Minería de Datos
Práctica de CHAMELEON – Algoritmo de Agrupación

Por:
Teófilo Lozano Apache – Jhon Adrián Cerón Guzmán

Objetivo:
Aplicar el algoritmo de agrupación CHAMELEON a dos conjuntos de datos de 8.000 y 10.000 puntos en dos dimensiones; estos puntos forman diferentes figuras geométricas, como se muestra a continuación.

Figura 1. Conjunto de datos # 1, compuesto por 10.000 puntos en dos dimensiones.

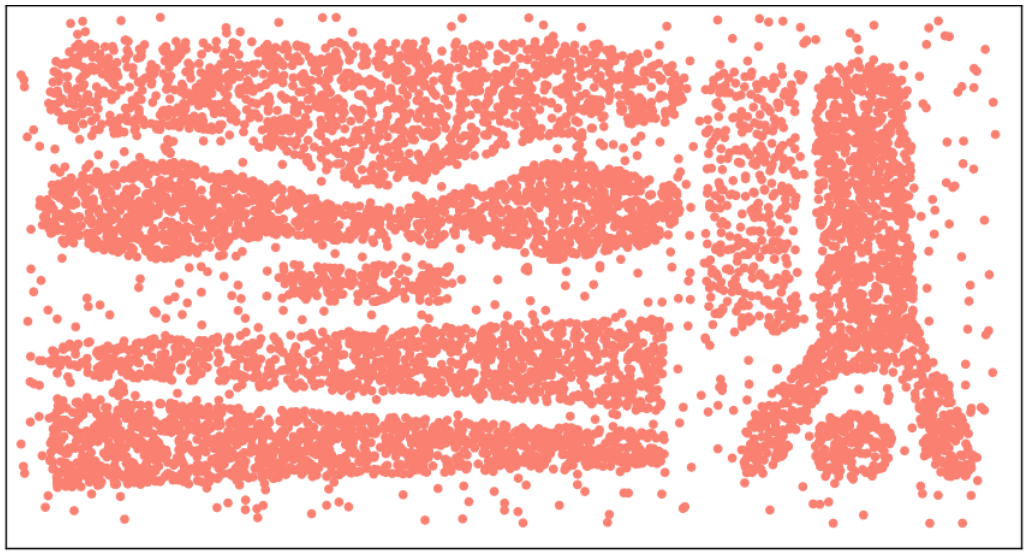

Figura 2. Conjunto de datos # 2, compuesto por 8.000 puntos en dos dimensiones.

Los resultados obtenidos mediante la aplicación del algoritmo CHAMELEON serán comparados con los resultados provistos por los algoritmos DBSCAN y Kmeans.

Prerrequisitos:

- Instalar *Python 2.7.8*.
- Instalar los siguientes módulos (librerías) para Python:
  - SciPy y
  - scikit-learn.
- Descargar y (descomprimir el archivo que contiene) la herramienta de minería de datos CLUTO (versión 2.1.1).
- Descargar el conjunto de datos "chameleon-data.tar.gz" de CLUTO (sección *Datasets*).

Conjunto de Datos # 1:
Las figuras geométricas que forma este conjunto de datos, de 10.000 puntos en dos dimensiones, pueden observarse en la figura 1.

A este conjunto de datos se aplicarán los algoritmos CHAMELEON, DBSCAN y KMeans. El archivo del conjunto de datos es "t7.10k.dat", contenido en el archivo "chameleon-data.tar.gz".

Antes de continuar, es necesario modificar el contenido del archivo del conjunto de datos así:

- *Agregar la linea al inicio del contenido "10000 2"*[1] *(sin comillas). El archivo ahora tendrá 10.001 lineas y su estructura debe ser igual a como se muestra a continuación.*

```
10000 2
539.512024 411.975006
542.241028 147.626007
653.468994 370.727997
598.585999 284.882996
573.062988 294.562988
139.570007 401.381012
228.970001 281.992004
305.747009 94.350998
```

De acuerdo a la ubicación de la herramienta CLUTO en el equipo, y el sistema operativo, se usará la misma para agrupar el conjunto de datos.

El sistema operativo del equipo donde se desarrolla esta guía es basado en GNU/Linux.

