Instituto Tecnológico y de Estudios Superiores de Monterrey

Campus Estado de México

Concentración de Inteligencia Artificial Avanzada para la Ciencia de Datos

Momento de Retroalimentación: Módulo 2 Análisis y Reporte sobre el desempeño del modelo.

Paula Sophia Santoyo Arteaga

A01745312

Profesor:

Jorge Adolfo Ramírez Uresti

11 de septiembre, 2023

Clasificación de Vinos con Árboles de Decisión

En este entregable voy a analizar el segundo código que tiene la implementación de árboles de decisión usando un framework, en este caso utilicé la librería de scikit-learn para generar el modelo. Decidí ocupar el dataset de wine (vinos) que la misma librería ofrece ya que es accesible para cualquier persona, es un problema de clasificación claramente definido dado que el objetivo es clasificar vinos en una de las tres categorías (clase 0, 1 o 2) en función de sus características químicas, cuenta con un número moderado de datos (178 muestras y 13 características) lo hace adecuado para probar y evaluar modelos de aprendizaje automático sin requerir un tiempo excesivo de entrenamiento.

Con la función `load_data()` se cargan los datos y se dividen en el conjunto x que contiene todos los datos menos la clasificación y en el conjunto y se guardan los datos de la clasificación. En ambos conjuntos se tienen 178 datos y se ven de la siguiente manera:

```python
# Cargar dataset de wine de scikit-learn
def load_data():
    # Almacena el dataset de vinos
    wine = load_wine()
    # Almacena los datos de los atributos
    x = wine.data
    print(f'Datos de los atributos\n{x}\n{len(x)}')
    # Almacena los datos de la clasificación del vino
    y = wine.target
    print(f'Datos de la clasificación del vino\n{y}')
    # Almacena los nombres de la clasificacion de vinos (clase 0, 1 o 2)
    target_names = wine.target_names
    return x, y, target_names
```

Parte del código donde se carga el dataset

Conjunto de datos en variable x

```
Datos de los atributos
[[1.423e+01 1.710e+00 2.430e+00 ... 1.040e+00 3.920e+00 1.065e+03]
 [1.320e+01 1.780e+00 2.140e+00 ... 1.050e+00 3.400e+00 1.050e+03]
 [1.316e+01 2.360e+00 2.670e+00 ... 1.030e+00 3.170e+00 1.185e+03]
 ...
 [1.327e+01 4.280e+00 2.260e+00 ... 5.900e-01 1.560e+00 8.350e+02]
 [1.317e+01 2.590e+00 2.370e+00 ... 6.000e-01 1.620e+00 8.400e+02]
 [1.413e+01 4.100e+00 2.740e+00 ... 6.100e-01 1.600e+00 5.600e+02]]
```

Conjunto de datos en variable y

```
Datos de la clasificación del vino
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2]
```

Para poder entrenar el modelo es necesario dividir estos datos en tres partes: entrenamiento, prueba, validación. Para poder dividir y entrenar el modelo se manda llamar a la función `train_model()` en la cual se dividen los datos primero en un 70% para entrenamiento y 30% que son temporales los cuales se dividen entre prueba y validación. Para esto se dividieron en 60% para validación y 40% para prueba, es decir, serían un 18% y 12% del total. Estos datos se ven de la siguiente forma:

```
Partes del Conjunto de Entrenamiento (70%):
Número de muestras en el Conjunto de Entrenamiento: 124
X_train:
[[1.349e+01 3.590e+00 2.190e+00 ... 8.100e-01 1.820e+00 5.800e+02]
 [1.251e+01 1.730e+00 1.980e+00 ... 1.040e+00 3.570e+00 6.720e+02]
 [1.233e+01 9.900e-01 1.950e+00 ... 1.060e+00 2.310e+00 7.500e+02]
 ...
 [1.438e+01 1.870e+00 2.380e+00 ... 1.200e+00 3.000e+00 1.547e+03]
 [1.269e+01 1.530e+00 2.260e+00 ... 9.600e-01 2.060e+00 4.950e+02]
 [1.234e+01 2.450e+00 2.460e+00 ... 8.000e-01 3.380e+00 4.380e+02]]

y_train:
[2 1 1 0 1 0 2 1 1 2 0 0 0 2 0 0 1 2 1 0 2 1 0 2 1 1 0 1 0 0 1 0 0 2 1 1 1
 0 1 1 1 2 2 0 1 2 2 1 1 0 1 2 2 1 2 1 1 1 0 0 2 0 2 0 0 1 1 0 0 0 1 0 1 2
 1 1 1 2 2 1 0 0 1 2 2 0 1 2 2 2 2 1 0 1 0 2 0 0 1 0 0 2 1 0 2 2 0 0 2 2 2
 1 1 1 1 1 1 2 0 1 1 0 1 1]
```

