Imagen que contiene Logotipo

Descripción generada automáticamente

## Universidad de Buenos Aires

Facultad de Ciencias Exactas y Naturales

Departamento de Computación

**Trabajo Práctico 02:**

**Clasificación y validación cruzada.**

**El caso EMNIST**

**Grupo ABC**

**Alumnos:**

Abbate, Lucas Ignacio (134/21)

Becker, Guillermo (616/90)

Cevasco Rieck, Jorge Augusto (230/23)

Profesor: Turjanski, Pablo

Materia: Laboratorio de Datos

Cuatrimestre: 1C2024

## Resumen

## Introducción

En el presente trabajo, se abordará la tarea de clasificación de imágenes utilizando el conjunto de datos EMNIST, específicamente el subconjunto que contiene las letras mayúsculas manuscritas. El objetivo principal de este TP es poner en práctica los conceptos de clasificación y selección de modelos mediante la utilización de técnicas de validación cruzada.

En 1995, el NIST (National Institute of Standards and Technology) estadounidense, desarrolla la “NIST Special Database 19” (SD19), una de las primeras bases publicadas de texto manuscrito. Para su desarrollo, 3.699 empleados del Instituto, estudiantes de colegio secundario y médicos, completaron a mano un formulario (Fig. 1) que contenía números y letras que tenían que replicar debajo, además de el inicio de la Constitución Nacional Americana.

Estos formularios fueron escaneados y sus caracteres fueron separados, etiquetados y procesados (corrección de intensidad, centrado, etc.) llegando a 810.000 imágenes de 28x28 píxeles. Años más tarde, la SD19 se convirtió en la “EMNIST database” (Extended Modified NIST database), una actualización de la ya popularizada MNIST, que, además de contar con los dígitos del 0 al 9, contaba con las letras del abecedario en mayúscula y en minúscula.

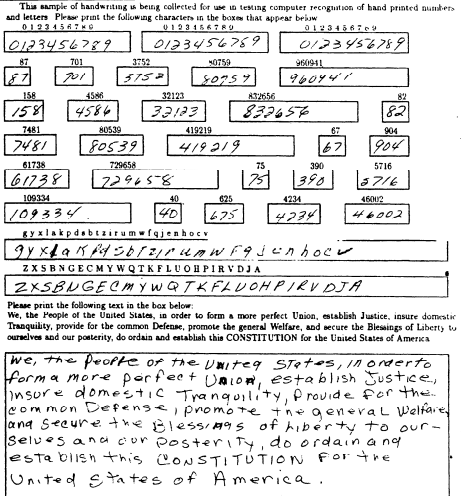


Fig. 1: Formulario para el Desarrollo de la base original del NIST. Extraído de https://www.nist.gov/system/files/documents/srd/nistsd19.pdf

Tanto MNIST como EMNIST se convirtieron en bases de datos estándar para el entrenamiento y desarrollo de modelos de aprendizaje automático: consisten en grandes cantidades de datos etiquetados, normalizados y aislados, y su procesamiento permite el desarrollo de herramientas de OCR (“optical character recognition”, reconocimiento óptico de caracteres), una aplicación concreta y de probada utilidad de estos modelos.

Desde hace décadas que distintas industrias utilizan modelos de este tipo para automatizar la categorización de textos. Uno de los ejemplos más comunes es el Servicio Postal de los Estados Unidos (USPS), que desde 2005 utiliza distintos algoritmos automáticos para “leer” las direcciones y códigos postales escritos a mano en los sobres de los correos. Se genera luego un puntaje de confiabilidad, que si supera cierto umbral es reenviado a la oficina correspondiente de dicho código postal. Caso contrario, es enviado a una oficina en la que personas leen y clasifican a mano el código postal y la dirección, generando además nuevos datos etiquetados para continuar el entrenamiento de los modelos.

En este trabajo se utilizaron distintos algoritmos para la generación de un clasificador de los caracteres. En primera instancia, se entrenó un clasificador binario, utilizando un modelo de kNN (“k -Nearest Neighbors”, o k-Vecinos más cercanos) para diferenciar las letras “L” de las letras “A”. Luego, se ajustó un clasificador multiclase, cuyo objetivo es distinguir todas las vocales. Para este último modelo se preparó un árbol de decisión, y se escogieron los hiperparámetros que mejor ajustan a nuestro set de datos utilizando validación cruzada (k-folding).

