DEPARTAMENTO DE COMPUTACION

Facultad de Ciencias Exactas y Naturales - UBA

Reconociendo Dígitos en su Forma Más Pura: Un Enfoque con Machine Learning y Validación Cruzada

Trabajo Práctico 2 sobre Machine Learning, Validación Cruzada y Clasificación Binaria-Multiclase

12 de noviembre de 2024 — Laboratorio de Datos

Grupo DataBuddies

Integrante	LU	Correo electrónico
Carriedo, Eliseo	392/23	carriedo.eliseo@gmail.com
Fage, Lila	235/24	fagelila@gmail.com
Laurido, Julián	1097/23	Jlaurido@outlook.com

Facultad de Ciencias Exactas y Naturales
Universidad de Buenos Aires
Ciudad Universitaria - (Pabellón I/Planta Baja)
Intendente Güiraldes 2610 - C1428EGA
Ciudad Autónoma de Buenos Aires - Rep. Argentina
Tel/Fax: (++54 +11) 4576-3300
http://www.exactas.uba.ar

Introducción

En el presente informe se llevará a cabo un análisis del conjunto de datos **TMNIST**, el cual contiene información sobre imágenes de dígitos representados en diversas tipografías. Este dataset será utilizado para evaluar y entrenar modelos de clasificación de imágenes.

El objetivo principal de este informe es aplicar y analizar técnicas de **validación cruzada** para evaluar la eficacia de los modelos de clasificación de manera más robusta y, a su vez, evitar problemas de **sobreajuste** (overfitting) durante el entrenamiento.

El trabajo se divide en varias secciones. En primer lugar, realizamos un **análisis exploratorio** de los datos. Luego, pasamos a un **análisis de clasificación binaria**, seguido de un **análisis de clasificación multiclase**. Finalmente, presentamos las **conclusiones** basadas en los resultados obtenidos de ambos análisis.

Análisis exploratorio

Durante este proceso, descubrimos que muchas de las columnas, que representan cada píxel de las imágenes, no presentan variación alguna entre las diferentes imágenes. Es decir, estas columnas eran iguales para todos los caracteres, sin distinción entre las distintas fuentes tipográficas. Debido a esto, decidimos no tener en cuenta esas columnas a la hora de entrenar el modelo, ya que a mayor cantidad de datos irrelevantes, más complejo se vuelve el modelo y su ejecución.

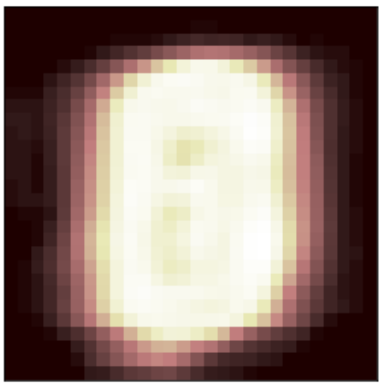

Figura 1 : combinación de variaciones de las imágenes.

En la figura 1 se observa la combinación de todas las imágenes del conjunto de datos. Las áreas de color oscuro indican zonas donde hay poca variación entre las imágenes, y se puede notar que en los bordes de las imágenes no existen atributos variables. Estos parámetros (pixeles) fueron los que decidimos dejar de lado.

Además, encontramos relaciones interesantes entre números. A simple vista, si graficamos la imagen promedio de cada número (**Figura 2**), observamos que son distinguibles. Luego, si deseamos analizar con más detalle la variación entre los atributos (píxeles) de dos números (**Figura 3**), como por ejemplo, entre los unos y cuatros, o entre los unos y ceros, podemos afirmar que hay clases que son más parecidas a unas que a otras. Entre los unos y ceros, se aprecia una mayor variación en sus atributos debido a que la forma del uno "rellena" la del cero; mientras que entre los unos y cuatros comparten muchos más atributos con valores similares. En los *violin-plot*, se muestra que las desviaciones estándar entre los cuatros y unos, se concentran más hacia el 0, lo que indicaría la relación ya mencionada.

