Implementación de una técnica de aprendizaje máquina sin el uso de un framework

Oskar Adolfo Villa López

Abstract

En este trabajo se utiliza un algoritmo de regresión logística para predecir el estado de una reservación de hotel. Se describe la metodología utilizada para el tratamiento de datos y para el desarrollo del modelo sin el uso de frameworks de aprendizaje máquina. Finalmente, se presenta un modelo entrenado capaz de realizar predicciones con un error de alrededor de 25 %.

Introducción

En este trabajo se busca implementar un modelo de aprendizaje máquina para predecir un fenómeno real. El conjunto de datos seleccionado es Hotel Reservations, obtenido de la plataforma Kaggle [4]. Las filas contienen datos correspondientes a una reservación de hotel, e incluyen columnas como número de habitaciones, días de anticipación de la reservación o tipo de plan de comida. También se incluye una columna que indica si la reservación fue cancelada o no. El objetivo del modelo es predecir si la reservación será cancelada utilizando los datos de ésta.

Obtención y tratamiento de datos

Extracción y descripción

El archivo de datos contiene 36,275 muestras, con 19 columnas. Cada fila de la tabla contiene los datos de una reservación. Las columnas contenidas son las siguientes:

- **Booking_ID**: Identificador único de cada reserva
- **no_of_adults**: Número de adultos
- **no_of_children**: Número de niños
- **no_of_weekend_nights**: Número de noches de fin de semana (sábado o domingo) que el huésped se quedó o reservó para quedarse en el hotel
- **no_of_week_nights**: Número de noches entre semana (lunes a viernes) que el huésped se quedó o reservó para quedarse en el hotel
- **type_of_meal_plan**: Tipo de plan de comidas reservado por el cliente:
- **required_car_parking_space**: ¿El cliente requiere un espacio de estacionamiento? (0 - No, 1- Sí)
- **room_type_reserved**: Tipo de habitación reservada por el cliente. Los valores están cifrados (codificados) por INN Hotels.
- **lead_time**: Número de días entre la fecha de la reserva y la fecha de llegada
- **arrival_year**: Año de la fecha de llegada
- **arrival_month**: Mes de la fecha de llegada
- **arrival_date**: Día del mes de la fecha de llegada
- **market_segment_type**: Designación del segmento de mercado.
- **repeated_guest**: ¿Es el cliente un huésped recurrente? (0 - No, 1- Sí)
- **no_of_previous_cancellations**: Número de reservas anteriores que fueron canceladas por el cliente antes de la reserva actual
- **no_of_previous_bookings_not_canceled**: Número de reservas anteriores que no fueron canceladas por el cliente antes de la reserva actual
- **avg_price_per_room**: Precio medio por día de la reserva; los precios de las habitaciones son dinámicos. (en euros)
- **no_of_special_requests**: Número total de solicitudes especiales realizadas por el cliente (por ejemplo, piso alto, vista desde la habitación, etc.)
- **booking_status**: Bandera que indica si la reservación fue cancelada.

Selección de features y label

De acuerdo con el objetivo del análisis, la columna a predecir (label) será **booking_status**, mientras que el resto de columnas se usarán como variables independientes (features). No todas las columnas dentro de los features contienen información relevante para el análisis, por lo que que las siguientes columnas fueron descartadas:

- **Booking_ID**: El identificador es aleatorio y no contiene información relevante.
- **arrival_month, arrival_year, arrival_date**: Las fechas no se consideraron para el análisis, ya que se encuentran fuera del objetivo de aprendizaje de éste.

Columnas no numéricas

Los datos contienen columnas no numéricas. Éstas fueron transformadas a columnas numéricas utilizando el método One Hot Encoding, el cual crea una columna binaria para cada una de las opciones dentro de una columna no numérica, y elimina la columna original. Se eligió este método porque funciona bien cuando la columna contiene más de dos opciones y éstas no cuentan con relaciones escalares. Para la columna de label se utilizó Label Encoding, el cual cambia los valores contenidos en la columna (en este caso True y False) por valores binarios. Se utilizó este método porque la columna contiene solo valores correspondientes a verdadero y falso, los cuales pueden ser representados con 0 y 1 dentro de la misma columna.

