Clasificación y Selección de Modelos

Aaron Cuellar, Sebastián Ceffalotti, Rodrigo Coppa

GrupoC

Laboratorio de Datos, Verano 2025
Departamento de Computación, FCEyN, UBA

1. Introducción

En este informe se analiza la clasificación de imágenes del dataset MNIST-C en su versión “Fog”, un conjunto de datos basado en dígitos escritos a mano con ruido agregado Se realizaron experimentos para clasificación binaria (dígitos 0 y 1) y multiclase (dígitos del 0 al 9) utilizando modelos de KNN y árboles de decisión. Se llevó a cabo un análisis exploratorio para evaluar las características del dataset, seleccionando atributos relevantes y visualizando datos mediante gráficos. Para la clasificación binaria, se evaluó el rendimiento de KNN con distintas configuraciones de atributos y valores de k. En la clasificación multiclase, se entrenaron modelos de árboles de decisión con diferentes profundidades y se aplicó validación cruzada para optimizar el desempeño. Finalmente, se compararon los resultados obtenidos y se presentaron conclusiones sobre el rendimiento de los modelos implementados.

2. Desarrollo

2.1 Análisis Exploratorio de Datos

El dataset de mnist-c consiste en datos de imágenes de números escritos a mano, las imagenes son de 28x28 pixeles, y estan en escala de grises (siendo 255 color blanco y 0 el color negro). En el dataset hay 786 columnas, en la primera columna hay un índice para identificar cada imagen, luego hay 784 columnas en las cuales se guarda la intensidad de cada píxel, y por último una columna con la etiqueta del número, que sirve para identificar cual es el número escrito. Los atributos relevantes para predecir el dígito son los de intensidad, el índice y la etiqueta no son útiles para predecir y pueden descartarse.
Se calculó la intensidad promedio de cada píxel para cada dígito, y el promedio para todos los dígitos. Los bordes mostraron valores mas cercanos a 0 (negro), mientras que las zonas centrales presentaron variaciones pero con mayor intensidad. Esto sugiere que los píxeles de los bordes (aproximadamente el 58 % para la suma de todos los dígitos en la figura 1) no son tan relevantes, ya que no aportan información útil para la clasificación. Se puede también ver que los píxeles más amarilllos tienen mayor intensidad y son los que brindan información sobre el dígito (ver figura 2).

***Figura 1**: Pixeles menos (Negro) y más relevantes (Blanco), para la imagen promedio de todos los digitos. Se definen como menos relevantes aquellos con intensidad menor a 100.*

