Resumen

Se incluye el análisis de la composición del dataset Fashion-MNIST, de los subconjuntos formados y de sus atributos, aquellos que consideramos relevantes. Se utiliza el modelo K-Nearest Neighbors (KNN) para una Clasificación Binaria, y árboles de decisión para una Clasificación Multiclase; ambos con el objetivo de poder identificar la clase de una prenda. En la conclusión analizamos las matrices de confusión, hablamos de cómo se ven afectadas las distintas métricas obtenidas según el modelo utilizado y las distintas características de las clases (promedio de intensidad de cada pixel, desviación estándar y diferencia entre los promedios de distintas clases).

Introducción

El dataset Fashion-MNIST recopila imágenes de prendas de ropa, los cuales son acompañados por una clase que denomina el tipo de prenda al que pertenece. Está conformado por 70.000 imágenes con 784 píxeles que indican la intensidad de su color en una escala gradual de grises. En un análisis exploratorio de los datos, hablaremos del balance existente de las clases, la manera en la que está conformado el dataset, y de la desviación estándar y promedio de los píxeles de cada clase, además de su utilidad. Luego, realizaremos distintos modelos de clasificación y los evaluaremos para elegir el que consiga una mayor exactitud.

Análisis Exploratorio de los Datos

El dataset Fashion-MNIST cuenta con 70.000 imágenes de 784 píxeles cada una, teniendo un tamaño 28x28. Cada atributo es un píxel distinto que lleva una etiqueta que denomina su ubicación, “pixel**N**” con **N = i * 28 + j**, siendo ambas variables números enteros que van del 0 al 27; **i** el número de fila comenzando arriba y **j** el número de columna desde la izquierda.

Cada píxel es un atributo de variable cuantitativa que indica su intensidad en una escala gradual de grises que varía entre el 0 (negro) y 255 (blanco). Además, contamos con un índice en la primera columna y la etiqueta de la clase en la última. Hay 10 clases distintas identificadas con un número entero entre 0 y 9:

- 0 = Remera
- 1 = Pantalón
- 2 = Suéter
- 3 = Vestido
- 4 = Abrigo
- 5 = Sandalia
- 6 = Camisa
- 7 = Zapatilla
- 8 = Bolsa
- 9 = Botita

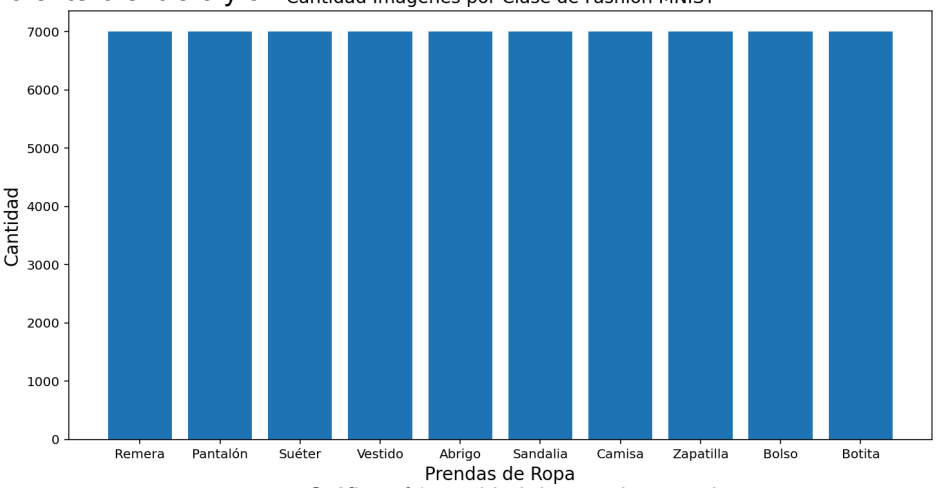

Gráfico n°1 cantidad de prendas por clase

Cada clase posee un total de 7.000 imágenes, por lo que el dataset está balanceado en cuanto a cantidad de prendas por clase. Los atributos poseen como valor un número que, si fuera visto individualmente y no como un conjunto redimensionado representado en una imagen, no sabríamos cómo interpretarlo. En cambio, en el dataset Titanic trabajado en clase encontrábamos variables categóricas como sexo, pasaje y edad, fácilmente interpretables por sí mismas. Por lo tanto, para detectar aquellos atributos de Fashion-MNIST que sean relevantes, en primer lugar tendremos que analizarlos con la ayuda de las herramientas de visualización, en conjunto.

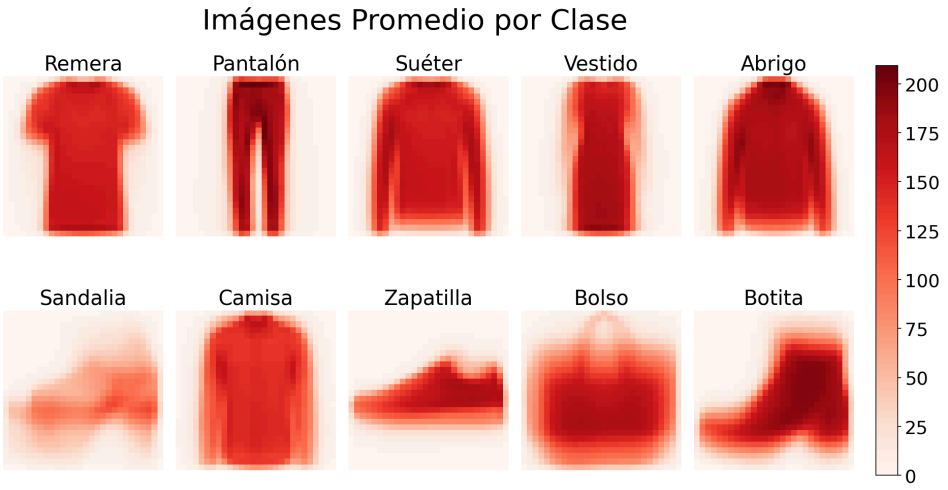

Figura n°1 Imágenes promedio por clase.

