DEPARTAMENTO DE COMPUTACION

Facultad de Ciencias Exactas y Naturales - UBA

Trabajo Práctico 02

Clasificación y validación cruzada

27 de diciembre de 2024 — Laboratorio de Datos

Grupo : El Peligroso

Integrante	LU	Correo electrónico
Cocú, Dante	1119/22	dcocu19@gmail.com
Navarro, Solana	906/22	solanan3@gmail.com
Said, Tomás Uriel	170/23	saidtomasur@gmail.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

1. Introducción

El conjunto de datos de Lenguaje de Señas MNIST es una adaptación del popular conjunto de datos MNIST, que originalmente contiene imágenes de dígitos manuscritos. El objetivo de esta adaptación es proporcionar un conjunto de datos para la visión por computadora y para poder reconocer gestos del lenguaje de señas.

Este conjunto de datos consta de imágenes de gestos de mano representando las 24 letras del alfabeto (excluyendo J y Z, que implican movimiento), lo que resulta en un problema multiclase con 24 clases. Cada imagen tiene una resolución de 28x28 píxeles y está representada en escala de grises, con valores de píxeles que van de 0 a 255 (Siendo el 0 el que representa el color negro y 255 al blanco).

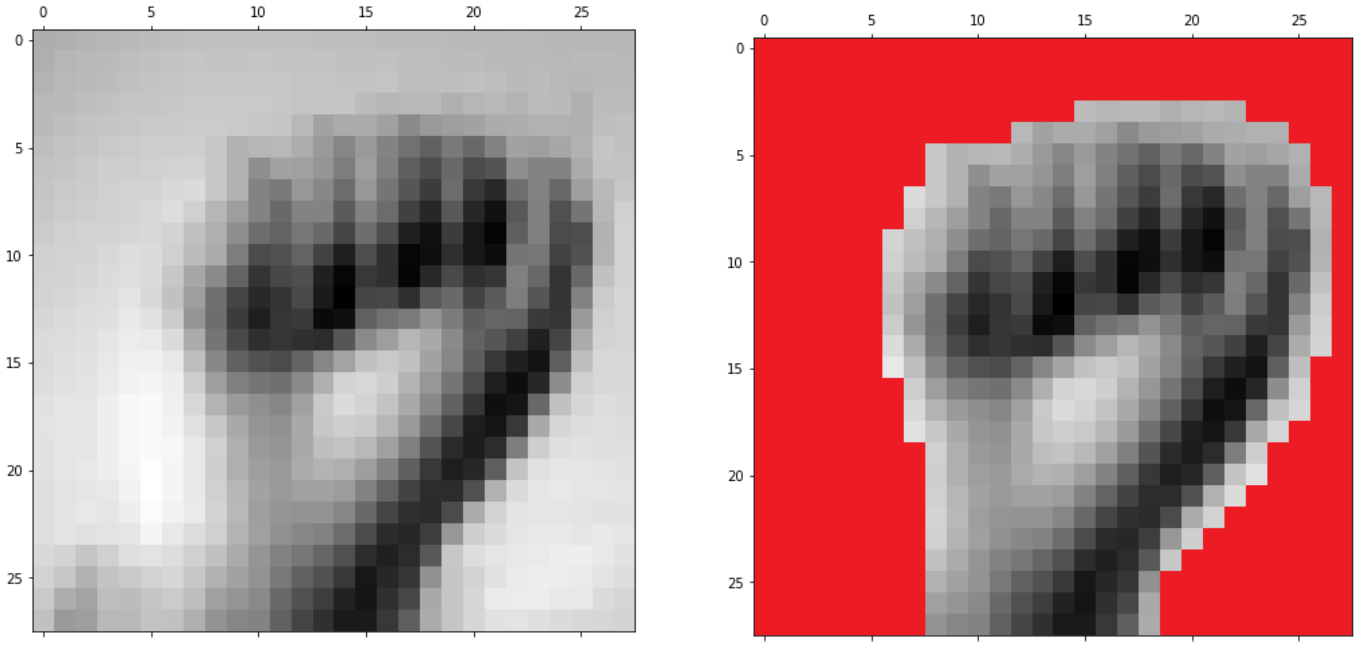
Figura 1: Foto general de todas las señas

2. Analísis Exploratorio

Para realizar un análisis exploratorio del conjunto de datos de Lenguaje de Señas MNIST, es fundamental abordar una serie de preguntas clave que nos permitirán comprender mejor la naturaleza de los datos y la viabilidad de desarrollar modelos de reconocimiento de gestos del lenguaje de señas.