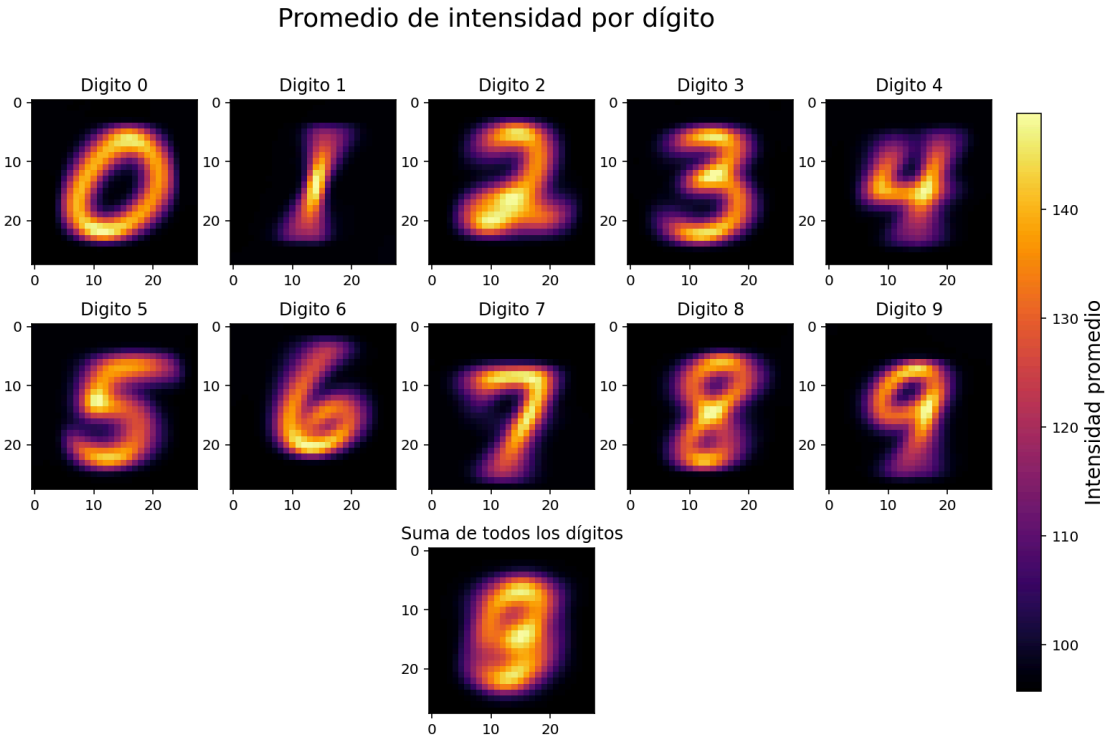

Figura 2: *Intensidad promedio de cada píxel para cada dígito y el promedio de la suma de todos los dígitos.*

Hay números más parecidos entre sí, por ejemplo se ve que el 9 se parece al 4, así como el 8 al 3, mientras que números como el 0 y el 1 son muy distintos. Para cuantificar el parecido se decidió medir las distancias entre las imágenes promedio de cada dígito, la distancia es la euclídea, pensando a cada imagen como un vector de $\mathbb{R}^{784}$ (una dimensión por pixel). Se calculó la distancia entre cada par de dígitos, y se puede ver en la matriz de la figura 3.

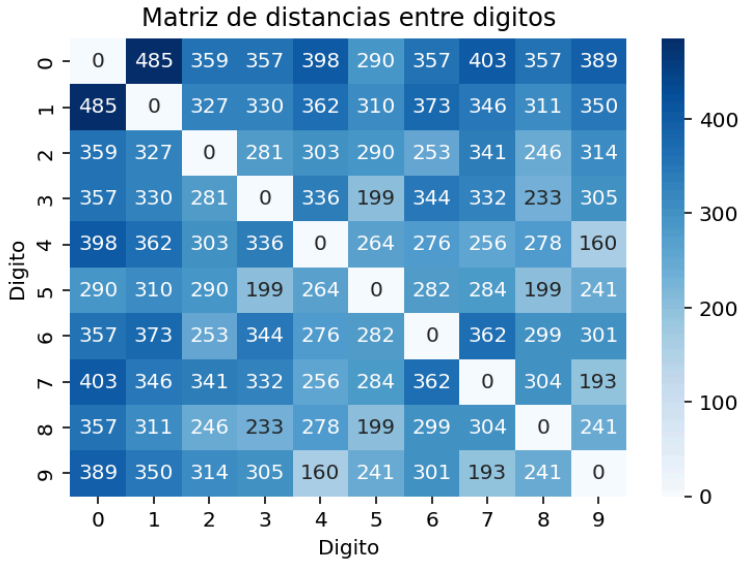

Figura 3: *Matriz de distancias entre las imágenes promedio de cada dígito*

De aquí se puede ver que los números con mayor distancia se parecen menos (como el 0 con el 1) y los de menores distancias se parecen más (como el 4 con el 9).

Tomamos el dígito 0 y elegimos 15 muestras al azar para observar las diferentes imágenes de una misma clase. Se puede observar que hay mucha variación entre imágenes (ver figura 4) pese a ser de la misma categoría. Para otros casos, como con la clase del 7 se apreció que difieren bastante al punto de confundirse con otras clases (como con la del 1).

