Laboratorio de Datos

Departamento de Computación
Facultad de Ciencias Exactas y Naturales
Universidad de Buenos Aires

Trabajo Práctico 2

Resumen:
En este informe, se emplean técnicas de aprendizaje automático, para abordar el problema de reconocimiento de letras manuscritas en imágenes del conjunto de datos EMNIST.Se concluye que para selección binaria entre letras el algoritmo de kNN funciona con una exactitud del 97 %; mientras que para clasificación de multiclase un árbol de decisión puede tener un buen rendimiento, cerca del 92 % de exactitud.

Palabras clave: análisis de datos, Entrenamiento de Modelos, Clasificación k-vecinos (KNN), Clasificación Binaria, aprendizaje automático

Grupo : 3

Integrante	LU	Correo electrónico
Pauletti, Guido	862/22	guido13pauletti@gmail.com
Catania, Juan Ignacio	840/22	juancatania.fceyn@gmail.com
Paredes, Gerson	1865/21	gerson.paredes@outlook.com

1 Introducción

El presente informe aborda el desafío de la clasificación multiclase de imágenes, específicamente enfocado en determinar a qué vocal corresponde una imagen dada. Esta tarea implica el uso de técnicas de aprendizaje automático para analizar y procesar un conjunto de datos compuesto por imágenes de vocales.
El objetivo principal de este trabajo es desarrollar y evaluar modelos de aprendizaje automático capaces de identificar con precisión la vocal representada en una imagen. Para lograr esto, adoptaremos un enfoque sistemático, que abarca desde la preparación de los datos hasta la evaluación del rendimiento de los modelos.

1.1 Análisis exploratorio de datos

Para comenzar con el análisis exploratorio de los datos del conjunto EMNIST, se examinaron diversas características que nos permitieron comprender la naturaleza del conjunto de datos y su potencial para la clasificación de letras.

1.1.1 Descripción del Dataset EMNIST

El conjunto de datos EMNIST contiene imágenes de letras etiquetadas, con cada imagen representada por una matriz de 784 píxeles (28x28). Cada fila en la tabla 1 representa una instancia del conjunto de datos, donde los siguientes valores corresponden a los píxeles de la imagen de una clase determinada.

Índice	**0**	**1**	**2**	**...**	**783**	**784**
1	F	0	0	...	0	0
2	Q	0	0	...	0	0
3	W	0	0	...	0	0
4	K	0	0	...	0	0
1	F	0	0	...	0	0
.	.	.	.	.	.	.
.	.	.	.	.	.	.
.	.	.	.	.	.	.
62398	N	0	0	...	0	0
62399	V	0	0	...	0	0

Table 1: Dataset EMNIST

1.1.2 Exploración de las Etiquetas

Notemos que las clases corresponden a letras del abecedario, por lo que hay un total de 24 letras en el conjunto de datos. Además, es importante observar que las letras se repiten, lo que sugiere la existencia de diferentes estilos o tipos de letra. Por ejemplo, consideremos la letra 'C'. Al observar la figura1

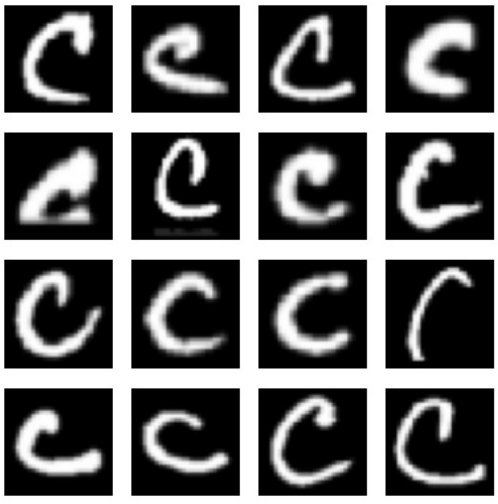

Figure 1: *Visualizacion de los diferentes tipos para la letra C*

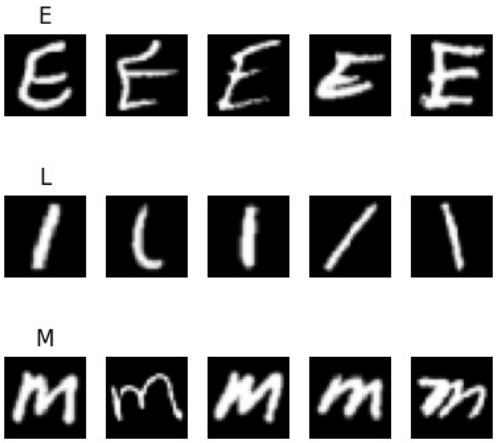

Figure 2: *Ejemplos de letras E, L y M*

Estas variaciones en los estilos de las letras son cruciales para el análisis, ya que afectan directamente al proceso de clasificación y reconocimiento de caracteres. En el análisis posterior, se explorarán estas diferencias en detalle para entender mejor cómo impactan en la precisión y efectividad de los modelos de clasificación.

