Trabajo Práctico

Laboratorio de Datos

Facultad de Ciencias Exactas y Naturales

Universidad de Buenos Aires

20/11/24

Breve introducción

Para este proyecto se proponemos construir un modelo de clasificación basado en árboles de decisión para lograr identificar qué dígito númerico (3, 4, 6, 8 o 9) corresponde a una imágen escrita a mano. El objetivo es analizar y encontrar cuáles son las características de interés y evaluar el rendimiento del modelo para establecer su capacidad de generalización.

El conjunto de datos utilizado es el MNIST-C (Motion Blur), una versión corrompida del dataset MNIST, que contiene imágenes de dígitos escritos a mano representadas en escala de grises con una resolución de 28x28 píxeles. Cada imagen corresponde a un dígito entre 0 y 9, y los valores de los píxeles oscilan entre 0 y 255. Este dataset contiene un alto nivel de ruido, lo que lo convierte en un desafío adicional para los modelos de clasificación.

Exploramos representaciones visuales, como mapas de calor que reperesentan el valor promedio en cada atributo de todas las imágenes, y luego agrupando por dígito, para analizar patrones y diferencias entre clases que puedan ser útiles en la construcción del modelo.

Para desarrollar el modelo, dividimos los datos en dos subconjuntos: desarrollo (80 % del total), utilizado para entrenar y ajustar el modelo, y validación (20 % restante), empleado para evaluar su rendimiento.

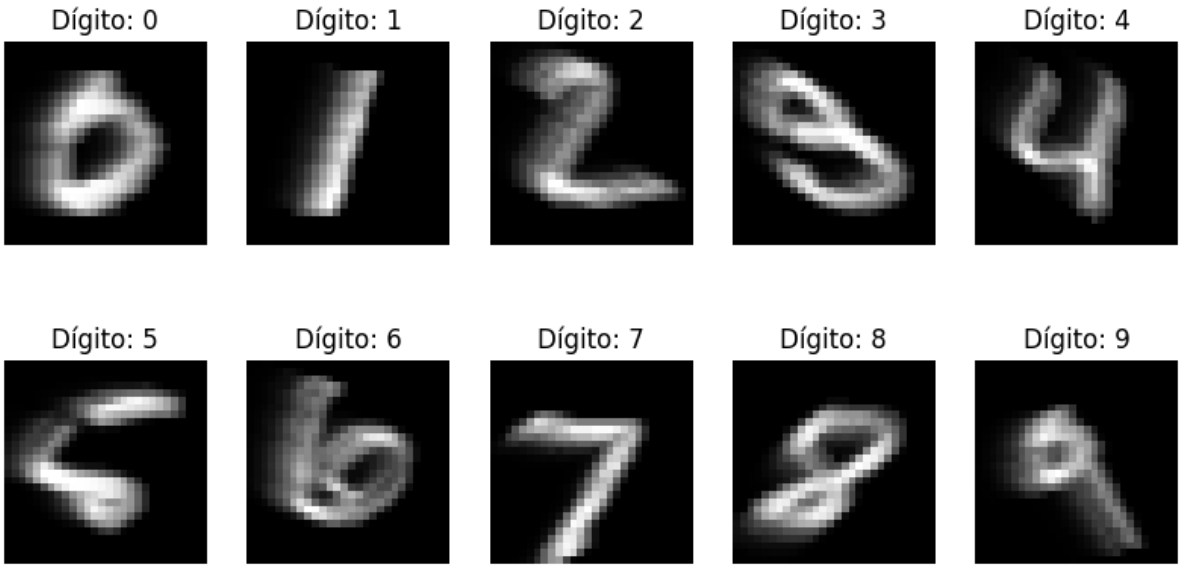
0.1 Analisis Exploratorio

Comenzamos determinando algunas características cuantitativas y cualitativas del dataset.

- Cantidad de datos: 10000
- Dimensiones de cada imagen: 28x28
- Cantidad de clases (dígitos): 10
- Clases de dígitos: [0 1 2 3 4 5 6 7 8 9]
- Frecuencia de cada clase (dígito): 0: 980, 1: 1135, 2: 1032, 3: 1010, 4: 982, 5: 892, 6: 958, 7: 1028, 8: 974, 9: 1009
- Rango de valores en las imágenes: 0 a 255
- No hay imágenes repetidas en el dataset