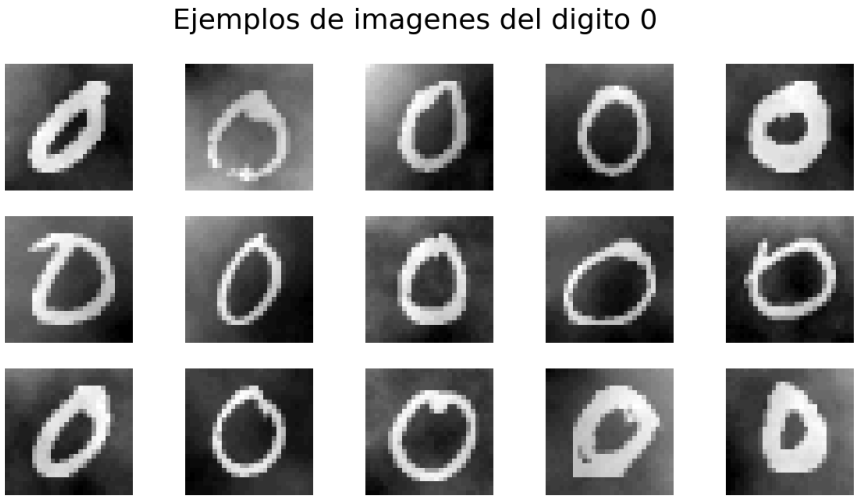

Figura 4: *15 imágenes elegidas aleatoriamente del conjunto de datos.*

El trabajar con imágenes como datos presenta diferencias importantes respecto a datasets como el de Titanic. Algunas complicaciones:

- Una imagen es una lista de intensidades para cada píxel, no hay atributos interpretables como en el dataset de titanic (por ejemplo edad, sexo y clase).
- Para entender la información de los atributos es necesario graficar la imagen, algo más complejo.

2.2 Clasificación Binaria (Dígitos 0 y 1)

Se creó un nuevo dataframe, el cual solo incluye los valores de ceros y unos. De allí se pudo ver que la cantidad de ceros es de 6903 y la cantidad de unos es de 7877, por lo que no está perfectamente balanceada. Entonces, primero hacemos un balanceo y luego separamos los datos en dos conjuntos. Uno TRAIN, del cual usamos el 80% y el resto lo usamos para TEST. Debido a que el balance de clases es bastante bueno usamos como métrica la accuracy para evaluar los modelos.

Para ver cómo entrenar el modelo, probamos tres métodos distintos. En principio elegimos píxeles que nos parecieran relevantes del análisis de las imágenes promedio (ver figura 5). Para entrenar el modelo KNN probamos utilizando 1, 3 y 14 píxeles.

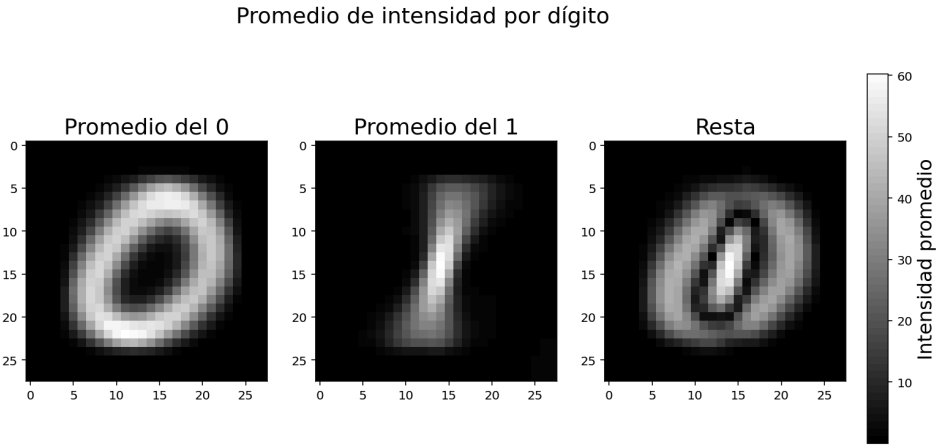

Figura 5: *Promedio de las imágenes del 0, 1 y su resta*

Entrenamos los modelos variando los k-vecinos desde 1 hasta 25, primero utilizamos un solo píxel central el cual no da información relevante ya que la exactitud da cerca del 60 % y es casi como elegir al azar. Luego, se puede ver que usando 3 píxeles, el central y dos en donde están los máximos de la imagen promedio del cero, da una exactitud más alta. Por último se usaron los 14 píxeles de la fila central dando una exactitud cercana a 1 (ver figura 6).