### Análisis exploratorio

La base de EMNIST utilizada en este trabajo está compuesta por 2.400 imágenes de cada letra del alfabeto inglés (62.400 en total). Cada una de esas imágenes consta de 784 valores, que representan la intensidad de cada píxel en una grilla de 28x28, en una escala que, para todas las observaciones, va de 0 a 255 (como se puede apreciar en la Fig. 2).

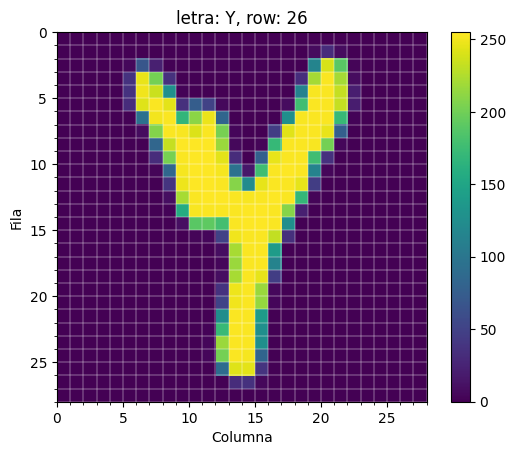


Fig. 2: representación en grilla de la 26ma observación. El color representa el valor de intensidad de dicho píxel.

Cada fila en nuestra base de datos representa una letra, y cada columna representa la intensidad de dicho píxel. Como la grilla (28x28) está representada en una serie de una dimensión, y la base utilizada no aclara si los datos vienen ordenados por filas o por columnas (es decir, si los primeros 28 valores son la primera fila o la primera columna), se utilizó la observación graficada en la Fig. 2 para deducirlo, ya que se sabe que una “Y” debería contar, por ejemplo, en la cuarta fila, con dos picos de intensidad separados por un vacío. En efecto, se ve en la Fig. 3 que no coinciden las intensidades de los valores correspondientes a la fila 4 de la figura anterior, y a los valores 112 a 139 (es decir, los que corresponderían a la cuarta fila, si la base estuviese ordenada de fila en fila). No obstante, este último gráfico sí coincide con las intensidades de la cuarta columna de la Fig. 2.

Habiendo concluido que los datos de la base estaban ordenados de “columna en columna”, se renombraron las variables con su número de fila y columna para su posterior análisis.

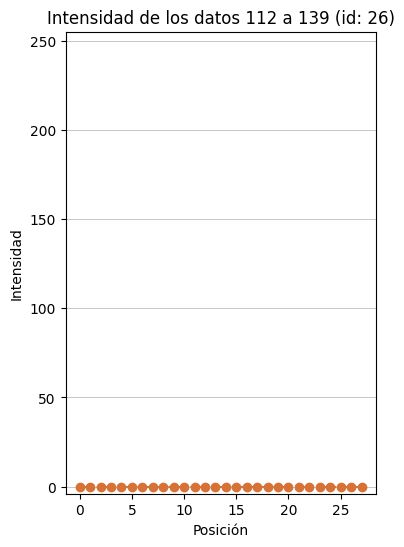
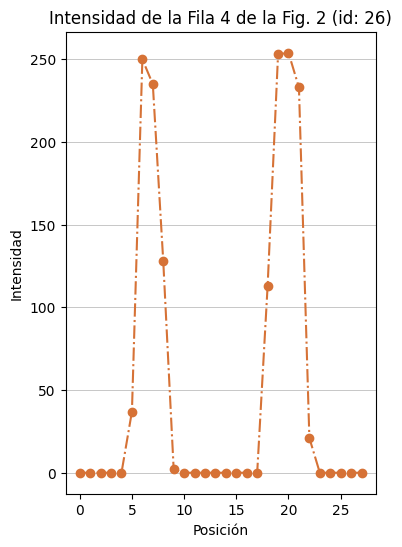


Fig. 3: gráficos correspondientes a la cuarta fila de la Fig. 2 y a los datos que deberían representar a la cuarta fila si estuviesen ordenados de fila en fila

En primera instancia, se muestran en la Fig. 4 la media, mediana y máximo de cada píxel entre todas las entradas. En el gráfico de los máximos, se ve que, aun teniendo una muestra de 86.400 imágenes, las dos primeras y últimas filas/columnas presentan pocos píxeles con intensidades significativas. Esta noción se observa aún más al analizar las medias y las medianas: las tres primeras y últimas filas/columnas dan valores prácticamente nulos, mientras que en las medianas esto se expande a mayores regiones.