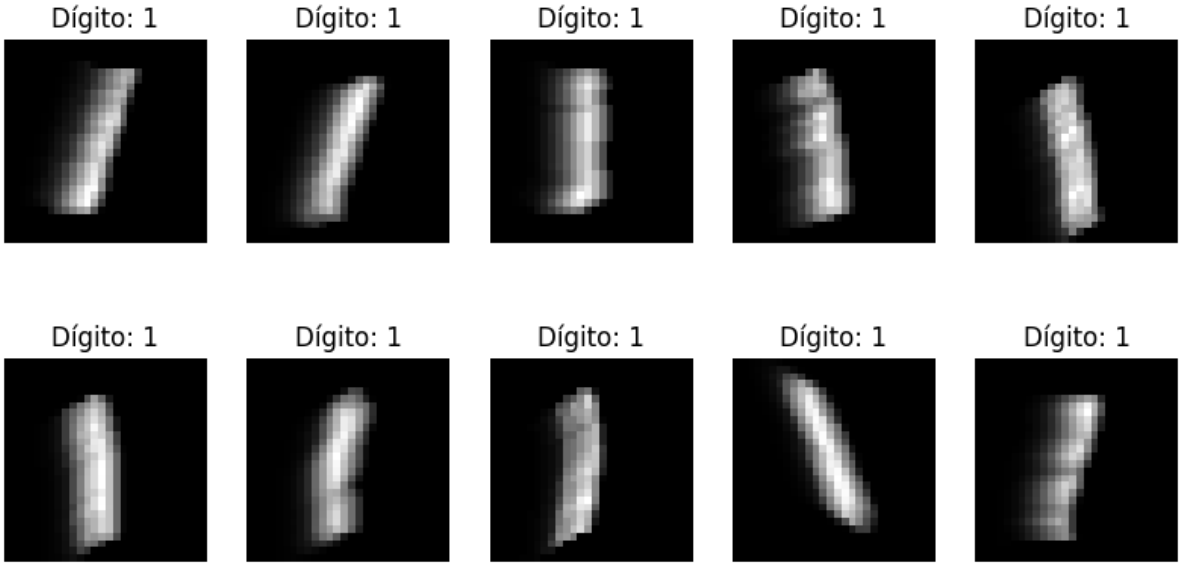
1. ¿Cuáles parecen ser los atributos (i.e., píxeles) más relevantes para predecir el dígito al que corresponde la imagen? ¿Cuáles no? ¿Creen que se pueden descartar atributos?

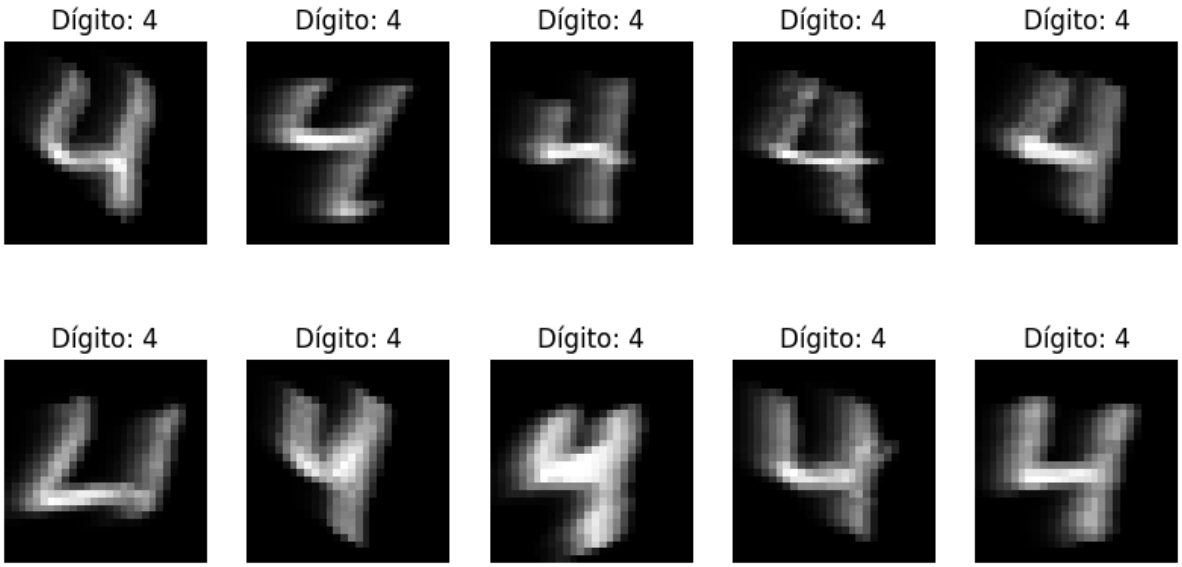
   Comenzamos con una visión general de distintos números para entender el dataframe con el que estamos trabajando.



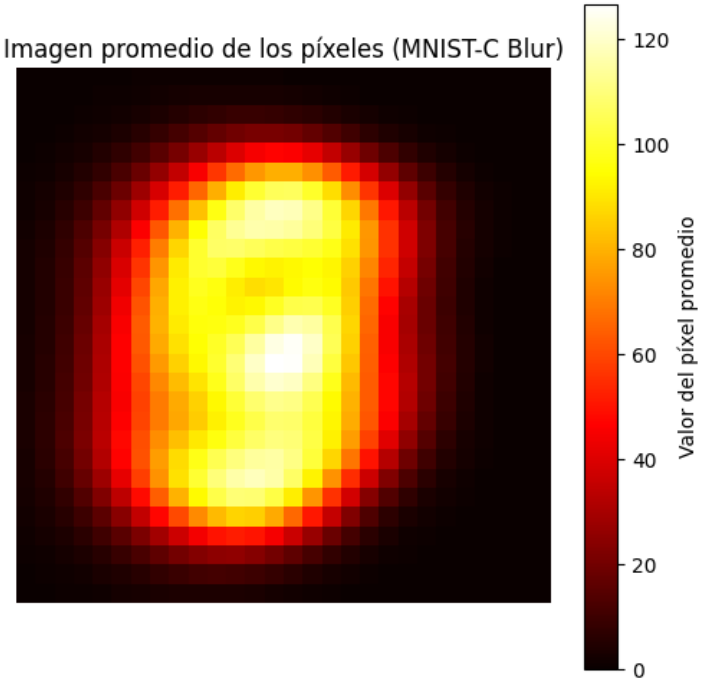
   Vemos como en las imágenes de distintos números los datos importantes para determinar el digito varian mucho, e incluso si lo vemos para un ejemplo, no queda claro que pixeles nos son de utilidad y cuales no. Ya que a cada número en particular se le puede recortar un margen distinto de pixeles negros.