Figura 2: Imagen promedio de cada número

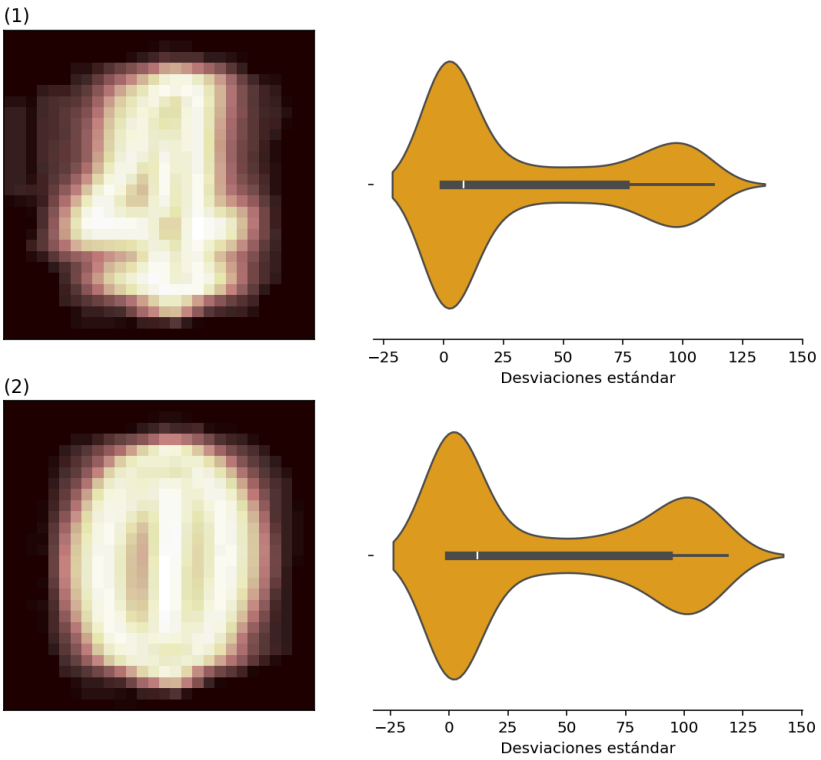

Figura 3: (1) Desviaciones estándar entre los píxeles los dígitos "4" y "1", *violin-plot* de las desviaciones estándar. (2) Desviaciones estándar entre los píxeles los dígitos "0" y "1", *violin-plot* de las desviaciones estándar

Por otro lado, nos resultó útil analizar la variación de un mismo número en diferentes tipografías. Observamos que, aunque la variación entre las representaciones de un mismo dígito no es muy grande, sí es lo suficientemente notoria como para que podamos generar un promedio que permita distinguir de qué número se trata. En las imágenes superiores de la **Figura 4** podemos ver el promedio de las representaciones, y en las inferiores, la variación entre las diferentes tipografías.

Figura 4: las imágenes superiores muestran el promedio de las representaciones, y las inferiores, la variación entre las diferentes tipografías.

En general, el estudio e identificación de imágenes suele ser más complejo que la predicción en un conjunto de datos más simple, como los que hemos trabajado anteriormente. Por un lado, estos conjuntos contienen menos información para el entrenamiento del modelo, y además, los parámetros eran fáciles de distinguir, como el sexo o la división entre adultos y niños (en el dataset Titanic). En cambio, en este dataset estamos trabajando con una cantidad mucho mayor de información, donde los parámetros son píxeles, lo cual es mucho más difícil de distinguir a simple vista, esencialmente imposible sin herramientas de visualización.

Análisis de clasificación binaria

Nos enfocaremos en el estudio de dos clases: la clase 0 y la clase 1. Luego de realizar la evaluación correcta en SQL, observamos que no solo la información de cada clase está balanceada, sino que **ambas clases están perfectamente equilibradas**, con 2990 datos en cada una.

Continuando con el entrenamiento del modelo binario, separamos los datos en conjuntos de entrenamiento (train) y prueba (test), y realizamos varias pruebas utilizando el método KNN. Primero entrenamos el modelo con tres atributos y cinco vecinos, y luego repetimos el proceso variando el número de atributos, entre 3 y 18. Es importante aclarar que como el número de vecinos es un hiperparámetro, no varía.

