INSTITUTO POLITÉCNICO NACIONAL
ESCUELA SUPERIOR DE CÓMPUTO
INTELIGENCIA ARTIFICIAL
ING. EN SISTEMAS COMPUTACIONALES

Laboratorio 8: datasets

Alumno: Hurtado Morales Emiliano - 2021630390

Maestro: Andrés García Floriano

Grupo: 6CV3

Fecha de entrega: 31/10/2024

Ciclo Escolar: 2025 – 1

Introducción

El análisis de datos es una disciplina fundamental en la ciencia de la computación y la inteligencia artificial, ya que permite a los investigadores y profesionales extraer información significativa de grandes volúmenes de datos. Entre los datasets más populares para iniciarse en esta área, se encuentra el Iris Plant Dataset, introducido por el botánico Ronald Fisher en 1936. Este conjunto de datos contiene 150 muestras de tres especies de flores del género Iris (Iris-setosa, Iris-versicolor, Iris-virginica), y cada muestra está compuesta por cuatro características: la longitud y el ancho de los sépalos y los pétalos.

El objetivo de este trabajo es implementar un programa que lea y organice el archivo del dataset Iris (bezdekIris.data) utilizando dos lenguajes de programación diferentes: Python y Java. En el caso de Python, se utiliza la popular librería pandas, que ofrece un entorno optimizado para la manipulación de datos tabulares. Para Java, se emplea una estructura manual utilizando clases estándar como BufferedReader para leer los datos del archivo y almacenarlos en un ArrayList. A través de esta comparación, se evaluará la facilidad de uso y flexibilidad de ambos lenguajes para tareas de procesamiento y organización de datos.

Explicación de los Algoritmos

Python

En Python, se utilizó la librería pandas para facilitar la manipulación y organización de los datos en un formato tabular. El enfoque principal es leer el archivo CSV utilizando el método read_csv() de pandas, definir las columnas del dataset y presentar los resultados de forma legible. El código sigue estos pasos:

```
import pandas as pd  # Importamos la librería pandas, que nos permite trabajar
con dataframes y manipular datos.

# Leemos el archivo CSV 'bezdekIris.data' y almacenamos los datos en un dataframe
llamado 'iris_df'.
# El archivo se lee con 'read_csv' y se le asignan nombres a las columnas usando
el argumento 'names'.
iris_df = pd.read_csv('bezdekIris.data', names=column_names)
```

1. **Lectura del archivo CSV**: Utilizamos pd.read_csv() para leer el archivo bezdekIris.data. A este método se le pasa el nombre del archivo y una lista con los nombres de las columnas.

```
# Definimos los nombres de las columnas para el dataset Iris.
column_names = ['sepal_length', 'sepal_width', 'petal_length', 'petal_width',
'class']
```

2. **Asignación de nombres de columnas**: Las columnas del dataset se definen como: sepal_length, sepal_width, petal_length, petal_width y class.

```
# Mostramos las primeras 5 filas del dataframe para verificar que los datos se
han cargado correctamente.
print(iris_df.head())
```

3. **Mostrar las primeras filas**: Para verificar que los datos se han leído correctamente, se imprime un pequeño subconjunto del dataframe utilizando el método head().

Ventajas del enfoque en Python

- Uso eficiente de la librería pandas, que facilita la manipulación de grandes cantidades de datos.
- Código sencillo y conciso con un mínimo esfuerzo para cargar y visualizar el dataset.

Java

El programa en Java adopta un enfoque manual, utilizando clases estándar como BufferedReader para leer el archivo línea por línea. Los pasos son los siguientes:

```
public class IrisDataset {
    public static void main(String[] args) {
        String file = "bezdekIris.data";  // Definimos la ruta del archivo que
vamos a leer.
        ArrayList<String[]> dataset = new ArrayList<>();  // Usamos un ArrayList
para almacenar cada fila como un arreglo de cadenas.
        String line;  // Variable que almacenará cada línea leída del archivo.

        try (BufferedReader br = new BufferedReader(new FileReader(file))) {
            // El bloque 'try-with-resources' abre el archivo y se asegura de
cerrarlo automáticamente.
            while ((line = br.readLine()) != null) {
                // Mientras existan líneas por leer, las procesamos.
```