   Miramos distintas imágenes del mismo número para ver en lineas generales su distribución.



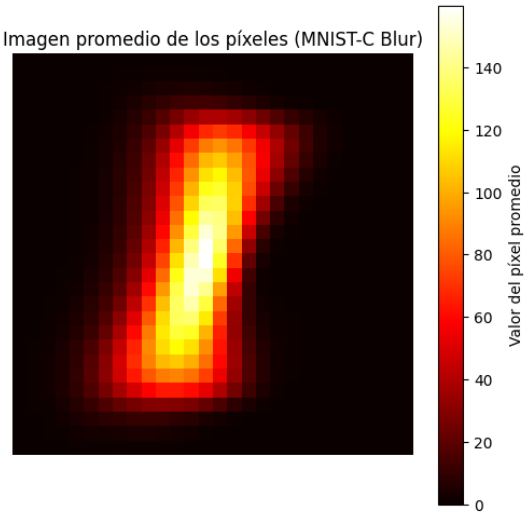


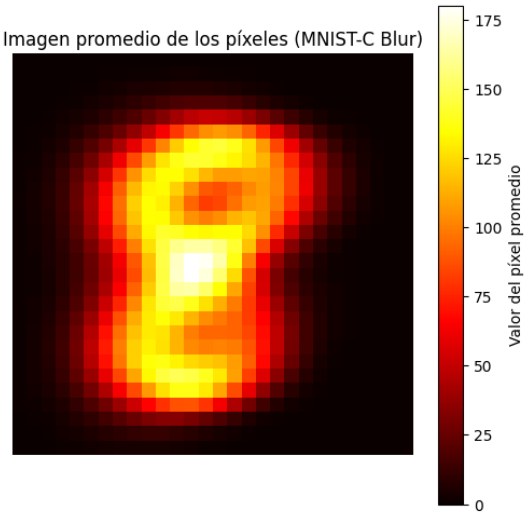
En estos dos ejemplos podemos ver que aunque sean similares, los pixeles de relevancia para determinar el numero difieren mucho entre si, y en un mismo numero, por lo tanto queriamos ver en lineas generales cuales son los pixeles mas y menos usados dentro de todas las imagenes del archivo, para eso generamos una imagen de calor de el promedio de los valores de los pixeles de todas las imagenes.



Al analizar en promedio todas las imagenes, podemos ver que hay claramente un conjunto de pixeles que se usan más que otros, los pixeles de los costados y los superiores pueden ser mas descartables a la hora de analizar un conjunto grande de imagenes, aunque en la practica no ayuda mucho a predecir el digito de una imagen. Vemos que para cada número podriamos hacer un mapa de calor de todas

las imagenes para determinar cuales pixeles son importantes para cada uno y así podriamos predecir con mas exactitud cada número. Por ejemplo:





2. II. ¿Hay dígitos que son parecidos entre sí? Por ejemplo, ¿qué es más fácil de diferenciar: las imágenes correspondientes a los dígitos 0 y 1, ó las imágenes de 5 y 6?

Viendo por encima las imagenes que obtuvimos en el primer inciso, podemos observar que hay algunos numeros que pueden llegar a tener imagenes muy similares y que a la hora de distingirlas una a una se puede hacer muy dificil el trabajo, por ejemplo, teniendo las siguientes imagenes solas, es muy dificil clasificar a que numero pertenece:

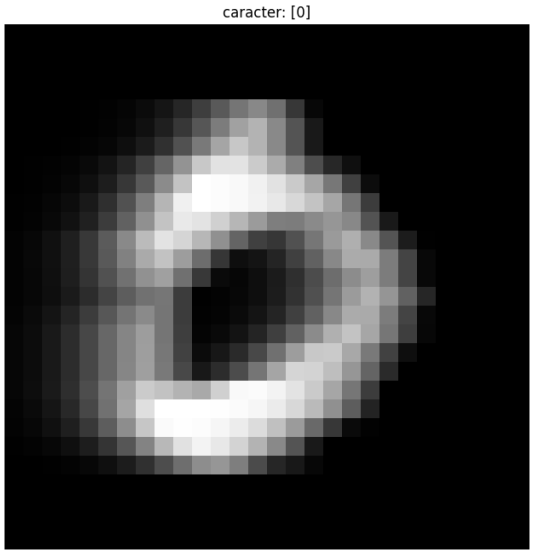


Figure 1: Es 0 o 6?