A partir de la imagen previa, notamos que las prendas tienden a estar centradas y alineadas, lo que facilita el entrenamiento de modelos. Como resultado, la mayoría de las imágenes tienen las esquinas completamente negras, por lo que esos píxeles con baja varianza no aportan información relevante para la clasificación. En cambio, serán más relevantes los atributos que marquen una diferencia entre las prendas a clasificar.

Además, calculamos la desviación estándar de cada clase. Con esto podemos observar que también existirá una gran diferencia entre prendas de una misma clase, lo cual podría complicar la clasificación al no hallar un patrón definido para cada una de ellas. Viendo la imagen inferior, notamos que hay prendas como las pertenecientes a la clase pantalón que poseen pocos píxeles con alta varianza en comparación a, por ejemplo, aquellas pertenecientes a sandalia.

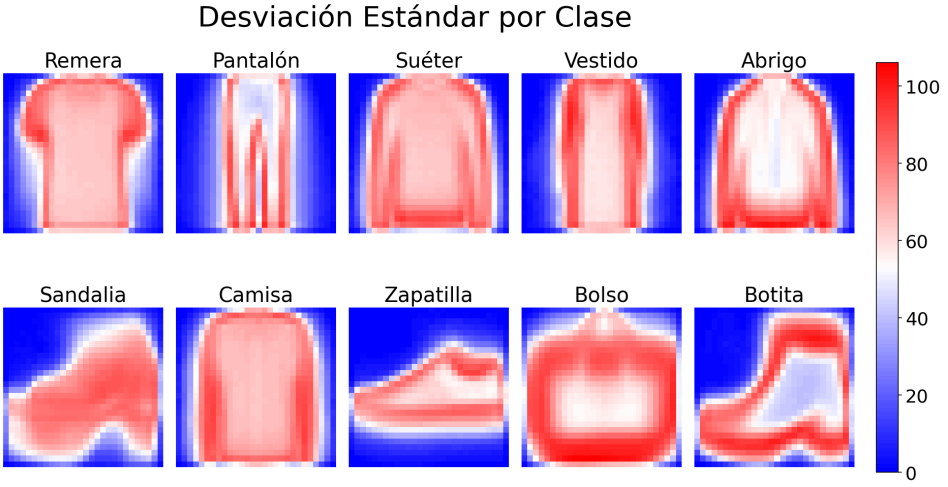

Figura n°2 Desviación estándar por cada clase.

A continuación veremos unos gráficos que muestran la diferencia de intensidad promedio de píxeles entre imágenes de distintas clases. Cada extremo será un color “azul” o “rojo”, que denominarán qué clase tiene mayor intensidad de color en cada píxel.

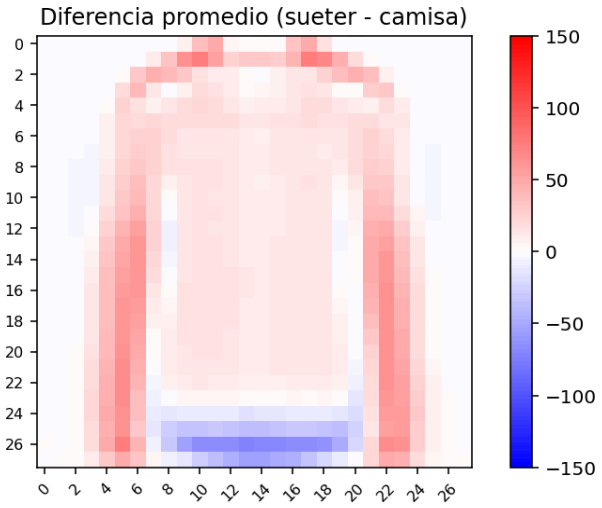

Figura n°3 Diferencia entre el promedio de la clase suéter (2) de la clase camisa (6).

En este caso, diferenciamos el promedio existente entre la clase Suéter (Rojo) y Camisa (Azul). Observamos que la diferencia de intensidad es en general baja, lo que indica que ambas clases presentan patrones visuales similares. Sin embargo, esto no implica que no existan diferencias, las mismas pueden darse en zonas puntuales que, aunque sutiles, resultan relevantes para distinguirlas.

Se entiende que al haber camisas de manga corta, se genera una diferencia con los suéters que son de manga larga en su totalidad. Otras distinciones que podemos ver son que la camisa suele llegar hasta la zona más baja de la imagen mientras que el suéter suele tener hombros más anchos o pronunciados.