Selección de conjuntos de entrenamiento y pruebas

Antes de seleccionar los subconjuntos, se utilizó la función *sample* de Pandas para reordenar de manera aleatoria el conjunto completo de datos, con el objetivo de evitar sesgos en caso de que los datos se encuentren ordenados. Es importante tener en cuenta aleatoriedad provoca que el código arroje resultados distintos cada vez que se ejecuta. Después de ser reordenados, se seleccionaron dos subconjuntos de datos, seleccionando el 80 % para entrenamiento y el 20 % para pruebas. Para la selección se utilizó la función *train_test_split* de Sklearn. Esto se debe a que el modelo tenía resultados inconsistentes entre ejecuciones por un balance incorrecto entre categorías presentes en el set de entrenamiento. La función se asegura de que las muestras se encuentren distribuidas correctamente.

Identificación de outliers y valores nulos

Se utilizaron las funciones *info* y *describe* de Pandas para poder identificar columnas con datos faltantes y outliers en los datos. Después de inspeccionar los resultados, no se encontraron valores nulos. Se detectaron outliers, presentes en la Figura 5 en los anexos, en las siguientes columnas:

- **no_of_week_nights**
- **lead_time**
- **no_of_previous_cancellations**
- **no_of_previous_bookings_not_canceled**

Para eliminarlos, se utilizó el método IQR, que etiqueta como valores atípicos los valores que se encuentran fuera de $Q3 + 1,5 * IQR$ o $Q1 - 1,5 * IQR$, donde $Q1$ y $Q3$ son los límites del primer y tercer cuartil, respectivamente, e IQR es el rango intercuartílico, o el rango que existe entre $Q1$ y $Q3$ [3]. Los valores atípicos fueron reemplazados por su límite más cercano.

Escalamiento de datos

Después de analizar los datos, se detectó que, a pesar de que cada columna tiene un rango aceptable de datos, los rangos difieren mucho entre columnas. Es por ello que se utilizó un escalamiento, con el objetivo de simplificar el proceso de entrenamiento al acercar todos los valores de los features a un mismo rango. Si los valores tienen una variación muy alta, es posible que el modelo tenga dificultad para adaptar la magnitud de cada uno de los parámetros para contrarrestar la diferencia en magnitud de los features.

Modelo de aprendizaje máquina

Selección del modelo

Se seleccionó un modelo de regresión logistica, el cual es útil para predecir resultados binarios mutuamente excluyentes [1], como es el caso de el estado de una reservación de hotel.

El modelo requiere una funciones de hipótesis, activación, costo y optimización, las cuales se describen a continuación. Utilizando la función de optimización, se ajusta la función de hipótesis en cada iteración de aprendizaje, llamada época, hasta alcanzar un costo mínimo.

Hipótesis (h)

La hipótesis es la función que busca predecir el resultado con relación a las variables independientes del modelo [2]. En la regresión logísitica se utiliza una función lineal con múltiples variables independientes, de la forma:

$$\hat{\mathbf{y}}(x_1, ..., x_n) = \theta_0 + \theta_1 x_1 + ... + \theta_n x_n$$

Donde $\hat{y}$ representa el label o variable dependiente, θ son los parámetros y las x representan los features o variables dependientes.

Activación

La función de hipótesis devuelve un valor con magnitud arbitraria que es poco interpretable. Es por ello que se utiliza una función de activación, que en el caso de la regresión logística tiene el objetivo de normalizar el resultado para acotarlo a un rango [2]. Para el modelo se utiliza la función sigmoidal, la cual acota los resultados a un valor entre 0 y 1, lo cual es útil para interpretar el resultado binario esperado. La función sigmoidal está definida como:

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

Costo

El costo es la función necesaria para calcular la diferencia entre la predicción del modelo y el valor real. Para el caso de clasificación con la función sigmoidal se puede utilizar la función cross-entropy, la cual resulta de la siguiente manera:

$$loss = -y(log(h(x)) - (1 - y)(log(1 - h(x)))$$

Donde $h(x)$ es el resultado de la función de hipótesis y y es el valor real de label.