1.1.3 Observaciones Clave del Análisis Exploratorio

- **Atributos Relevantes para la Predicción:**
  Para identificar atributos relevantes para predecir la letra a la que corresponde la imagen, se observó que los píxeles de la imagen son los atributos principales y unicos para esta distencion, ya que contienen información sobre la forma y la estructura de la letra.
  Por lo que son estos atributos (o columnas) fundamentales para lograr nuestra tarea. Sin embargo, se puede notar que los valores correspondientes a las esquinas y bordes de la imagen son en su mayoria igual a cero. Quizas podriamos reducir la cantidad de piexeles por imagen y prescindir de esos datos durante el proceso de preprocesamiento de datos.
- **Similitud entre Letras:**
  Las letras no son parecidas, incluso varian mucho las imagenes de una misma letra. Sin embargo cabe notar que si se comparan de a tres (como con la E, la M y la L), se podría decir que hay un par con mas similitudes entre sí. En la figura 2 podemos observar mayores similitudes entre la M y la E, pues ambas presentan un patron de tres rayas, la E hacia la derecha y la M los presenta hacia abajo.
- **Complejidad de Datos de Imágenes:**
  En este caso la exploración de datos se debe tomar por otro enfoque, gráficos de caja, histogramas, etc. servirían para visualizar el rendimiento de los modelos, pero lo importante a la hora de explorar en una base de datos de imágenes es poder verlas, ver patrones en ellas, identificar la cantidad de colores tal vez, el tamaño de cada imagen; y por sobre todas las cosas, ver que valores puede tomar cada imagen (que letras representan), para responder la cantidad de posibles valores distintos, y que porcentaje representa cada uno de estos en el conjunto de datos.
- **Proporción en la cantidad de instancias de cada letra.**
  Cada letra aparece 2400 veces en los datos, por lo que cada dataframe que filtremos según las letras que necesitemos van a estar perfectamente balanceados.

2 Metodologia

En esta sección, describiremos el enfoque metodológico utilizado para abordar el problema de clasificación binaria y multiclase de imágenes de letras. Nos centraremos en el proceso de preparación de datos, ajuste de modelos, selección de hiperparámetros y evaluación del rendimiento de los modelos. A continuación, detallaremos cada paso del proceso.

2.1 Evaluación de Modelos de Clasificación Binaria

Para evaluar los modelos de clasificación binaria, seguimos un enfoque sistemático que involucra los siguientes pasos:

1. **Filtrado de Datos y Separación en Conjuntos de Entrenamiento y Prueba:** Iniciamos seleccionando únicamente las observaciones correspondientes a las letras "A" y "L" de nuestro conjunto de datos. Luego, dividimos los datos en características y etiquetas, y separamos el conjunto en conjuntos de entrenamiento y prueba utilizando la función `train_test_split` de *scikit-learn* [1].

2. **Ajuste de Modelos de KNN con Diferentes Conjuntos de Atributos:**
   Para explorar el impacto de diferentes conjuntos de atributos en el rendimiento del modelo, ajustamos múltiples modelos de clasificación KNN utilizando combinaciones aleatorias de tres atributos. Se realizaron 15 iteraciones para evaluar una variedad de combinaciones de atributos.

   - **Conclusiones del experimento:**
     Notamos que las columnas del medio nos dan un buen resultado, esto se nos puede sugerir que hay información relevante concentrada osea, que los datos están estructurados de manera que la información relevante está concentrada en ciertas partes. O ademas, puede darnos una idea de menor ruido, es decir, que las características en las extremidades pueden contener más ruido o información irrelevante que las características en el medio.

3. **Selección Manual de Atributos y Evaluación del Modelo:**
   Basándonos en los resultados obtenidos de los modelos ajustados anteriormente, seleccionamos manualmente un conjunto de tres atributos que estan en el centro del Dataset, en este caso son: 391,392,393 y ajustamos un nuevo modelo KNN utilizando estos atributos. Posteriormente, evaluamos el rendimiento del modelo en el conjunto de prueba.

