Machine Learning y políticas públicas-2024

Trabajo práctico N°1

Aprendizaje automatizado supervisado: Árboles de decisión

Alumna

Fernández, Silvia silvia2484@gmail.com

Introducción

En el siguiente trabajo práctico se propone implementar un modelo supervisado automatizado utilizando árboles de decisión, aplicado al conjunto de datos Wine. disponible en la url "https://archive.ics.uci.edu/ml/machine-learning-databases/wine/wine.data".

A partir del modelo automático supervisado proporcionado en el notebook del módulo 2, se propone modificar diversos parámetros del árbol de decisión, como los criterios de división y el tamaño permitido. El objetivo principal es analizar cómo estos ajustes afectan la organización de los datos y el rendimiento del modelo, permitiendo una comprensión más profunda del impacto de los parámetros en la estructura y la efectividad del árbol de decisión.

Para evaluar el modelo se evaluaron métricas estándar como la matriz de confusión, la precisión, recall y f1-score. Además, se aplicó la validación cruzada para medir la robustez y un enfoque ensemble learning mediante Bagging Classifier con árboles de decisión para mejorar el desempeño.

Desarrollo

En un principio, cabe destacar que el dataset Wine es ampliamente utilizado en problemas de clasificación supervisada debido al equilibrio en sus variables. El conjunto de datos contiene tres categorías de vinos, cada una representada por un grupo de observaciones con características específicas que describen propiedades químicas y físicas, cómo la acidez, el contenido de alcohol y otras medidas relevantes. Nuestra variable target va a ser **“class”**, quien encierra las 3 categorías de vinos, a partir de esta, vamos a probar nuestro modelo.

Se sugiere analizar los siguientes árboles:

Nombre	Criterio	Profundidad	Semilla	Mínimo de muestras
default	**gini**	**1**	**42**	**1**
depth_3	**gini**	**3**	**42**	**1**
gini_depth_5	**gini**	**5**	**42**	**5**
entropy_depth_5	**entropy**	**5**	**42**	**5**

Los árboles se construyen de la siguiente manera:

- Primero cargamos el dataset Wine utilizando la librería pandas.
- Especificamos nuestra variable target, “class”.
- Seleccionamos nuestras variables características y el objetivo.
- Definimos el modelo según el tipo de árbol qué precisamos
- Dividimos el conjunto de entrenamiento con un 80% y dejó el 20% para el test
- Entrenamos al modelo
- Obtenemos las predicciones

El modelo se evalúa de la siguiente manera:

- Se calcula una matriz de confusión para comparar etiquetas verdaderas del y_test con las predicciones del modelo.
- Se estima el puntaje en el conjunto de prueba para ver la proporción de predicciones correctas sobre el total de observaciones en el conjunto de prueba.
- Se calculan las métricas de precisión, recall y f1-score para evaluar el balance entre predicciones correctas y errores
- Se realiza validación cruzada para garantizar que el modelo tiene un desempeño consistente en diferentes particiones de datos.
- Se realiza un Bagging Classifier con el árbol de decisión para evaluar su desempeño en comparación con un modelo individual.
- Se estima un puntaje en el conjunto de prueba Bagging
- Se generan nuevas observaciones para cada clase y se obtienen las predicciones para evaluar el desempeño del modelo fuera de la muestra original