Tengamos en cuenta que explorar un dataset compuesto por imágenes puede ser más complicado. A diferencia de conjuntos no visuales, las imágenes requieren herramientas específicas para la visualización, ya que revela directamente valores numéricos. A pesar de estos desafíos, existen herramientas y técnicas diseñadas para facilitar la exploración y comprensión de datos visuales.

1. **¿Cuáles parecen ser atributos relevantes para predecir la letra a la que corresponde la seña? ¿Cuáles no? ¿Se pueden descartar atributos?**



(a) Mediana de la letra "A"

(b) Atributos no relevantes pìntados de Rojo

Figura 2: Letra A

En cuanto a los atributos relevantes para predecir la letra correspondiente a la seña, se destacan aquellos que muestran cambios significativos en el modelo, como se evidencia en la Figura 2(b). Específicamente, los píxeles que corresponden a la mano en la imagen son los más relevantes, ya que capturan información *crucial* sobre la forma y la estructura de la seña. Por otro lado, los píxeles resaltados en rojo se consideran *irrelevantes*, ya que introducen ruido en el conjunto de datos y no contribuyen de manera significativa a la predicción. Por lo tanto, un modelo que tenga en cuenta solo los píxeles correspondientes a la mano sería mejor.

2. **¿Hay señas que son parecidas entre sí? Por ejemplo, ¿Qué es más fácil de diferenciar: la seña de la E de la seña de la L o la seña de la E de la seña de la M?**

   Al observar todas las imágenes, se puede apreciar que algunas tienen similitudes notables, mientras que otras son más distinguibles. Un ejemplo evidente de estas similitudes se encuentra entre la letra E y la M: aunque no son idénticas, la diferencia entre ellas es mínima. En contraste, las letras E y L son más fáciles de diferenciar. Estas similitudes en las señales pueden representar un desafío al intentar identificar la letra correspondiente a cada señal, ya que podrían generar confusiones en la clasificación.

   Formamos los siguientes graficos para analizar cuan parecidas son las letras M y L en comparacion con la E. Generamos el promedio del valor de todos los píxeles de las imagenes de cada letra.

