DEPARTAMENTO DE COMPUTACION

Facultad de Ciencias Exactas y Naturales - UBA

Trabajo práctico 2

Desarrollo de modelos de clasificación sobre el conjunto de datos de imágenes MNIST-C Fog.

10 de marzo de 2025 — Laboratorio de Datos

Malfatti

Integrante	LU	Correo electrónico
Sandoval, Francisco	1212/23	franciscodtsandoval@gmail.com
Illescas, Marcos	390/14	marcosillescas90@gmail.com
Mannino, Jeremias	611/24	jere7005_@hotmail.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

El objetivo de esta actividad es, dado un dataset de imágenes de dígitos del 0 al 9, lograr realizar modelos de aprendizaje automático que identifiquen de forma correcta, para la mayor cantidad de imágenes, el número que estas contienen. Este dataset llamado MNIST-C, está en su versión corrompida “Fog”. Contiene, por cada imagen, el número que ésta representa. Al conocer la respuesta de la variable que queremos predecir, están dadas las condiciones para utilizar modelos de aprendizaje supervisado.
Se escogerán 2 modelos. Un modelo intentará responder la pregunta: ¿la imagen corresponde al dígito 0 o al dígito 1?. El siguiente modelo, responderá la siguiente pregunta: ¿A cuál de los 10 dígitos corresponde la imagen?.
Para esto, se comenzará haciendo un análisis exploratorio de datos, para comprenderlos mejor e incorporar información útil, y así poder configurar nuestros modelos de forma óptima.

Análisis exploratorio

Se cuenta con un dataset de 70.000 imágenes, donde cada imágen tiene 784 atributos, cada uno siendo un numero entero entre 0 y 255 que representa la luminosidad del pixel (0 es negro, 255 es blanco). Al estar compuesto por imágenes, la exploración de los datos no es directa, sino que se debe realizar una tarea de entendimiento, procesamiento y filtrado de los datos, para así poder configurar nuestros modelos de forma correcta.
Como primer análisis gráfico, se tiene el histograma de la Figura 1, con las frecuencias de los datos.

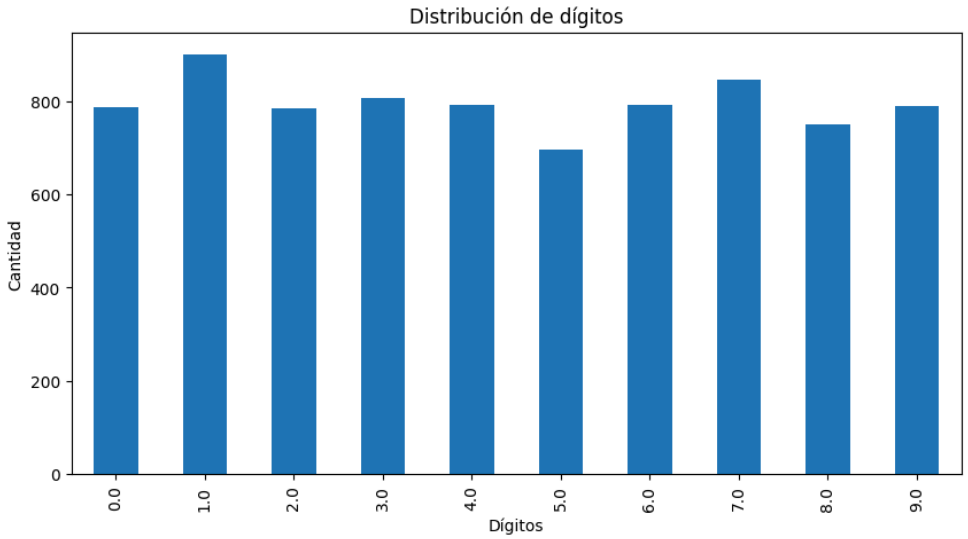

Fig 1. Distribución de frecuencias de los dígitos

Se puede observar que no hay grandes diferencias en la cantidad de datos por cada dígito, lo cual es bueno porque se tiene información bien distribuída.