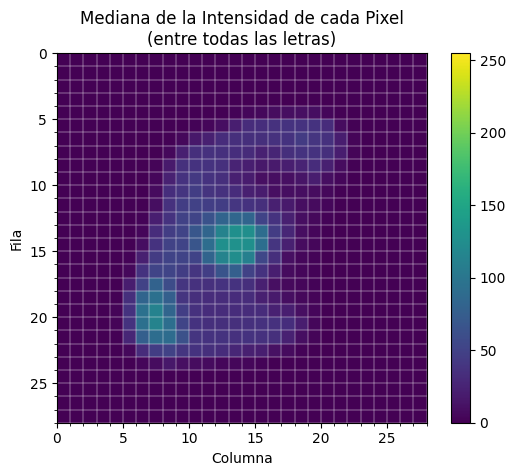
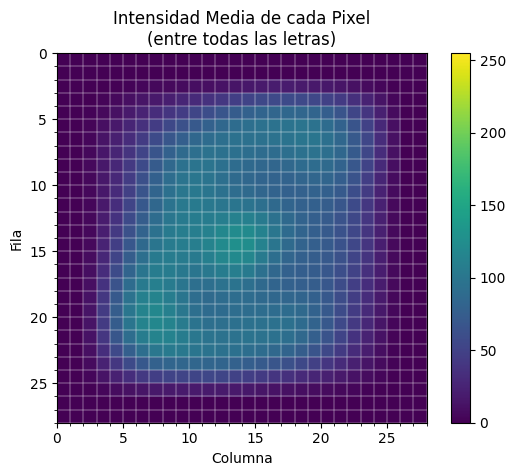
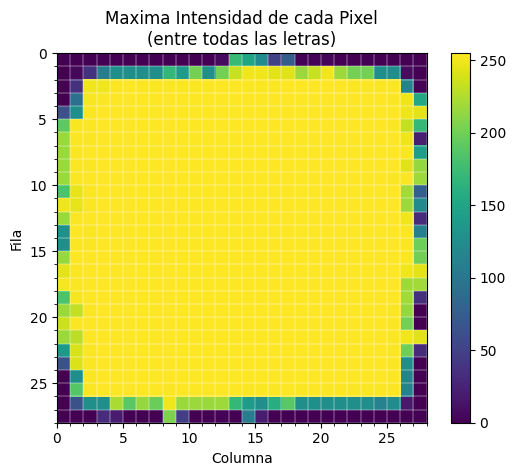


Fig. 4: Intensidad mediana, máxima y media por píxel entre todas las letras de la base.

Esto seguramente se deba a que, en la mayoría de las letras, la mayoría de los píxeles están vacíos (su intensidad es nula), pero aquellos que tienen valor tienen intensidad alta (ya que todas las imágenes están normalizadas a 255), generando entonces que la media se vea muy afectada por los valores extremos. A modo de ejemplo, se muestran en la Fig. 5 histogramas de algunos píxeles para todas las letras. Se observa que, como explicamos antes, la media suele ser significativamente mayor a la mediana, exceptuando en el caso de la columna 14 y la fila 15, donde la mediana es mucho mayor a la de los otros casos, casi empatando a la media.

