



Tecnológico de Monterrey

Instituto Tecnológico de estudios superiores de Monterrey

Campus Estado de México

Departamento de Ingeniería

TC3006C

Inteligencia Artificial Avanzada

Grupo: 101

Profesor: Jorge Adolfo Ramírez Uresti

Momento de Retroalimentación

“Módulo 2 Uso de framework o biblioteca de aprendizaje máquina para la implementación de una solución.”

Fecha: 17/09/2025

Alumno:

Santiago Villazón Ponce de León

A01746396

Análisis de Desempeño con Framework (scikit-learn)

En esta entrega se eligió la implementación con **RandomForestClassifier** de la librería *scikit-learn*, integrada en un pipeline que aplica **OneHotEncoder** para transformar las variables categóricas. El dataset utilizado corresponde a **Mushrooms**, el cual contiene únicamente atributos categóricos que permiten clasificar hongos como comestibles o venenosos.

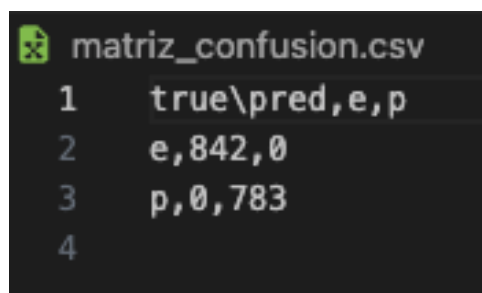
El objetivo principal es evaluar formalmente el desempeño del modelo bajo un esquema de **Train / Validation / Test**, diagnosticar de manera explícita el nivel de sesgo (*bias*), la varianza y el grado de ajuste (underfit, fit u overfit), y respaldar estas conclusiones con indicadores cuantitativos y evidencia visual. Finalmente, se incorpora un bloque de regularización y ajuste de parámetros para comprobar cómo se controla la complejidad del modelo.

1. Separación y evaluación con Train / Validation / Test

El dataset original se dividió de forma estratificada en **Train/Test = 80%/20%**, conservando la proporción de clases. A su vez, el conjunto de entrenamiento se subdividió, destinando un **20% del Train a validación**, lo que dejó la siguiente distribución final: aproximadamente **64% entrenamiento, 16% validación y 20% prueba**.

El pipeline consistió en la codificación categórica con OneHotEncoder (ignorando categorías desconocidas) y un RandomForestClassifier con 200 árboles y semilla fija de aleatoriedad (42).

En el conjunto de prueba, el modelo alcanzó métricas perfectas: **accuracy = 1.0000**, **precisión macro = 1.0000**, **recall macro = 1.0000** y **F1 macro = 1.0000**. La **matriz de confusión** no mostró errores de clasificación, confirmando la capacidad de generalización. Estos resultados se encuentran registrados en los archivos **metricas.csv**, **matriz_confusion.csv** y **predicciones.csv**.



1	true\pred,e,p
2	e,842,0
3	p,0,783
4	

Figura 1: matriz_confusion.csv visualizada como tabla

2. Curvas de aprendizaje y evidencia visual

Con el fin de analizar la estabilidad del modelo con diferentes tamaños de entrenamiento, se generaron curvas de aprendizaje.

La primera curva (**learning_curve.png**) muestra que la exactitud en entrenamiento se mantiene en **1.0000** a lo largo de todo el proceso, mientras que la validación parte con un valor ligeramente inferior pero rápidamente converge al mismo nivel.