Las métricas obtenidas muestran una mejora en el desempeño del modelo a medida que aumentamos el número de atributos. Es interesante notar que los datos utilizados para entrenar el modelo también presentan un rendimiento significativamente superior. Para medir la performance de cada modelo, utilizamos el criterio de **exactitud**.

$$\text{Accuracy} = \frac{(TP + TN)}{(TP + FP + TN + FN)}$$

Figura 5: Fórmula binaria para la exactitud. Puede ser extrapolada a clasificación multiclase, con respectivos cambios.

Estas comparaciones pueden observarse debajo en las **Figuras 6 y 7**. Notemos que la exactitud utilizando siempre 3 atributos es muy variable. Esto se debe a que depende mucho de cuáles son los 3 atributos elegidos en cada iteración, podemos pensar que algunos píxeles sirven más para distinguir dos números que otros.

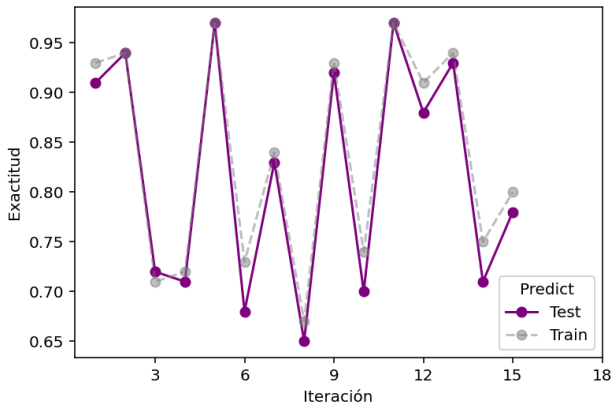

Figura 6: Evaluación de modelos con 3 atributos.

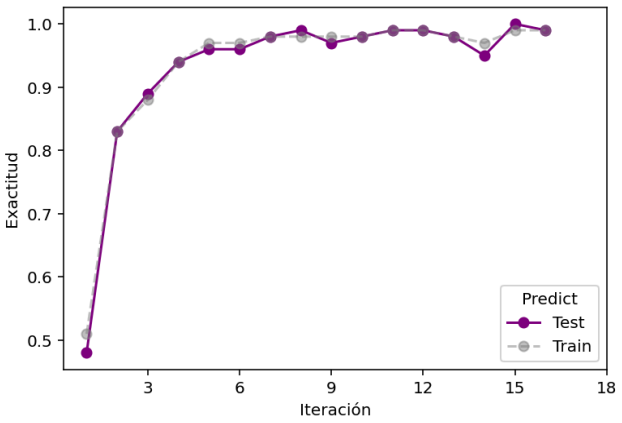

Figura 7: Evaluación de modelos con atributos variantes.

Otro aspecto clave fue el estudio y comparación de modelos con diferentes hiper parámetros.

En esta segunda fase, nos centramos en explorar la relación entre la cantidad de atributos y el número de vecinos (hiperparámetro). Observamos que la exactitud del modelo mejoraba a medida que aumentaba el número de atributos, alcanzando un valor cercano al máximo. Aunque el número de vecinos también juega un papel importante en la mejora de la exactitud, es evidente que, incluso con un número elevado de vecinos, si la cantidad de atributos es baja, el modelo no logrará un buen desempeño en términos de exactitud.

En la **Figura 8** podemos observar estas relaciones y sus métricas. Notemos que cuánto más amarilla sea la burbuja, mejor exactitud tiene el modelo al que representa.