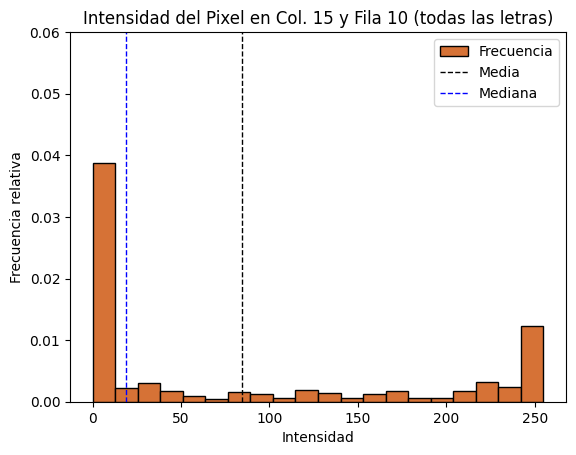
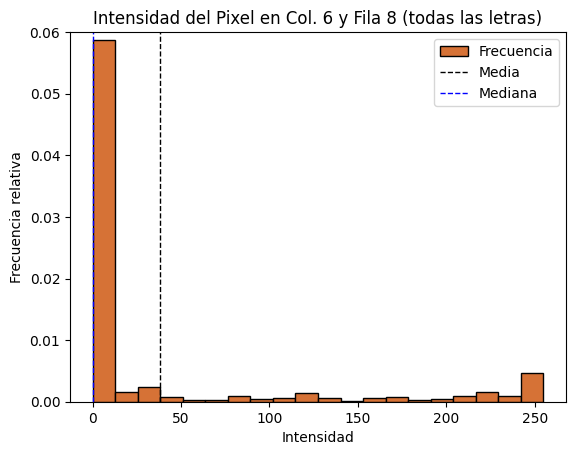
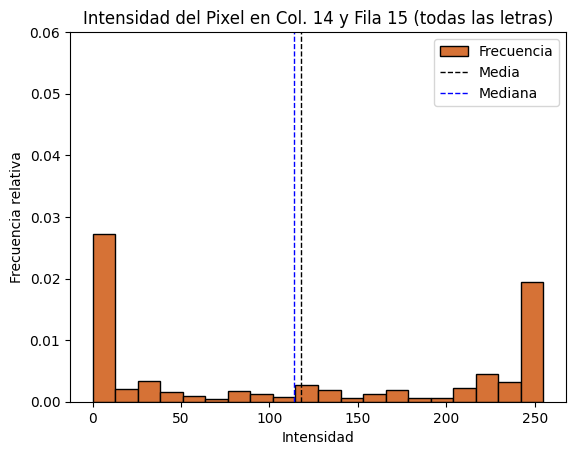
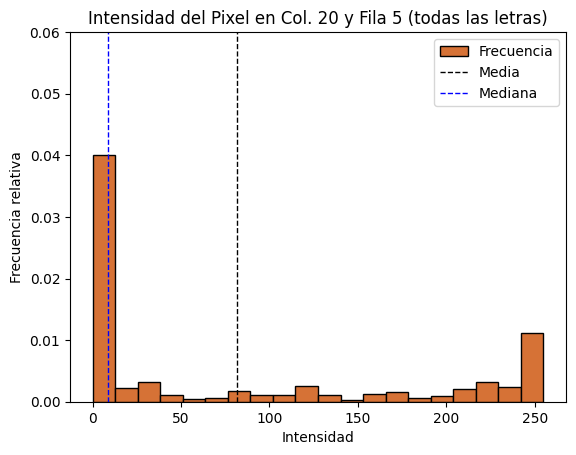


Fig. 5: Histogramas de la intensidad entre todas las letras para distintos pixeles, junto a su respectiva media y mediana

Esta diferencia entre medias y medianas se observa no sólo en las primeras y últimas filas/columnas, sino también en la intensidad de las zonas centrales: en el gráfico de las medianas se observa una figura similar a una ε, mientras que en el de las medias esta figura está mucho más difuminada por toda la zona central. Esto puede interpretarse como que, al analizar las medias, los píxeles son más similares entre sí, mientras que analizando las medianas, se observa con mayor intensidad una separación entre clases.

En efecto, explorando la diferenciabilidad entre clases, se realizaron gráficos de la resta entre medias y medianas para distintos pares de letras. En la Fig. 6, se puede ver que las diferencias de intensidades medias, tanto para la comparación de “E” con “M”, como para “L” con “I”, son de menor magnitud que las mismas comparaciones entre medianas. Se muestran en la Tabla 1 los desvíos de las diferencias entre medias, medianas y su diferencia en porcentaje para estos y otros pares de letras. Se ve que, en todos los casos, el desvío entre las diferencias de medianas es mayor al mismo para las medias.

|  |  |  |  |
| --- | --- | --- | --- |
| Par de letras | Desvío de la diferencia entre medianas | Desvío de la diferencia entre medias | Diferencia porcentual |
| E y M | 85.26 | 61.10 | 39.6% |
| I y L | 16.14 | 25.24 | 56.4% |
| E y L | 73.97 | 44.46 | 66.4% |
| E y I | 78.97 | 53.83 | 46.7% |