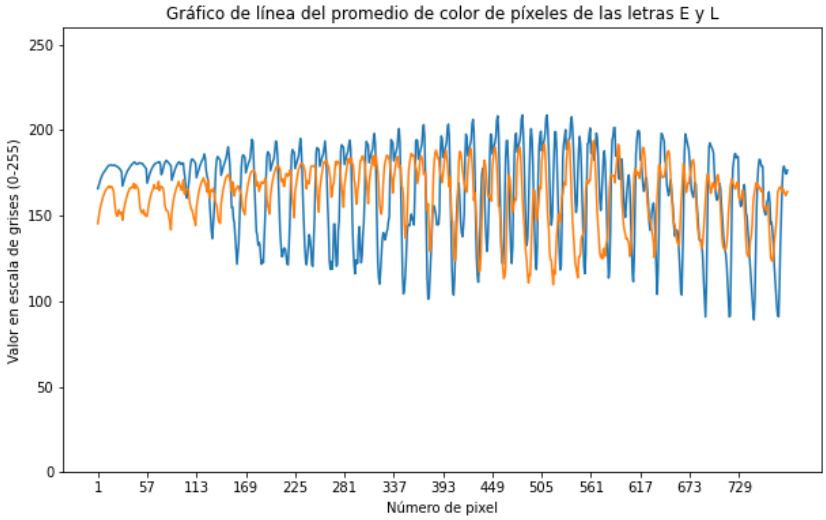


Figura 3: Promedio de las letras E y L

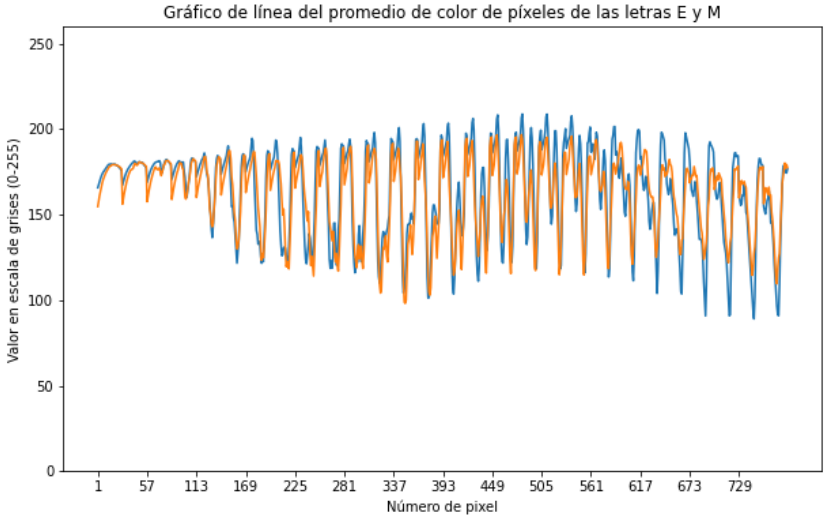
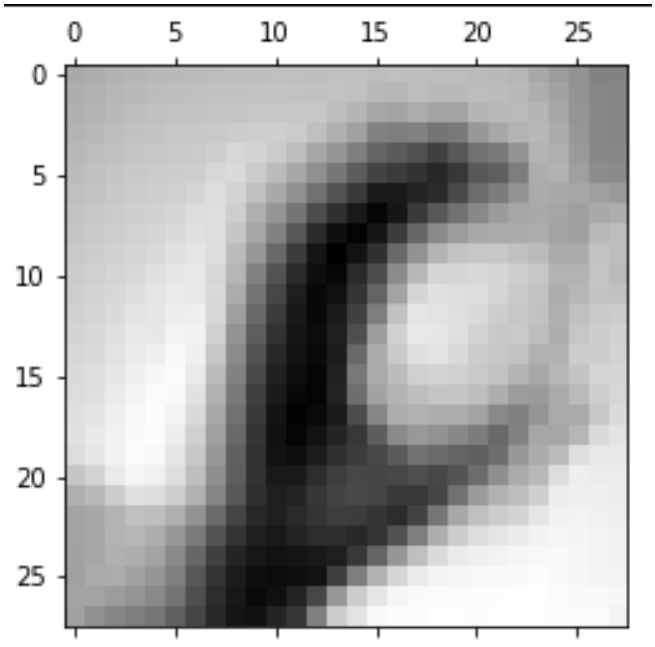


Figura 4: Promedio de las letras E y M

En la Figura 4, se observa que las curvas prácticamente coinciden, mostrando una separación apenas perceptible debido a las variaciones de tono. Esto sugiere que el promedio de los píxeles en las imágenes de las letras E y M es muy similar entre sí. Por otro lado, en la Figura 3, se aprecia una diferencia más marcada entre las curvas, indicando que el promedio de los píxeles en las letras E y L presentan tonos distintivos. Esta diferencia en los tonos sugiere que estas señas no son tan similares entre sí como la E y M.