Gradiente descendiente

Finalmente, se requiere una función para entrenar el modelo y darle la capacidad de aprender cuando se alimenta con datos. La función seleccionada para este modelo es gradiente descendiente, que consiste en una función de optimización que busca minimizar el costo de las predicciones [2]. La función es la siguiente:

$$\theta_j = \theta_j - \frac{\alpha}{m}\sum_{i=1}^{m}[(h_\theta(x_i) - y)x_{ij}]$$

Donde α es learning rate y m es la cantidad de muestras. Esta función debe aplicarse para cada una de las tetas, guardando el valor resultante y actualizando todos los valores al finalizar el cálculo.

Hiperparámetros

El modelo contiene los hiperparámetros learning rate y número de épocas. Para encontrar los valores óptimos de éstos, se entrenó y probó el modelo con diferentes combinaciones hasta encontrar un mínimo local en el error, lo que corresponde con un valor mayor en accuracy. El mejor modelo resultante se entrenó con un learning rate de 0.1 en 10000 épocas. La Figura 1 muestra una comparación de accuracy y error entre las diferentes combinaciones. La combinación 2 corresponde a la descrita anteriormente.

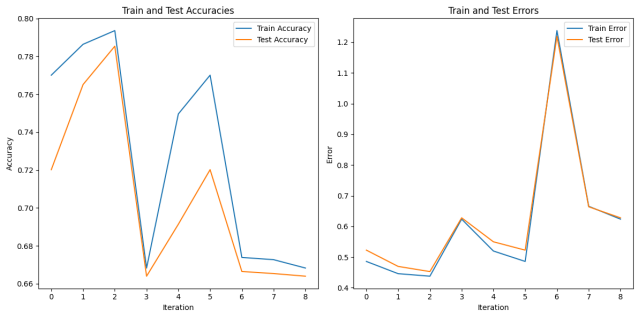

Figura 1: Accuracy y error con distintos hiperparámetros.

Resultados

Error en entrenamiento

Como se puede observar en la Figura 2, el error del modelo se redujo con cada época, lo que indica que el modelo fue ajustando los parámetros para predecir con más certeza el label. La magnitud del error no es interpretable, pero el comportamiento es el esperado.

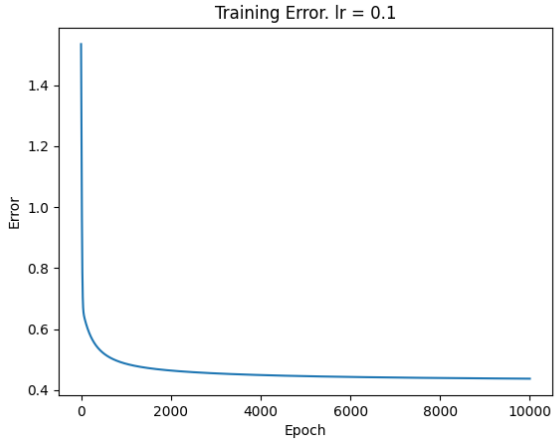

Figura 2: Error a lo largo del entrenamiento.

Matriz de confusión

La Figura 3 muestra la matriz de confusión para el conjunto de entrenamiento, mientras que la Figura 4 muestra la matriz para el conjunto de prueba. Como se puede observar, el modelo tiene respuestas correctas para la mayoría de las muestras del conjunto de entrenamiento. En el caso de las pruebas, el modelo predice correctamente los resultados 1, pero tiene dificultad con los valores 0. Con mucha frecuencia, el error produce falsos positivos.