Tabla 1: Desvío de la diferencia entre medias y medianas de distintos pares de letras

---- Análisis de la variabilidad dentro de cada letra, mostrar (¿media mediana y varianza?) de la C y alguna otra letra mas compleja----

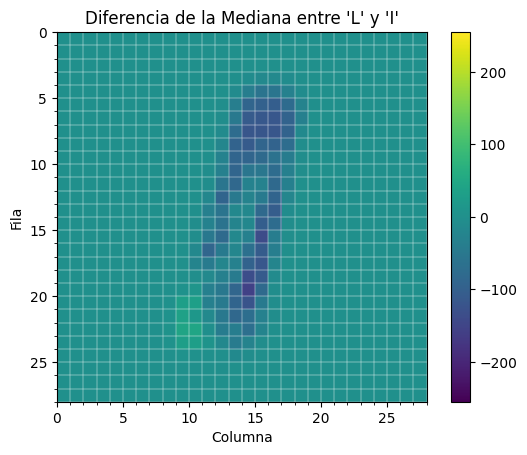
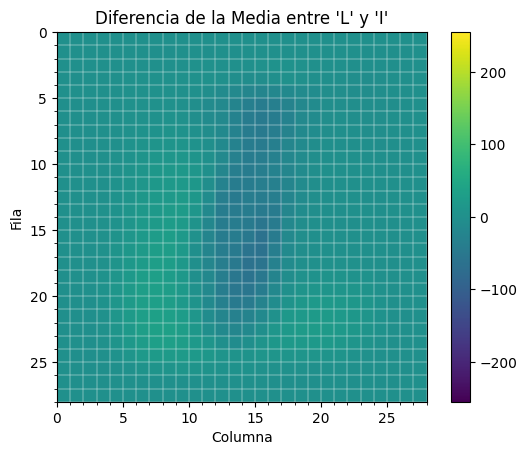
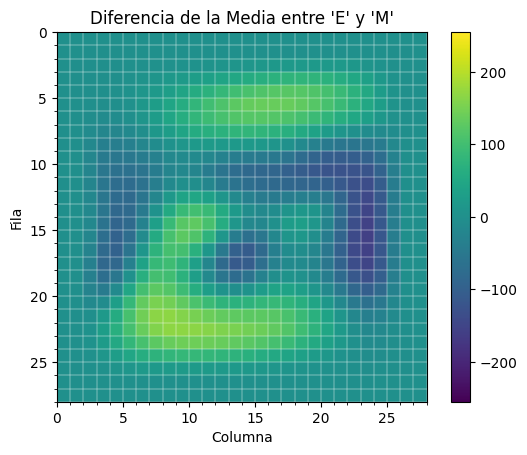
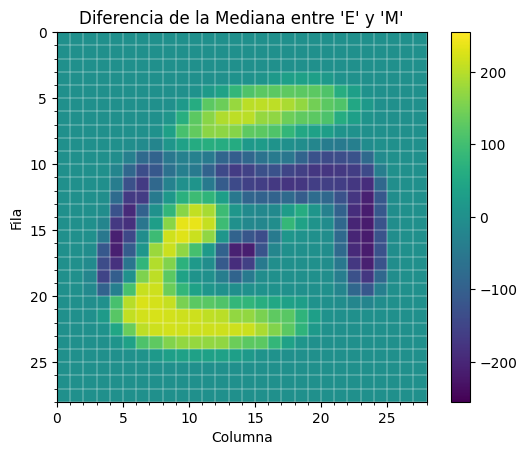


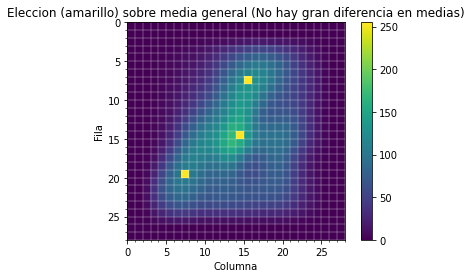
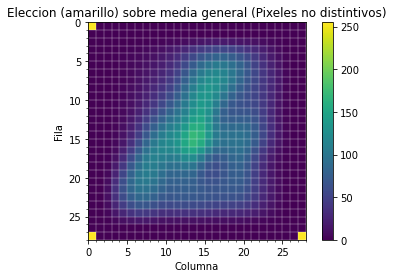
Fig. 6: Diferencias entre medias y medianas de “E” con “M” y “L” con “I”. Se puede apreciar que las diferencias entre medianas son más importantes