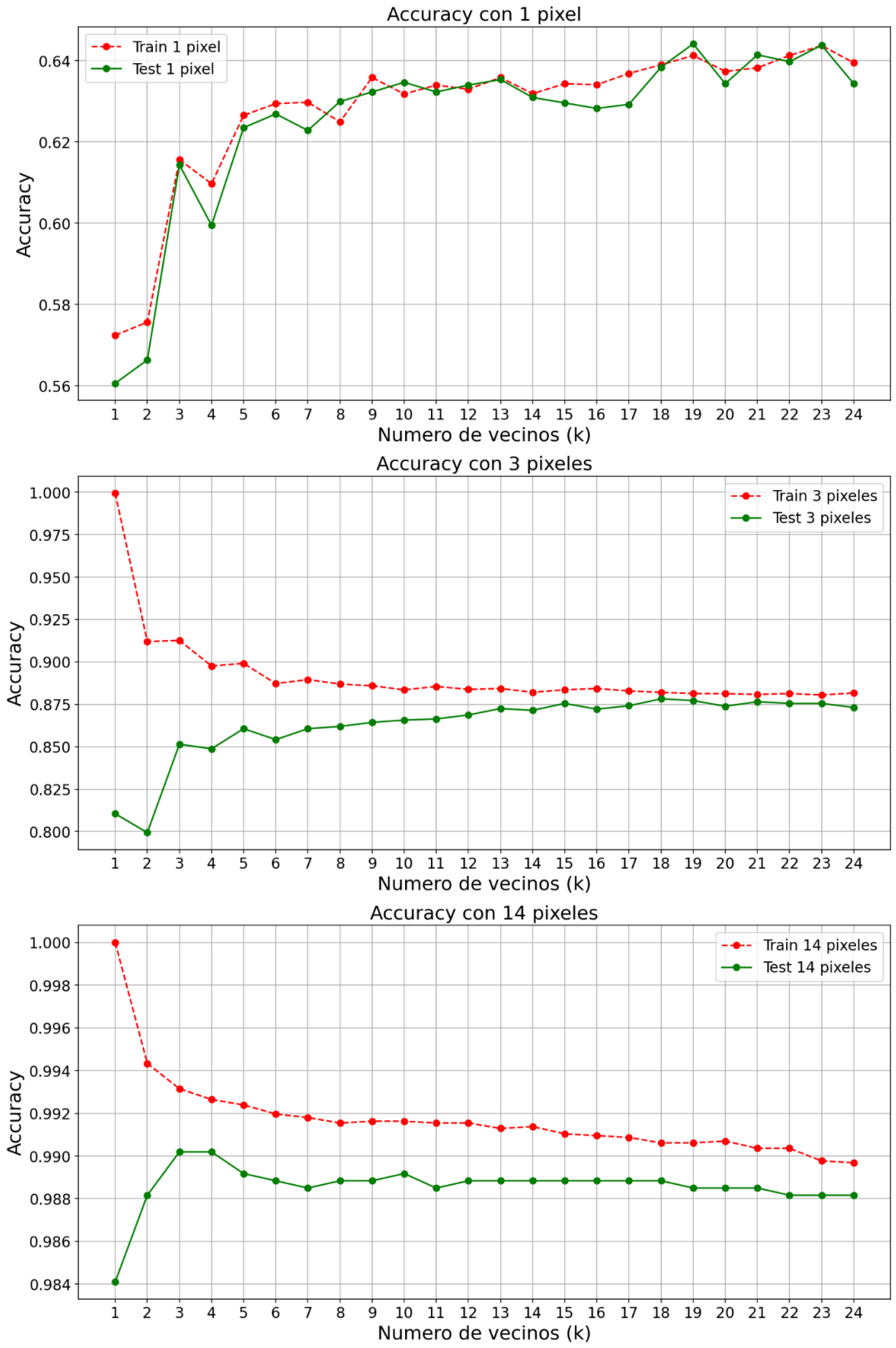

Figura 6: *Valor de accuracy para modelos entrenados usando 1, 3 o 14 pixeles elegidos arbitrariamente.*

Para k = 5 las curvas de train y test de 1 píxel se estabilizan, mientras que para 3 y 14 pixeles lo hacen en k = 3.