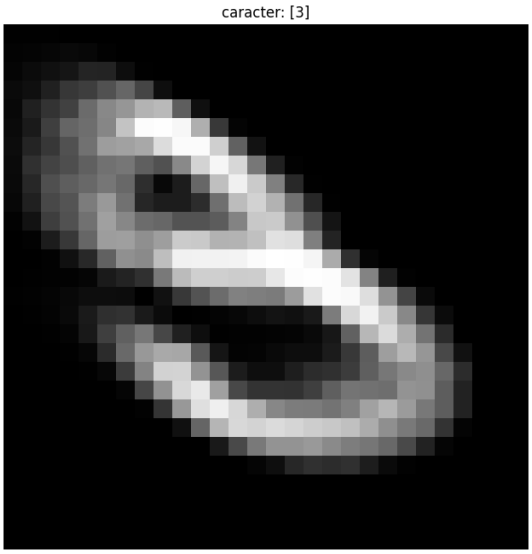


Figure 2: Es 3 o 9?

Es por eso que pensamos en encarar el analisis de la similitud entre numeros usando la distancia Euclidiana, que va a analizar la similaridad entre los pixeles de diferentes imagenes.

¿Cómo funciona la distancia euclidiana entre todos los pares de imágenes?

(a) Flattening de las imágenes:

Cada imagen en el conjunto tiene forma 28x28 píxeles, lo que equivale a 784 valores individuales por imagen. Al hacer .flatten(), cada imagen se convierte en un vector de 784 elementos.

(b) Calcular la distancia Euclidiana entre todos los pares:

Para cada imagen de un determinado digito, el código calcula su distancia Euclidiana con cada imagen del digito a comparar. Como resultado, se genera una lista de todas las distancias posibles entre las imágenes de ambos digitos.

(c) Promedio de las distancias:

Al final, el código calcula el promedio de todas las distancias obtenidas. Esto da una idea general de la "similitud" entre las dos clases. En otras palabras, obtienes una medida de cuán "diferentes" son, en promedio, los dígitos en este espacio de características de 784 dimensiones.

La distancia euclidiana entre dos imágenes A y B, representadas por sus vectores de píxeles $A = [a_1, a_2, \ldots, a_n]$ y $B = [b_1, b_2, \ldots, b_n]$, se calcula como:

$d(A, B) = \sqrt{\sum_{i=1}^{n}(a_i - b_i)^2}$

donde n es el número total de píxeles en cada imagen, a_i y b_i son los valores de los píxeles correspondientes en A y B.

Resultados Obtenidos:

I. Distancia Euclidiana promedio entre todas las imágenes de 0 y 1:

$$d(0, 1) = 2755.3682529628836 \tag{1}$$

II. Distancia Euclidiana promedio entre todas las imágenes de 5 y 6:

$$d(5, 6) = 3308.8110927657617 \tag{2}$$

Dados estos resultados podemos intuir que en promedio la similitud entre los datos de las imagenes de 0 y 1 son mas cercanos que los datos de las imagenes de 5 y 6. Pero para poder cuantificar mejor la facilidad al diferenciar dos digitos diferentes, podemos tener en cuenta las distancias Euclideanas ya calculadas y comparar esos números con unos nuevos. Pj calculamos la distancia entre el 1 y el 5:

$$d(1, 5) = 3344.21711919125 \tag{3}$$

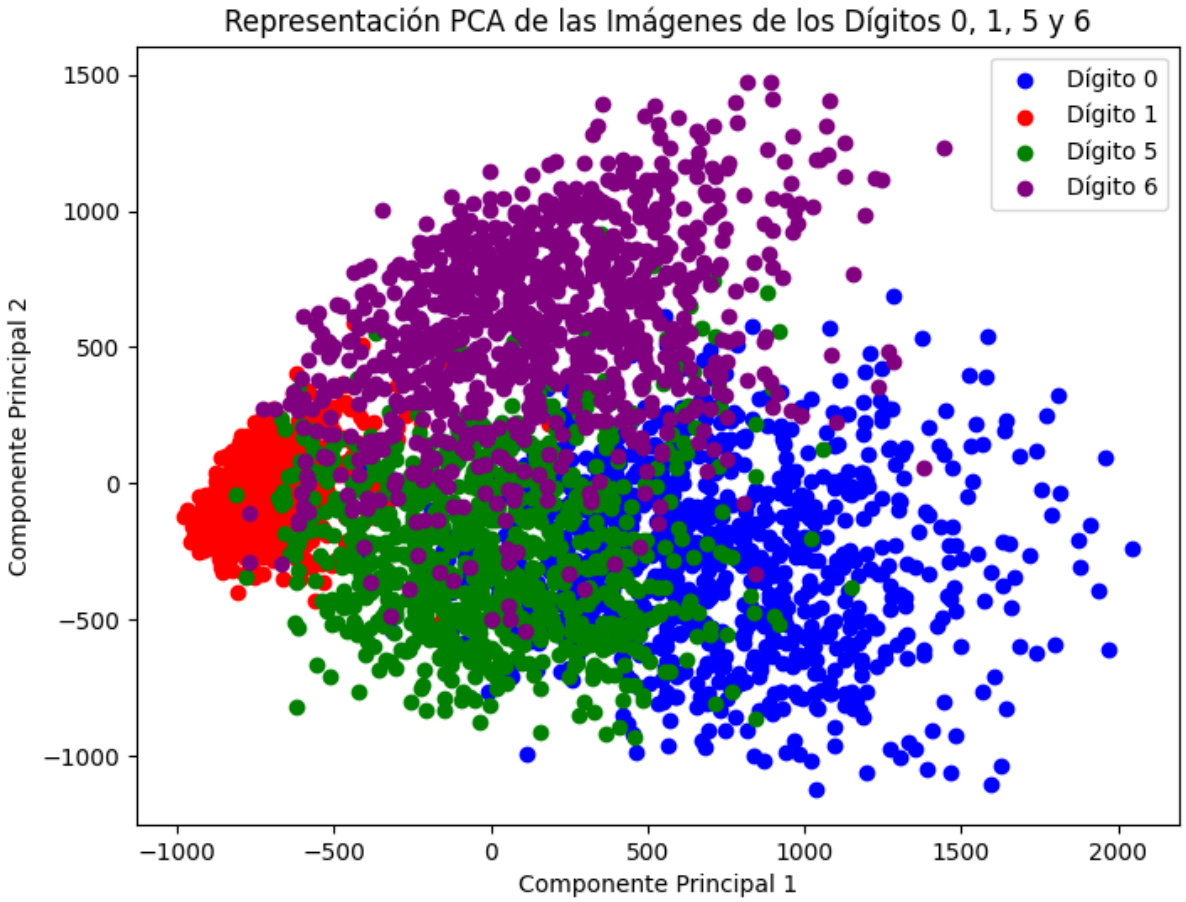
Podemos ver que las imageens correspondientes al numero 1 son mas parecidas en promedio a las imagenes del 0 que a las del 5, y tambien que las imagenes del 5 tiene mejor similitud con las imagenes del 6 que con las imagenes del 1.

Visualización de los Datos Redimensionados con PCA

El PCA es un algoritmo estadístico que transforma un conjunto de datos de alta dimensión a un espacio de menor dimensión, preservando la mayor parte de la variabilidad de los datos. A través de este proceso, se identifican las combinaciones lineales de las características originales, conocidas como componentes principales, que capturan la mayor parte de la varianza en los datos.

En el siguiente gráfico se ha utilizado PCA para reducir las imágenes de los dígitos 0, 1, 5 y 6, que originalmente tienen 784 características (una por cada píxel), a solo dos componentes principales. Esto permite representar los datos en un plano bidimensional. Los puntos del gráfico corresponden a las imágenes de los dígitos, y cada color representa un edígito diferente (azul para el 0, rojo para el 1, verde para el 5 y morado para el 6).

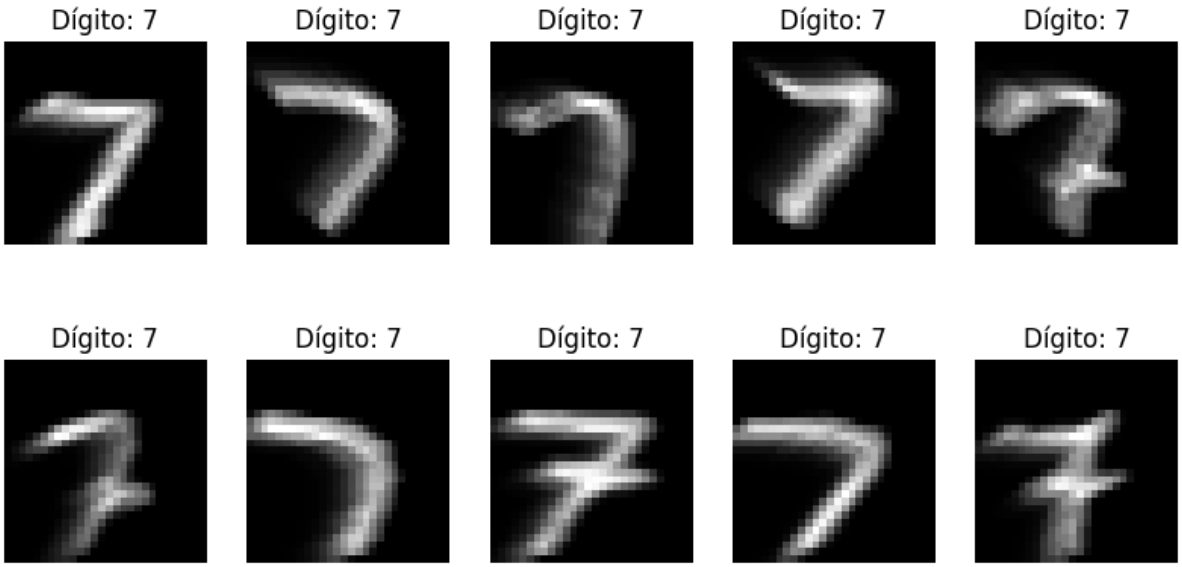
El eje horizontal y el eje vertical representan las dos primeras componentes principales, que son las direcciones de máxima varianza en el espacio de los datos.