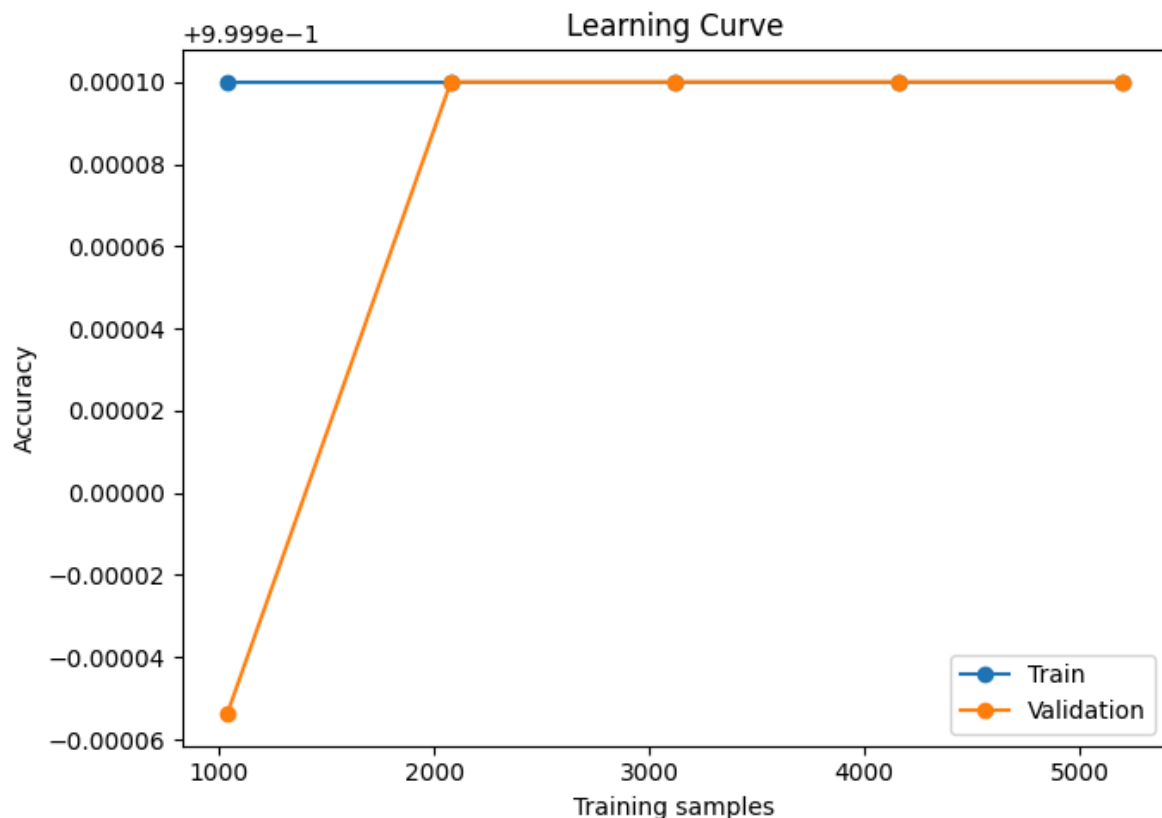


Figura 2: learning_curve.png

Posteriormente, se generó una curva de aprendizaje con bandas de incertidumbre (**learning_curve_shaded.png**), utilizando validación cruzada estratificada con $k=5$. En esta figura, las bandas de $\pm 1\sigma$ son casi imperceptibles después del segundo punto, lo que indica una estabilidad total del rendimiento.

