Ingeniería del Conocimiento

Práctica 3: Métodos de clasificación

Autores

- Pedro Antonio Martín Peláez
- Álvaro Ray Martínez

Detalles de Implementación

La solución se ha desarrollado íntegramente en Python 3.x, agrupando todo el código en un único fichero (main.py) que organiza la práctica en tres bloques principales: funciones de lectura de datos, implementaciones de los algoritmos (K-Medias Fuzzy, Naive Bayes y Lloyd Competitivo) y la capa de interfaz gráfica con Tkinter. En la primera parte se definen tres rutinas para cargar, respetivamente, los datos de entrenamiento con etiqueta, sin etiqueta y las muestras de test, traduciendo cada línea de texto a vectores de tipo float y estructurándolos en diccionarios o listas según convenga. A continuación, cada algoritmo reside en su propia función con parámetros configurables (número de clusters, tasa de aprendizaje, tolerancias, etc.), y comparte un mecanismo común para convertir el resultado de clustering en predicciones de clase mediante votación mayoritaria. Finalmente, el bloque de GUI enlaza todos estos mecanismos, ofreciendo controles para seleccionar ficheros, desplegar el método elegido y presentar los resultados en un área de texto desplazable, mientras gestiona excepciones y muestra alertas en caso de error.

Lenguaje Utilizado y Librerías

En esta práctica se ha utilizado el lenguaje de programación **Python**, por su versatilidad y facilidad para el procesamiento de datos y la creación de algoritmos de aprendizaje automático.

En esta práctica se ha utilizado el lenguaje de programación Python, ampliamente reconocido por su simplicidad y eficiencia en el desarrollo de algoritmos de aprendizaje automático. Las librerías empleadas son:

- **csv:** Utilizada para la lectura de los datos de entrada desde ficheros en formato CSV, donde se almacenan los ejemplos con los que se entrena y evalúa el clasificador.
- **math:** Empleada para realizar operaciones matemáticas como logaritmos y exponenciales, esenciales en el cálculo de las probabilidades durante la clasificación.
- **tkinter**: Es una biblioteca estándar de Python para crear la interfaz gráfica (ventanas, botones, cuadros de texto).
- **collections** (defaultdict, Counter): para agrupar muestras por clase y calcular etiquetas por mayoría.
- **random**: para la inicialización aleatoria de centroides en K-Medias Fuzzy.

- **numpy**: procesamiento numérico eficiente: operaciones vectoriales, cálculo de distancias y estadísticas.

Procedimiento de Implementación

El desarrollo comenzó con el análisis de requisitos: lectura de ficheros de Iris, definición de las estructuras de datos y los parámetros de los algoritmos. Se diseñó una función genérica de carga que pudiese servir a todos los métodos y se implementaron cada uno de los tres algoritmos de forma independiente, probándolos primero en modo consola para verificar su correcto funcionamiento. A continuación, se integró la GUI en Tkinter, creando los cuadros de entrada de ruta de fichero, el selector de algoritmo y el botón de ejecución, de modo que al pulsar "Ejecutar" se encadenasen automáticamente la lectura de datos, el entrenamiento o agrupamiento, la asignación de etiquetas y la presentación de resultados. Durante la integración se añadieron controles de error —por ejemplo, alerta si falta algún archivo o si se generan valores `NaN`— y se definieron valores por defecto para facilitar las pruebas (centros iniciales fijos, tolerancias estándar). Finalmente, la aplicación se empaquetó en un único ejecutable para Windows, incluyendo todas las dependencias necesarias.

Ampliaciones Realizadas

Además de los requisitos iniciales, se han añadido varias mejoras. La interfaz gráfica ofrece un acceso más intuitivo que la línea de comandos y alerta mediante cuadros de diálogo cualquier error de fichero no encontrado o dato mal formateado. Se incluye la posibilidad de fijar manualmente los centros iniciales para reproducir experimentos de clustering, y la estructura de funciones independientes facilita la incorporación de nuevos métodos o métricas en el futuro. El manejo de excepciones garantiza que el programa no finaliza abruptamente ante entradas inesperadas.