3. **Tomamos una de las clases, por ejemplo la seña correspondiente a la C, ¿Son todas las imágenes muy similares entre sí?**

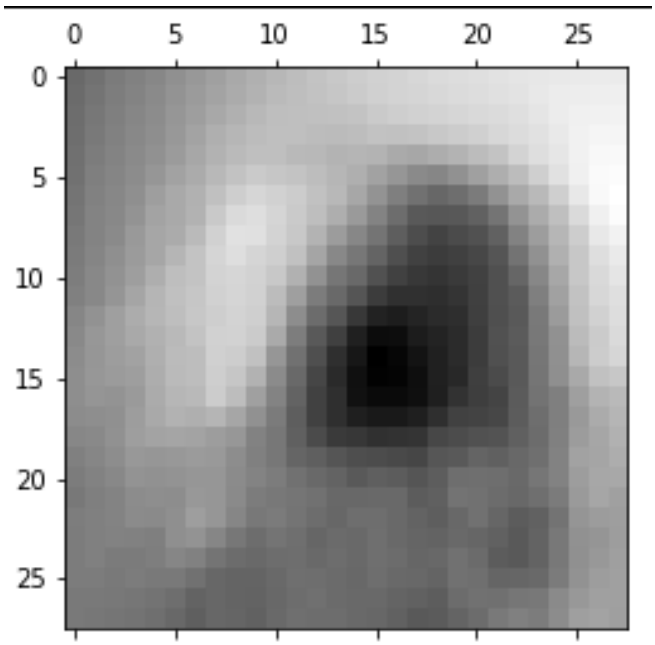
   Esta diferencia de imagenes depende completamente de la clase, ya que algunas de ellas tienen la forma mas clara que en otras. Se nos ocurrio generar una imagen con el promedio de todos los pixeles de una misma clase. En nuestro caso elegimos la seña de la C y la seña de la Q y obtuvimos los siguientes resultados:



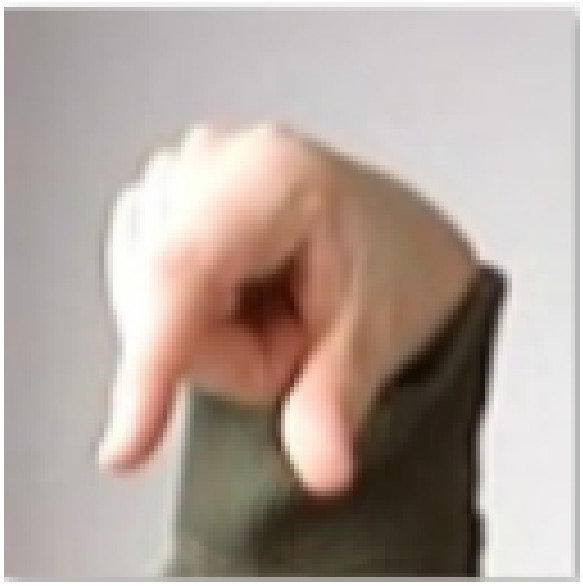
(a) Promedio de la letra C

(b) Foto original de la letra C

Figura 5: Letra C



(a) Promedio de la letra Q



(b) Foto original de la letra Q

Figura 6: Letra Q

Pudimos notar que, las imágenes de la clase C son mucho más similares entre sí. El promedio de todas las imágenes de la C se parece mucho a la figura original. Esto se puede ver en la Figura 5, donde se muestran una de las imágenes individuales de la clase C.

En contraste, todas las imágenes de la clase Q son menos similares entre sí. Se puede ver que la Figura 6 (a) no es tan clara la seña que estamos utilizando, esto ocurre ya que hay una gran variabilidad en las imágenes individuales.

Para ver esta diferencia en numeros, decidimos calcular el promedio del desvio estandar de cada pixel. De esta forma podriamos comparar cual es el resultado que mejor da. Cuanto mas chico, menor variacion hay. Obtuvimos los siguientes resultados:

```
In [21]: conseguir_desvio_estandar_promedio(sign_mnist, 'C')
Out[21]: 39.4751001607521

In [22]: conseguir_desvio_estandar_promedio(sign_mnist, 'Q')
Out[22]: 50.10678921222328
```

Figura 7: Promedio de desvio estandar de la C y la Q

La diferencia en la variabilidad entre las clases C y Q probablemente se deba a la forma de las dos señas. La seña C es una forma simple y geométrica, mientras que la seña Q es más compleja y tiene más detalles. Esto hace que la seña C sea más fácil de reproducir de manera consistente, mientras que la seña Q es más susceptible a la variación individual.