Es esperable que haya números similares entre sí, como el tres y el ocho, a diferencia, por ejemplo, del uno y el tres, donde sus formas deberían estar bien diferenciadas. Esto queda claro al ver el gráfico de la figura 2, que además, nos da una idea de lo que contiene el dataset.

Fig 2. Imágenes promedios de algunos dígitos.

También se observa una variabilidad intra clase, por ejemplo al observar la figura 3, donde imágenes de una misma clase (dígito 1) tienen formas de escrituras heterogéneas.

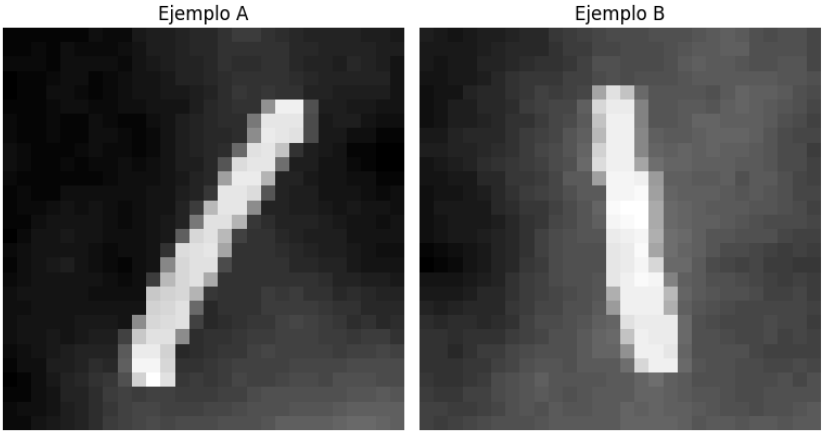

Fig 3. Ejemplos de imágenes con el dígito 1.

Esta variabilidad existente entre diferentes ejemplares de un mismo dígito no permite una elección obvia de píxeles característicos de ese dígito.
Es deseable también obtener información de la importancia de cada píxel. Para esto, parece conveniente guiarse por la varianza de cada píxel, como muestra la figura 4.

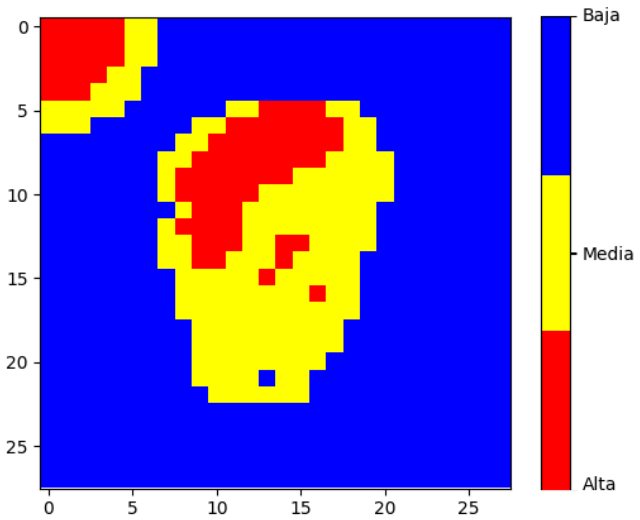

Fig 4. Mapa de calor de varianza de los píxeles.

Sin embargo, notar que, en la esquina superior izquierda, la varianza es muy alta. No es intuitivo pensar que la esquina de una imágen pueda ayudar a detectar un número, por ende, esta métrica por sí sola no es suficiente para determinar la importancia de un píxel.
Se decidió utilizar una métrica más compleja para medir la importancia de los píxeles. La métrica por píxel de cada dígito se menciona en el anexo como f_{in}. Esta métrica permite detectar los píxeles más importantes en un dígito, teniendo en cuenta la variación con respecto al resto de números. Se calcula por cada píxel en cada dígito, y luego se calcula el promedio de todos los dígitos por píxel. De esta forma, se mide la importancia, como se encuentra graficado en la Figura 5.