Manual de Usuario

En la carpeta entregable encontrará un ejecutable llamado **iris_classifier.exe** junto con los archivos de datos **Iris2Clases.txt**, **TestIris01.txt**, **TestIris02.txt** y **TestIris03.txt**. Para utilizar la aplicación, basta con:

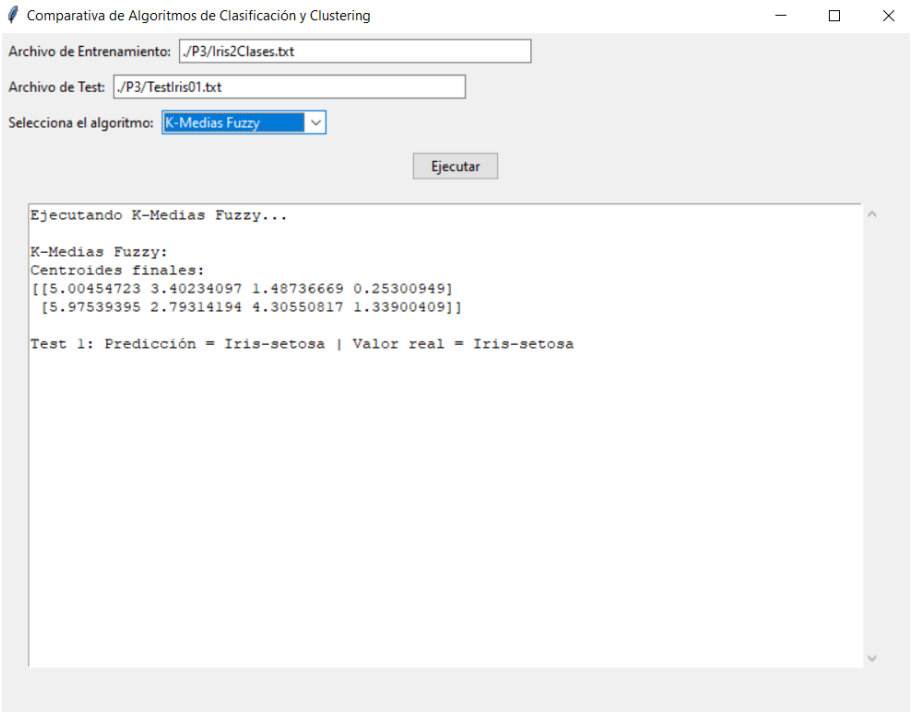
1. Abrir la carpeta y hacer doble clic en **iris_classifier.exe**.
2. En la ventana que aparece, comprobar que los campos
    - **Archivo de Entrenamiento** apunten a **Iris2Clases.txt**.
    - **Archivo de Test** al fichero deseado (por ejemplo **TestIris01.txt**).
3. Seleccionar el algoritmo (K-Medias Fuzzy, Bayes o Lloyd Competitivo) desde el menú desplegable.
4. Pulsar el botón **Ejecutar**.

Automáticamente se leerán los datos, se aplicará el método elegido y se mostrará en el panel inferior tanto la información de los centroides (si aplica) como la predicción para cada muestra de test, acompañada de su etiqueta real (cuando esté disponible). Puede cambiar de fichero de

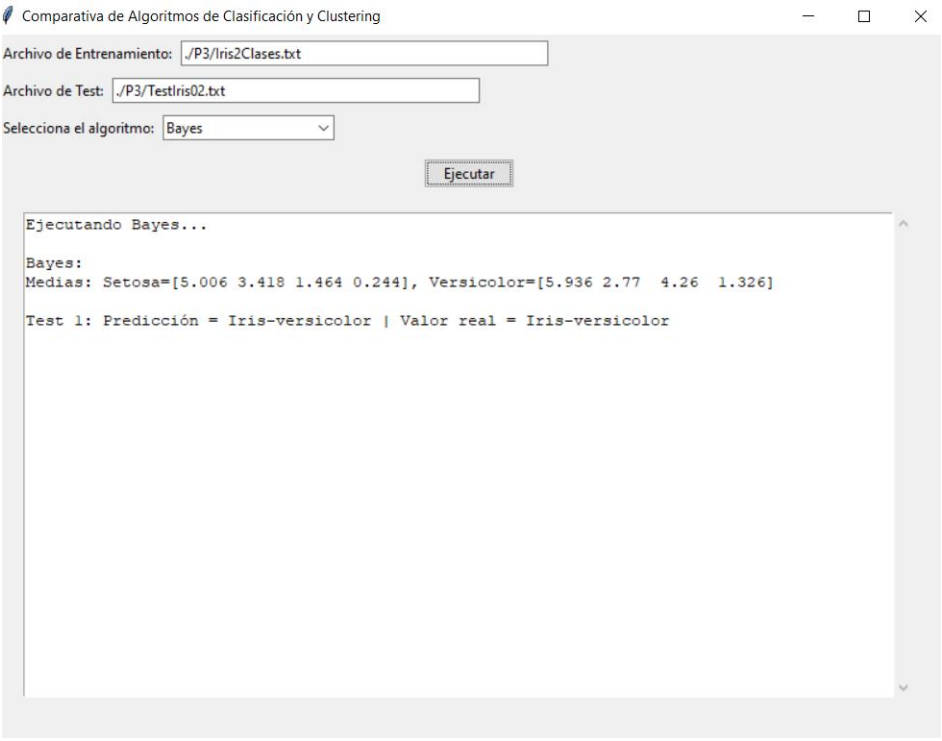
test o de algoritmo tantas veces como desee para comparar resultados de forma rápida e intuitiva.