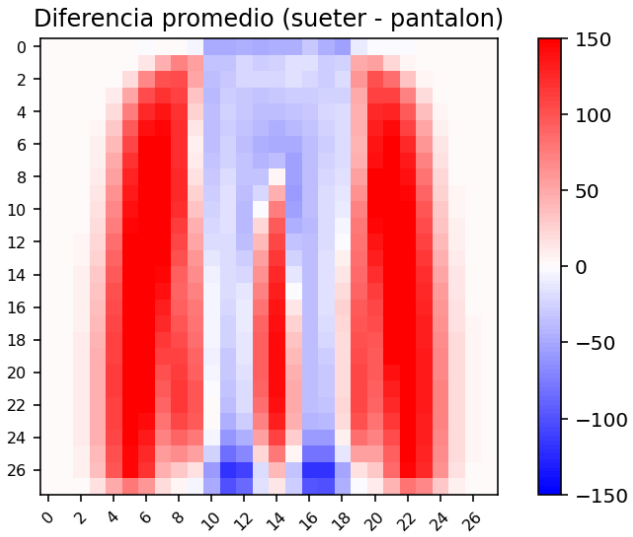

Figura n°4 Diferencia entre el promedio de la clase suéter (2) de la clase pantalón(1)

Sin embargo, existen casos donde la diferencia es mayor. En este caso, las clases a diferenciar son Suéter (Rojo) y Pantalón (Azul).
Los colores en el gráfico son más intensos, más que nada el Rojo que rodea a la “sombra” de un pantalón que se forma en el centro. De igual modo, cerca de la zona inferior, podemos ver un Azul más intenso que representa el hecho de que los pantalones ocupan todo el largo, donde pareciera que los suéteres (o al menos la mayoría) no llegan. En conclusión, los pantalones ocupan una parte más angosta de la imagen, mientras que el suéter tiene más píxeles intensos a lo ancho con sus mangas.

Clasificación Binaria

Con el objetivo de poder clasificar una imagen según su clase, considerando únicamente las clases 0 y 8, comenzamos armando el subconjunto con el que trabajaremos en esta sección. El mismo contiene un total de 14.000 prendas, siendo cada mitad una clase distinta pues está balanceada al igual que el dataframe original. Luego, manteniendo el balance de clases, separaremos el 80% de datos del subconjunto para formar el conjunto de entrenamiento y el 20% restante para el conjunto test.

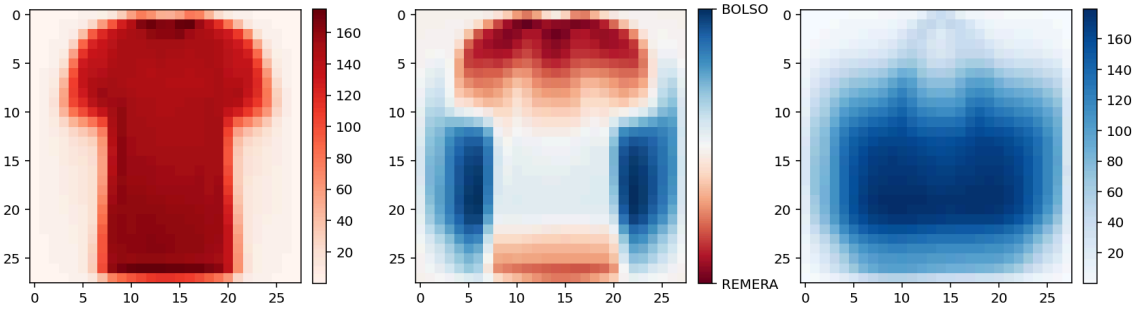

Figura n°5 Promedio de intensidad de los píxeles de la clase 0, Diferencia de promedios entre las clases 0 y 8, y Promedio de intensidad de los píxeles de la clase 8, en este orden.

Al superponer los píxeles promedio de ambas clases, podremos hallar esos que diferencian una clase de otra. Los mismos tendrán una intensidad mayor al tratarse de una clase específica, lo cual nos ayudará a seleccionar los atributos que usaremos para el modelo KNN. En total, experimentamos con tres conjuntos:

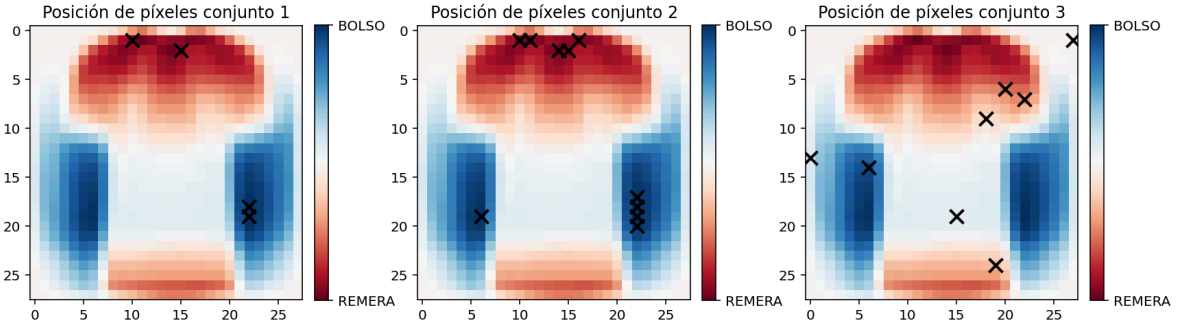

Figura n°6 Diferencia de promedios entre las clases 0 y 8, donde las cruces de cada uno determina la posición de los píxeles que forman parte de cada conjunto. El tercer conjunto, al contener píxeles aleatorios, varía en cada ejecución. Utilizaremos uno en particular para este informe por temas de simpleza.