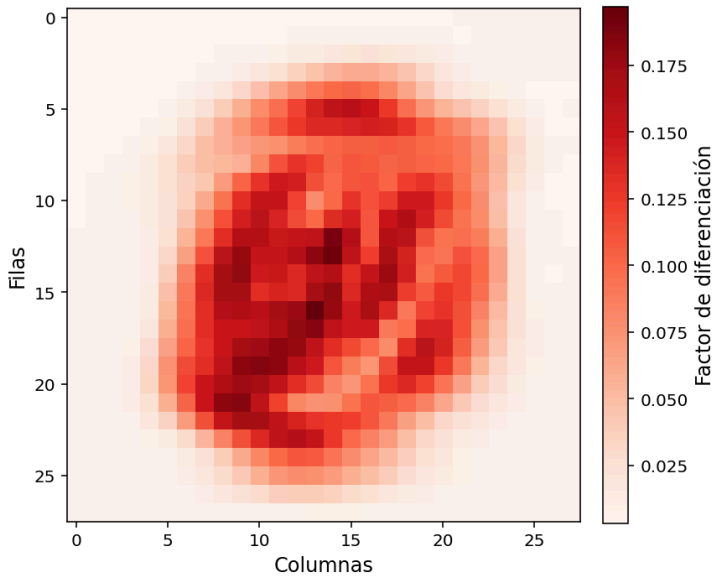

Fig 5. Mapa de importancia de los píxeles.

Se puede observar que los píxeles más importantes se encuentran en el medio de la imágen.

Experimentación

Se desean encontrar 2 modelos de aprendizaje automático, para distintas tareas, de forma tal que puedan tener la mejor exactitud posible en sus predicciones, sin sacrificar su simplicidad.
Para los modelos, se decidió filtrar los píxeles que obtendrán en su entrenamiento, y así serán menos costosos computacionalmente, sin perder información importante. Esto se hizo de forma distinta para cada uno.
A continuación, se detallan las metodologías utilizadas para encontrar los modelos con mejor performance posible.

Clasificación Binaria con KNN

En primer lugar, se plantea la tarea de distinguir las imágenes correspondientes a los dígitos ceros y unos de la base de datos implementando un modelo KNN. Para esto, como primer paso, se filtran los datos de la base de datos original aquellos registros que corresponden a los dígitos ceros y unos. En la figura 6 se muestra la proporción de los dígitos ceros y unos luego de filtrar los datos. Si bien hay una mayor proporción de unos que de ceros, no consideramos la diferencia importante y vamos a operar con los datos obtenidos luego del filtrado.

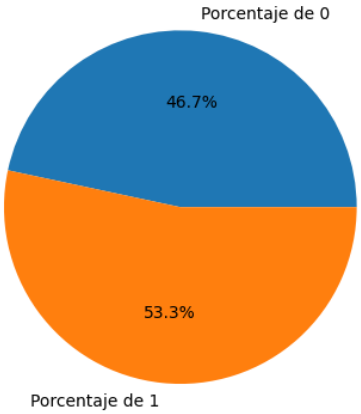

Fig 6. Distribución de datos luego de filtrado.

Para elegir los píxeles más importantes para el modelo que diferenciará los ceros y unos, se utilizó como métrica el factor $f_{01} = \frac{|\mu_2 - \mu_1|}{\sigma_2 + \sigma_1}$, donde μ son las medias de la intensidad de los píxeles y σ son sus desvíos estándar. Esta métrica permite detectar los píxeles más frecuentes en un número y menos frecuentes en el otro, acorde a los que la maximicen. El gráfico de la

figura 7-A muestra un mapa de calor de los píxeles en función del factor de diferenciación entre ceros y unos, mientras que la figura 7-B muestra los píxeles elegidos para los experimentos.