1. **Lectura del archivo**: Se utiliza un BufferedReader para leer el archivo bezdekIris.data línea por línea.

```
                // Dividimos cada línea en partes usando la coma como separador.
                String[] data = line.split(",");

                // Agregamos el arreglo resultante (una fila del dataset) al
ArrayList.
                dataset.add(data);
            }
```

2. **Almacenamiento en una estructura**: Los datos se almacenan en un ArrayList que contiene arreglos de cadenas. Cada fila de datos se separa utilizando split() para dividir las cadenas por comas.

```
        // Imprimimos las primeras 5 filas del dataset con un formato tabulado.
        for (int i = 0; i < 5; i++) {
            System.out.println(String.format("%-15s %-15s %-15s %-15s %-20s",
                dataset.get(i)[0], dataset.get(i)[1], dataset.get(i)[2],
dataset.get(i)[3], dataset.get(i)[4]));
        }
```

3. **Impresión de los datos**: Para visualizar los resultados, el programa imprime las primeras cinco filas del dataset con los valores alineados utilizando String.format() para crear una tabla legible.

Ventajas del enfoque en Java

- El enfoque es flexible y se puede personalizar para la lectura y procesamiento de archivos de formato diverso.
- Permite control detallado sobre cómo se manejan los datos en cada paso, útil para fines educativos.

Resultados Esperados

En ambos programas, se espera obtener una representación estructurada de las primeras cinco filas del dataset Iris. Las columnas estarán etiquetadas con los nombres de las características y se mostrarán valores numéricos correspondientes a la longitud y el ancho del sépalo y el pétalo, seguidos por la clase de la flor.

```
PS D:\ESCOM\9 Semestre\IA\Bender-IA\Lab_8> python Lab_8.py
   sepal_length  sepal_width  petal_length  petal_width        class
0           5.1          3.5           1.4          0.2  Iris-setosa
1           4.9          3.0           1.4          0.2  Iris-setosa
2           4.7          3.2           1.3          0.2  Iris-setosa
3           4.6          3.1           1.5          0.2  Iris-setosa
4           5.0          3.6           1.4          0.2  Iris-setosa
PS D:\ESCOM\9 Semestre\IA\Bender-IA\Lab_8> java IrisDataset.java
sepal_length    sepal_width     petal_length    petal_width     class
------------------------------------------------------------------------
5.1             3.5             1.4             0.2             Iris-setosa
4.9             3.0             1.4             0.2             Iris-setosa
4.7             3.2             1.3             0.2             Iris-setosa
4.6             3.1             1.5             0.2             Iris-setosa
5.0             3.6             1.4             0.2             Iris-setosa
```

Conclusión

El desarrollo de este proyecto permite comparar dos enfoques diferentes para procesar datos de un archivo CSV en Python y Java, destacando tanto las ventajas como las desventajas de cada uno. En Python, la simplicidad y el poder de la librería pandas permiten que la implementación sea rápida y concisa, con herramientas especializadas para la manipulación de datos. Este enfoque resulta ideal cuando el objetivo principal es trabajar de manera eficiente con grandes volúmenes de datos y ejecutar análisis estadísticos o de machine learning.

Por otro lado, el enfoque en Java proporciona un control más detallado sobre el proceso de lectura y manejo de los datos. Aunque el código es más largo y manual, es útil en situaciones donde es necesario ajustar minuciosamente cómo se procesan y organizan los datos. Java ofrece una base sólida para aplicaciones que requieren una mayor personalización y control, especialmente en contextos industriales o de backend.

En conclusión, ambos lenguajes son capaces de procesar el dataset Iris con éxito, pero Python sobresale en términos de simplicidad y eficiencia para manipular datos. Java, aunque más laborioso, permite una mayor flexibilidad cuando se requiere un control exhaustivo sobre las operaciones de lectura y escritura. La elección de un lenguaje dependerá del contexto y los requisitos del proyecto, pero para tareas específicas de análisis de datos, Python resulta la opción más conveniente.