Las combinaciones básicas serán las siguientes:

- **Conjunto 1: [554, 526, 71, 38]**
- **Conjunto 2: [554, 526, 538, 582, 498, 71, 38, 44, 70, 39]**
- **Conjunto 3: [55, 691, 270, 398, 364, 547, 188, 218]**

Además usaremos dos combinaciones que hicimos con los conjuntos mencionados y algunos píxeles aleatorios, que en el caso de este informe los fijaremos en:

- **Conjunto 2 & nro random: agregamos los píxeles 736, 333 y 594 en conjunto 2**
- **Conjunto unión: unión entre conjunto 2 y conjunto 3**

Experimentamos con k que varían entre el 1 y el 20, y estos fueron nuestros resultados:

A partir de los siguientes gráficos notaremos que cuando los valores de k son menores a 3 ocurre una mayor diferencia de exactitudes entre los conjuntos Test y Train de cada combinación, por ende, se produce un sobreajuste. Para los resultados limitaremos los valores de k a considerar con un rango entre 3 y 20, para luego analizar los mejores casos de cada conjunto con el motivo de hallar el conjunto de atributos óptimo que no produzcan un sobreajuste.

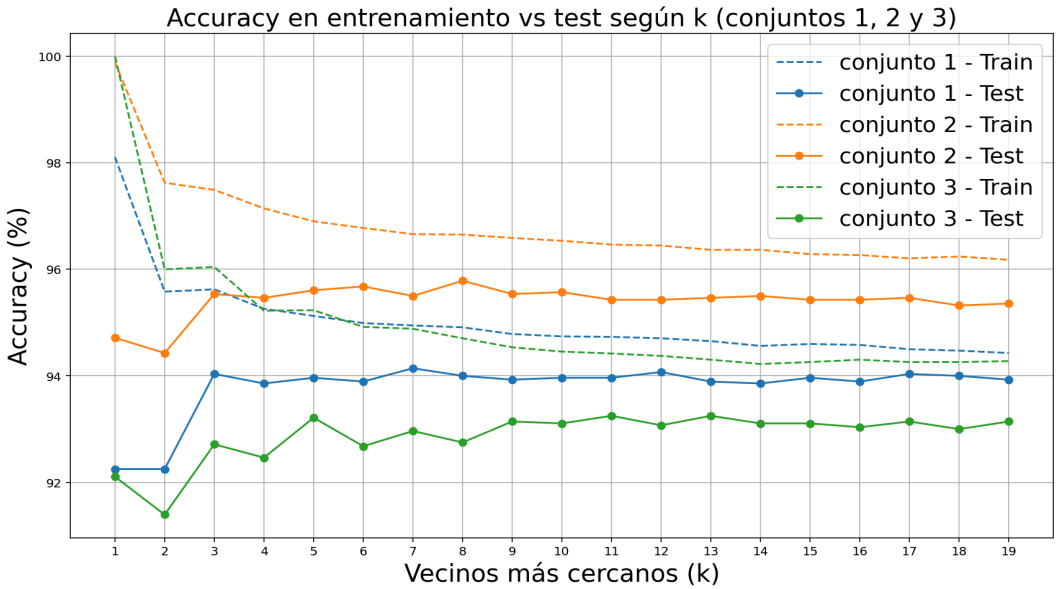

Gráfico n°2 Exactitud evaluada en train y en test de las combinaciones básicas en función del número de vecinos (k)

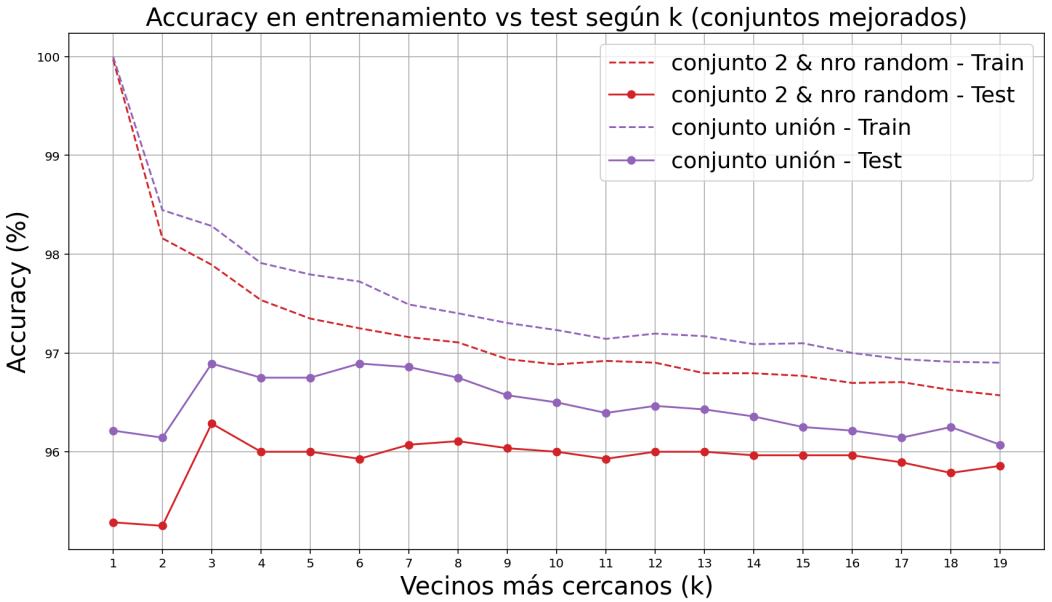

Gráfico n°3 Exactitud evaluada en train y en test de las combinaciones con atributos aleatorios en función del número de vecinos (k). En este caso notamos buenos resultados al agregar atributos adicionales a pesar de haberlos seleccionado aleatorios. Pensamos que puede deberse a que la cantidad de atributos, además de los píxeles en sí, son importantes a la hora de predecir.