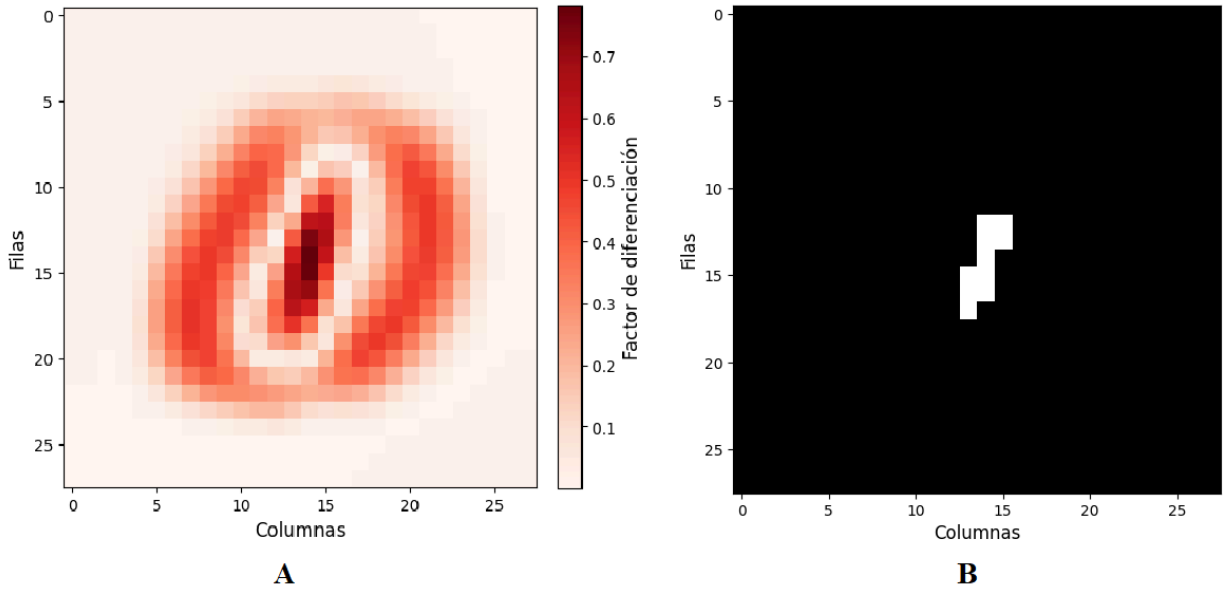

Fig 7. Visualización de los 10 píxeles elegidos por factor de diferenciación.

Para evaluar los modelos desarrollados durante los experimentos, se dividió el conjunto de datos de ceros y unos en un subconjunto de entrenamiento y de test.
Para los modelos KNN, en una primera instancia, se usó $k = 3$ y se eligieron 3 atributos por cada modelo. Para encontrar los mejores atributos, se usaron los diez píxeles con mejor factor de diferenciación, mostrados en blanco en la figura 6-B.
Con el objetivo de hacer un análisis exhaustivo, se evaluaron todas las posibles combinaciones de tres píxeles dentro de los 10 píxeles mencionados anteriormente. Mediante la evaluación sistemática de estas 120 combinaciones, se alcanzó una exactitud de 0,82 usando los píxeles correspondientes a las columnas de índices 406, 433 y 462. En la figura 8 se muestra la matriz de confusión del mejor modelo, evaluado según su exactitud.

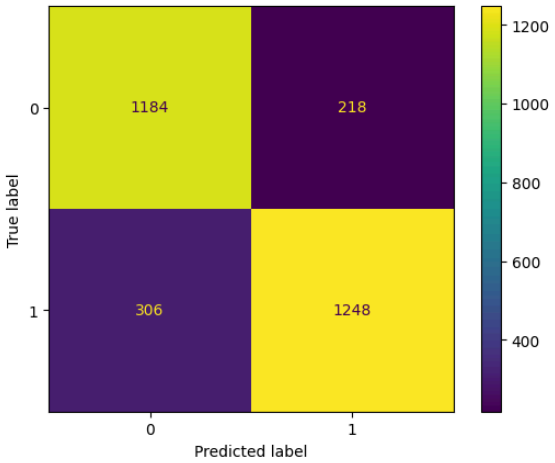

Fig 8. Matriz de confusión del modelo KNN (k=3) con mayor exactitud.