Resultados

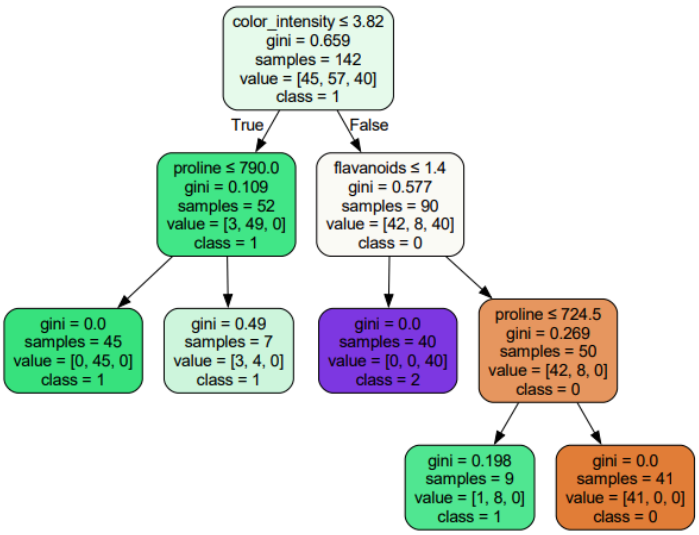

Imagen 1- Modelo gini_depth_5

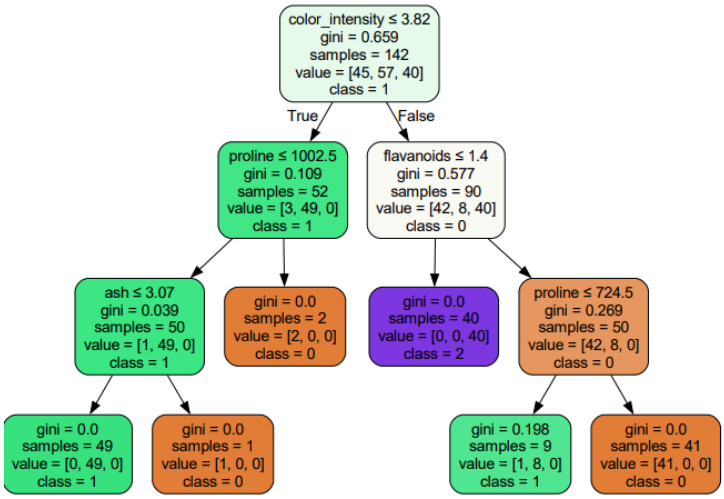

Imagen 2- Modelo depth_3

Imagen 3- Modelo default

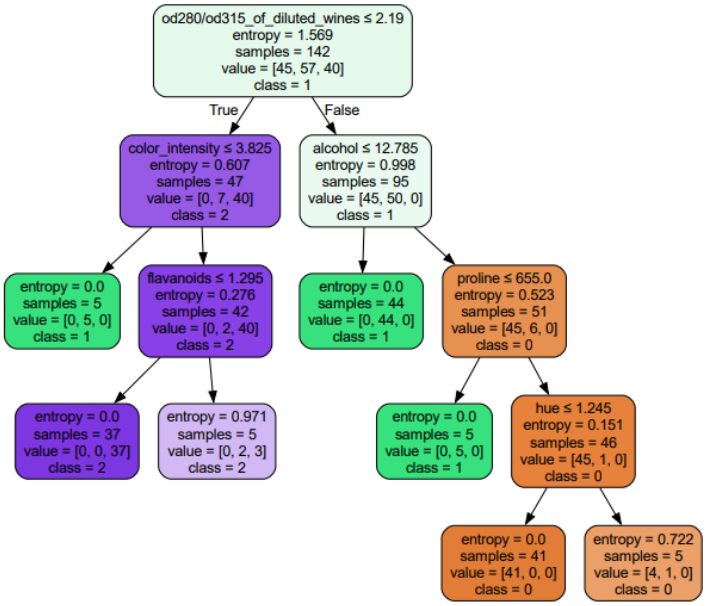

Imagen 4- Modelo entropy_depth_5

Tablas de métricas.