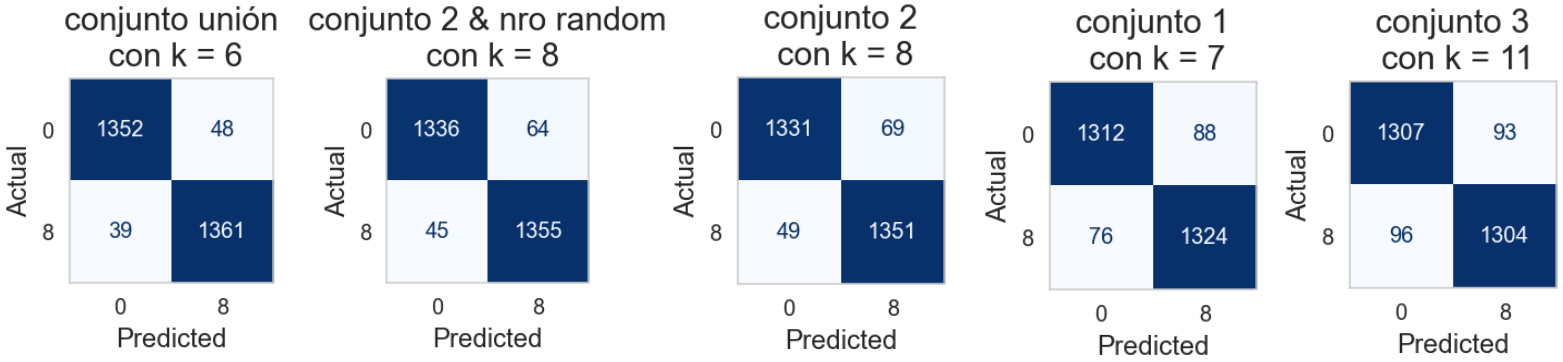

Figura n°7 Matrices de confusión de las combinaciones en el caso de su k con mayor exactitud (considerando únicamente k mayores a 3) sobre el conjunto test, ordenados según exactitud de manera descendente.

COMBINACIÓN	K	EXACTITUD EN TEST	EXACTITUD EN TRAIN	PRECISIÓN	RECALL	F1
Conjunto 1	7	94,14%	94,95%	94,15%	94,14%	94,14%
Conjunto 2	8	95,79%	96,65%	95,8%	95,79%	95,79%
Conjunto 3	11	93,25%	94,42%	93,25%	93,25%	93,25%
Conjunto 2 & nro random	8	96,1%	97,1%	96,11%	96,1%	96,1%
Conjunto unión	6	96,9%	97,7%	96,9%	96,9%	96,9%

Tabla n°1 métricas de cada combinación en el caso de su mejor k (considerando únicamente k mayores a 3). Marcadas en gris las métricas de nuestros atributos seleccionados.

El conjunto unión, que combina cantidad de atributos con calidad es el que presenta mejores métricas, además de una menor diferencia en la evaluación train/test. Le sigue el conjunto 2 & nro random, el cual posee menos atributos pero en su mayoría relevantes. Luego los conjuntos 2 y 1 que tienen la totalidad de atributos seleccionados según su relevancia y, por último, el conjunto 3 que a pesar de contar con una gran cantidad de atributos tiene la menor exactitud por componerse únicamente de selecciones aleatorias. Además de su baja exactitud, se ve la poca robustez de este conjunto ante la variación del número de vecinos, es más inestable. El conjunto con mayor diferencia en la exactitud train/test es el conjunto 2, lo cual podría estar indicando un sobreajuste dentro de este rango de valores de k seleccionados.

Clasificación Multiclase

Buscamos determinar la clase de cualquier prenda del dataset Fashion-MNIST, siendo una clasificación con 10 posibles clases en total. Comenzamos dividiendo al conjunto de datos, un 80% será para el conjunto de datos en desarrollo(dev) y el 20% restante para validación(held-out). Luego, al conjunto dev lo subdividimos en conjuntos train y test para la búsqueda del mejor modelo variando los hiperparámetros del árbol de decisión.

Probamos modelos entrenados variando únicamente su profundidad entre 1 y 10.

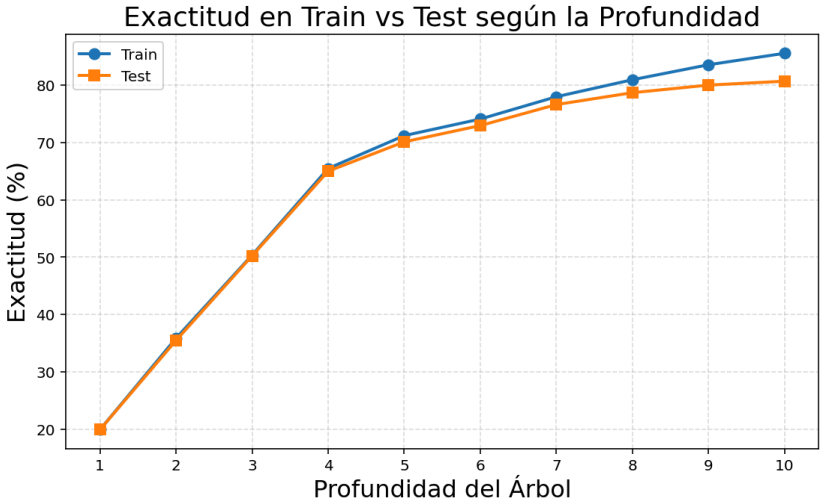

Gráfico n°4 Exactitud del árbol según su profundidad. Notamos una mayor exactitud del modelo a medida que aumenta su profundidad. Sin embargo, a su vez aumenta la brecha entre la exactitud de ambos subconjuntos en una misma profundidad. Consideramos que seguir aumentando podría provocar un sobreajuste.