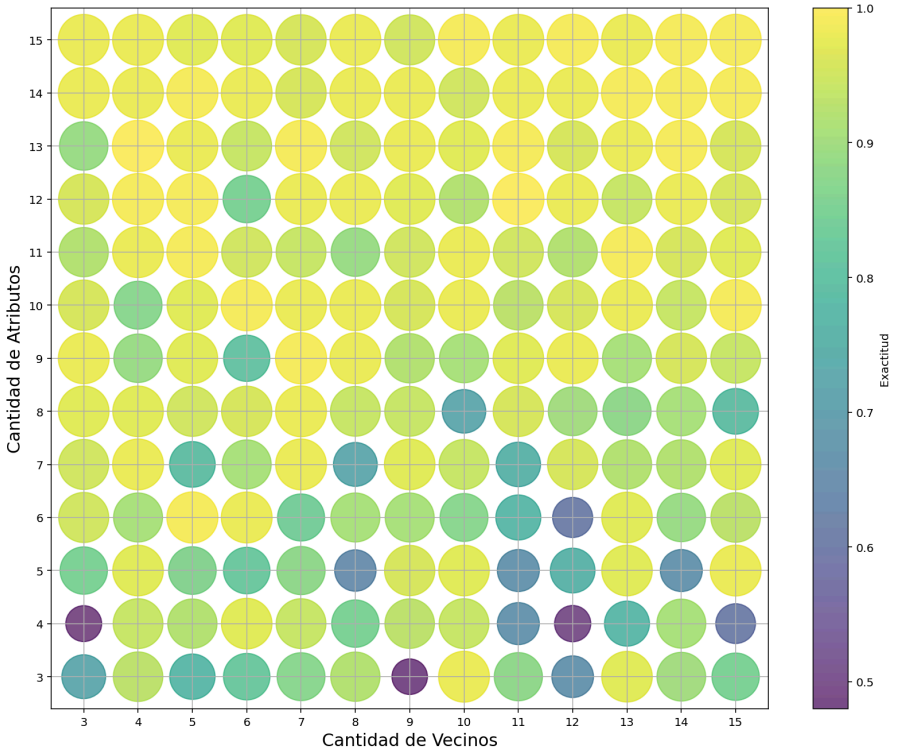

Figura 8: Bubble chart que relaciona cantidad de vecinos, cantidad de atributos y exactitud.

Análisis de clasificación multiclase

Para comenzar, planteamos un objetivo claro: entrenar un modelo que sea capaz de diferenciar imágenes y clasificarlas según el dígito que representan, en el rango de 0 a 9.

A continuación, procedimos, como en el inciso anterior, a dividir los datos disponibles. Primero, separamos un conjunto del **10%**, que dejamos apartado como conjunto **held-out**, esencial para la evaluación final del modelo. El 90% restante fue tratado como si fuera el conjunto completo de datos, es decir, como si fuese el 100%.

En este caso, utilizamos el **método de árboles de decisión** y aplicamos **validación cruzada con k-fold**.

- **Árbol de decisión:** Ajustamos un modelo utilizando diferentes valores de profundidad, desde 1 hasta 10, y evaluamos su exactitud. En la **Figura 9** se muestra su comportamiento. Como se observa, la exactitud mejora a medida que aumenta la profundidad del árbol, sin llegar a producir overfitting.

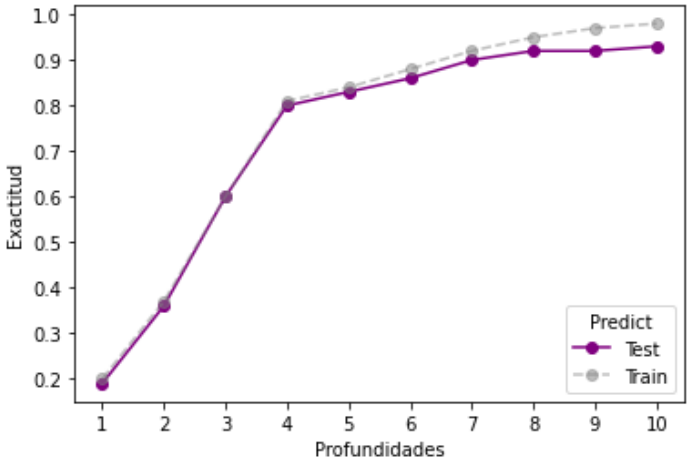

Figura 9: Relación entre profundidades del árbol y exactitud.

- **K-Fold cross-validation**: En esta sección, nos enfocamos en comparar diferentes árboles de decisión con profundidades que varían de 1 a 10, utilizando el método de **validación cruzada** con **K-Fold**. Consideramos dos medidas de impureza clave: **entropía** y **Gini-gain**, para evaluar la calidad de las divisiones en los árboles.

Decidimos variar los siguientes hiperparámetros: profundidad del árbol (valores entre 1 y 10); mínimo de muestras por nodo (variando según 2^k, con k entre 1 y 5); número de atributos a considerar (variando entre 5 y 10).