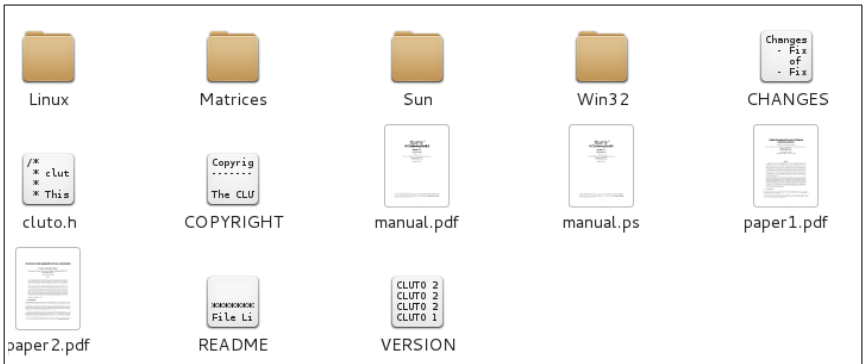

Figura 3. Estructura de directorios y ficheros de la herramienta CLUTO.

Con el siguiente comando vía línea de comandos del sistema operativo, se ejecuta el algoritmo CHAMELEON.

"*/ruta/a/cluto*/Linux/vcluster */ruta/al/dataset*/t7.10k.dat 9 -clmethod=graph -sim=dist -agglofrom=30"

1 El primer valor indica el número de registros del conjunto de datos. El segundo, separado por un carácter de espacio, el número de atributos.

Los parámetros pasados para ejecutar la agrupación se describen a continuación:

- "*/ruta/al/dataset*/t7.10k.dat": nombre del archivo que contiene los datos.
- "9": número de *clusters* (o grupos) deseado.
- "-clmethod=graph": la solución es obtenida modelando los datos usando una gráfica de *nearest-neighbor*, y entonces un algoritmo particional es utilizado para dividir la gráfica en los *k clusters* deseados.
- "-sim=dist": función de similaridad usada para el agrupamiento. En este caso, se usa la distancia Euclideana entre los datos.
- "-agglofrom=30": este parámetro hace que la solución del agrupamiento se obtenga combinando los métodos particional y aglomerativo. 30 indica el número de *clusters* iniciales en el *dataset*, y los 9 finales se obtienen fusionando los mismos usando un algoritmo aglomerativo.

Como resultado se obtiene el archivo "t7.10k.dat.clustering.9" de 10.000 lineas y un sólo atributo, cuyo rango de valores va desde cero (0) a ocho (8). El valor en la linea *i* del archivo resultado indica el *cluster* al cual el punto *i* del archivo inicial pertenece. El reporte de la solución *clustering* se presenta a continuación.

```
vcluster (CLUTO 2.1.1) Copyright 2001-03, Regents of the University of Minnesota

Matrix Information ------------------------------------------------------------
  Name: /var/unal/dm/chameleon/dataset/t7.10k.dat, #Rows: 10000, #Columns: 2, #NonZeros: 20
000

Options -----------------------------------------------------------------------
  CLMethod=GRAPH, CRfun=Cut, SimFun=EuclDist, #Clusters: 9
  RowModel=None, ColModel=None, GrModel=SY-DIR, NNbrs=40
  Colprune=1.00, EdgePrune=-1.00, VtxPrune=-1.00, MinComponent=5
  CSType=Best, AggloFrom=30, AggloCRFun=SLINK_W, NTrials=10, NIter=10

Solution ----------------------------------------------------------------------

------------------------------------------------------------------------------
9-way clustering: [Cut=1.64e+05] [10000 of 10000]
------------------------------------------------------------------------------
cid  Size  ISim  ISdev   ESim  ESdev  |
------------------------------------------------------------------------------
  0   294 +0.106 +0.029 +0.000 +0.000 |
  1   350 +0.095 +0.019 +0.000 +0.000 |
  2   425 +0.075 +0.016 +0.000 +0.000 |
  3   668 +0.048 +0.012 +0.000 +0.000 |
  4   682 +0.047 +0.012 +0.000 +0.000 |
  5  1159 +0.028 +0.007 +0.000 +0.000 |
  6  1179 +0.027 +0.007 +0.000 +0.000 |
  7  2328 +0.014 +0.003 +0.000 +0.000 |
  8  2915 +0.011 +0.003 +0.000 +0.000 |
------------------------------------------------------------------------------

Timing Information ------------------------------------------------------------
   I/O:                             0.020 sec
   Clustering:                      3.580 sec
   Reporting:                       0.004 sec
```

Figura 4. Resultado del clustering aplicando CHAMELEON al conjunto de datos # 1.