En un siguiente experimento, continuaremos con una exploración que utiliza la técnica de validación cruzada con K-folding usando k = 5, variando los siguientes hiperparámetros: *'max_depth': [5,7,10], 'min_samples_split': [5,10,15], min_samples_leaf: [2,4] y 'max_features': [None, 'sqrt', 'log2', 0.5]*. Elegimos dejar el criterio de Gini por default ya que hicimos la prueba con Entriopy y, aunque la exactitud mejoraba un 0,20%, el costo era aún mayor. Sin saber aún cuánto varía la exactitud según la profundidad para cada tipo de mf, nos preguntamos si existe una relación entre la cantidad de atributos así como ocurría en la Clasificación Binaria. Estos fueron los resultados:

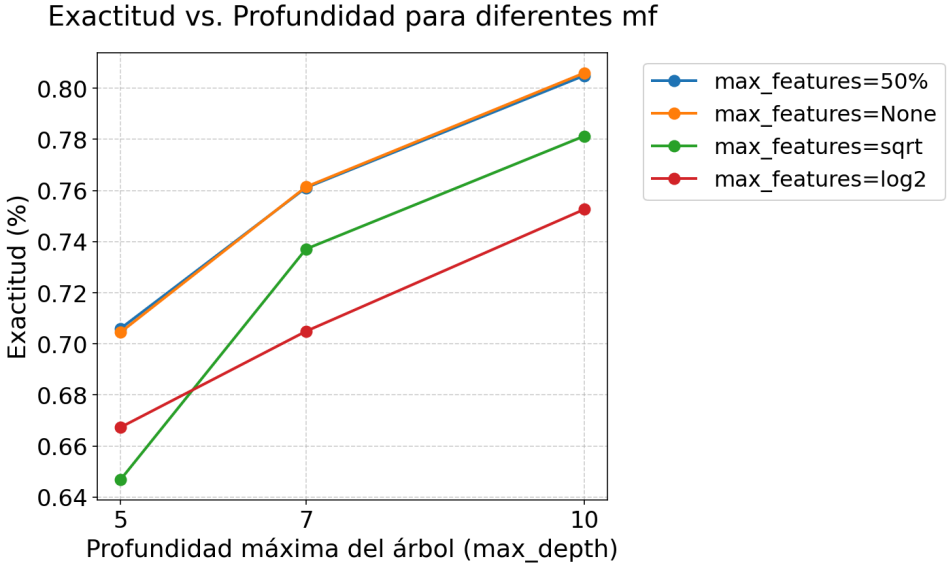

Gráfico n°5 Exactitud de cada tipo de mf en función de la profundidad máxima del árbol. Una observación a destacar es que para cada punto elegido se utilizaron los mss y msl que mayor exactitud daban en tal profundidad, así es posible observar los mejores casos. Por ende son distintos en cada caso.

Esta enorme similitud entre utilizar todos los atributos (mf:None) y la mitad (mf:50%), nos hizo preguntar cuál nos convendría considerar como hiperparámetro final. Hallamos que ambos pueden ser buenos casos al combinarse con distintos hiperparámetros, además de algunas observaciones curiosas: a pesar de que mf:None en su caso óptimo supera en exactitud del dev a mf:50% por 0,82%, al comparar las exactitudes del subconjunto train y test en ambos casos, vemos una menor diferencia al utilizar mf:50%.

MÁXIMA CANT. DE ATRIBUTOS	None: el programa considera una cantidad total por default	50%: consideramos la mitad de atributos (sin pensar de cuáles píxeles consideramos en sí)
Resultados (exactitudes) para evaluar posible sobreajuste	*train_acc : 85.35%* *test_acc : 80.66%* Diferencia de 4,69%	*train_acc : 84.36%* *test_acc : 80.52%* Diferencia de 3,84%
Mínimo de muestras por hoja (msl)	2	4
Mínimo de divisiones (mss)	5	10

Tabla n°2 Comparación de resultados en el caso de profundidad 10 para las dos mejores cantidades de atributos a considerar hallados. Notamos que si seleccionamos la mitad de atributos, para llegar a un valor de exactitud similar a si usaramos el total, debemos duplicar los valores de mss y msl.

Tomamos como modelo final la cantidad de atributos al 50%, con una profundidad 10 mss: 10 y msl: 4, como resultado obtenemos un porcentaje de exactitud Held-Out: 80.29%.