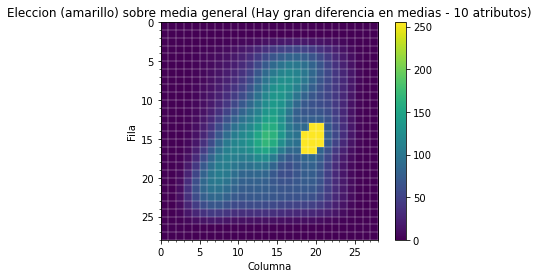
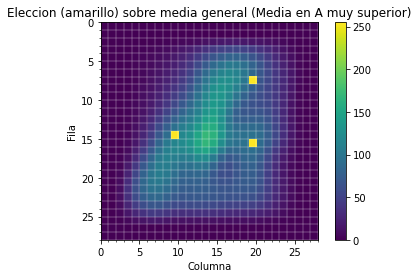
---- Parrafito de que el análisis exploratorio es distinto a analizar features sueltas de una tabla común y que acá tuvimos que hacer mas análisis de distribuciones que de variables en si (quizás podría ir antes de analizar los gráficos -----

### Experimentos realizados

Clasificación binaria:

Nos quedamos con las imágenes de las letras L y A y separamos el nuevo DataFrame, aislando un 30% como conjunto de test. Observamos las medias, medianas y distancias entre ellas para las letras para decidir distintas formas de escoger atributos. Trabajamos sobre el conjunto de entrenamiento seleccionando cada conjunto de atributos y ajustamos para cada uno modelos de KNN, los cuales evaluamos utilizando métricas binarias sobre las predicciones respecto al conjunto aislado de test.





*Fig 7: Pixeles elegidos representados con intensidad máxima respecto de la media de las imágenes, con el criterio de elección de cada uno.*

Clasificación multiclase:

Nos quedamos con las imágenes de las vocales y dividimos es un conjunto de entrenamiento y uno de test guardando un 30% del contenido para este último. Luego, empleamos K-folding para ajustar modelos de árbol de decisión variando los hiperparámetros de altura entre 3 y 14 y de criterio entre Gini y Entropy. Calculamos métricas multiclase para cada altura según su criterio y nos quedamos con el mejor de cada criterio, para luego compararlos y conservar los hiperparámetros que resultaron mejores.

### Conclusiones

Clasificación binaria:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Atributos | n\_neighbors | Accuracy | Matriz conf | Precision | Recall | F1 |
| Pixeles no distintivos | 5 | 0.49… | [[711 0]  [729 0]] | 0.49… | 1 | 0.66... |
| Pixeles no distintivos | 15 | 0.49… | [[711 0]  [729 0]] | 0.49… | 1 | 0.66... |
| Media en A muy superior | 5 | 0.95… | [[693 18]  [44 685]] | 0.94… | 0.97… | 0.95... |
| Media en A muy superior | 15 | 0.94… | [[694 17]  [56 673]] | 0.92… | 0.97… | 0.95... |
| Media en L muy superior | 5 | 0.85… | [[598 113]  [95 634]] | 0.86… | 0.84… | 0.85… |
| Media en L muy superior | 15 | 0.86… | [[593 118]  [84 645]] | 0.87… | 0.83… | 0.85... |
| Media en A muy superior + pixeles no distintivos | 5 | 0.95… | [[693 18]  [ 44 685]] | 0.94… | 0.97… | 0.95... |
| … | … | … | … | … | … | ... |

*Tabla 2: Algunas entradas de la tabla de resultados de la clasificación binaria disponible en el Anexo*

Podemos observar que la elección de píxeles no distintivos, correspondiente a las esquinas, no resulta para nada útil, pues el modelo resulta sumamente sencillo: clasifica todos los casos como la misma letra. Esto no cambia al aumentar el número de vecinos, pues se trata de píxeles donde, como vimos en los máximos de la base de datos entera, no hay ninguna letra allí, por lo que no hay vecinos para comparar.