En segundo lugar se decidió utilizar para entrenar el modelo las distancias de cada imagen a la imagen promedio del cero y del uno (figura 7). Se puede ver, para distinto número de vecinos el valor de accuracy. Se obtuvo un buen accuracy, pero es necesario preprocesar los datos.

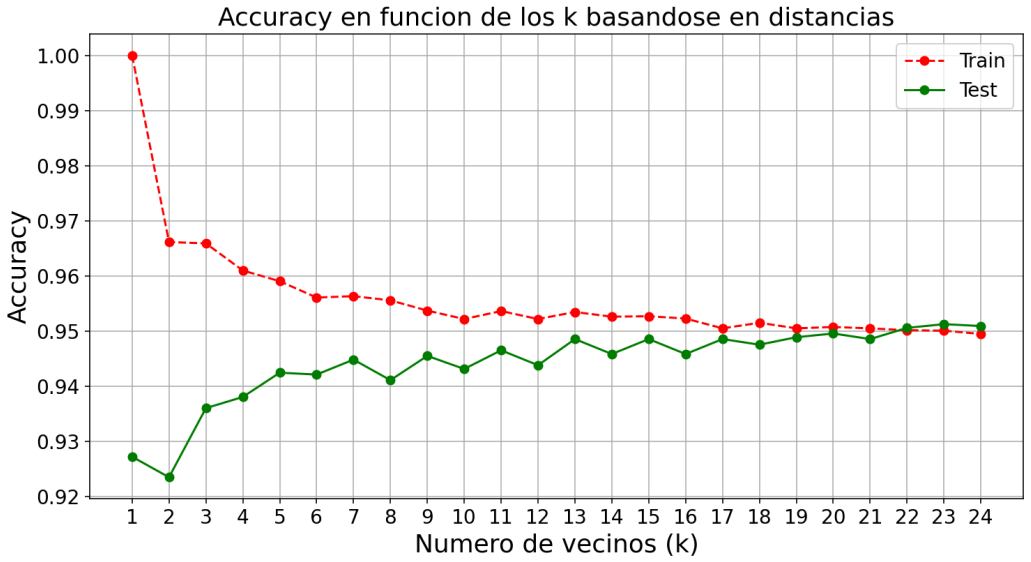

Figura 7: *Valor de accuracy para un modelo entrenado usando las distancias al dígito promedio 0 o 1.*

Se puede ver que para k = 5 las curvas de train y test se estabilizan, por lo que este sería un buen modelo a utilizar.

Por último se entrenó un árbol de decisión para obtener cuales son los píxeles más relevantes para identificar si el dígito es un 0 o un 1, y se entrenó un modelo KNN con los 3 píxeles más importantes, en la figura 8 se pueden ver los píxeles más importantes según su ubicación. Los resultados de las métricas son mejores a comparación de elegir los píxeles arbitrariamente, estos valores son de Accuracy del 95%, Recall del 98% y Precisión del 94% (se puede ver el valor de Accuracy en la figura 9).

