Clasificación de ingreso anual mayor a 50k

José Pablo Martínez Valdivia

7 de septiembre de 2024

Abstract

This paper presents a predictive model based on the Random Forest algorithm to classify income levels using the Census Income (50k) dataset. The objective of the study is to predict whether an individual's income exceeds $50,000 per year based on demographic and work-related attributes. Furthermore, I discuss the various preprocessing decisions made, such as handling encoding categorical variables, and the choice of hyperparameters to optimize model performance.

1. Introducción

El dataset "Census Income" fue proveido por el "UCI Machine Learning Repository". Contiene 14 variables y 48842 instancias de datos provenientes de un censo realizado en 1994 para determinar si un adulto genera un ingreso anual mayor a $50k. Emplearemos el modelo de bosque aleatorio (random forest) para emplear una predicción categórica y disminuir el overfit en el modelo.

2. ETL

2.1. Extracción

Los datos fueron descargados por medio de la librería de "ucimlrepo". Estos proveen datos demográficos como edad, estado civil, educación, ocupación, raza, sexo, pais de origen, etc.

Los datos se encuentran completos en su mayoría, pero hay datos faltantes en las columnas de tipo de trabajo, ocupación y pais de origen como se puede ver en 1. Se tomó la decisión de deshacernos de las filas con datos faltantes las cuales representan 1221 instancias, lo cual es el 2.4 % de los datos; sobrándonos 47621 instancias. Esta decisión se tomó considerando que algún tipo de inputación podría introducir errores para la taréa de clasificación.

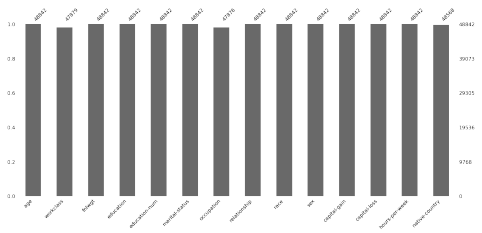

Figura 1: Valores faltantes.

2.2. Transformación

Para poder general el bosque aleatorio necesitamos convertir las columnas categóricas en valores numéricos. Para esto haremos uso de *one-hot encoding*, esta técnica consta de tomar todas las categorías en una columna y crear una columna por categoría, dejando un uno en la categoría que prsenta la instancia y ceros en el resto.

Colour
Green
Red
Blue

Green	Red	Blue
0	1	1
1	1	1
1	0	1
0	0	0
0	1	0

Figura 2: Método de one-hot encoding.

3. Marco teórico

3.1. Árboles de decisión

Un arbol de decisión, específicamente para clasificación, es un modelo que nos permite producir una salida discreta. Este modelo consta de una estructura de árbol binario en la cual cada nodo padre presenta una comparación lógica de una de las entradas y las hojas representan una clase de salida.

Este modelo nos trae una serie de problemas. Para empezar, la complegidad de tiempo para obtener el resultado de un arbol es $O(n \log n)$ dónde n se refiere al número de muestras con el que hayamos entrenedo el árbol, por otro lado la complegidad para crear el arbol es de:

$$O(n_{muestras} n_{variables} \log n_{muestras}) \qquad (1)$$

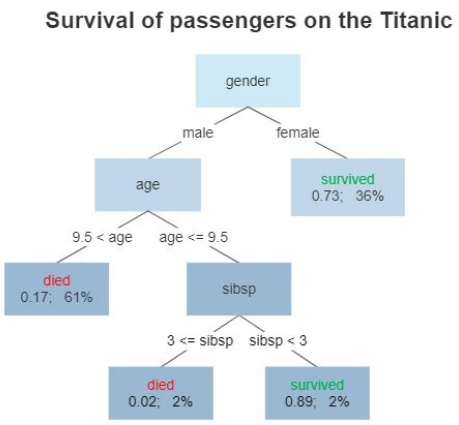

Figura 3: Estructura de un árbol de decisión.

lo cual indica que los tiempos de entrenamiento escalaran $O(n \log n)$ en base a la cantidad de muestras que tengamos en el dataset. Esto quizá no presenta un problema tan grave ya que solo contamos con 48 mil entradas, pero el problema más grande de los árboles de decisión es que el generar un árbol muy complejo esta sujeto sobreajuste, es decir, que el modelo se adapte a predecir los datos de entrenamiento y por lo tanto no generalice y tenga una menor precisión al predecir datos nuevos.

3.2. Bosque Aleatorio

Una de las formas de combatir el sobreajuste inevitable de los árboles de decisión es haciendo uso de un bosque aleatorio, este es un método de ensambe en el cual, en lugar de generar un solo árbol complejo. Generamos múchos árboles pequeños y tomamos como salida la clase que haya sido más votada por todos los árboles. Este método nos permite en general mitigar el sobreajuste dada la aleatoreidad de cada árbol.

Train score: 0.833841 Test score: 0.832231

4. Entrenamiento

El set de datos fue separado en tres conjuntos: entrenamiento, validación y pruebas constando respectivamente del 60 %, 20 % y 20 % de los datos del set. Para el modelo se decidió usar los siguientes híperparametros:

- n_estimators: 400
- max_leaf_nodes: 10
- max_depth: 20

Estos parámetros serán usados inicialmente para probar como se desempeña el modelo contra los datos de validación.