3. Clasificación Binaria

Dada una imagen se desea responder la siguiente pregunta: ¿La imagen corresponde a una seña de la L o a una seña de la A?

Para determinar si una imagen corresponde a la seña de la letra L o la A, primero seleccionamos del conjunto de datos original solo las imágenes de esas dos letras. Luego, verificamos la cantidad de imágenes por cada letra para ver si están balanceadas.

```
In [35]: letras_LyA['label'].value_counts()
Out[35]:
11    1241
0     1126
Name: label, dtype: int64
```

Figura 8: 0 corresponde a la A (47,57 %), 11 corresponde a la L (52,43 %). No están completamente balanceados pero el desbalanceo no es significativo

Posteriormente, dividimos el data set en conjuntos diferentes de train (80 %) y test(20 %) para poder entrenar y evaluar un modelo de clasificación de KNN. De esta manera poder asignar a una seña la letra A y L correctamente.

Despues, generamos una funcion donde indicamos la cantidad de KNN que queremos que busque y cuantos atributos queremos que use. Para asi poder hacer un scatter plot y comparar con distintos KNN y atributos. De esta forma pudimos ver cual de ellos conviene utilizar.

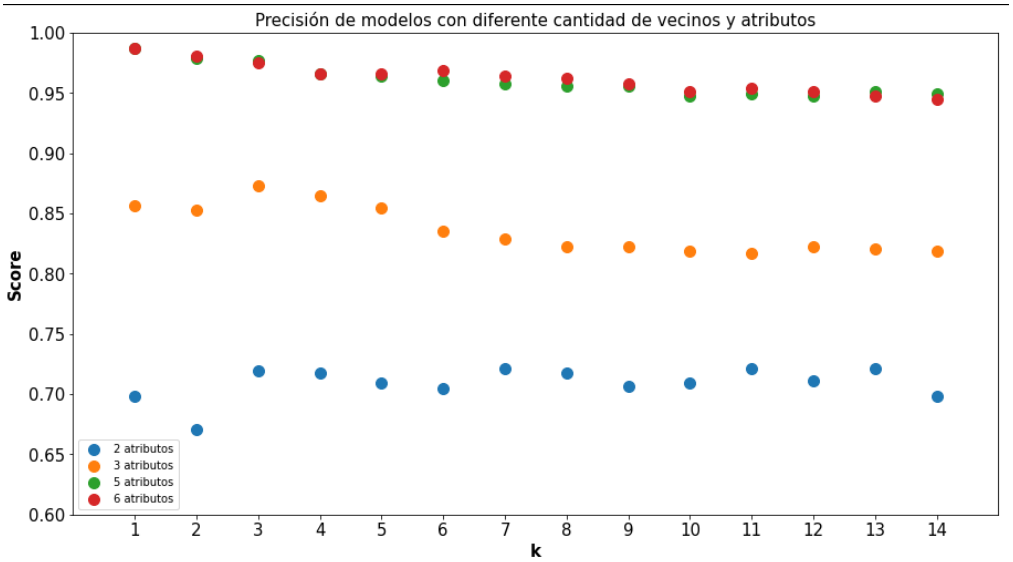
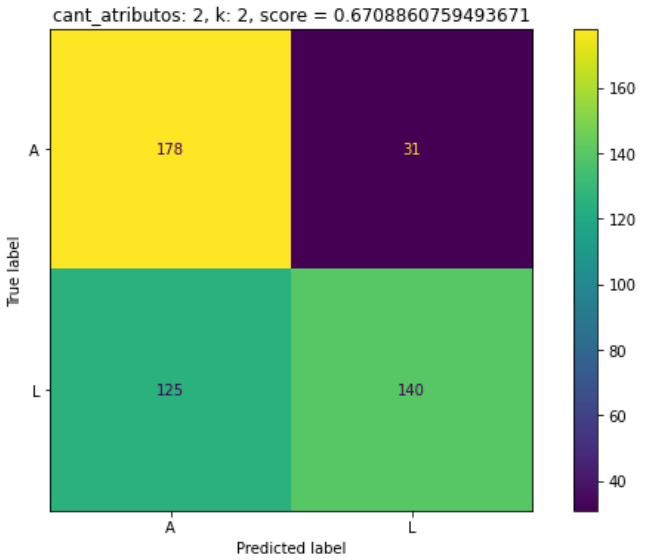