Conjunto de datos de entrenamiento

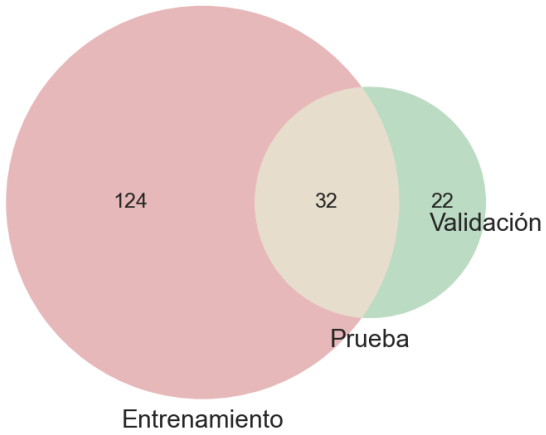
Conjunto de datos de prueba

```
Partes del Conjunto de Prueba (12%):
Número de muestras en el Conjunto de Prueba: 22
X_test:
[[1.208000e+01 1.390000e+00 2.500000e+00 2.250000e+01 8.400000e+01
  2.560000e+00 2.290000e+00 4.300000e-01 1.040000e+00 2.900000e+00
  9.300000e-01 3.190000e+00 3.850000e+02]
 [1.383000e+01 1.650000e+00 2.600000e+00 1.720000e+01 9.400000e+01
  2.450000e+00 2.990000e+00 2.200000e-01 2.290000e+00 5.600000e+00
  1.240000e+00 3.370000e+00 1.265000e+03]
 [1.258000e+01 1.290000e+00 2.100000e+00 2.000000e+01 1.030000e+02
  1.480000e+00 5.800000e-01 5.300000e-01 1.400000e+00 7.600000e+00
  5.800000e-01 1.550000e+00 6.400000e+02]
 [1.341000e+01 3.840000e+00 2.120000e+00 1.880000e+01 9.000000e+01
  2.450000e+00 2.680000e+00 2.700000e-01 1.480000e+00 4.280000e+00
  9.100000e-01 3.000000e+00 1.035000e+03]
 [1.362000e+01 4.950000e+00 2.350000e+00 2.000000e+01 9.200000e+01
  2.000000e+00 8.000000e-01 4.700000e-01 1.020000e+00 4.400000e+00
  9.100000e-01 2.050000e+00 5.500000e+02]
 [1.430000e+01 1.920000e+00 2.720000e+00 2.000000e+01 1.200000e+02
  2.800000e+00 3.140000e+00 3.300000e-01 1.970000e+00 6.200000e+00
  1.070000e+00 2.650000e+00 1.280000e+03]]

y_test:
[1 0 2 0 2 0 1 0 0 2 1 1 1 1 0 1 2 2 0 0 1 1]
```

