Laboratorio de Datos
Trabajo Práctico 02

TP 02: Clasificación y validación cruzada

Grupo: JVC

Integrantes:

- *Carlos Pórcel,*
- *Valentino Pons,*
- *Ricardo Javier Suarez*

Introducción

Contamos con un data frame llamado Sign Language MNIST el cual básicamente contiene imágenes de letras representadas con lenguaje de señas. En base a los datos proporcionados por el dataframe el siguiente informe trata de resolver diversos problemas, como por ejemplo dada una la imagen de una seña poder predecir a qué letra corresponde, entre otros.
Para responder esas preguntas se realizó un análisis, experimentación, entrenamiento y evaluación de modelos de machine learning como K-nearest neighbors y Árboles de decisión.

Resumen

Se realizó la observación del data frame Sign Language MNIST para ver con qué tipos de datos nos enfrentamos. Se cuentan con numerosas columnas cuyos valores representan píxeles de una imagen y una en específico que representaba el tipo de letra a la que corresponde la imagen (es decir, una letra del abecedario). A continuación se realizara:

- Análisis exploratorio
- Experimentación y evaluación de modelos
- Decidir si una imagen corresponde a una seña de la letra L o A con un modelo KNN
- Responder a qué vocal corresponde la seña de la imagen con el modelo de Árbol de Decisión

Análisis Exploratorio de Datos

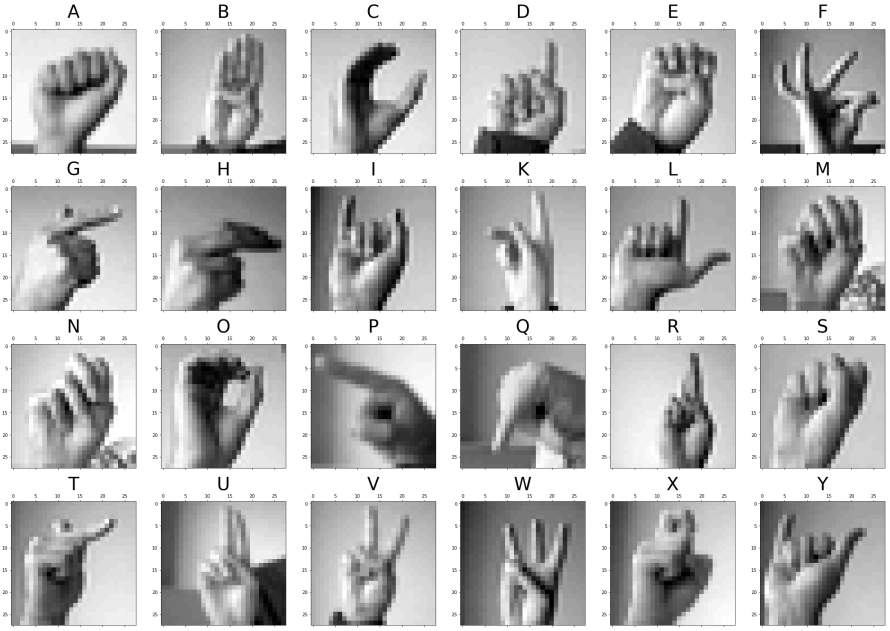
El data frame Sign Language MNIST presenta 785 atributos. Cada fila está conformada por la primera columna llamada ‘label’, la cual contiene un número que representa una letra y las demás columnas las cuales almacenan píxeles. Estos al ser combinados generan la imagen de una seña de la letra correspondiente a la columna ‘label’ anteriormente mencionada en lenguaje de señas estadounidense.
Contamos con un total de 24 clases distintas las cuales corresponden a las 24 letras del abecedario estadounidense, excluyendo J y Z las cuales necesitan movimiento para representarlas.

Haciendo una breve exploración de las imágenes que representan las señas nos dimos cuenta que los atributos (píxeles) de mayor interés e importancia para poder identificar qué letra es, son aquellos que componen la mano. Normalmente son los píxeles del centro los que proporcionan mayor distinción ya que es donde realmente varían de una seña a otra. También la columna ‘label’ es importantísima ya que como se dijo más arriba identifica a qué letra corresponde cada tupla .