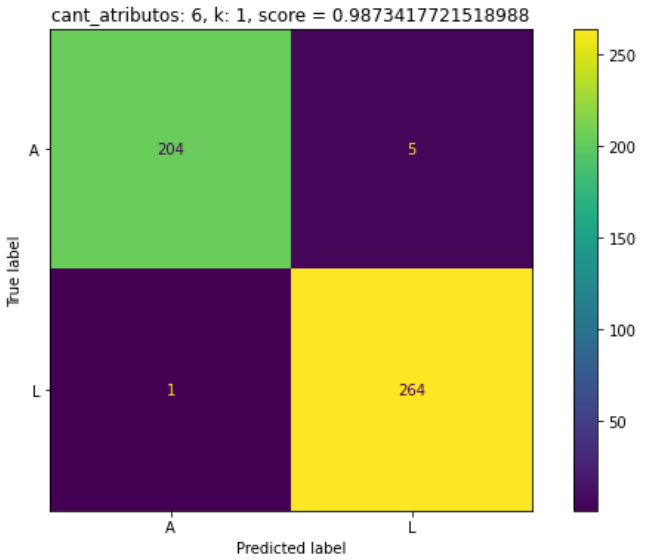
Figura 9: Comparación entre modelos de KNN

En la Figura 9 se realiza una comparación de distintos atributos y cantidades de vecinos más cercanos (KNN) para determinar la mejor combinación en términos de rendimiento. Se observa claramente que los mejor resultado son utilizando 5 o 6 atributos y 1 vecino más cercano. Es evidente que a partir de 5 atributos, las diferencias en el rendimiento son mínimas. Además, hay casos en los que el número óptimo de vecinos no es 1; por ejemplo, al utilizar 2 o 3 atributos, el mejor rendimiento se alcanza con 3 vecinos más cercanos en lugar de 1. Esto sugiere que la elección de la cantidad de vecinos depende de la configuración específica de atributos utilizados. Una observación es que los atributos los elegimos de manera aleatoria, con lo cual cuando hay pocos atributos, además de generar un modelo subajustado, la precisión depende mucho de los atributos que hayamos elegido, ya que no es lo mismo elegir atributos irrelevantes que relevantes. Muy probablemente si nuestro modelo decidió aleatoriamente atributos no relevantes, su rendimiento va a ser malo.

Para poder comparar el rendimiento de nuestro modelo basandonos en distintas cantidades de KNN y de atributos, decidimos hacer matrices de confusión:



(a) Matriz de confusión de un modelo KNN impreciso



(b) Matriz de confusión de un modelo KNN con buena precisión

Figura 10: Matrices de confusión

En la Figura 10 (a), se presenta el modelo con 2 atributos y 2 vecinos más cercanos (KNN), donde se detallan los True Positives, True Negatives, False Positives y False Negatives obtenidos. La matriz de confusión revela que el modelo alcanza una precisión del 67.09 % al distinguir entre las letras A y L. Sin embargo, se observa un número considerable de errores al clasificar imágenes de la letra A, se confunde 31 veces con la letra L, mientras que 125 imágenes de la letra L se confunden con la letra A. Esto sugiere que al utilizar un número limitado de atributos y solo 2 vecinos, es posible que el modelo esté subajustado, lo que podría explicar su rendimiento no óptimo.

En cambio, la matriz de confusión de la Figura 10 (b) revela que el modelo con 6 atributos y 1 KNN supera ampliamente al modelo con 2 atributos y 2 KNN. El Accuracy del modelo ha aumentado a un 98 %, con una notable reducción en los errores de clasificación de la letra A y L. La especificidad y la sensibilidad también han mejorado, lo que indica una mejor capacidad del modelo para identificar correctamente ambas letras. Este avance significativo se atribuye al aumento en la cantidad de atributos y a la disminucion de KNN utilizados.