   - **Conclusiones del experimento:**
     Es interesante ver que al seleccionar manualmente las tres columnas del medio (columnas 391, 392 y 393), el rendimiento del modelo fue bastante malo. Esto sugiere que estas columnas pueden ser redundantes o no contener información significativa para la predicción del modelo.

4. **Visualización de Ejemplos de Imágenes:**
   Para proporcionar una representación visual del proceso, se muestra la figure 3 que ilustra ejemplos de imágenes de las letras "A" y "L". En esta figura, se resaltan con señales rojas tres atributos cercanos seleccionados en cada imagen, lo que ayuda a comprender cómo se ven estos atributos en las imágenes y cómo pueden influir en la clasificación. La exactitud obtenida en este modelo fue menor al 60 %, esto lo podemos entender por la redundancia de información que conlleva tomar 3 pixeles contiguos.

Figure 3: Al tomar 3 pixeles contiguos hay mucha información redundante.

5. **Evaluación con Diferentes Cantidades de Atributos y Vecinos:**
   Para ampliar nuestra evaluación, realizamos experimentos adicionales variando la cantidad de atributos y el número de vecinos en el algoritmo KNN. Comenzamos evaluando combinaciones de 9 atributos, luego aumentamos a 21, y finalmente exploramos el impacto del número de vecinos en el rendimiento del modelo.

   Realizamos experimentos con combinaciones de 9 y 21 atributos, respectivamente, para evaluar cómo afecta la cantidad de atributos al rendimiento del modelo KNN. Para cada experimento, se ajustaron 15 modelos de KNN utilizando una selección aleatoria de atributos y se evaluó el rendimiento en el conjunto de prueba. Los resultados obtenidos se muestran a continuación:

   - **Con 9 Atributos:**
     La mejor combinación de atributos logró una exactitud máxima de 0.938 en el conjunto de prueba, con una exactitud media de 0.842.
   - **Con 21 Atributos:**
     Al aumentar la cantidad de atributos a 21, la mejor exactitud obtenida fue también de 0.971, con una exactitud media de 0.888.

Estos resultados indican que 21 atributos es un buen número para usar en el entrenamiento del modelo kNN.

2.1.1 Comparación de Modelos KNN con Diferentes Vecinos

Para evaluar cómo afecta el número de vecinos en el algoritmo KNN al rendimiento del modelo, realizamos experimentos con diferentes valores de k. Utilizando la mejor combinación de 21 atributos identificada previamente, ajustamos modelos de KNN con valores de k variando desde 3 hasta 32. Los resultados obtenidos son los siguientes:

- La mejor exactitud se obtuvo con k = 9, con una exactitud máxima de 0.973 en el conjunto de prueba. Los atributos utilizados para este modelo fueron [407 574 728 99 684 726 547 739 613 462 643 769].

Basándonos en estos resultados, concluimos que el mejor modelo de KNN para la clasificación de vocales tiene un valor de k igual a 9, utilizando una combinación de 12 atributos.

2.2 Evaluación de Modelos para Clasificación de Vocales

En este apartado, se busca realizar la clasificación multiclase de imágenes correspondientes a cinco vocales diferentes (por ejemplo, A, E, I, O, U). El objetivo es entrenar un modelo de árbol de decisión para identificar a qué vocal corresponde una imagen dada.

2.2.1 Exploración y Entrenamiento del Modelo

1. **Preparación de datos:** Inicialmente, se filtraron los datos para incluir solo las imágenes correspondientes a las vocales seleccionadas. Luego, se separaron los atributos de las etiquetas, creando conjuntos de características y etiquetas.
2. **Entrenamiento del modelo de árbol de decisión:** Se definió una función para entrenar un modelo de árbol de decisión con diferentes profundidades. Se exploraron distintas profundidades del árbol y se evaluó la exactitud del modelo en el conjunto de entrenamiento. Las mejores profundidades en términos de exactitud fueron 9 y 11 con un puntaje de 0.98 y 0.99 respectivamente.
3. **Validación cruzada con k-folding:** Se implementó la validación cruzada con k-folding para comparar modelos de árbol de decisión con diferentes alturas y criterios de división. Los resultados se registraron en un DataFrame.
4. **Selección del mejor modelo:** Se seleccionó el mejor modelo de árbol de decisión basado en los resultados de la validación cruzada, es decir con los siguientes hiperparámetros: profundidad máxima = 10; criterio = entropy. Luego, se evaluó el rendimiento del modelo seleccionado en el conjunto de prueba.