Por último , los atributos no mencionados corresponden al fondo de la imagen(píxeles más cercanos a los bordes) los cuales no aportan mucho valor ya que no se diferencian entre las distintas imágenes y podrían ser eliminados, aunque es un proceso muy costoso de realizar pues no todos los píxeles que generan la mano están centrados.

Fuente:. https://www.kaggle.com/datasets/datamunge/sign-language-mnist

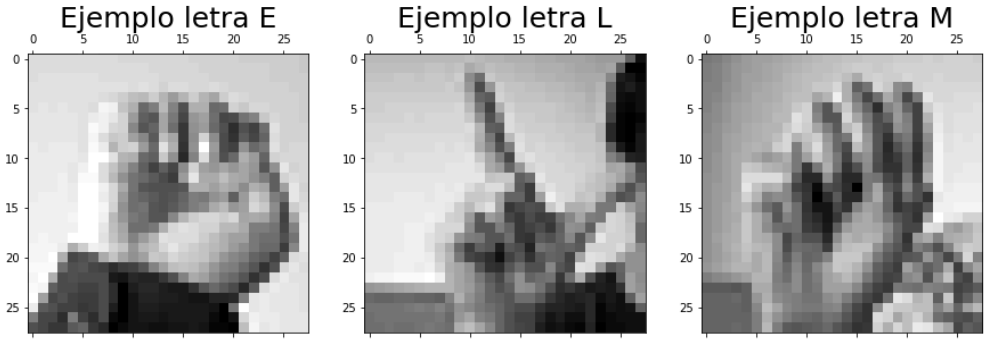
Columnas : 785
Cantidad de datos(ejemplos): 27455



Esta imagen muestra ejemplos de las señas de las letras presentes en Sign Language MNIST con su correspondiente letra.

Similitudes entre señas de distinta letra

Como se observa más abajo hay señas las cuales son parecidas entre sí, como el caso de la E con la M. A simple vista es más fácil diferenciar la señas de la letra E con la de la letra L que con la M.



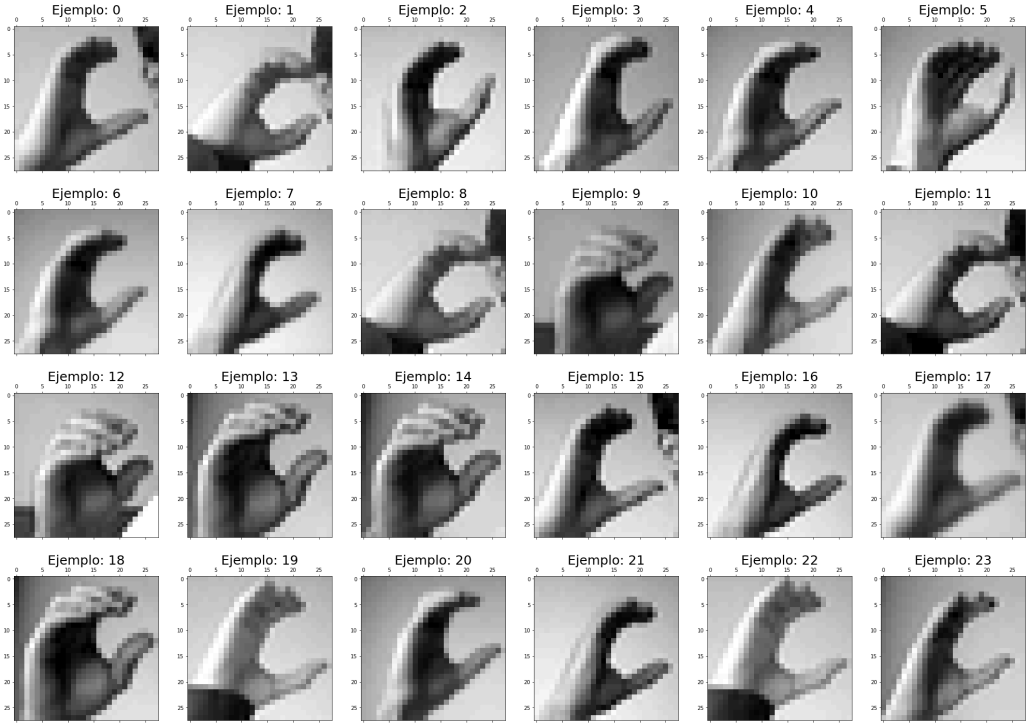
Muestra de las letras E, L y M del conjuntos de imágenes de Sign Language MNIST

Similitud entre señas de la misma letra

Para poder evaluar las similitudes de los datos de las señas para la misma letra tomaremos a la letra C en particular.