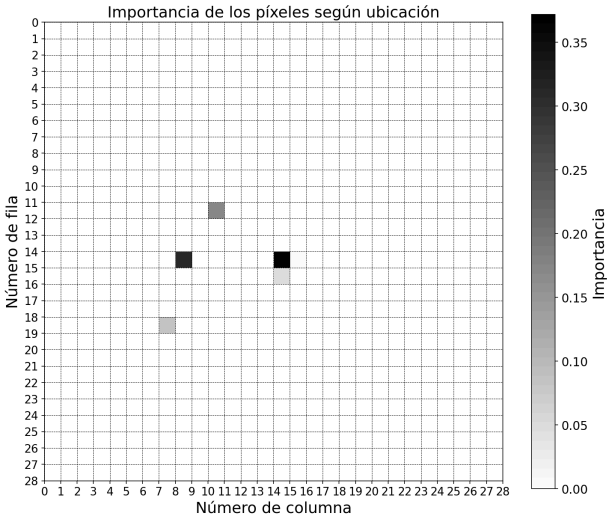

Figura 8: *Importancia de los píxeles según su ubicación para los dígitos 0 y 1.*

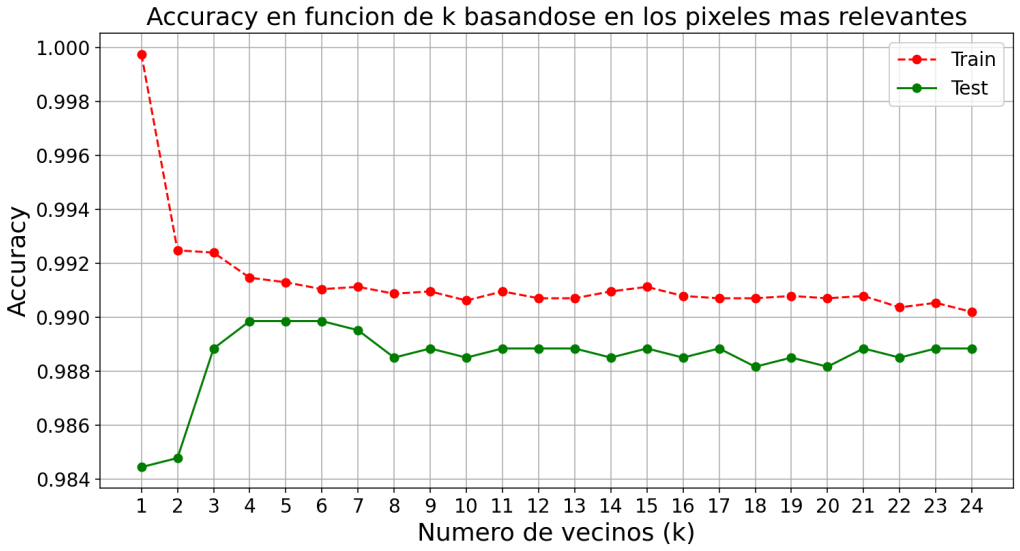

Figura 9: *Valor de accuracy para un modelo entrenado usando los 3 píxeles más relevantes según un modelo de árbol de decisión, en función del número de vecinos.*

Se puede observar que para k = 4 hay una buena relación entre el parecido de train y test y además se estabilizan las curvas, será, por lo tanto, el modelo elegido para responder la pregunta del enunciado.

2.3 Clasificación Multiclase (Dígitos 0 al 9)

Para esta parte primero se separó el conjunto de datos en desarrollo (dev) y en validación (held-out), además de definir los parámetros de entrenamiento, como las alturas del árbol que se van a usar y el número de folds (se decidió usar 3 debido al tamaño del dataset). Luego, se entrenaron los árboles usando como criterio Gini, para profundidades del 1 al 10, obteniéndose la accuracy como se muestra en la figura 10. También se entrenaron árboles pero esta vez utilizando el criterio de entropía, obteniendo las métricas de la figura 11.