2.2.2 Análisis de errores

Después de entrenar y evaluar el modelo seleccionado con todo el conjunto de datos de entrenamiento y prueba, obtuvimos una exactitud del 92%, como se esperaba por los resultados obtenidos durante la validación cruzada. Esta alta tasa de precisión indica que nuestro modelo de árbol de decisión es capaz de clasificar correctamente la mayoría de las imágenes de las vocales.

Índice	hmax	criterio	promedio_exactitud
0	10	entropy	0.920952
1	10	gini	0.916905
2	15	log_loss	0.916548
3	10	log_loss	0.915833
4	15	entropy	0.915833
.	.	.	.
.	.	.	.
.	.	.	.

Table 2: Dataframe modelos

Sin embargo, para comprender mejor el rendimiento del modelo y analizar los errores cometidos, realizamos una matriz de confusión. Esta matriz nos muestra cómo se distribuyen las predicciones del modelo en comparación con las etiquetas reales para cada una de las cinco vocales.

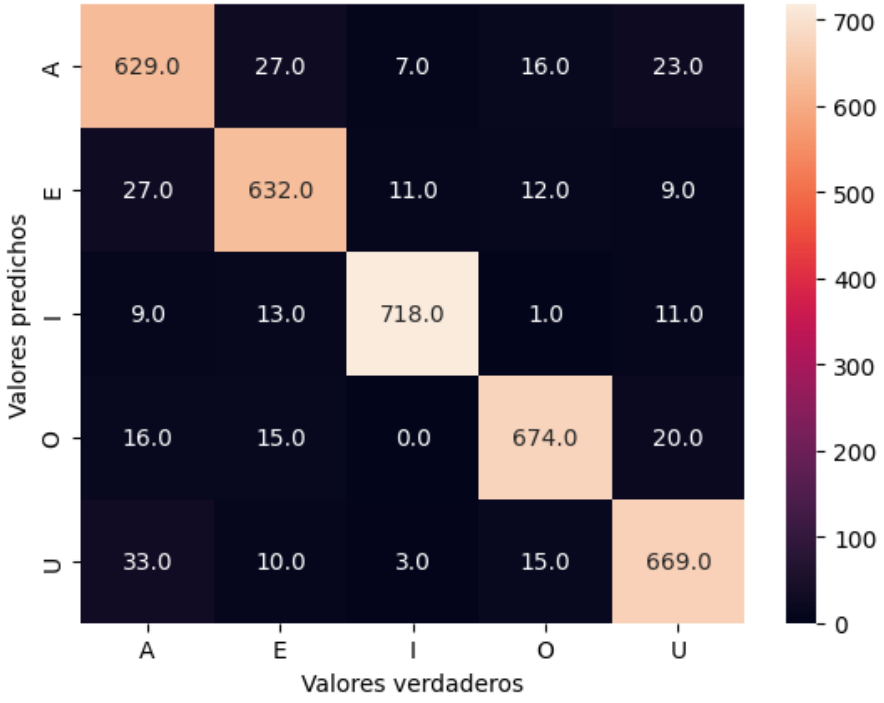

Figure 4: *Matriz de confusión para la métrica de clasificación de modelos*

Al observar la matriz de confusión, podemos notar que la diagonal principal, que representa las predicciones correctas, tiene valores altos, lo que indica que el modelo clasifica correctamente la mayoría de las imágenes. Sin embargo, también podemos observar algunos valores fuera de la diagonal principal, lo que indica errores en la clasificación. Estos errores pueden deberse a la similitud visual entre algunas de las vocales o a la presencia de ruido en las imágenes. Observamos por ejemplo que la A es confundida muchas veces con la U y con la E, y viceversa. También notamos que la I es la que menos veces es confundida por otra letra.

3 Conclusiones

Ambos modelos tuvieron buenos resultados, kNN logró una mayor exactitud que el árbol de decisión; lo cual se explica probablemente por el hecho de que se ocupó de una clasificación binaria y no de un caso multiclase. Concluimos que es posible identificar una letra usando técnicas de aprendizaje automático, incluso con pocos datos (pixeles); sin embargo, se encuentran dificultades cuando dos letras tienen similitudes, disminuyendo la exactitud en estos casos.

References

[1] Scikit-learn: Machine Learning in Python.
`https://scikit-learn.org/stable/`