Conclusions del grafico: Este grafico refleja de gran manera lo similares que son los numeros. Lo que nos dice que va a ser de mucha dificultad lograr un modelo de arbol de decision que tenga una alta precision. Tanto los digitos del 0, 1, 5 y 6 se superponen en gran medida y esto va a ocurrir para la mayoria de grupos posibles

3. **Tomen una de las clases, por ejemplo el dígito 7. ¿Son todas las imágenes muy similares entre sí?**

   Ya vimos que no solo los números son diferentes entre si, sino que dentro de un conunto de imagenes de un mismo numero, tambien van a haber amplias diferencias entre las imagenes. Para el caso del 7 podemos observarlo rapidamente extrayendo un conjunto de sus imagenes:

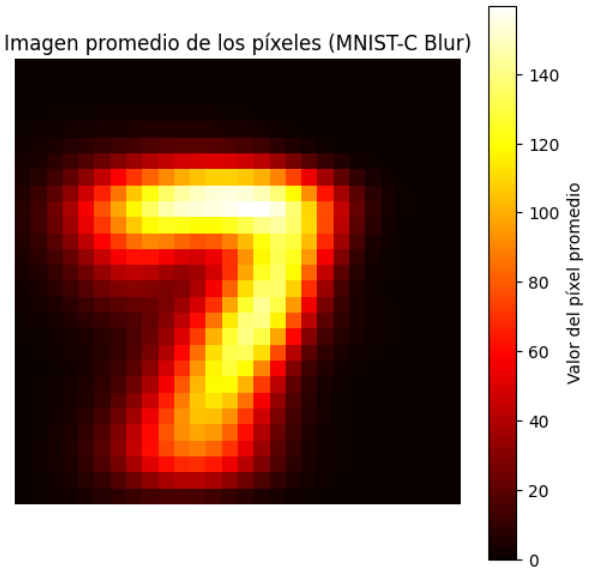


   Aunque estas imagenes estan bien para analizar un par de casos, para responder a la pregunta de manera mas general vamos a utilizar los metodos que usamos en los ítems anteriores.

   Miramos cual es la distancia promedio entre todos las imágenes de un mismo dígito, específicamente para el 7:

$$d(0,1) = 185.84275884072986 \tag{4}$$

Tambien podemos analizar su respectiva imagen del mapa de calor:



Con esto podemos ver claramente que aunque haya una tendencia a que el 7 este representado en color amarillo, hay varias imagenes que se desvian de esto, generando los pixeles rojos de alrededor del 7. Aunque no se puede decir que las imagenes son muy parecidas, hay una relativa similitud entre los que generan que haya un claro 7 en la imagen de calor.

Ademas, obsevando el grafico PCA del punto anterior. Podemos ver como las imgenes del digito 1 son las mas similares entre si por como se agrupan los puntos de color rojo. Al contrario el digito 0 y 6 tienden a dispersarse hacia los bordes, lo que implica una menor similitud entre ellos y entre si mismos.

0.2 Clasificacion multiclase

I. Con los datos analizados en la seccion 1, podemos tener una mejor idea de como implementar un arbol de decision para poder determinar el numero que representa una nueva imagen, nos limitamos a incluir solo los numeros: (3, 4, 6, 8, 9).

Entrenamos los arboles de decision usando el K-folding, que trabaja eligiendo una parte de entrenamiento y otra de evaluacion K veces y promediando su exactitud para diferentes alturas posibles. Para analizar la efectividad de los arboles creados, vamos a utilizar el metodo de Gini gain y la Entropia para determinar la efectividad del árbol.

Al experimentar con restringir las alturas posibles del arbol de entrenamiento, observamos que la exactitud del arbol incrementaba con la altura, hasta que llega a las alturas despues de 10 permanece relativamente constante alrededor de 0.8 para la exactitud de la evaluacion con el metodo de Gini gain y el de entropia:

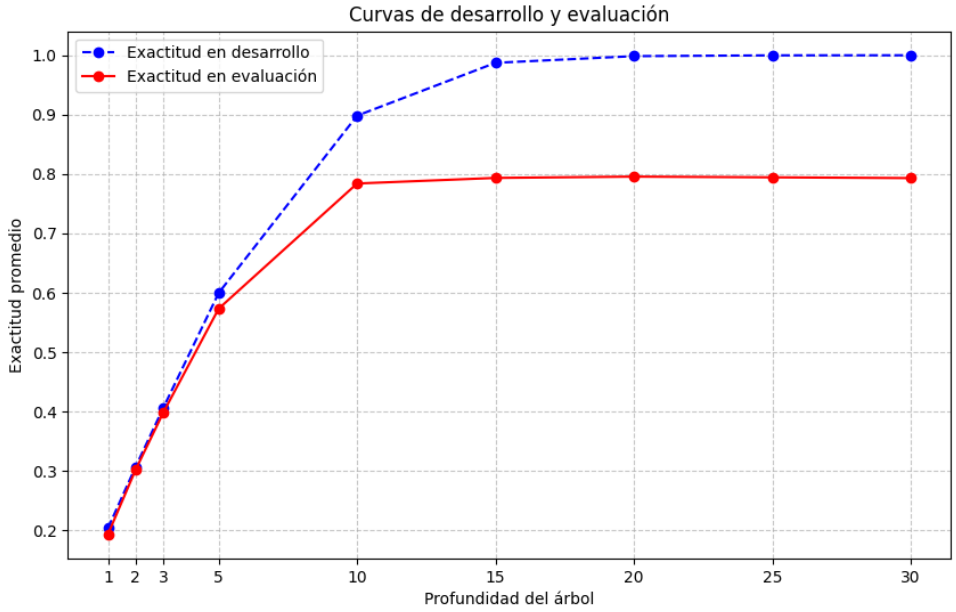


Figure 3: Gini gain

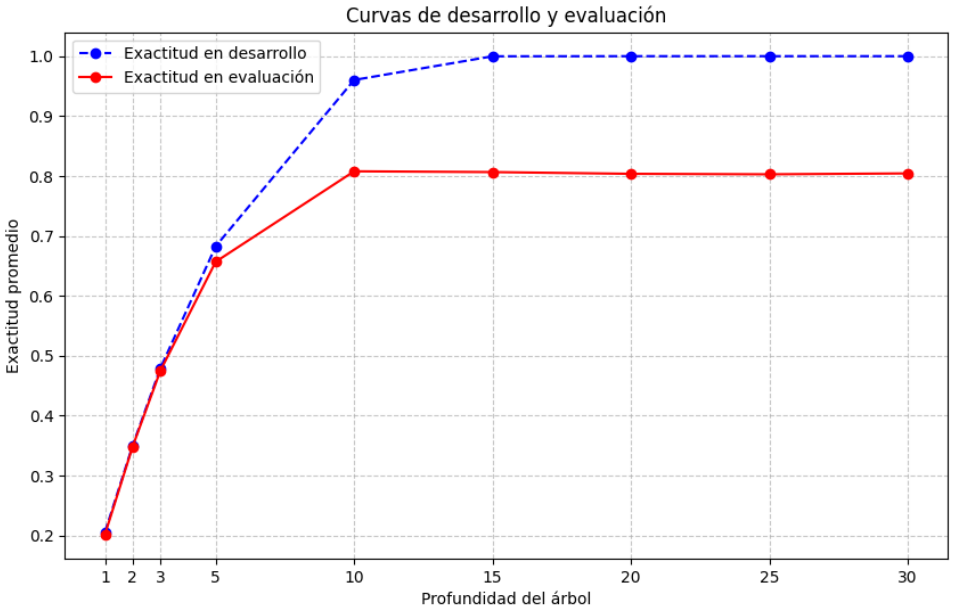


Figure 4: Entropia

Al ver que para ambas metricas, la exactitud para la evaluacion varia muy poco despues del 10, nos parece mejor elegir un numero cercano a 10 porque cuanto mas grande es el arbol mas se empiezan a desviar las curvas de entrenamiento y de evalucion.

Evaluamos mejor las profundiades alrededor de 10 para elegir la mejor altura:

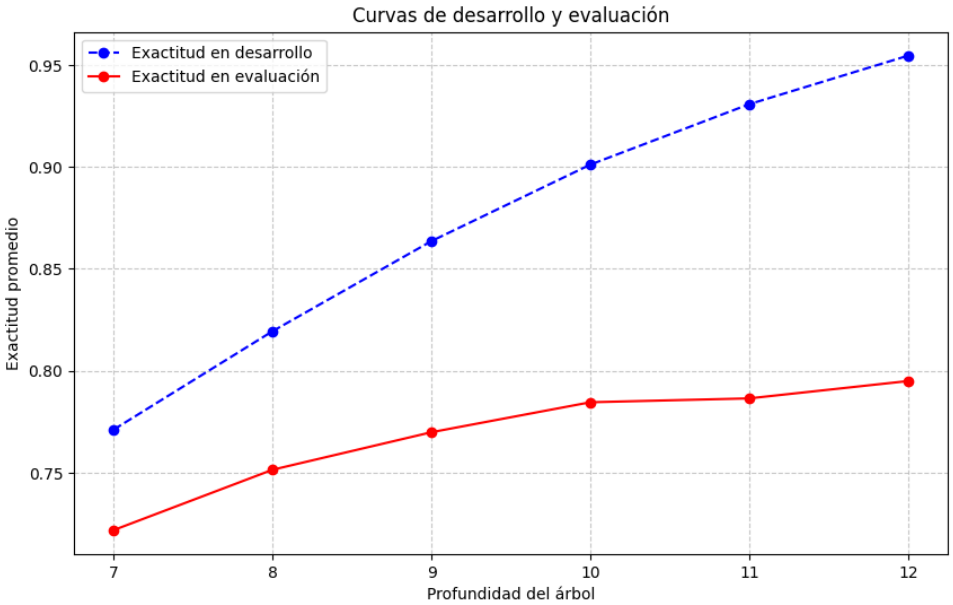


Figure 5: Gini gain

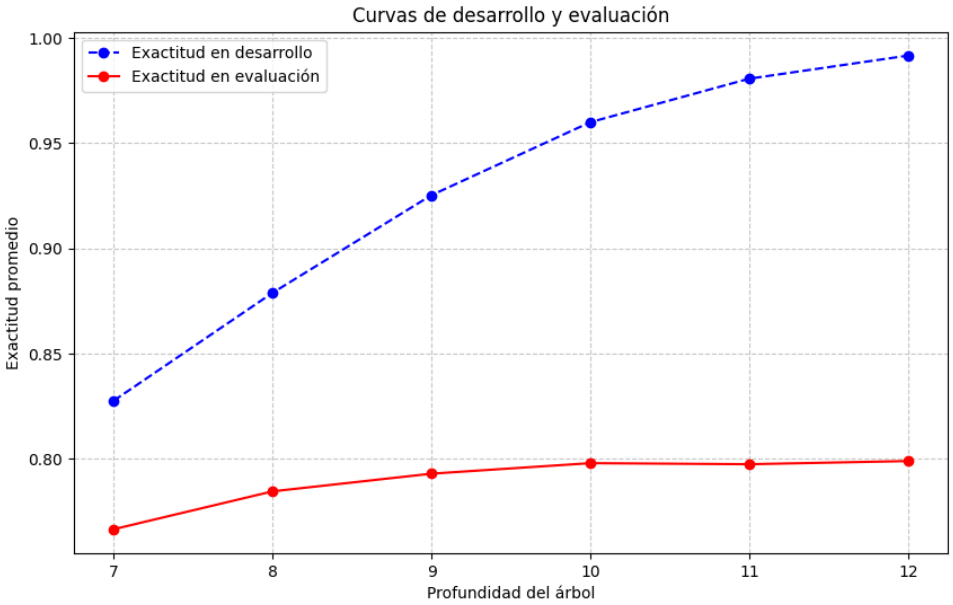


Figure 6: Entropia

Como el mayor punto de exactitud de evaluacion aparece en 10, vamos a usarlo como altura del arbol y vamos a evaluar usando el metodo Gini gain que nos genera menos diferencia entre la exactitud de entrenamiento y de evaluación.

Usando los datos del *hold-out*, evaluamos la performance del arbol, computando métricas relevantes, y luego graficando una matriz de confusion multiclase.

Las métricas más relevantes resultaron satisfactorias a lo largo del total de las clases, con exactitud, recall, y f-1 score alcanzando el 0.85 en promedio. Sin embargo se ven ciertas clases que presentan mayores dificultades, como se puede ver en sus valores más bajos en las métricas ya mencionadas.

	precision	recall	f1-score	support
3	0.90	0.84	0.87	202
4	0.86	0.85	0.86	196
6	0.90	0.91	0.91	192
8	0.78	0.80	0.79	195
9	0.81	0.84	0.82	202
accuracy			0.85	987
macro avg	0.85	0.85	0.85	987
weighted avg	0.85	0.85	0.85	987

Figure 7: Metricas de performance de nuestro modelo

Particularmente a los números 4 y 3 con frecuencia les son asignadas las clases 9 y 8 erróneamente. Esto se puede ver claramente en la matriz de confusion del test.

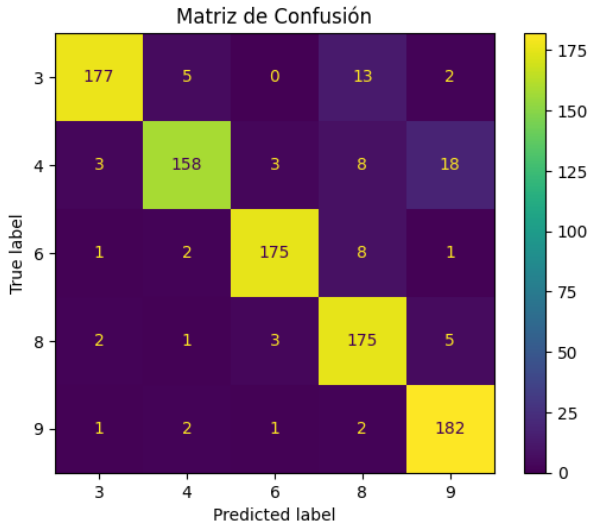


Figure 8: Matriz de Confusion de Clases 3,4,6,8,9

Los resultados de la matriz de confusion nos dan a entender que el arbol tiene generalmente buena efectividad en determinar el digito representado por cada imagen, dada su diagonal fuerte, más allá de los déficits ya mencionados.

En conclusion, se logró contruir un arbol de decision con una profundidad adecudada, que permita clasificar de forma correcta la mayoria de las instancias de las clases presentes en el dataset.