En un segundo experimento, se entrenaron varios modelos de KNN en los cuales se varía tanto la cantidad de atributos (n) que reciben estos modelos como la cantidad de vecinos (k). Se decidió variar tanto la cantidad de atributos como de vecinos en un rango entre 3 y 10. Pero esta vez, a diferencia del primer experimento, no se evaluó exhaustivamente todas las combinaciones para que el tiempo y costo de cómputo no sea mayor. Por eso, para obtener los n atributos en cada modelo, se decidió tomar en orden de importancia los primeros n píxeles de una lista ordenada según el factor de diferenciación f_{01}.

Entonces, para cada combinación de atributos y vecinos, se entrenó un modelo KNN y se midió la exactitud sobre el conjunto de prueba. Los resultados se almacenaron en una matriz y se representaron gráficamente en el mapa de calor de la figura 9.

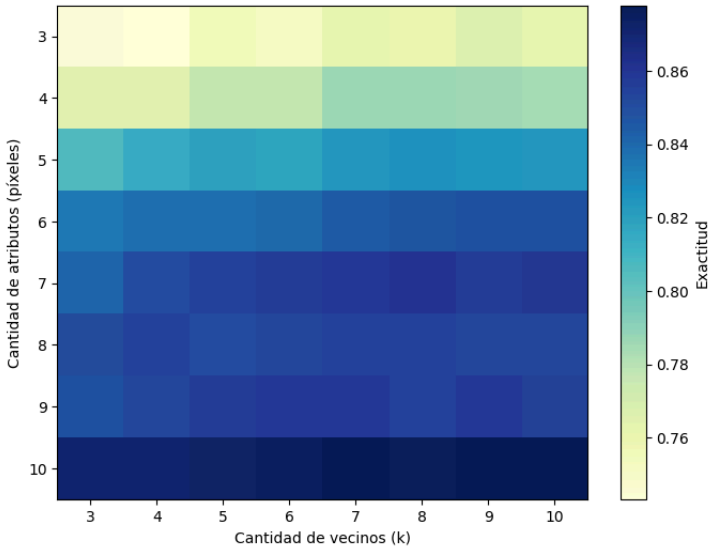

Fig 9. Mapa de calor de exactitud en función de cantidad de atributos y de k.

Como se observa en la figura 9, la mayor exactitud se logra cuando se aumenta la cantidad de vecinos y la cantidad de atributos en el modelo. Sin embargo, parece que la cantidad de

atributos tiene un impacto mayor en la exactitud que logra el modelo. El mayor valor de exactitud obtenido fue de 0,88 alcanzada por las combinaciones $(n = 10,\ k = 9)$ y $(n = 10,\ k = 10)$.

Clasificación de dígitos con árboles de decisión

Se busca un modelo de árbol de decisión que identifique correctamente los dígitos del 0 al 9.
Se comienza escogiendo aquellos píxeles que serán utilizados para entrenar el modelo.
Para este modelo, los píxeles elegidos fueron los que maximizaron la métrica f_{in} mencionada en el anexo. Se eligieron de esta forma los 5 píxeles más importantes por cada dígito, obteniendo así un total de 39 píxeles, como muestra la figura 10.

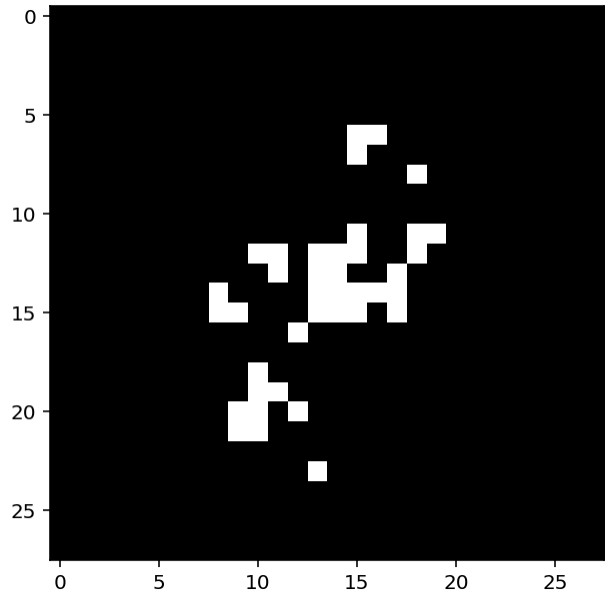

Fig 10. Visualización de los 39 píxeles elegidos por factor de diferenciación.