Seleccionamos 24 muestras de la letra C y pudimos notar que hay diferencias entre unas y otras señas. Si bien todas tienden a realizar el mismo símbolo, es decir, que los píxeles que componen la mano están ubicados generalmente en los mismos lugares, se pueden notar diferencias como la distancia de los dedos con el pulgar o la inclinación de la mano. Sin

embargo, las diferencias no son tan graves. Por lo que siguen siendo útiles para el entrenamiento de un modelo predictivo.



Muestras de la letra C del conjunto de imágenes Sign Language MNIST.

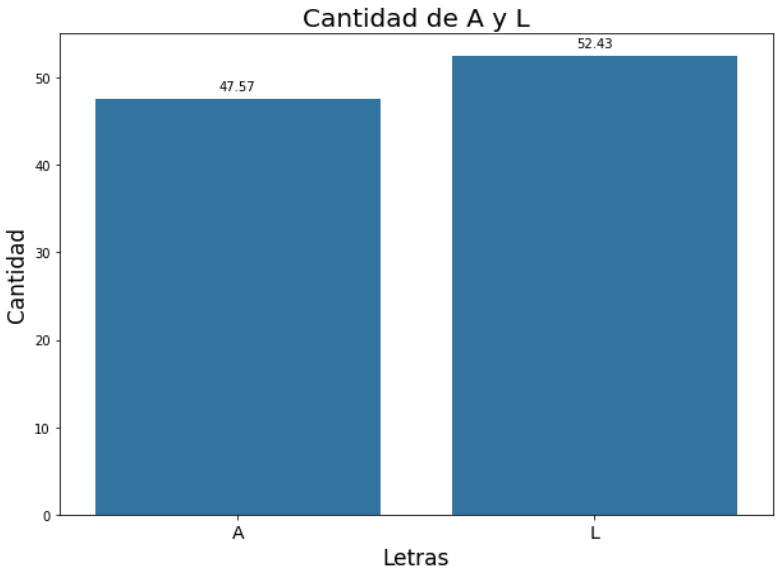
Dificultades encontradas en la exploración de datos

Trabajar con este dataset, el cual cada uno de sus atributos representa pixeles y no datos concretos genera una incomodidad a la hora de manipular los datos, ya que cada pixel por sí solo no es representativo; en el momento que se combinan es cuando ganan valor ya que se puede dar forma a una imagen.
Por esta razón hay que estar trabajando con un gran número de atributos y a la vez tener en cuenta lo que representan esos píxeles. Seleccionar píxeles específicos genera una dificultad ya que requiere pensar de forma matricial a los datos para su manipulación.

Comparando con el dataset del Titanic, donde podíamos elegir un atributo en particular (como Sexo, Edad, etc.), trabajar con las imágenes conlleva un mayor trabajo el analizar y explorar.Trabajar con datos cuyos atributos poseen una característica en particular es mucho más sencillo que hacerlo con datos que individualmente dicen muy poco, ya que nos ahorramos todo el proceso de unir valores para recién obtener algo que nos sirva.

¿La imagen corresponde a una seña de la L o a una seña de la A?

Para resolver la primera pregunta seleccionamos solo las muestras de la letra A y L, dejando un total de 2367 muestras. Del total de muestra hay 47.57% de muestras correspondiente a la letra A (en particular 1126 muestras), mientras que la L tiene 52.43% de las muestras (en particular 1241 muestras). Si bien hay que tener presente esta diferencia se puede decir que las muestras están balanceadas, ya que no hay diferencia significativa (115 muestras).



Porcentaje de muestras para las letra A y L respecto al conjunto de muestra de ambas letras juntas.