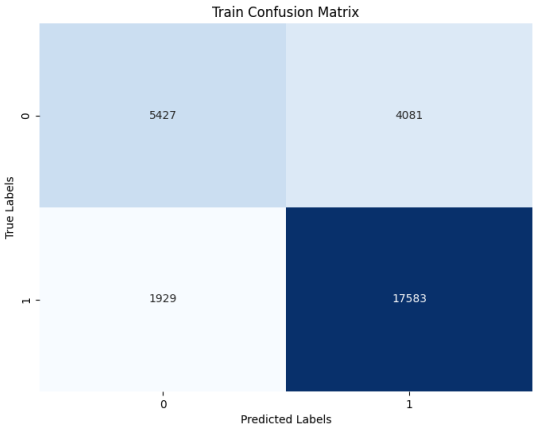

Figura 3: Matriz de confusión para conjunto de entrenamiento.

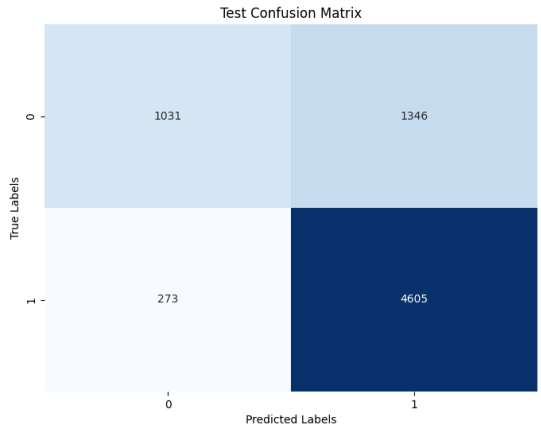

Figura 4: Matriz de confusión para conjunto de pruebas

Accuracy y F1

Las métricas de acierto utilizadas son accuracy y F1. Accuracy es el procentaje de acierto, el cual se calcula dividiendo el número de predicciones correctas entre las predicciones totales. Los valores obtenidos son los siguientes:

- **Accuracy entrenamiento:** 0.792901
- **Accuracy pruebas:** 0.776843
- **F1 entrenamiento:** 0.854041
- **F1 entrenamiento:** 0.850494

Ambas métricas tienen magnitudes similares para entrenamiento y pruebas. Los valores para entrenamiento son ligeramente más altos, lo cual es esperado ya que el modelo conoce con anterioridad los datos de entrenamiento.

Evaluación del modelo

Varianza

Debido a que las predicciones de los conjuntos de entrenamiento y pruebas son similares en cuanto a accuracy y F1, se puede deducir que la varianza es baja, ya que el modelo realiza predicciones estables. La varianza baja es correcta y esperada.

Sesgo

El sesgo del modelo es medio, ya que el modelo es incorrecto en aproxiamadamente 20 % de las ocasiones. Un sesgo maś bajo otorgaría mejores resultados en accuracy y F1, lo que significaría que el modelo es más adecuado. Este sesgo medio puede deberse a que el modelo no es lo suficientemente complejo para seguir la tendencia de los valores reales.

Ajuste

El ajuste del modelo está determinado por su capacidad de predecir resultados. En este caso, el modelo puede considerarse como *underfit*, ya que tanto la métrica accuracy como F1 pueden mejorarse para llegar a un modelo más robusto, y los valores fueron similares en el entrenamiento, lo que descarta una clasificación de *overfit*.

Conclusiones

A pesar de que el modelo no es 100 % preciso, se puede considerar que es adecuado para el problema que busca resolver, ya que la predicción de cancelaciones no involucra riesgos críticos de seguridad y de salud. Las predicciones con certeza cercana a un 80 % son aceptables en este contexto. Sin embargo, en caso de buscar mejorar el modelo, se deben considerar aproximaciones más complejas, ya que no se logró mejorar la eficacia del modelo lineal modificando los hiperparámetros.

Referencias

[1] Michael P. LaValley. "Statistical Primer for Cardiovascular Research". En: *Circulation* 117.18 (2008). DOI: https://doi.org/10.1161/CIRCULATIONAHA.106.682658.

[2] Peter Norvig. *Artificial Intelligence A Modern Approach*. Prentice Hall, 2015.

[3] Amerah Alabrah. "An Improved CCF Detector to Handle the Problem of Class Imbalance with Outlier Normalization Using IQR Method". En: *MDPI* 23.9 (2023). DOI: https://doi.org/10.3390/s23094406.