Conjunto de datos de validación

```
Partes del Conjunto de Validación (18%):
Número de muestras en el Conjunto de Validación: 32
X_val:
[[1.336e+01 2.560e+00 2.350e+00 2.000e+01 8.900e+01 1.400e+00 5.000e-01
  3.700e-01 6.400e-01 5.600e+00 7.000e-01 2.470e+00 7.800e+02]
 [1.141e+01 7.400e-01 2.500e+00 2.100e+01 8.800e+01 2.480e+00 2.010e+00
  4.200e-01 1.440e+00 3.080e+00 1.100e+00 2.310e+00 4.340e+02]
 [1.402e+01 1.680e+00 2.210e+00 1.600e+01 9.600e+01 2.650e+00 2.330e+00
  2.600e-01 1.980e+00 4.700e+00 1.040e+00 3.590e+00 1.035e+03]
 [1.434e+01 1.680e+00 2.700e+00 2.500e+01 9.800e+01 2.800e+00 1.310e+00
  5.300e-01 2.700e+00 1.300e+01 5.700e-01 1.960e+00 6.600e+02]
 [1.378e+01 2.760e+00 2.300e+00 2.200e+01 9.000e+01 1.350e+00 6.800e-01
  4.100e-01 1.030e+00 9.580e+00 7.000e-01 1.680e+00 6.150e+02]
 [1.350e+01 3.120e+00 2.620e+00 2.400e+01 1.230e+02 1.400e+00 1.570e+00
  2.200e-01 1.250e+00 8.600e+00 5.900e-01 1.300e+00 5.000e+02]
 [1.364e+01 3.100e+00 2.560e+00 1.520e+01 1.160e+02 2.700e+00 3.030e+00
  1.700e-01 1.660e+00 5.100e+00 9.600e-01 3.360e+00 8.450e+02]
 [1.242e+01 4.430e+00 2.730e+00 2.650e+01 1.020e+02 2.200e+00 2.130e+00
  4.300e-01 1.710e+00 2.080e+00 9.200e-01 3.120e+00 3.650e+02]
 [1.237e+01 1.210e+00 2.560e+00 1.810e+01 9.800e+01 2.420e+00 2.650e+00
  3.700e-01 2.080e+00 4.600e+00 1.190e+00 2.300e+00 6.780e+02]
 [1.237e+01 1.630e+00 2.300e+00 2.450e+01 8.800e+01 2.220e+00 2.450e+00
  4.000e-01 1.900e+00 2.120e+00 8.900e-01 2.780e+00 3.420e+02]
y_val:
[2 1 0 2 2 2 0 1 1 1 0 0 1 0 1 2 1 2 0 2 1 0 1 0 2 0 2 1 0 1 0 1]
```

División de Datos entre Conjuntos de Entrenamiento, Prueba y Validación



Para poder evaluar el modelo se crea una matriz de confusión por cada conjunto y cada profundidad máxima del árbol. También se genera un informe de clasificación que incluye las métricas de precisión, recall, f1-score y support. También para evaluar el modelo se genera una gráfica que muestra la curva de aprendizaje en cada una de las profundidades máximas del árbol. Una ejecución se ve de la siguiente forma:

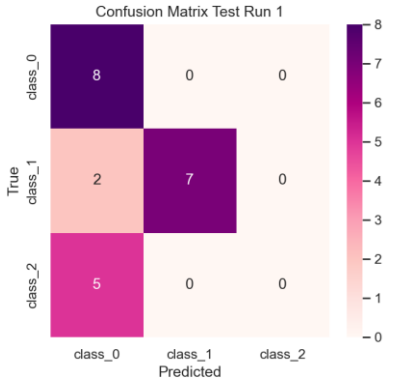
```
*****   RUN 1   *****
Max Depth: 1

                    ~ TRAINING DATA ~

Accuracy: 0.66
Classification Report:
              precision    recall  f1-score   support

     class_0       0.49      0.95      0.64        40
     class_1       0.96      0.88      0.92        50
     class_2       0.00      0.00      0.00        34

    accuracy                           0.66       124
   macro avg       0.48      0.61      0.52       124
weighted avg       0.54      0.66      0.58       124
```

~ TEST DATA ~

Accuracy: 0.68

Classification Report:

	precision	recall	f1-score	support
class_0	0.53	1.00	0.70	8
class_1	1.00	0.78	0.88	9
class_2	0.00	0.00	0.00	5
accuracy			0.68	22
macro avg	0.51	0.59	0.52	22
weighted avg	0.60	0.68	0.61	22