Ejemplos de Ejecución

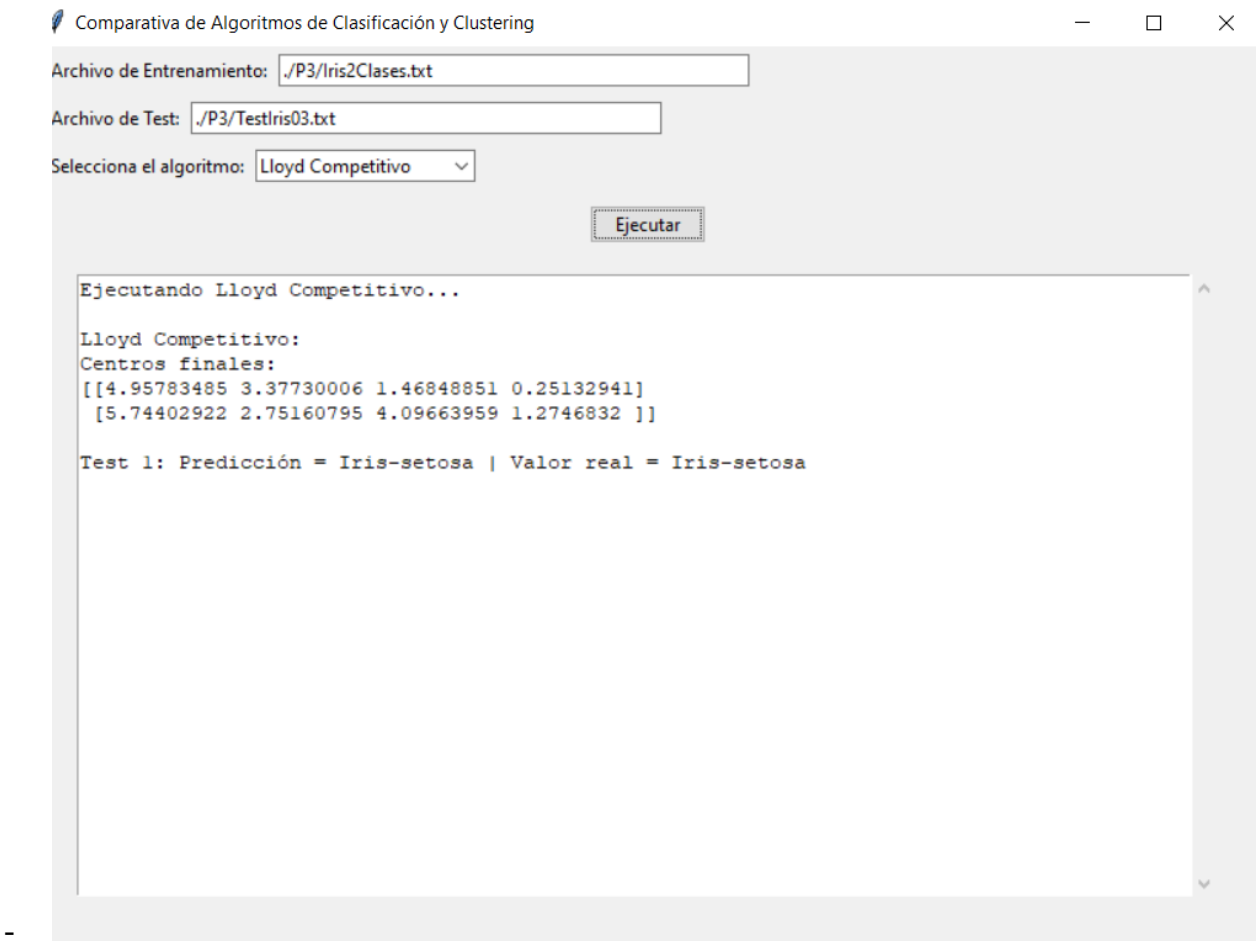
- K-Medias usado con TestIris01



- Bayes con TestIris02



- Lloyd con TestIris03


Comparativa de Algoritmos de Clasificación y Clustering
Archivo de Entrenamiento: ./P3/Iris2Clases.txt
Archivo de Test: ./P3/TestIris03.txt
Selecciona el algoritmo: Lloyd Competitivo
Ejecutar
Ejecutando Lloyd Competitivo...
Lloyd Competitivo:
Centros finales:
[[4.95783485 3.37730006 1.46848851 0.25132941]
[5.74402922 2.75160795 4.09663959 1.2746832]]
Test 1: Predicción = Iris-setosa | Valor real = Iris-setosa

-