default				
Clase	**Precisión**	**Recall**	**f1-score**	**Support**
0	**1.00**	**0.93**	**0.96**	**14**
1	**0.88**	**1.00**	**0.93**	**14**
2	**1.00**	**0.88**	**0.93**	**8**
Accuracy			**0.94**	**36**
macro avr	**0.96**	**0.93**	**0.94**	**36**
weighted avg	**0.95**	**0.94**	**0.94**	**36**
validación cruzada	**[0.94444444 0.94444444 0.88888889 0.85714286 0.94285714]**			
promedio	**0.9155555555555555**			
Bagging	**0.9722222222222222**			

gini_5				
Clase	**Precisión**	**Recall**	**f1-score**	**Support**
0	**1.00**	**0.93**	**0.96**	**14**
1	**0.88**	**1.00**	**0.93**	**14**
2	**1.00**	**0.88**	**0.93**	**8**
Accuracy			**0.94**	**36**
macro avr	**0.96**	**0.93**	**0.94**	**36**
weighted avg	**0.95**	**0.94**	**0.94**	**36**
validación cruzada	**[0.94444444 0.91666667 0.88888889 0.88571429 0.88571429]**			
promedio	**0.9042857142857142**			
Bagging	**0.9722222222222222**			

depth_3				
Clase	**Precisión**	**Recall**	**f1-score**	**Support**
0	**1.00**	**0.93**	**0.96**	**14**
1	**0.88**	**1.00**	**0.93**	**14**
2	**1.00**	**0.88**	**0.93**	**8**
Accuracy			**0.94**	**36**
macro avr	**0.96**	**0.93**	**0.94**	**36**
weighted avg	**0.95**	**0.94**	**0.94**	**36**
validación cruzada	**[0.94444444 0.91666667 0.80555556 0.88571429 0.88571429]**			
promedio	**0.8876190476190476**			
Bagging	**0.9722222222222222**			

entropy_depth_5				
Clase	**Precisión**	**Recall**	**f1-score**	**Support**
0	**0.93**	**1.00**	**0.97**	**14**
1	**0.93**	**1.00**	**0.97**	**14**
2	**1.00**	**0.75**	**0.86**	**8**
Accuracy			**0.94**	**36**
macro avr	**0.96**	**0.92**	**0.93**	**36**
weighted avg	**0.95**	**0.94**	**0.94**	**36**
validación cruzada	**[0.94444444 0.91666667 0.88888889 0.94285714 0.97142857]**			
promedio	**0.9328571428571429**			
Bagging	**1.0**			

Conclusión

En general, todos los modelos que vemos en las imágenes de los resultados muestran divisiones razonables, logrando identificar correctamente las clases principales 0, 1 y 2, en la mayoría de los casos. Sin embargo hay dos modelos, **gini_depth_5** y **entropy_depth_5**, que presentan más divisiones y un mayor riesgo de especializarse en el conjunto de entrenamiento, afectando su capacidad de generalización. Los modelos más simples, **depth_3** y **default**, logran una separación efectiva con menor complejidad, mostrando que un árbol menos profundo es suficiente para el conjunto de datos Wine

A partir de las tablas, podemos ver que con el criterio **“gini”** y una profundidad máxima ajustada, el desempeño fue consistente, con una precisión, recall y F1-score elevados en toda las clases. Esto nos indica que, por un lado, **“gini”** proporciona una buena división de los datos en términos de pureza en el nodo, lo que favorece la clasificación. Por otro lado, limitar la profundidad máxima en 3 niveles, ayuda a evitar el sobreajuste, manteniendo un buen balance entre simplicidad y precisión del modelo. También podemos notar que el modelo con “entropy” y con una máxima profundidad mostró diferencias en el comportamiento de las clases, particularmente en la clase 2, su precisión de 1.0 y un recall bajo de 0.75, esto sugiere que el modelo está subestimando esta clase, clasificando algunos de estos a otra categoría. Por lo tanto se concluye que los modelos más adecuados para este conjunto de datos son **depth_3** y **default**, ya que **entropy_depth_5** parece ser menos robusto para clases con menos representatividad, como la 2.

Anexo

Fernandez, Silvia, Jupyter Notebook
Aprendizaje Automático Supervisado - Clasificación - Árboles de decisión - TP_Fernandez_Silvia