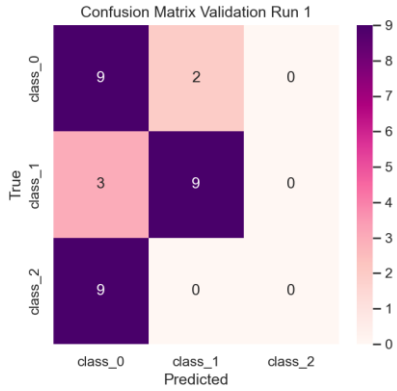


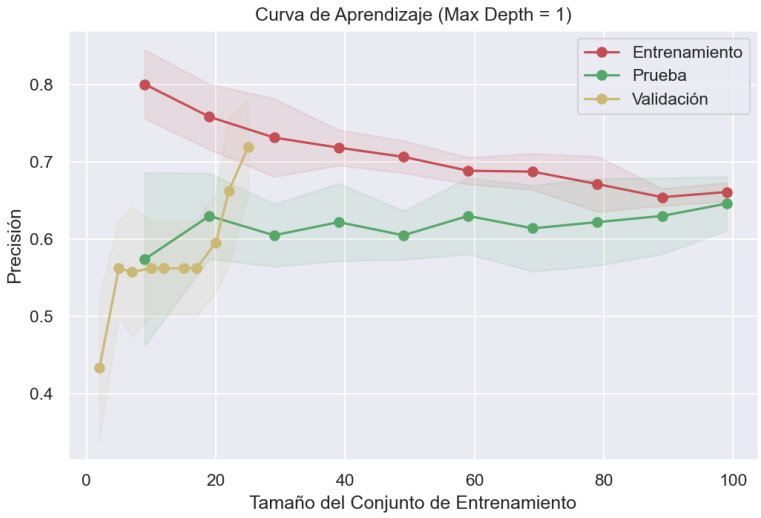
~ VALIDATION DATA ~

Accuracy: 0.56

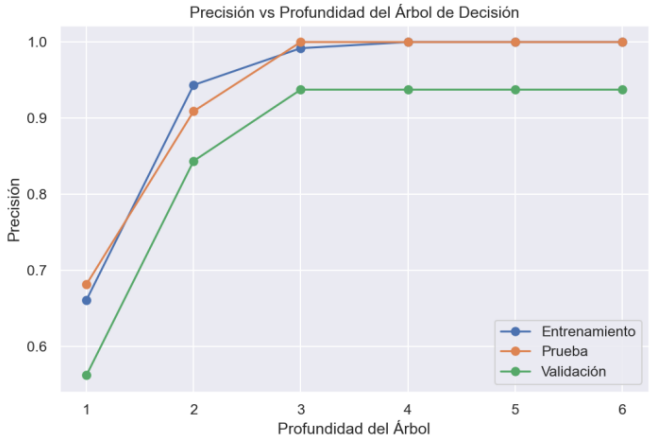
Classification Report:

	precision	recall	f1-score	support
class_0	0.43	0.82	0.56	11
class_1	0.82	0.75	0.78	12
class_2	0.00	0.00	0.00	9
accuracy			0.56	32
macro avg	0.42	0.52	0.45	32
weighted avg	0.45	0.56	0.49	32



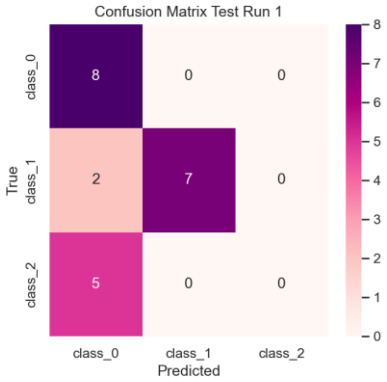


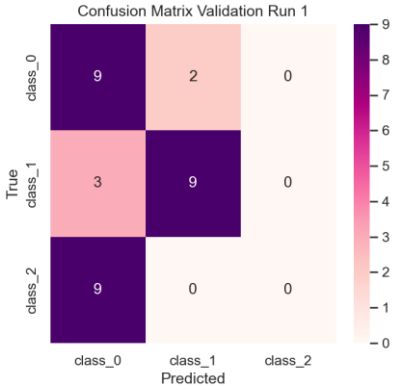
Por lo que, al ejecutarse 6 veces, se tendrán 18 gráficas de confusión, 6 informes de clasificación y 6 gráficas de curva de aprendizaje. Al finalizar el código, se obtiene una gráfica comparando las precisiones de cada uno de los conjuntos en cada una de las profundidades.

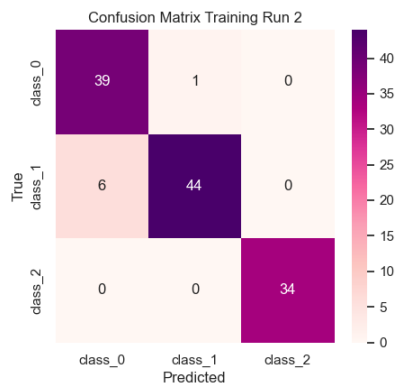


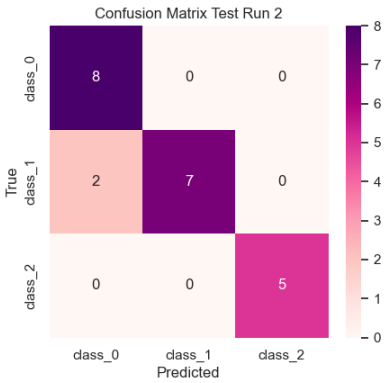
Con esto podemos hacer los análisis para los grados de sesgos y varianza que a su vez nos van a servir para saber qué nivel de ajuste se tiene.