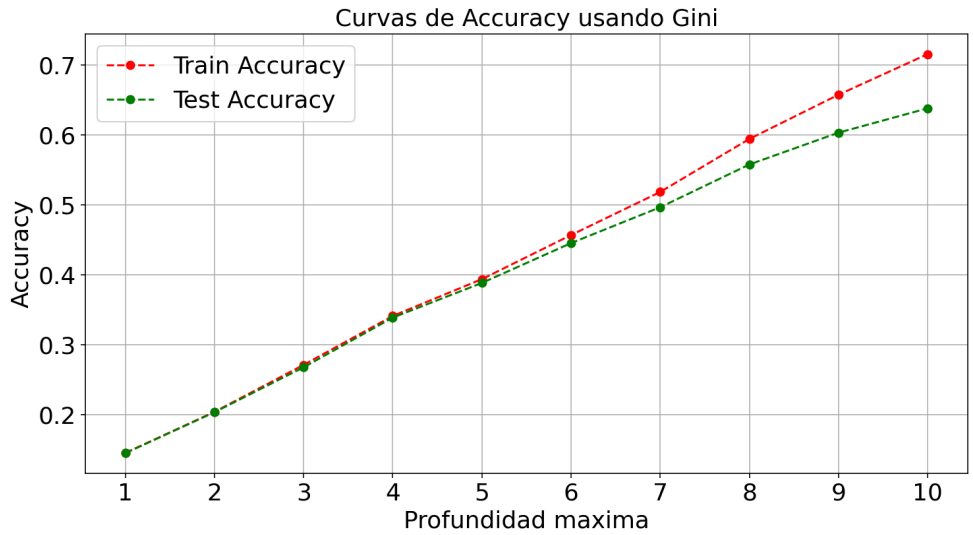

Figura 10: *Accuracy en train y test, a partir de diferentes profundidades del Árbol de Decisión usando el criterio de Gini.*

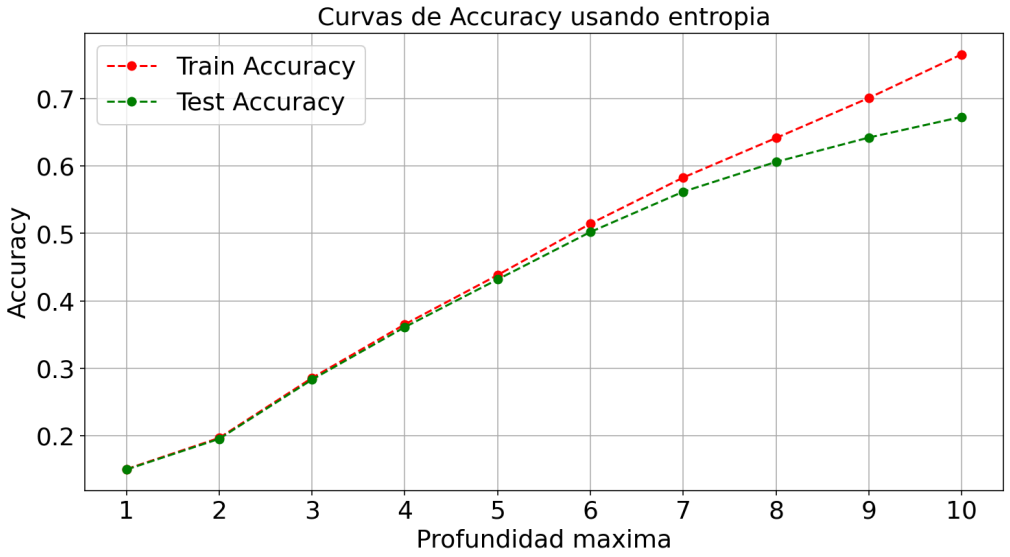

Figura 11: *Accuracy en train y test, a partir de diferentes profundidades del Árbol de Decisión usando el criterio de entropía.*

Se puede ver que para el valor 7 de la profundidad máxima del árbol de decisión, empiezan a divergir las curvas de train y test indicando que el árbol está empezando a sobreajustar. Por ello se entrenó finalmente el modelo usando como hiperparametros una profundidad máxima de 6 y el criterio de entropía ya que reportó un mejor valor. El entrenamiento se hizo con todos los datos de dev, y se evaluó en los datos de Held Out, obteniendo la matriz de confusión de la figura 12.