Además, generamos 5 instancias de **K-Fold**, que utilizamos para promediar la **performance** de cada modelo y obtener una evaluación más robusta. En la Figuras 8 se pueden observar las tendencias correspondientes a cada variación.

En la **Figura 10.1**, se muestra cómo los datos de entrenamiento (train) son predichos perfectamente, mientras que las predicciones para los datos de prueba (test) mejoran a medida que aumentamos significativamente la cantidad de atributos máximos. Sin embargo, luego de superar los 40 atributos, vemos que la exactitud se “estanca” en 0.925.

Por el contrario, en la **Figura 10.2**, la exactitud supera el umbral de 0.9. Aunque la perfección en las predicciones sobre los datos de entrenamiento sigue siendo evidente, se aprecia un aumento significativo en la exactitud de las predicciones para los datos de prueba. Es decir, a medida que la profundidad del árbol aumenta, la exactitud también se eleva. Es importante destacar que este crecimiento en la exactitud se vuelve cada vez más leve, lo que sugiere que aumentar la profundidad de manera innecesaria no beneficiaría al modelo, sino que podría llevar a un **overfitting**.

También es interesante estudiar cómo varía la exactitud en función del mínimo de ejemplares para *splitear* por nodo. En este caso, en la **Figura 10.3** observamos que la exactitud disminuye a medida que aumenta el número mínimo de ejemplares. Esto es lógico, ya que un valor muy alto de ejemplares (por ejemplo, 32) puede restringir posibles

splits que mejoren la exactitud del modelo. Para evitar este problema, en este caso es recomendable utilizar un valor mínimo de ejemplares menor o igual a 4. Como se observa en el gráfico, después de estos valores se produce una pérdida de exactitud, no sólo en los test, sino que también en los train. Lo curioso de este caso es que las predicciones sobre los datos de entrenamiento alcanzan su pico de exactitud con un valor de 2 ejemplares.