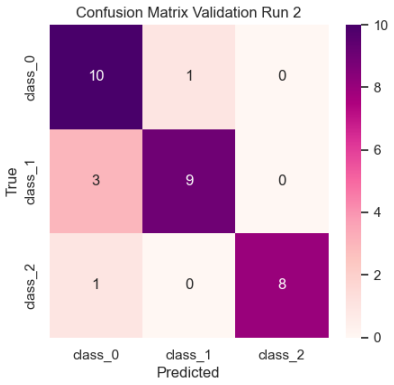
Gráficas de matriz de confusión

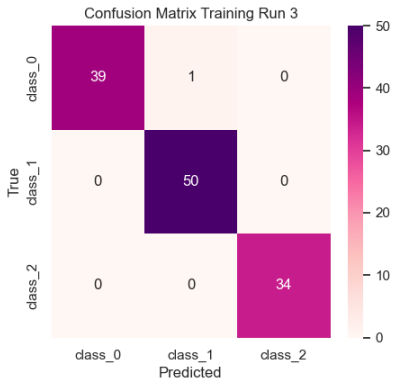


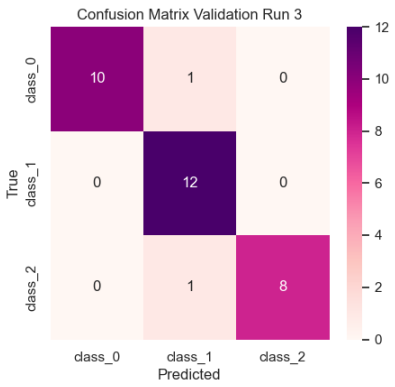


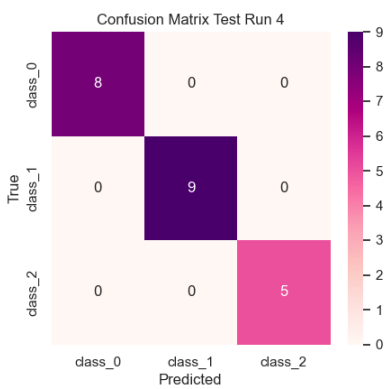


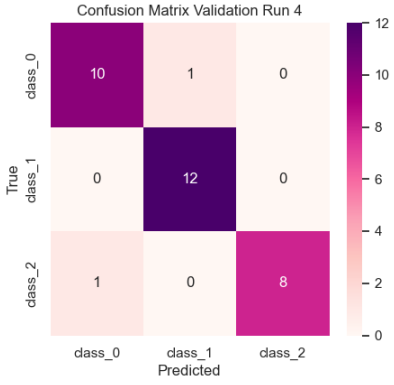


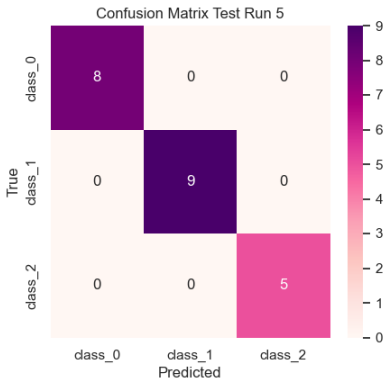


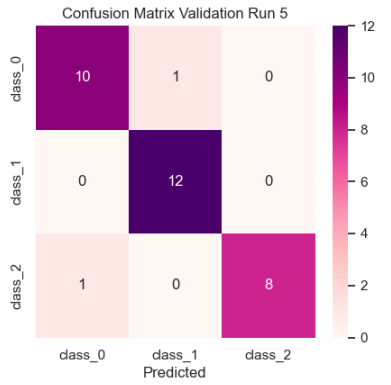


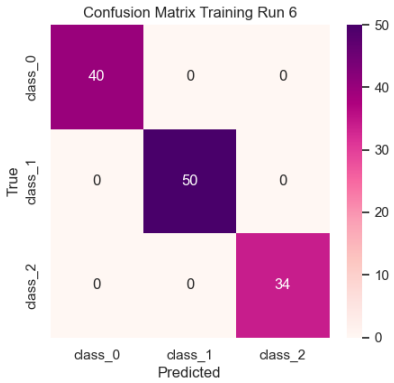


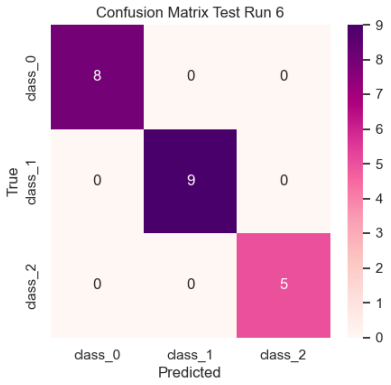


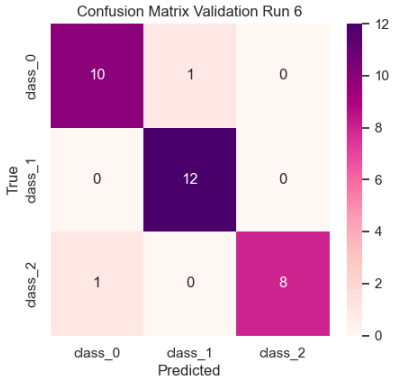




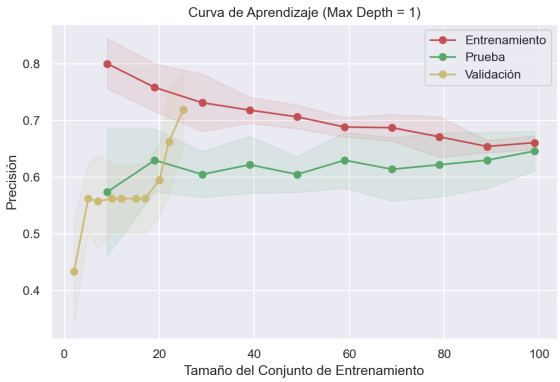


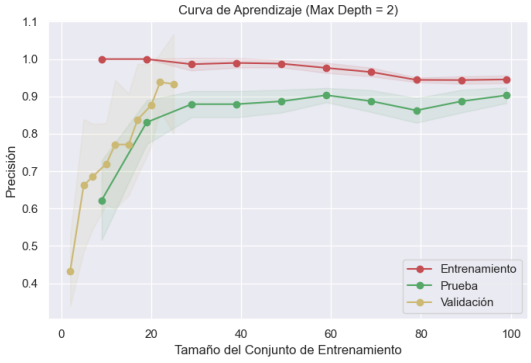


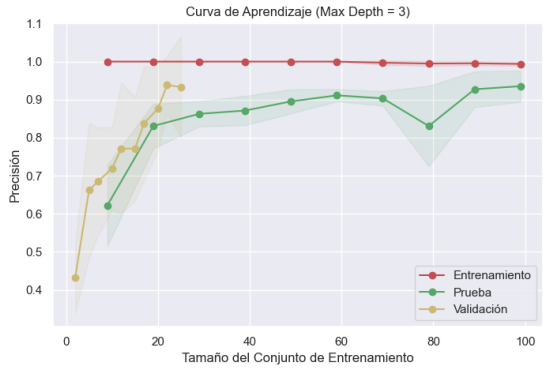


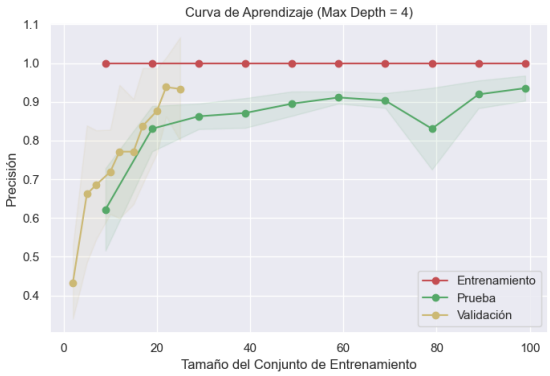


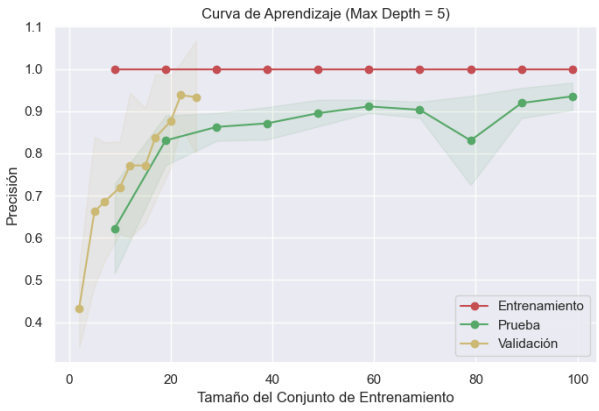
Gráficas de curva de aprendizaje

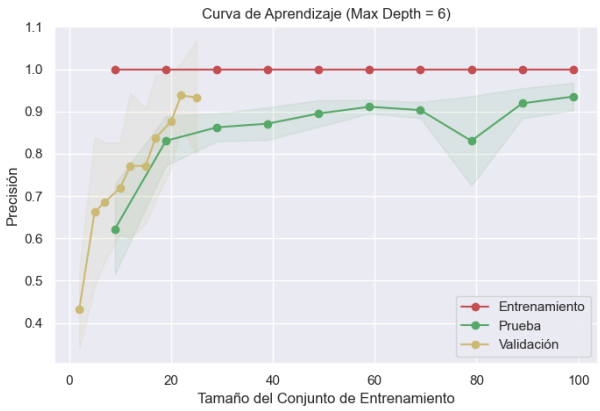




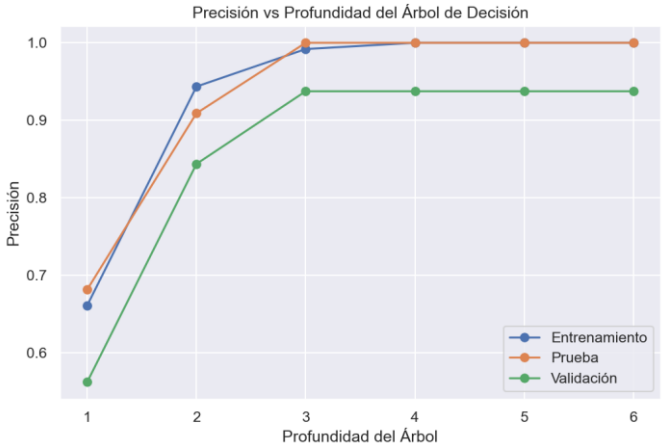








Gráfica de Precisión vs Profundidad



Analizando tanto las gráficas de la matriz de confusión como las de la curva de aprendizaje podemos observar que el grado de bias o sesgo no es alto ya que