4. Clasificación Multiclase

Dada una imagen se desea responder la siguiente pregunta: ¿A cuál de las vocales corresponde la seña en la imagen?

Para clasificar imágenes de señas de las vocales A, E, I, O y U, seleccionamos las imágenes de esas vocales del conjunto original y las dividimos en DEV y EVAL. Para luego, entrenar un modelo de árbol de decisión con diferentes profundidades y seleccionar el mejor árbol usando validación cruzada con k-folding en el conjunto de DEV.

A la hora de seleccionar la profundidad, no sabiamos que altura usar, por lo que definimos alturas al azar. Utilizamos GridSearch de sklearn para averiguar que criterio y que profundidad era la mejor. Al usar el comando ***search.cv_results_*** obtuvimos el siguiente resultado:

```
'params': [{'criterion': 'gini', 'max_depth': 3},
 {'criterion': 'gini', 'max_depth': 5},
 {'criterion': 'gini', 'max_depth': 7},
 {'criterion': 'gini', 'max_depth': 9},
 {'criterion': 'gini', 'max_depth': 10},
 {'criterion': 'gini', 'max_depth': 11},
 {'criterion': 'gini', 'max_depth': 12},
 {'criterion': 'gini', 'max_depth': 13},
 {'criterion': 'gini', 'max_depth': 14},
 {'criterion': 'entropy', 'max_depth': 3},
 {'criterion': 'entropy', 'max_depth': 5},
 {'criterion': 'entropy', 'max_depth': 7},
 {'criterion': 'entropy', 'max_depth': 9},
 {'criterion': 'entropy', 'max_depth': 10},
 {'criterion': 'entropy', 'max_depth': 11},
 {'criterion': 'entropy', 'max_depth': 12},
 {'criterion': 'entropy', 'max_depth': 13},
 {'criterion': 'entropy', 'max_depth': 14}],
```

Figura 11: Parametros evaluados

```
'rank_test_score': array([17, 16, 14, 12, 11,  8,  9, 10,  7, 18, 15, 13,  6,  5,  1,  4,  2,
        3])}
```

Figura 12: Ranking de los parametros

Con esta información, llegamos a la conclusión de que el criterio ‘entropy ’con una profundidad de 11 obtiene el mejor puntaje. Le siguen ‘entropy ’con profundidades 13 y 14. Observamos que algunos modelos, al aumentar la profundidad, tienden a sobreajustarse, lo que deteriora su score. Por esta razón, el mejor valor de profundidad no es necesariamente el más alto.

Optamos por el modelo que mejor rendimiento mostró, para asi poder evaluarlo con los datos EVAL. Al hacer esto obtuvimos un score de 0.9625. A continuación, decidimos elaborar una matriz de confusión para visualizar los errores cometidos por nuestro modelo seleccionado.

