización del F1_score, el cual fue seleccionado debido a la naturaleza del problema en el que no había un motivo particular para seleccionar una métrica específica (como recall o precision). Esto es debido a que el F1 score es una métrica que busca el equilibrio entre las métricas anteriormente mencionadas. Con este árbol obtuvimos un valor levemente mejor de f1_score. Sin embargo el mayor beneficio es que simplifica de manera importante las reglas utilizadas para predecir

Por último, para evaluar el árbol obtenido utilizamos la técnica de Cross Validation con 10 folds. Así comprobamos que en una generalización del modelo, este se comporta de manera similar a como lo venía haciendo. Esto se puede ver en, por ejemplo, los valores de todas las métricas, las cuales mantienen (aproximadamente).

REPORTEE LALALALALA

— MAXIMO 1 CARILLA

Checkpoint 2 : Árbol de decisión

Construir árboles de decisión y optimizar sus hiperparámetros mediante k-fold Cross Validation para obtener la mejor performance. ¿Cuántos folds utilizaron? ¿Qué métrica consideran adecuada para buscar los parámetros?

b. Graficar el árbol de decisión con mejor performance encontrado en el punto anterior. Si es muy extenso mostrar una porción representativa.

c. Analizar el árbol de decisión seleccionado describiendo los atributos elegidos, y decisiones evaluadas (explicar las primeras reglas obtenidas).

d. Evaluar la performance del modelo en entrenamiento y validación, explicar todas las métricas y mostrar la matriz de confusión.

e. Generar predicciones con el conjunto de test y realizar los submits correspondientes en la competencia de Kaggle.