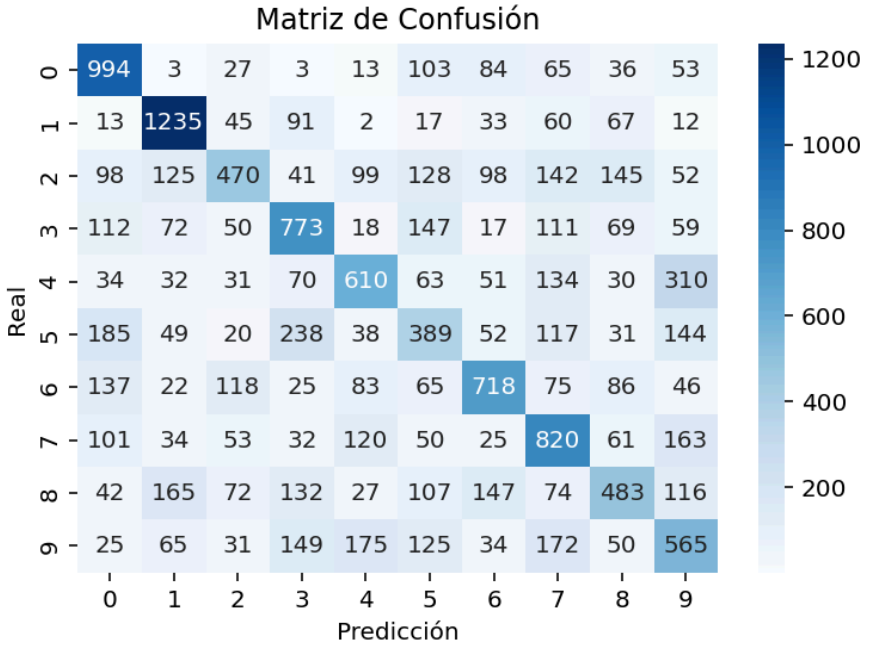

Figura 12: *Matriz de confusión para el modelo evaluado en Held out.*

En la matriz de confusión, los valores en la diagonal representan los aciertos del modelo. Se observa que, por ejemplo, entre los dígitos 4 y 9 hay una alta cantidad de falsos negativos, algo esperado según el análisis previo. Además, el dígito 5 es el peor clasificado, presentando tanto muchos falsos negativos como falsos positivos.

Por último, en la tabla 1 se muestran los valores de las métricas para el modelo utilizado con Held Out.

Métrica	***Valor***
Accuracy	*0.504*
Recall	*0.498*
Precisión	*0.497*

Tabla 1: *Valor de las Métricas para el modelo evaluado en Held Out para el árbol entrenado con una profundidad máxima de 6 y criterio de entropía.*

Al analizar estos resultados, se concluye que al considerar todas las clases (y no solo el 0 y el 1), los valores de Accuracy, Recall y Precisión disminuyen. Esto se debe a que ciertos dígitos, como el 4 y el 9 o el 3 y el 5, tienen trazos similares, lo que aumenta los errores de predicción. Aunque la accuracy general no es alta y hay cierto desbalance en las predicciones, el modelo logró clasificar bien algunos dígitos como el 0 y el 1, pero falló significativamente en otros, como el 5, además se puede decir que no sobreajustó ya que la evaluación en held-out fue muy similar a en dev y test.

3. Conclusiones

Se pudo concluir que no todos los píxeles son igualmente relevantes para la clasificación a través del análisis exploratorio y el uso de modelos de árboles de decisión, además se identificaron regiones clave en las imágenes para diferenciar los dígitos.

Para la clasificación binaria utilizando pocos píxeles estratégicos se pudo obtener una buena precisión en la clasificación. El mejor modelo se obtuvo usando KNN, este se obtuvo usando los píxeles más importantes según un árbol de decisión, alcanzando un accuracy alto sin estar sobreajustando, esto para un número de vecinos de k igual a 4.

Para la clasificación multiclase se encontró que los árboles de decisión con profundidad máxima de 6 poseen el mejor balance entre sobreajustar y generalizar los resultados, en particular se vieron resultados mejores usando el criterio de entropía en lugar de Gini. Del modelo evaluado en los datos de held out la matriz de confusión reveló que algunos dígitos, como el 5, son más difíciles de clasificar debido a similitudes con otros números. La exactitud general del modelo es baja por lo que se podría plantear otro modelo que logre mejores resultados para la clasificación.