A continuación, se aplicarán los algoritmos DBSCAN y KMeans al mismo conjunto de datos. Note que, a diferencia de CHAMELEON y KMeans, DBSCAN no requiere como parámetro el número de *clusters* deseado. En este algoritmo son requeridos los parámetros *eps* y *min_samples*. El primero se refiere a la distancia máxima entre dos puntos para ser considerados en la misma "vecindad". El segundo por su parte, se refiere al número mínimo de puntos requeridos para formar un región densa.

Los valores de los parámetros de DBSCAN son calculados utilizando heurísticas[2][3].

2 La descripción del algoritmo DBSCAN, así como las heurísticas, están fuera del alcance de esta práctica.

3 Los valores de los parámetros *eps* y *min_samples* para el conjunto de datos # 1, son tomados de:

La aplicación de los algoritmos DBSCAN y KMeans, así como los resultados (en una gráfica de dispersión) de estos mas los obtenidos a través de CHAMELEON pueden ser obtenidos utilizando el programa en Python “t7.10k.dat.py”. Las siguientes figuras muestran los resultados por cada algoritmo.

Para ejecutar este programa, ejecute en la línea de comandos del sistema operativo la siguiente instrucción: “python t7.10k.dat.py” (sin comillas).

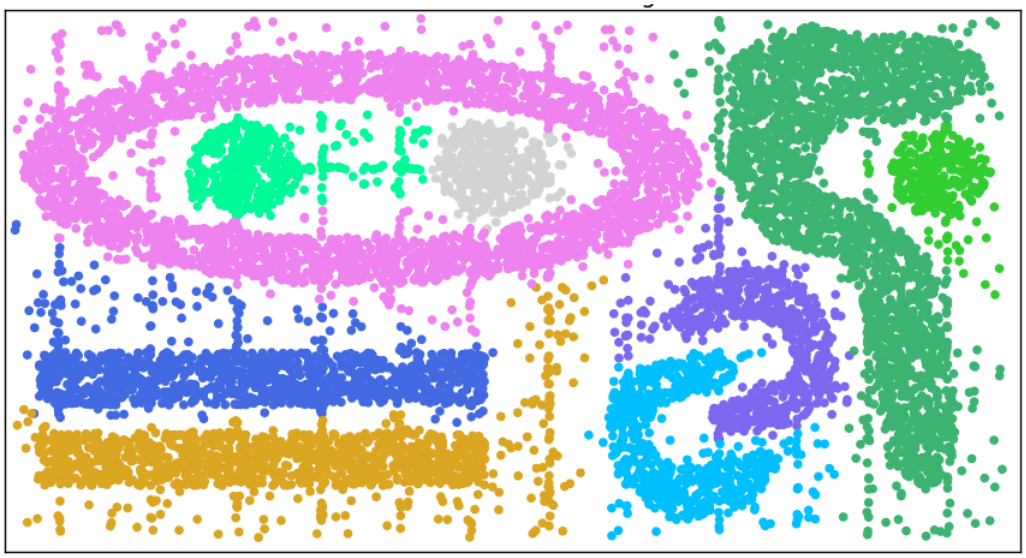

Figura 5. *Clusters* obtenidos aplicando el algoritmo CHAMELEON.