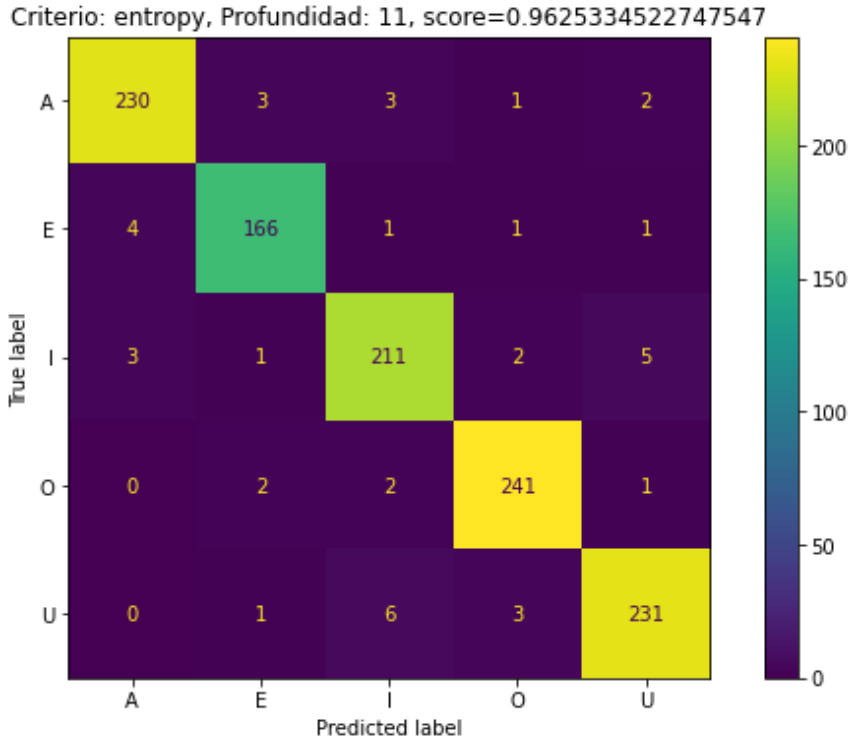


Figura 13: Matriz de confusión

La matriz de confusión varía desde el color violeta hasta el amarillo, donde el violeta representa los valores más bajos y el amarillo los más altos. Con la métrica de entropía, encontramos el mejor puntaje con una profundidad de 11, lo que resulta en una matriz de confusión con un 96 % de precisión.

Si bien se observan algunos errores en las predicciones del modelo, por ejemplo cuando la letra real es ”U” y el modelo

predice "I" (6 errores) o cuando la letra real es "I" y el modelo predice "U" (5 errores), estos errores no son relevantes para determinar la utilidad del modelo, ya que son los menos frecuentes en comparación con los "True Positives", que representan la mayoría de las predicciones.

Adicionalmente, se observa que el modelo tiene una tasa de error muy baja al predecir la letra "O" con solo 5 errores. Tambien notamos que la letra mas dicha erroneamente es la "I".

5. Conclusión

En conclusion, el análisis exploratorio del conjunto de datos del Lenguaje de Señas MNIST nos proporcionó una comprensión detallada de la naturaleza de los datos y nos guió en el desarrollo de modelos efectivos para la clasificación de gestos del lenguaje de señas. Abordamos preguntas clave sobre la relevancia de los atributos, la similitud entre las señas y la variabilidad dentro de las clases. Utilizando herramientas de visualización adecuadas, identificamos atributos relevantes para la clasificación y observamos similitudes y diferencias entre algunas señas, lo que destacó la importancia de considerar características distintivas.

El proceso de clasificación binaria entre las letras L y A se llevó a cabo tras seleccionar exclusivamente las imágenes correspondientes a estas dos letras del conjunto de datos original y verificar que la distribución entre ambas clases estuviera balanceada, se dividió el conjunto en datos de entrenamiento y prueba. La construcción del modelo se basó en una función que permitía ajustar la cantidad de vecinos más cercanos (KNN) y la cantidad de atributos a utilizar, lo que facilitó la exploración de diferentes configuraciones. La evaluación del rendimiento reveló que un modelo con 6 atributos y 1 vecino KNN superó significativamente al modelo de 2 atributos.

Para la clasificación multiclase de las vocales, se implementaron modelos de árbol de decisión y se evaluaron diferentes profundidades para encontrar la configuración óptima, utilizando validación cruzada con k-folding en el conjunto de desarrollo (DEV). Para seleccionar la profundidad óptima y el mejor criterio, se utilizó GridSearch de sklearn, lo que proporcionó una lista de parámetros evaluados y su respectivo ranking de rendimiento. Con esta información, se determinó que el criterio 'entropy 'con una profundidad de 11 obtiene el mejor puntaje. Se observó que algunos modelos tendían a sobreajustarse al aumentar la profundidad, lo que deterioraba su puntaje, destacando así la importancia de encontrar un equilibrio entre la complejidad del modelo y su capacidad predictiva. Si bien se observaron algunos errores en las predicciones del modelo, fueron poco frecuentes y no afectaron significativamente su utilidad general.