Modelo: k-nearest neighbors(KNN)

Para poder realizar la predicción utilizaremos el modelo KNN e iremos experimentando con los distintos atributos de las muestras y los k necesarios para el modelo.

Experimentacion

Queremos evaluar qué tan importante es la posición de los dedos contra la inclinación de la muñeca por lo que elegimos 3 píxeles que representan los dedos y otros que representan la muñeca. También queremos ver si es mejor que los píxeles estén de forma vertical u horizontal.

Finalmente para los mejores atributos (pixeles de cada experimento), con respecto a la exactitud en los datos de test, evaluaremos la cantidad de n_neighbors (k) que mejore el modelo.

Primer Experimento

Tomamos los píxeles del medio de la imagen de forma horizontal que generalmente representan (en la seña de la A y L) el dedo medio e índice, siendo los atributos `'pixel392','pixel393','pixel394'`. Estos dedos son los que más cambian al hacer la seña.

Entreno modelo con estos atributos [`'pixel392','pixel393','pixel394'`] y k = 2
Exactitud con datos de train: 0.90
Validación cruzada con 5 k-folds: 0.81
Exactitud con datos de test: 0.81

Segundo Experimento

En el segundo experimento tomamos píxeles del medio pero de forma vertical siendo los siguientes: `'pixel365','pixel393','pixel421',` que generalmente conforman el dedo medio.

Entreno modelo con estos atributos [`'pixel365','pixel393','pixel421'`] y k = 2
Exactitud con datos de train: 0.81
Validación cruzada con 5 k-folds: 0.67
Exactitud con datos de test: 0.66

Tercer Experimento

Para el tercer experimento tomamos los atributos `‘pixel769’, ‘pixel770’, ‘pixel771’,` que se ubican en la parte media inferior de forma horizontal (que en general pertenecen a la muñeca de la mano o cerca de ella).

Entreno modelo con estos atributos ['pixel769', 'pixel770', 'pixel771'] y k = 2
Exactitud con datos de train: 0.91
Validación cruzada con 5 k-folds: 0.83
Exactitud con datos de test: 0.83

Cuarto Experimento

Para el cuarto experimento tomamos los atributos `‘pixel714’, ‘pixel742’, ‘pixel770’,` que se ubican en la parte media inferior de forma vertical (que en general pertenecen a la muñeca de la mano o cerca de ella).

Entreno modelo con estos atributos [`‘pixel714’, ‘pixel742’, ‘pixel770’]` y k = 2
Exactitud con datos de train: 0.91
Validación cruzada con 5 k-folds: 0.81
Exactitud con datos de test: 0.83

Observaciones de los experimentos: con estos 4 primeros experimentos se puede concluir que la muñeca mejora la predicción de la letra que representa la seña, además que la mejor forma de seleccionar los píxeles es horizontal.

Quinto Experimento

El mejor modelo hasta ahora es el que toma píxeles de la muñeca de la mano. Ahora la pregunta sería si mejora o empeora al cambiar la cantidad de atributos (pixeles) que tomamos que representen la muñeca.

Los resultado serian los siguientes:

Experimento A:

Entreno modelo con estos atributos `[‘pixel769’, 'pixel770']` y k = 2
Exactitud con datos de train: 0.86
Validación cruzada con 5 k-folds: 0.73
Exactitud con datos de test: 0.73

Experimento B

Entreno modelo con estos atributos `['pixel768', 'pixel769', 'pixel770', 'pixel771']` y k = 2
Exactitud con datos de train: 0.95
Validación cruzada con 5 k-folds: 0.88
Exactitud con datos de test: 0.88

Experimento C

Entreno modelo con estos atributos `['pixel767', 'pixel768', 'pixel769', 'pixel770', 'pixel771', 'pixel772']` y k = 2
Exactitud con datos de train: 0.98
Validación cruzada con 5 k-folds: 0.94
Exactitud con datos de test: 0.93

Observamos que al aumentar la cantidad de atributos, la exactitud también aumenta favorablemente, y esto tiene que ver con que iremos teniendo más píxeles y en el mejor de los casos tendremos la imágen completa.