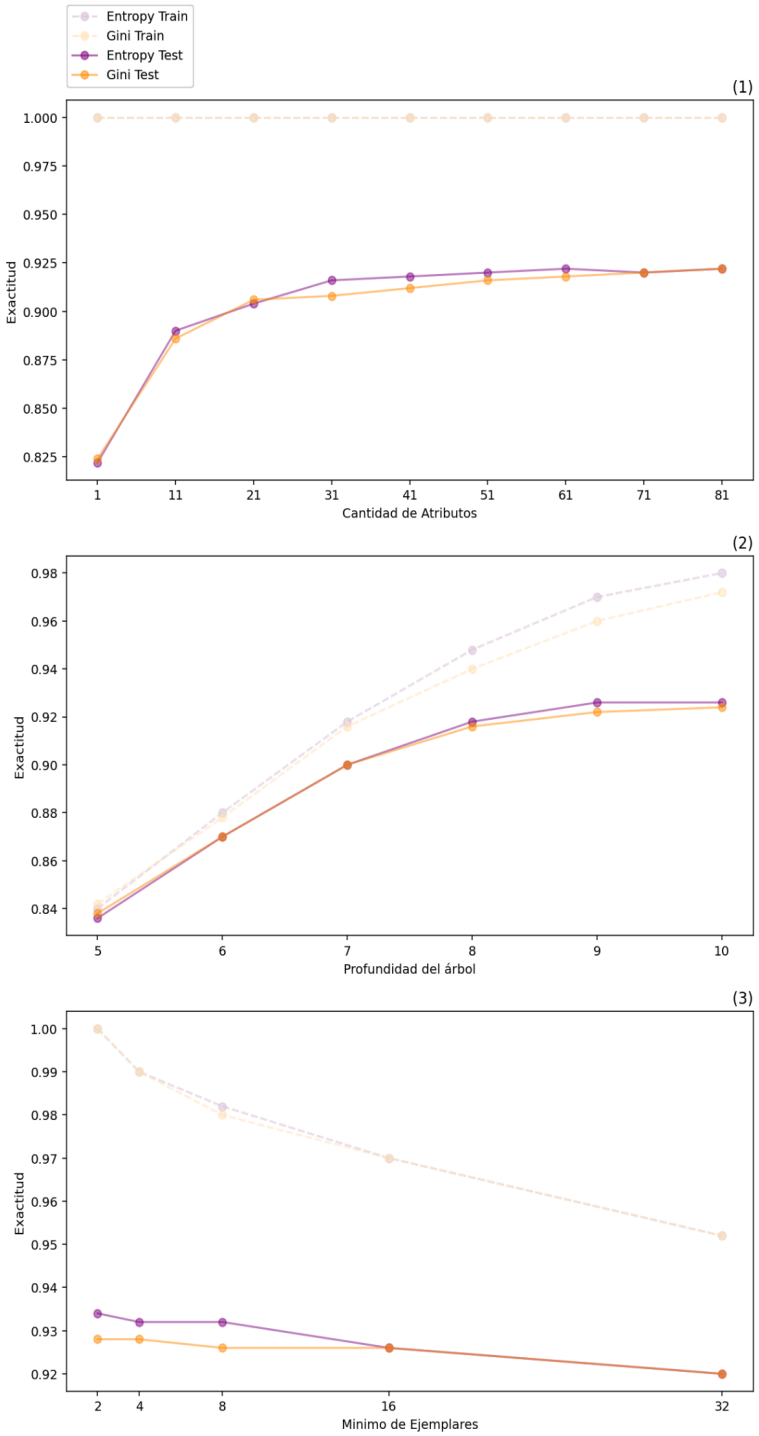

Figura 10: (1) Variación de la exactitud a mayor cantidad de atributos. (2) Variación de la exactitud a mayor profundidad del árbol. (3) Variación de la exactitud a mayor cantidad de mínima de ejemplares para realizar un *split*.

Luego de realizar las pruebas anteriores, decidimos que nuestro modelo final iba a utilizar el criterio ***entropy***, con un máximo de **50** atributos, una profundidad máxima de **10** nodos para el árbol predictivo, y una cantidad mínima de **2** atributos (default) para realizar un *split* en el mismo.

Resultados

Antes de exhibir los resultados, debemos ver que cantidad de muestras contenemos en nuestro *Held-Out* de cada clase. A continuación en la **Figura 11**, vemos una comparativa entre los datos separados. Podemos notar que la cantidad de datos de cada clase no están perfectamente equilibradas, sin embargo no hay una gran diferencia entre ellas.

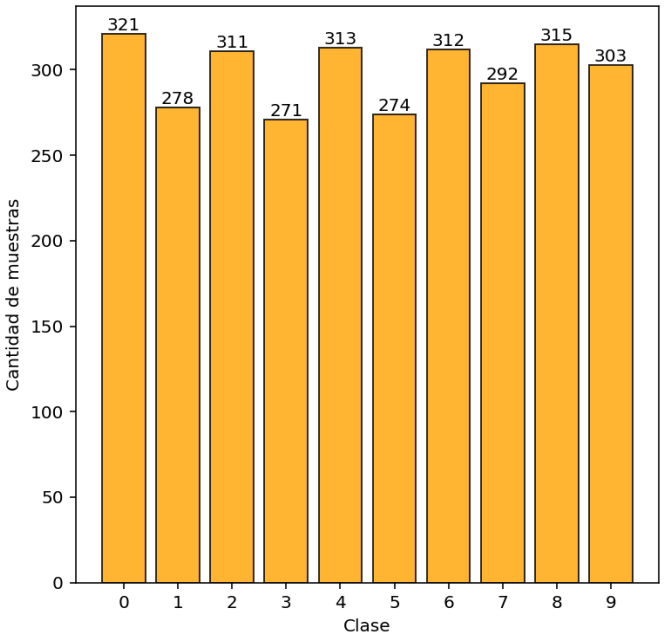

Figura 11: Conteo de muestras por clase en el *Held-Out*.

En la **Figura 12** se pueden observar las métricas obtenidas después de ejecutar el conjunto *Held-Out*. A simple vista, se destaca el buen rendimiento de nuestro modelo, el cual logró un **92%** de *accuracy*, aunque también se pueden identificar algunos errores pequeños, generalmente causados por clases tipograficamente similares (como el *1* y *7*).