[4] Ahsan Raza. *Hotel Reservations Dataset*. URL: https://www.kaggle.com/datasets/ahsan81/hotel-reservations-classification-dataset (visitado 15-08-2024).

Anexos

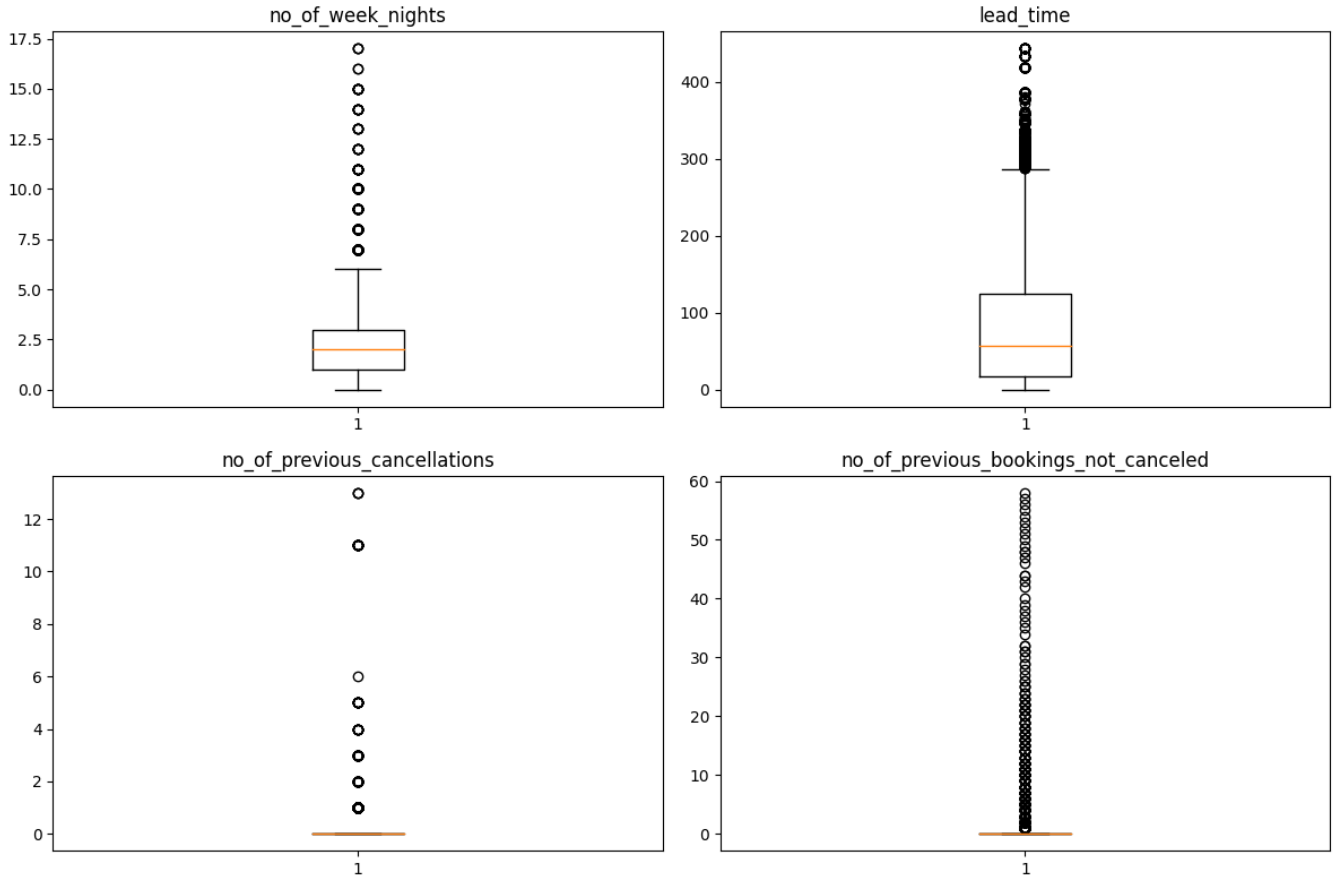

Figura 5: Distribución de datos para identificación de outliers.