Mejorar el Hiper Parámetro: n_neighbors

Probamos más modelos con el hiper parámetro n_neighbors(k) desde 1 al 9 para los atributos del tercer experimento, y del quinto experimento sólo los atributos del primer y último experimento.

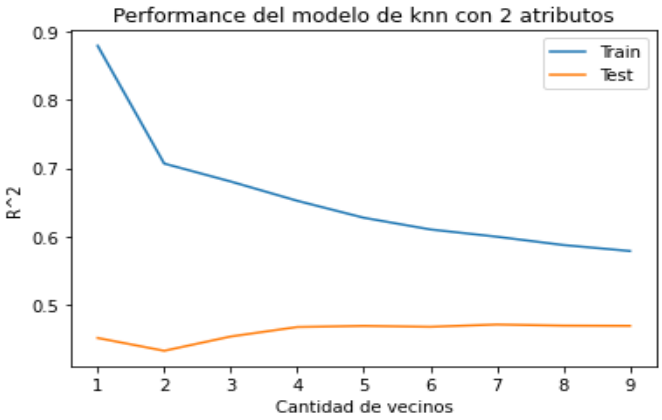
Con esas variaciones pudimos observar, de manera general, que a medida que aumentamos los ‘k’ la exactitud con los datos de test disminuye, y que si aumentamos la cantidad de atributos la exactitud con los datos de test aumenta. En el caso de ir variando los ‘k’, encontramos que el pico con mayor exactitud en la mayoría de las veces se suele encontrar con k=1 o k=2.

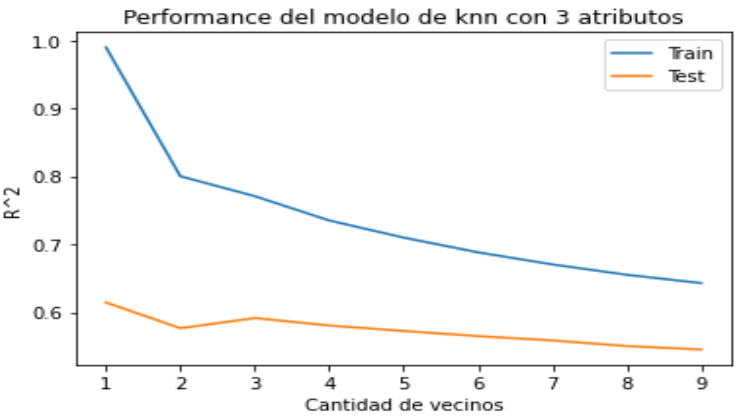
Por ejemplo: para un caso donde tenemos 2 atributos como `'pixel769'` y `'pixel770'` con k=2, obtenemos una exactitud de 0.73 en los datos de test. En cambio si usamos 6 atributos como `'pixel767', 'pixel768', 'pixel769', 'pixel770', 'pixel771'` y `'pixel772'` con el mismo k de antes, es decir k=2, obtenemos una exactitud de 0.93 en los datos de test.

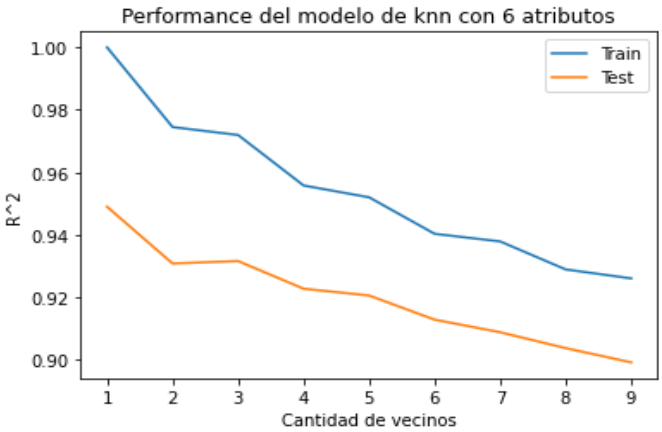
De esta manera, vemos claramente que al aumentar la cantidad de atributos, en ese caso de píxeles, la exactitud aumenta notablemente.