el modelo se ajusta adecuadamente a los datos de entrenamiento y se puede ver más a detalle en las gráficas de curva de aprendizaje ya que no se tiene mucha separación entre los datos de entrenamiento y los de prueba. Dado a que se tiene un alto rendimiento en el conjunto de entrenamiento y el rendimiento en los datos de prueba es similar al del de entrenamiento se puede decir que el grado de sesgo que tiene el modelo es bajo.

Para la varianza, podemos observar que, al tener un buen rendimiento en los datos de entrenamiento, los conjuntos de validación y prueba son consistentes con el rendimiento, sin embargo, podemos notar un ligero sobreajuste, aunque el mismo no es muy grave por lo que podemos decir que se tiene una varianza media a baja.

Por último, para saber el nivel de ajuste del modelo podemos observar en las gráficas que el modelo tiene buen rendimiento tanto en el entrenamiento como en las pruebas por lo que no tiene sobreajuste. Tampoco se ve un bajo rendimiento en los datos de entrenamiento como en los de prueba así que no se tiene underfitting por lo que se llega a la conclusión de que se tiene un buen ajuste ya que se cuenta con un equilibrio entre los datos de entrenamiento y los de prueba. Por esto mismo se puede decir que se puede generalizar bien nuevos datos que se le den al modelo.

Para lograr estos niveles se usó un número determinado de profundidad máxima para el árbol de decisión, en este caso se tuvo una profundidad de 1 a 6. Para evaluar el modelo recurrí a obtener el puntaje de precisión, así como las métricas como recall o f1-score que vienen dentro del informe de clasificación. Además, se construyeron las matrices de confusión para ver cuantos aciertos y errores se tuvieron durante la evaluación del modelo. Usando la profundidad máxima en el modelo fue crucial ya que, si vemos las gráficas de la curva de aprendizaje, el modelo fue más acertado mientras el valor de la profundidad iba aumentando regresando una buena evaluación del modelo.