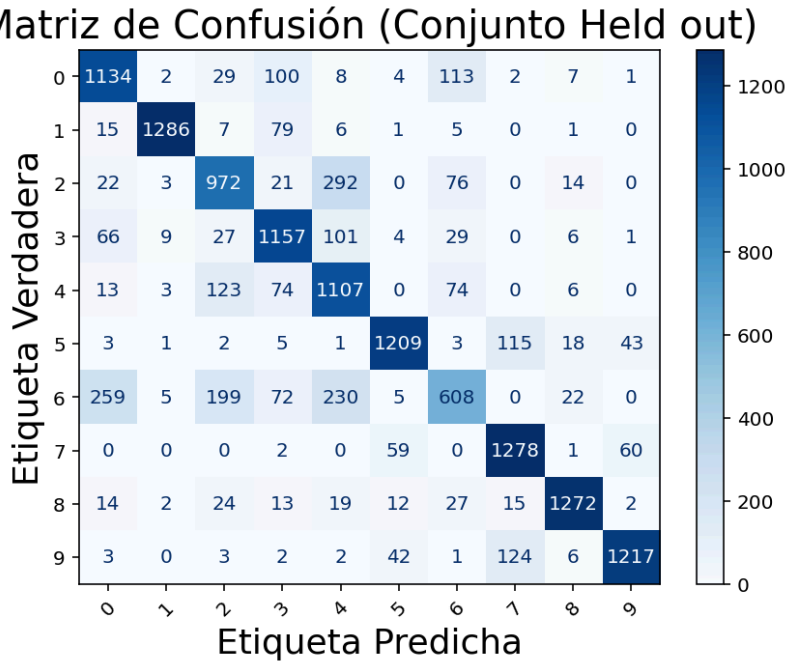

Figura n°8 Matriz de confusión para clasificación multiclase del caso mf:50%. Notamos que la clase con menores predicciones correctas fueron la clase Camisa y Suéter sin poder llegar a las 1000 y que, por otro lado, existen clases con predicciones falsas igual a cero, lo cual es positivo para nuestro objetivo. En conclusiones ampliaremos al respecto.

Conclusiones

En Clasificación Binaria comenzamos con la pregunta de si podíamos hallar aquellos atributos óptimos específicos a partir de los píxeles con intensidades más similares a alguna clase específica, en esa misma posición. Nos encontramos con que es cierta su importancia, pero que además debemos considerar la cantidad de píxeles en sí, sin abusar en cantidad (tanto de atributos como de vecinos) para evitar sobreajuste.

En la Clasificación Multiclase, notamos que las exactitudes eran menores al 85%, a diferencia de la Clasificación Binaria que pudo sobrepasar el 90% en cada conjunto mencionado. Esto ocurrió a pesar de que KNN es un modelo más simple que el árbol binario, el cual considera todos los atributos teniendo muchos hiperparámetros modificables.

Un primer motivo a considerar es que la Clasificación Binaria buscó realizar una distinción entre sólo dos clases que, si bien poseen mayor desviación estándar, fue posible hallar un cierto patrón, que se complementa con la alta diferencia entre el promedio de intensidad de sus píxeles. Pero analicemos los resultados de la Clasificación Multiclase. ¿Qué influyó en la exactitud de la misma, además de la cantidad de clases?

A partir de la Matriz de Confusión de la Clasificación Multiclase hecha con el árbol binario, armamos una tabla que relaciona cada clase con aquella que fue más confundida y menos confundida a la hora de etiquetar.

Clase	Más confundida con	Menos confundida con
0. Remera	Camisa	Botita
1. Pantalón	Vestido	Zapatilla y Botita
2. Suéter	Abrigo	Sandalia, Zapatilla y Botita

3. Vestido	Abrigo	Zapatilla
4. Abrigo	Suéter	Sandalia, Zapatilla y Botita
5. Sandalia	Zapatilla	Pantalón y Abrigo
6. Camisa	Remera	Zapatilla y Botita
7. Zapatilla	Botita	Remera, Pantalón, Suéter, Abrigo y Camisa
8. Bolso	Camisa	Pantalón y Botita
9. Botita	Zapatilla	Pantalón

Tabla n°3. Muestra, para cada clase, con cuál existe una mayor o menor confusión a la hora de predecir

A grandes rasgos parecieran formarse ciertos grupos con clases que se parecen más entre sí. Entre ellos, podemos identificar al grupo de “Calzados” con Sandalia, Zapatilla y Botita, y al de las “Prendas de Arriba” con Camisa, Remera, Suéter y Abrigo; o incluso aquellas clases que tienen una similar relación entre ancho y largo de la prenda como ocurre con los Pantalones y Vestidos. La formación de estos grupos podría haber afectado la exactitud del modelo. Más que nada el grupo de las “Prendas de Arriba” que, si observamos el porcentaje de principales confusiones, notaremos que está principalmente conformado por ellos.

Principales confusiones (>1 % de la clase):

Suéter → Abrigo: 20.86%
Camisa → Remera: 18.50%
Camisa → Abrigo: 16.43%
Camisa → Suéter: 14.21%
Botita → Zapatilla: 8.86%
Abrigo → Suéter: 8.79%
Sandalia → Zapatilla: 8.21%
Remera → Camisa: 8.07%
Vestido → Abrigo: 7.21%
Remera → Vestido: 7.14%

En un gráfico que relaciona a cada clase con el valor alcanzado de su Precisión y Recall, podemos observar cómo afectan estas confusiones a la hora de clasificar.