Por último, si combinamos las dos conclusiones, tenemos que cuanto menor sea el ‘k’ y mayor sea la cantidad de atributos, mayor va a ser la exactitud con los datos de test y de esta forma podremos decidir si se trata de una imagen que corresponde a una seña de la letra **L** o a una de la **A**.

Dejamos a continuación material que ayudará a la comprensión del análisis de los distintos modelos:







Rendimiento del modelo KNN con 2, 3 y 6 atributos para n_neighbors *desde 1 a 9*

¿A cuál de las vocales corresponde la seña en la imagen?

Luego de filtrar los datos que no corresponden a vocales en un nuevo data frame llamado solo_vocales se separaron 70% de los datos en training y los restantes en test.

Se evaluaron 3 modelos distintos de árboles con criterion = 'entropy' , pero cambiando la profundidad de cada uno. Arbol2 con max_depth = 2 , Arbol4 con maxdepth = 4 y Arbol8 con max_depth = 8.

Luego de entrenarlos se observó que el Arbol8, el cual tenía mayor profundidad entre los tres, retornaba mejores resultados de score en comparación a los otros 2 modelos. Luego le seguía Arbol4 y por ultimo Arbol2.

Para tener datos de score más reales se sometieron los 3 modelos a una validación cruzada k-folding (con k = 5) con los datos de train y evaluando la métrica exactitud. A continuación se muestran los promedios de los resultados de las evaluaciones para cada modelo.

modelos_evaluados_en_train:

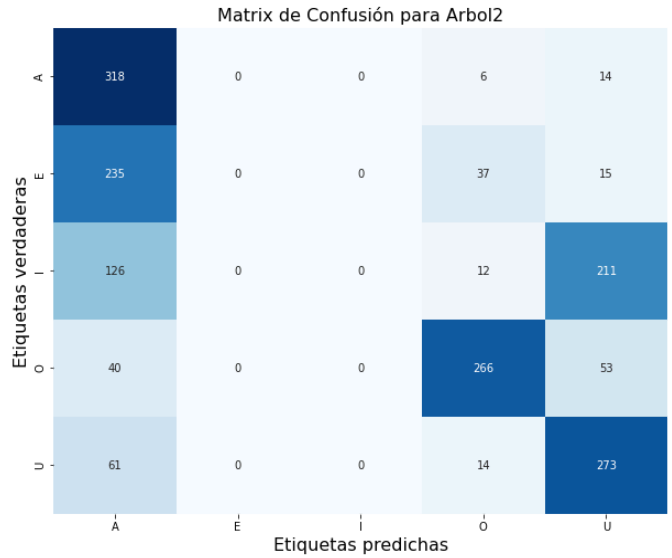
Metrica	Arbol2	Arbol4	Arbol8
Exactitud	0.507	0.736	0.943

Luego se sometieron los modelos al testeo utilizando los datos de testing y evaluando no solo exactitud, sino también recall y precisión. A continuación se muestra la tabla obtenida.

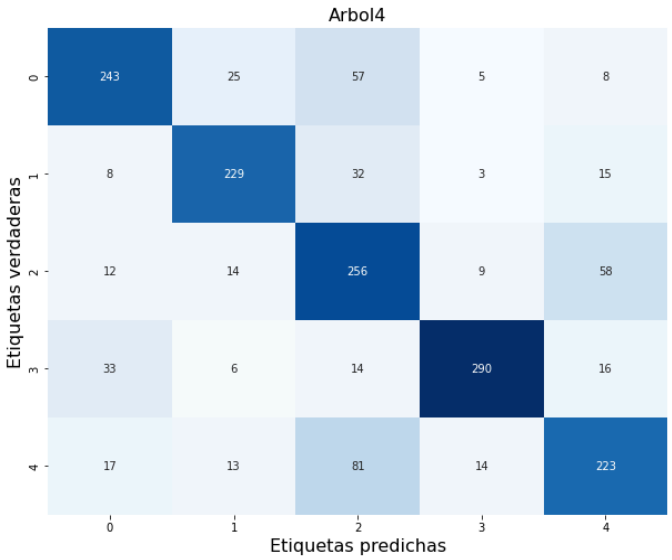
modelos_evaluados_en_test:

Metrica	Arbol2	Arbol4	Arbol8
Exactitud	0.509	0.738	0.944
Recall	0.509	0.738	0.944
Precisión	0.351	0.750	0.945

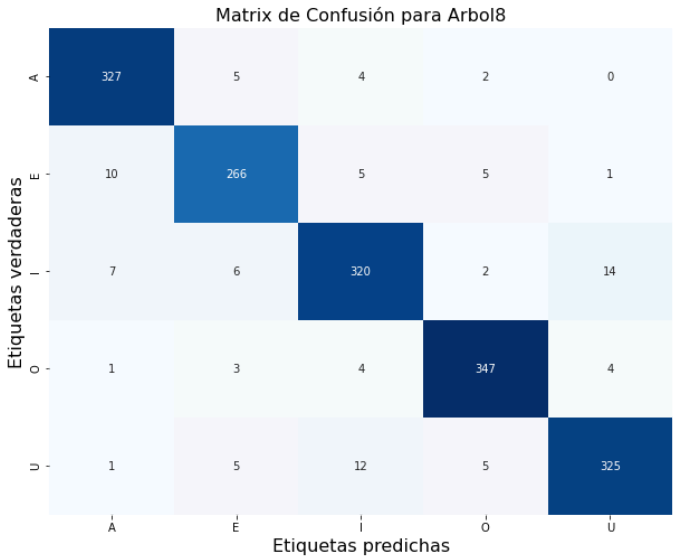
Si bien todo pareciera apuntar que el mejor modelo es el Arbol8 se realizó una última comparación utilizando matrices de confusión.



Matrices de Confusión para Árbol de Decisión con max_deep = 2 para las vocales.



Matrices de Confusión para Árbol de Decisión con max_deep = 4 para las vocales.



Matrices de Confusión para Árbol de Decisión con max_deep = 8 para las vocales.

Analizando las matrices, el Arbol8 es el que posee una diagonal más marcada y por lo tanto es el más efectivo a la hora de predecir. Todo indica que el aumento de profundidad en los modelos planteados mejoró drásticamente la clasificación y predicción de datos.

Conclusiones

A lo largo de este proyecto se trabajó con imágenes, por lo que, creemos que trabajar con un conjunto de datos donde cada valor por sí solo no representa o dice muy poco individualmente, como el caso de un píxel, es complicado. Es más conveniente trabajar con datos que posean atributos diferentes o de distinta clase, como el caso de ‘Sexo’, ‘Edad’, etc.

Para poder predecir con muy pocos pixeles una imagen, primero había que decidir cuales tomábamos y de que zona de la imagen. Experimentamos con píxeles que se encuentren en el medio de la imagen, y luego con los del medio inferior lo que sería pixeles a la altura de la muñeca de la mano, y observamos con nuestros modelos que los mejores resultados de test para poder predecir se encontraban en los pixeles del medio inferior.

A la hora de comparar modelos K-nearest Neighbors (KNN) utilizando distintos atributos y distintos valores de K, observamos que cuanto mayor sea la cantidad de atributos y menor sea el valor de K, mejores resultados tenemos en las medidas de evaluación, para el caso del valor de K, en la mayoría de los casos funciona mejor con K=1 o K=2, y en cuanto a la cantidad de atributos, concluimos que cuantos más píxeles tengamos más información tendremos de la imagen; en nuestro caso serían más partes de la mano.

Finalmente, el mejor modelo KNN al que llegamos se entrenó con estos atributos `['pixel767', 'pixel768', 'pixel769', 'pixel770', 'pixel771', 'pixel772']` y k = 2.

Exactitud con datos de train: 0.98
Validación cruzada con 5 k-folds: 0.94
Exactitud con datos de test: 0.93

Por último, para detectar e intentar predecir a cuál de las vocales corresponde una imagen se evaluaron 3 modelos distintos de árboles, cuyas profundidades elegidas fueron 2, 4 y 8.

Se observó que a medida que aumentaban las profundidades, los resultados de test mejoraban notablemente. Estos resultados se analizaron testeando métricas multiclase y matrices de confusión. Con todo esto dicho es que nos quedamos como modelo con el árbol de profundidad 8 el cual arrojó mejores resultados y que fueron detallados anteriormente de manera explícita.