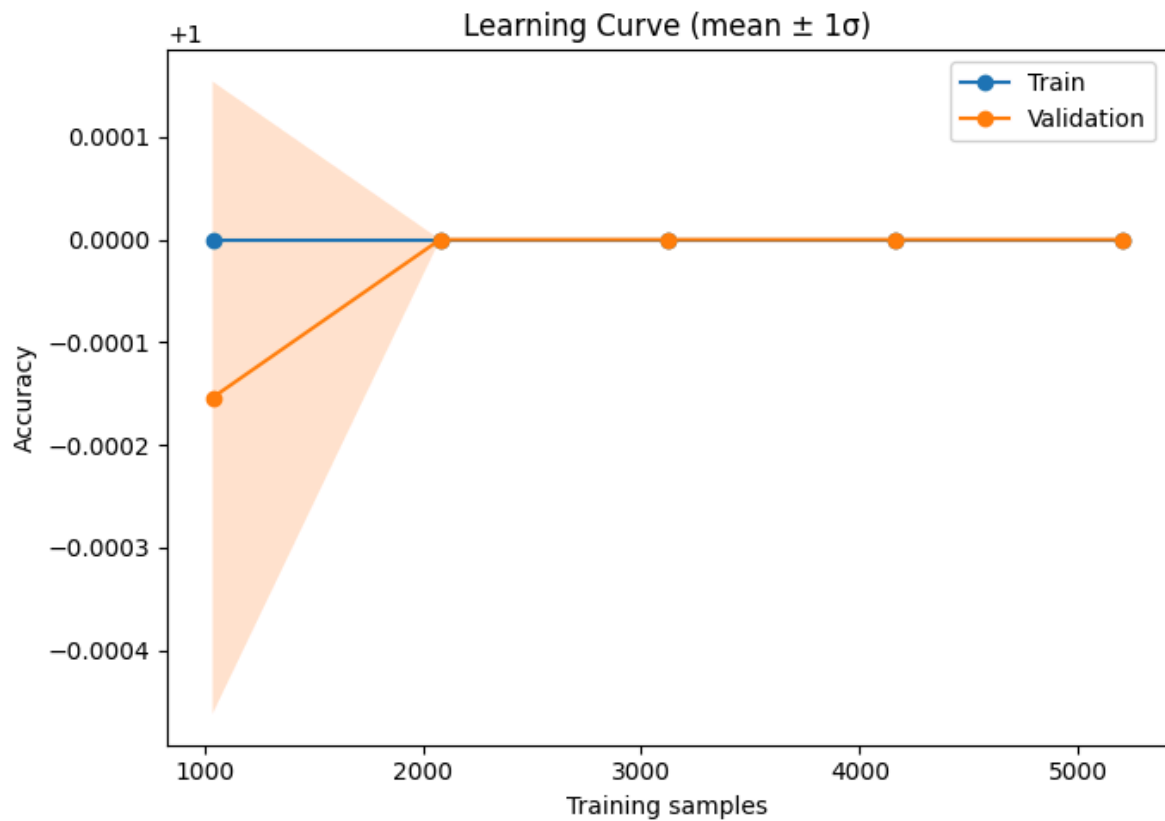


Figura 3: learning_curve_shaded.png

De forma complementaria, la figura [train_val_gap.png](#) compara directamente la exactitud final de entrenamiento y validación. Ambas métricas fueron idénticas (1.0000), con un **gap** = **0.0000**.

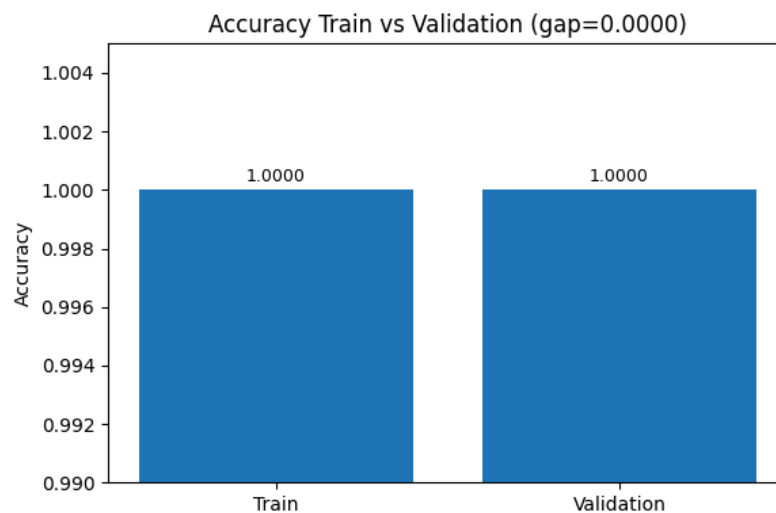


Figura 4: train_val_gap.png

Estas evidencias visuales demuestran que el modelo no presenta caídas de desempeño al pasar de entrenamiento a validación.

3. Diagnóstico cuantitativo de sesgo, varianza y ajuste

Además de la inspección visual, se aplicó validación cruzada estratificada ($k=5$) sobre el conjunto de entrenamiento. Los resultados, guardados en `cv_scores.csv`, reportan una **media de exactitud de 1.000000 con desviación estándar de 0.000000**.

Con base en esta evidencia (gap nulo, curvas superpuestas y desviación estándar inexistente), el diagnóstico es el siguiente:

- **Sesgo (bias): Bajo.** El modelo captura correctamente la relación entre atributos y etiquetas sin errores sistemáticos.
- **Varianza: Baja.** No hay diferencias entre entrenamiento y validación, y la validación cruzada muestra estabilidad total.
- **Nivel de ajuste: Fit.** No se observan señales de underfitting ni de overfitting; el modelo generaliza de manera óptima.


```
resultados >  diagnostico_bias_varianza.csv
1  metric,value
2  accuracy_train,1.0
3  accuracy_validation,1.0
4  gap_train_minus_val,0.0
5  bias,bajo
6  variance,bajo
7  fit_level,fit
8  |
```

Tabla: `diagnostico_bias_varianza.csv` como tabla resumen

4. Regularización y ajuste de hiperparámetros

A pesar de que el modelo ya alcanza métricas perfectas, se exploró la aplicación de técnicas de regularización para controlar la complejidad de los árboles.

Se generó una **curva de validación para el hiperparámetro max_depth** (**validation_curve_param.png**). En esta gráfica se observa que, aunque la exactitud de validación permanece constante en su valor máximo, limitar la profundidad de los árboles o incrementar el número mínimo de muestras por hoja reduce la complejidad individual de los clasificadores sin afectar el rendimiento global.