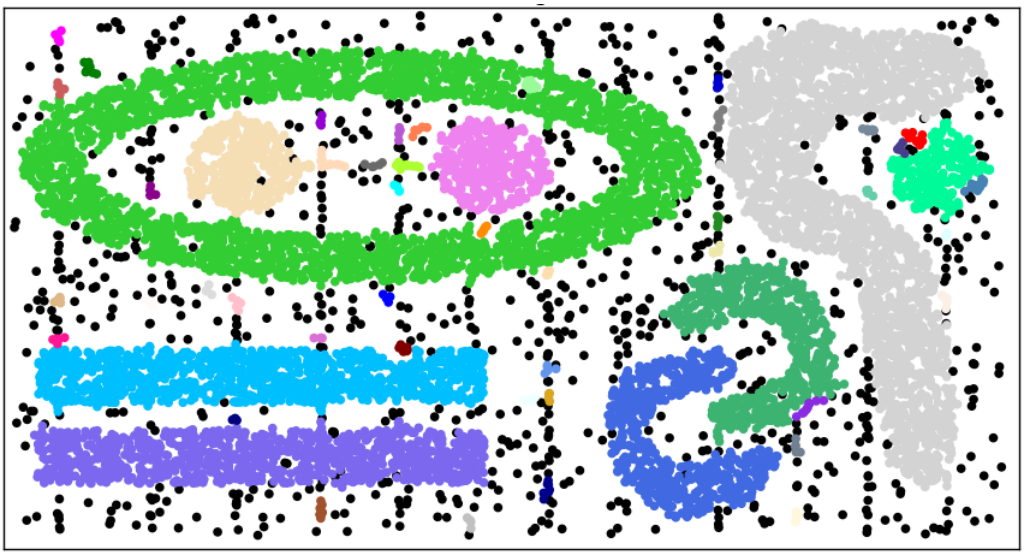

Figura 6. *Clusters* obtenidos aplicando el algoritmo DBSCAN.

Note de la figura 6 que, los puntos de color negro representan ruido en el conjunto de datos. Sin embargo, puede observarse que las formas geométricas son capturadas.

http://glaros.dtc.umn.edu/gkhome/node/152

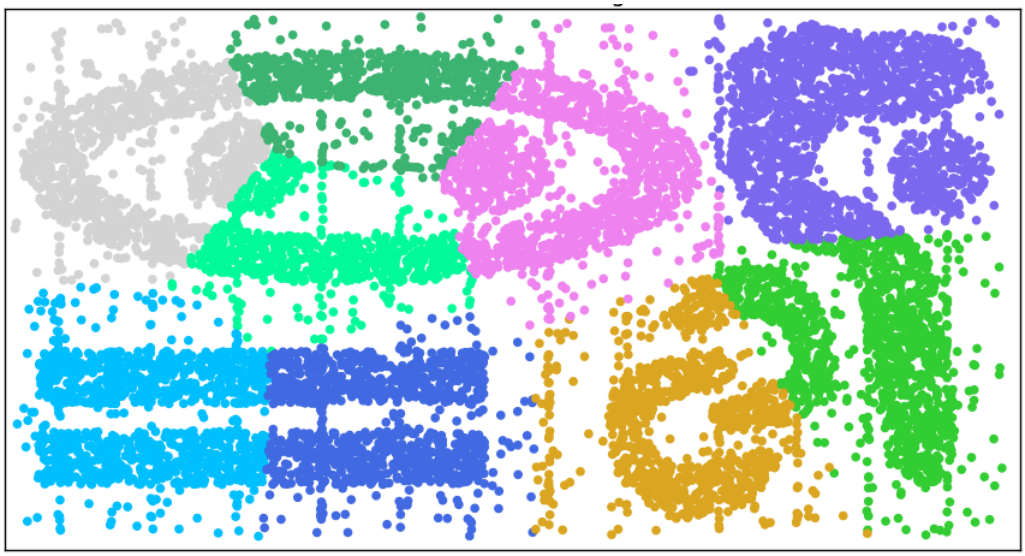

Figura 7. *Clusters* obtenidos aplicando el algoritmo KMeans.

Con base en los resultados obtenidos utilizando KMeans, puede notarse que algunas de las limitaciones de esta técnica tradicional de *clustering* son (por lo menos en el conjunto de datos utilizado para ejemplificar): (1) no captura formas geométricas diferentes a elipses o círculos, ni la variación en tamaño que puedan tener los *clusters*.

Conjunto de Datos # 2[4]:
Repite el procedimiento realizado al conjunto de datos # 1, y utilice el programa en Python "t8.8k.dat.py" para graficar los resultados de la aplicación de CHAMELEON y KMeans[5]. El número de *clusters* en este conjunto de datos es 8.

Finalmente, discuta sobre los resultado obtenidos.

4 El archivo que contiene los datos es "t8.8k.dat".
5 Para este conjunto de datos, no se aplicará el algoritmo DBSCAN.