Luego, se separaron los datos entre un conjunto held-out que será utilizado únicamente para evaluar la performance de los modelos finales, y un conjunto de desarrollo. A su vez, este conjunto de desarrollo se separó en uno de entrenamiento, que será utilizado para entrenar el modelo inicialmente, y uno de validación, para evaluar parcialmente las performance del mismo modelo con distintos hiperparámetros.
Se comenzó entrenando el modelo, variando el hiperparámetro profundidad entre 1 y 10, obteniendo una mejora progresiva en la exactitud en el conjunto de validación, desde 0.15 hasta 0.60.
Luego se probaron varias combinaciones de hiperparámetros, con 2 valores de profundidades, 2 de mínimo de decrecimiento de impureza, y 2 de poda, y se eligió el de mejor performance con el método k-folding, con k=5, siendo la mejor combinación la que tiene los hiperparámetros: ccp_alpha=1e-05, max_depth=10,min_impurity_decrease=0.00005. Luego, este modelo se entrenó con todos los datos de entrenamiento, y se midió su exactitud, la cual fue de 0.60, igual que la mejor de las exactitudes reportadas al modificar las profundidades teniendo el resto de

los parámetros por default. Es decir, evidentemente el hiperparámetro máxima profundidad es demasiado importante en comparación con el resto.
Por último, este modelo con esos hiperparámetros fueron entrenados con todo el conjunto de desarrollo, y se midió su performance con el conjunto held-out, la cual fue de 0.64, cuya matriz de confusión se ve en la Figura 11. Cabe recalcar que esta performance mejora cuando se aumenta el hiper parámetro de máxima profundidad, pero se prefirió que el máximo sea 10 para mantener la simpleza del modelo.
Esta exactitud no fue la deseada, por eso se tomó la decisión de experimentar con un segundo modelo que, en lugar de tomar como datos de entrenamiento los píxeles más importantes, tome un conjunto de datos el cual se le haya aplicado un procesamiento previo. Para esto, primero se detectan los bordes internos de cada número. Luego, por cada fila y cada columna, se calcula la cantidad de bloques de píxeles juntos que tienen la misma clasificación borde o no borde. Por ejemplo, si se tiene una fila con un solo píxel borde que está en el medio, esta tendrá 3 bloques. Estos datos de bloques por fila y columna se utilizarán para entrenar el nuevo modelo.
Se repitió el mismo proceso que para el modelo anterior. De forma similar, sucede que obtiene mejor performance cuanto mayor sea el parámetro profundidad, que varía del 1 al 10, llegando a 0.81 con 10. Esta no varía al cambiar otros hiperparámetros, y cuando se evalúa su performance en el held-out, obtiene una puntuación de 0.82, y su matriz de confusión está en la Figura 11. Esta performance también mejora cuando se aumenta el hiper parámetro de máxima profundidad.