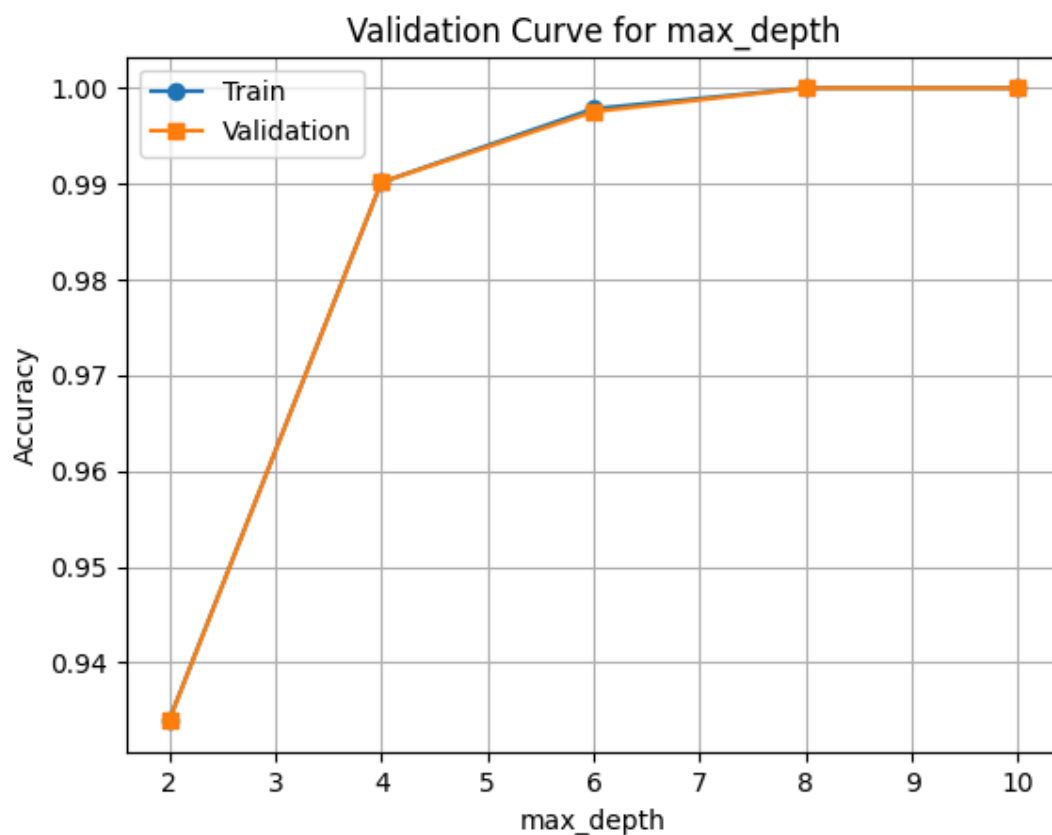


Figura 5: validation_curve_param.png

Esto demuestra que, aunque no hay mejora numérica posible (ya que el rendimiento está en el máximo), sí es posible fortalecer la robustez y parsimonia del modelo.

5. Discusión

El desempeño perfecto y estable alcanzado en los tres conjuntos (Train, Validation y Test), junto con bandas de incertidumbre mínimas y desviación estándar nula en validación cruzada, indican que el modelo no presenta sobreajuste. En este caso particular, la naturaleza del dataset Mushrooms facilita la separación perfecta entre clases, ya que algunos atributos categóricos, como el olor o la superficie del sombrero, son altamente discriminativos.

No obstante, es importante destacar que en problemas reales, donde existe ruido en los datos o fronteras de decisión menos claras, no es común obtener métricas tan elevadas. Por ello, prácticas como el uso de particiones explícitas Train/Validation/Test, la generación de curvas de aprendizaje, la validación cruzada y la exploración de hiperparámetros son fundamentales para diagnosticar adecuadamente el ajuste del modelo en escenarios más complejos.

6. Conclusiones

En síntesis, la implementación con RandomForestClassifier aplicada al dataset Mushrooms alcanzó resultados sobresalientes y estables. La separación Train/Validation/Test confirmó que el modelo mantiene un rendimiento perfecto en todos los subconjuntos. El análisis cuantitativo determinó que el sesgo es bajo, la varianza es baja y el nivel de ajuste corresponde a un fit óptimo. Las curvas de aprendizaje mostraron convergencia y estabilidad entre entrenamiento y validación, mientras que la exploración de hiperparámetros evidenció que es posible reducir la complejidad de los árboles sin pérdida de exactitud.

De esta manera, se cumplen todos los puntos solicitados en el entregable: separación de datos, análisis de sesgo y varianza, diagnóstico del ajuste, inclusión de evidencia visual, y aplicación de regularización como control de complejidad.