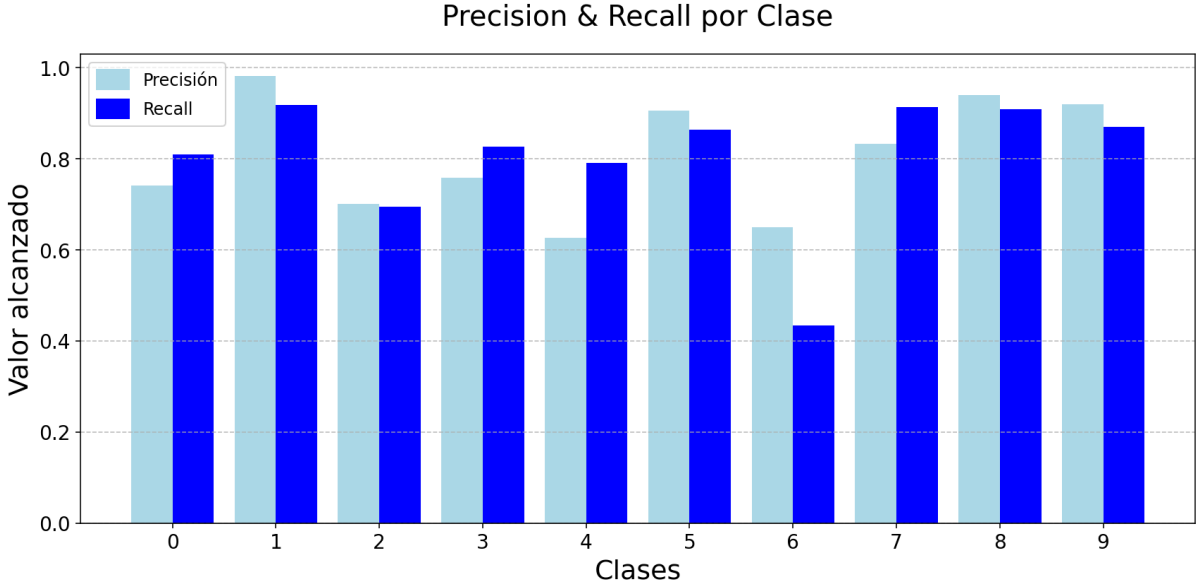

Gráfico n°6 Métricas precisión y recall obtenidas para cada clase

Podemos observar que las clases 1, 5, 7, 8 y 9 (nótese como ninguna forma parte del grupo “Prendas de arriba”) poseen las métricas más altas, lo que indica que son las mejores clasificadas por nuestro modelo. Por otro lado, la clase 6 (Camisa) posee las métricas más bajas, en este caso el modelo tiene peor desempeño de clasificación. Pudimos ver en los porcentajes de mayores confusiones como Camisa era la prenda más nombrada, siendo la misma perteneciente al grupo de “Prendas de Arriba”.

Hay casos donde se ve que la métrica de recall es mayor a la de precisión, como en las clases 0, 3, 4 y 7 (Remera, Vestido, Abrigo y Zapatilla). Esto sugiere que el modelo identifica correctamente la mayoría de los TP, pero también comete más FP. En la clase 6, sucede lo opuesto: el recall es más bajo que la precisión. Es decir, el modelo acierta con mayor frecuencia pero se le “escapan” muchos TP, lo que genera muchos FN.

Podemos decir, como conclusión, que el rendimiento del modelo no es homogéneo entre las distintas clases, a pesar de que el dataset haya estado siempre balanceado.

Una posible explicación a la formación de los grupos que dificultan a la clasificación es lo que mencionamos en el Análisis Exploratorio: existen clases con una gran desviación estándar tal que sería posible que contengan alguna prenda similar al promedio de otra clase. El desempeño depende mucho de la similitud y diferencias entre las clases, además del propio patrón que se pueda deducir de cada una de ellas.

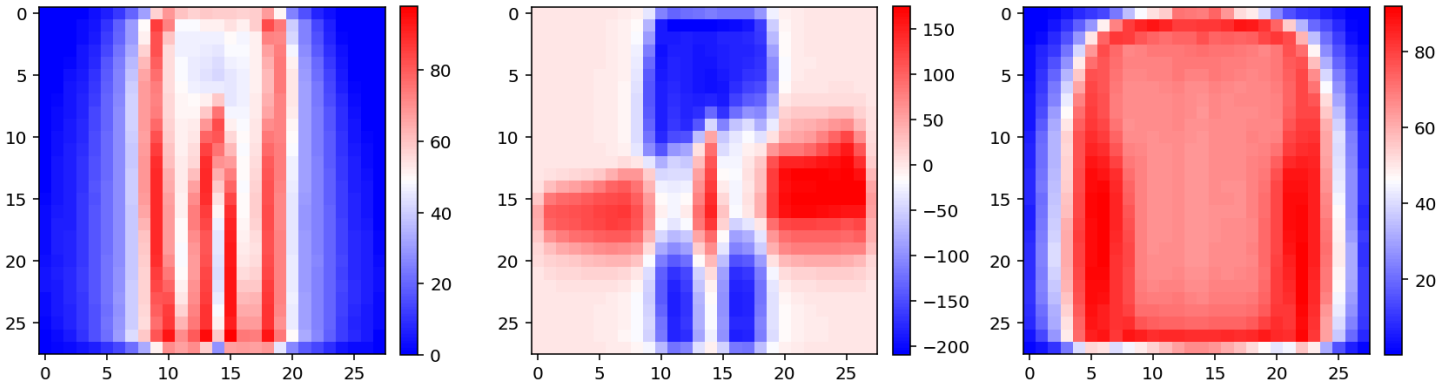

Figura n°9 Desviación Estándar de Pantalón, Diferencia Promedio de los píxeles de las clases Pantalón y Zapatilla, y Desviación Estándar de Camisa, en este orden.

Como posible solución, pensamos que sería interesante armar un modelo cuyos atributos no fueran únicamente píxeles individuales, sino algo más complejo. Operar con conjuntos de píxeles que aporten una información que caracterice a cada clase, así como la relación entre el ancho y largo que ocupa una prenda en la imagen, el promedio de píxeles negros que contiene dentro de la prenda, sectores de la imagen con mayor intensidad, entre otros. De esta forma, podríamos buscar atributos que logren identificar, por ejemplo, la presencia de una “tira” de un Bolso, para diferenciar esta clase de la clase Remera. Además, notamos que por lo general si un bolso **no** posee una tira, entonces éste ocupa casi todo el ancho de la imagen, diferencia clave con la remera que en su lugar ocupa el largo, pero no el ancho total.

Entonces, considerando distintas características que presentan mayor complejidad, tal vez la exactitud del mejor modelo posible para la Clasificación Multiclase podría minimizar las clasificaciones falsas y llegar a un mejor resultado.