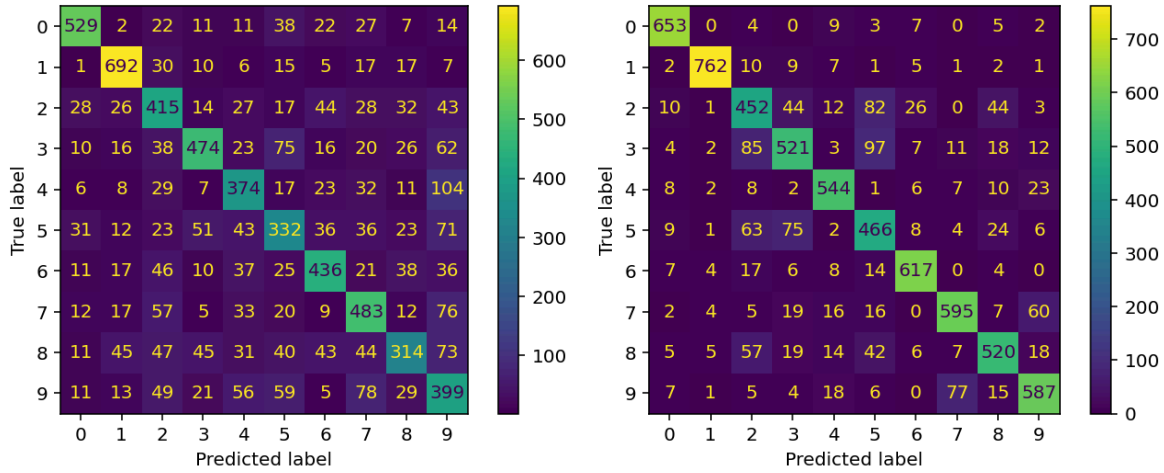

Fig 11. A la izquierda, la matriz de confusión del modelo cuyos atributos fueron los píxeles filtrados. A la derecha, la del modelo con atributos pre-procesados.

Conclusiones

En primer lugar, una inspección de la distribución de los dígitos de la base de datos, determinó que los dígitos se encuentran en cantidades similares. Esto es importante porque evitó sesgos cuando se entrenaron los modelos. Se observa, en la figura 5, que no todos los píxeles tienen la misma importancia. Por eso, fue clave la elección de píxeles para no perder información relevante sobre los datos.

Para encontrar el mejor modelo que pueda clasificar ceros y unos, se entrenaron modelos de KNN con distintos hiperparámetros. Primero, se utilizaron diferentes configuraciones de 3 atributos, y k = 3 (número de vecinos), evaluando su eficacia mediante la métrica de exactitud. El mejor resultado obtenido fue una exactitud de 0,82.

Luego, investigamos cómo varía la exactitud modificando tanto la cantidad de vecinos como la cantidad de atributos. Se observa una mejora con el aumento de ambos, logrando una exactitud de 0,88 cuando tanto k, como el número de atributos, es 10. Sin embargo, se debe considerar que ésta configuración supone un aumento en la complejidad del modelo. Es por esto que lo mejor sería encontrar un balance entre la simplicidad y el rendimiento, para lo cual es conveniente tomar ambos en 3.

Para la clasificación de todos los dígitos mediante árboles de decisión, luego de filtrar los píxeles, y experimentar con distintos hiperparámetros, se obtuvo una performance de 0.64. Esto no parece una buena performance, entonces fue pensado otro modelo que toma otro tipo de atributos. Para este modelo, donde se realizó un procesamiento previo de los datos, el resultado de la performance fue 0.82. Este segundo es mejor ya que la diferencia de exactitud con respecto al anterior es bastante significativa. En concordancia con esto, es notable que el pre-procesamiento de los datos puede jugar un papel clave a la hora de realizar el modelo.

Adicionalmente se puede ver en la figura 9 que ambos modelos tienden a equivocarse con algunos dígitos. Por ejemplo, el modelo cuyos datos de entrenamiento fueron los píxeles filtrados, tiene una leve tendencia a etiquetar los dígitos cuatro como el dígito nueve. Por otra parte, el modelo que se entrenó con datos pre-procesados, tiene una tendencia menor a etiquetar los dígitos 3 como el dígito 5. Esto sucede porque hay dígitos que son bastante parecidos entre sí.

Como conclusión, el modelo elegido para responder la pregunta “¿la imagen corresponde al dígito 0 o al dígito 1?” Es un modelo de clasificación con KNN que toma 3 atributos y el hiperparámetro k=3.

Luego, el modelo elegido para responder la pregunta “¿A cuál de los 10 dígitos corresponde la imagen?” Es un modelo de clasificación con árboles de decisión, con el hiperparámetro max_depth=10, y un pre-procesamiento de los datos.

Anexo

La siguiente métrica es mencionada en secciones anteriores: $f_{in} = \frac{|\mu_i - \mu_n|}{\sigma_i + \sigma_n}$, donde μ_i es la media de la intensidad de los píxeles del dígito i, μ_n es la media de la intensidad de los píxeles de los dígitos distintos a i, σ_i es el desvío estándar de los píxeles del dígito i, σ_n es el desvío estándar de los píxeles de los dígitos distintos a i.