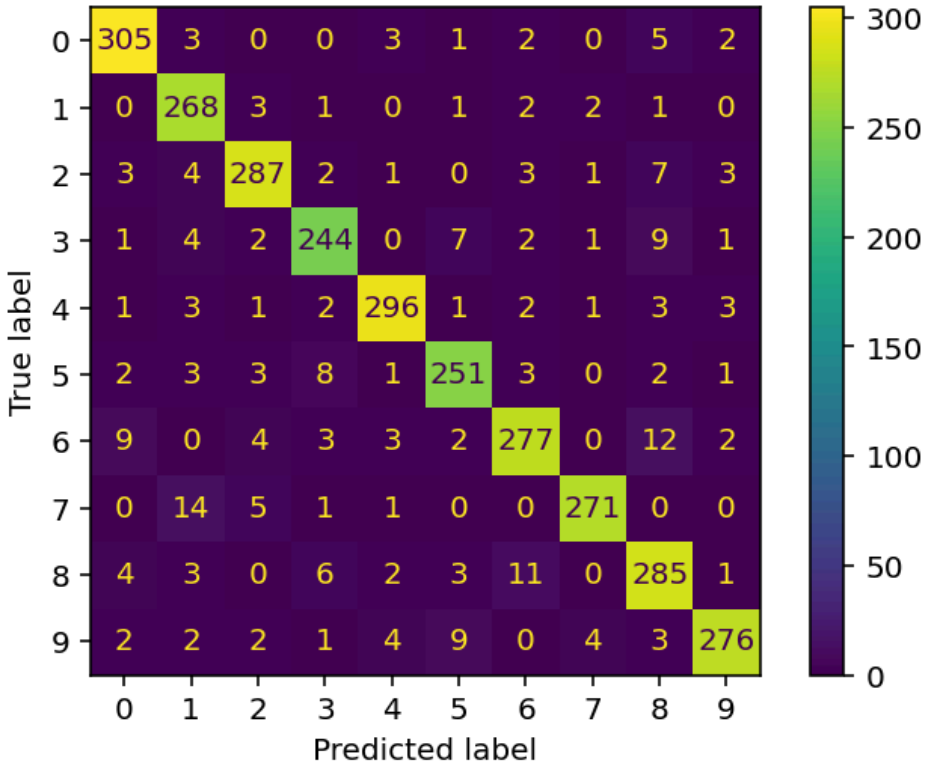

Figura 12: Matriz de confusión, conformada por la ejecución del conjunto *Held-Out*.

Otro resultado interesante de analizar, es que la *accuracy* obtenida de predecir los datos de desarrollo (con los que entrenamos el modelo final) fue de **96%**, significativamente mayor a la del *Held-Out*.

Conclusiones

El estudio demuestra que, para lograr un buen desempeño en la predicción de datos, es fundamental contar con características adecuadas en el conjunto de datos utilizado para entrenar el modelo. Un paso crucial es eliminar aquellos datos que no aportan información relevante para la predicción, ya que podrían introducir errores durante el entrenamiento y afectar la precisión del modelo.

Además, es importante mantener un equilibrio en las clases de datos, ya que un desequilibrio podría llevar a un modelo sesgado, que prediga con mayor acierto ciertas clases y menos eficientemente otras. En este caso, el conjunto de datos estaba equilibrado, con la misma cantidad de ejemplos para cada dígito, lo que permitió entrenar un modelo más robusto y generalizable.

El entrenamiento de un modelo también requiere atención al elegir los hiperparámetros adecuados. Como se demuestra en el análisis, tomar decisiones incorrectas puede provocar tanto subajuste como sobreajuste (*min_samples_split*, *max_depth*, *max_features*), lo que a su vez, genera un mal rendimiento. Para evitar estos problemas, se utilizan técnicas de validación cruzada, como las implementadas en este

proyecto, que proporcionan métricas útiles para seleccionar el modelo más adecuado según el objetivo que se desea alcanzar.

El modelo final seleccionado (árbol de decisión con criterio de entropía y un valor y cantidad adecuada de hiperparámetros) logró un rendimiento equilibrado, alcanzando una alta exactitud sin caer en el sobreajuste. Además, se concluye que, en general, la clasificación de imágenes es una tarea más compleja que otros tipos de clasificación, lo que exige una mayor sofisticación en los modelos y en las técnicas de validación utilizadas.