El caso donde escogimos píxeles donde la media en A es muy superior resultó muy eficaz bajo todas las métricas, por arriba de aquella elección sobre la media en L muy superior. Esto tiene sentido, pues vimos en gráficos que la diferencia de medias en los píxeles elegidos sobre A tiene mayor intensidad sobre la de píxeles elegidos sobre L, es decir, estos últimos no eran tan distintos en módulo.

Al mezclar criterios el modelo parece dar prioridad a aquellos atributos donde los valores están más diferenciados, “heredando” las métricas calculadas cuando solo elegimos esos. Debido a que el modelo KNN se basa en distancias, prioriza estos atributos ya que provocan más distancia euclídea sobre el aporte del eje de los atributos más diferenciados entre las clases. Es decir, al realizar el cuadrado de la resta de estas coordenadas, habrá más peso en la clasificación.

En este caso, la diferencia al cambiar el número de vecinos de 5 a 15 no resulta significativa; en algunos casos aporta, en otros resta, pero siempre en muy baja medida.

Clasificación multiclase:

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Altura | Criterio | Acc | P\_A | P\_E | P\_I | P\_O | P\_U | R\_A | R\_E | R\_I | R\_O | R\_U |
| 3 | Gini | .69 | .78 | .56 | .92 | .48 | .79 | .83 | .03 | .92 | .92 | .72 |
| 9 | Gini | .91 | .91 | .89 | .95 | .91 | .90 | .88 | .91 | .95 | .93 | .90 |
| 14 | Gini | .91 | .90 | .89 | .94 | .91 | .89 | .88 | .91 | .95 | .92 | .89 |
| 3 | Entropy | .81 | .89 | .85 | .88 | .73 | .73 | .70 | .70 | .96 | .87 | .80 |
| 8 | Entropy | .91 | .89 | .90 | .95 | .91 | .90 | .88 | .91 | .95 | .93 | .89 |
| 11 | Entropy | .91 | .90 | .92 | .94 | .91 | .88 | .87 | .91 | .95 | .92 | .90 |
| 14 | Entropy | .91 | .90 | .91 | .94 | .91 | .89 | .87 | .91 | .95 | .92 | .90 |
| … | … | … | … | … | … | … | … | … | … | … | … | ... |

*Tabla 3: Métricas de Accuracy (acc), Precisión (P\_vocal) y Recall (R\_vocal) sobre el conjunto de train para cada par de hiperparámetros*

Como se puede observar en la Tabla 3, obtenemos que, como es de esperar, la altura más baja (para clasificar 5 clases es necesario al menos 3 decisiones) no obtiene resultados muy buenos. En esta altura el criterio de Gini brindó resultados muy bajos para algunas letras y muy altos para otras, como se puede observar con la precisión y recall, mientras que el criterio de Entropy fue bastante parejo. Es explicable, pues Gini prioriza aislar una clase antes de continuar con otra, mientras que Entropy intenta formar árboles más balanceados en general. A alturas mayores, la diferencia entre ambos criterios se mitiga considerablemente.

Eventualmente, aumentar la altura no cambia significativamente las métricas, por lo que los árboles más altos suelen ser diferenciables desde terceros decimales en adelante.

|  |  |
| --- | --- |
| Criterio | Entropy |
| Altura | 8 |
| Accuracy | 0.91… |
| Promedio Precisión | 0.91… |
| Promedio Recall | 0.91... |
| Precisión A | 0.90… |
| Recall A | 0.89… |
| Precisión E | 0.90... |
| Recall E | 0.90... |
| Precisión I | 0.95... |
| Recall I | 0.94... |
| Precisión O | 0.91... |
| Recall O | 0.93... |
| Precisión U | 0.89... |
| Recall U | 0.90... |

*Tabla 4: Métricas del árbol con mejores resultados*

Como se puede ver en la Tabla 4, nos encontramos con que el árbol de altura 8 y criterio Entropy resulta mejor comparando el accuracy, el promedio de precisiones y el promedio de recalls de todos los árboles. Si bien los árboles muy altos tienden a provocar overfitting, no es un fenómeno muy observable en estas alturas, pues se trata de una base de datos con muchos atributos y unas alturas proporcionalmente bajas. Es debido a la calidad de la base de datos y al bajo número de letras elegidas que nos encontramos con que estas alturas nos dan resultados tan altos.