Tutorial sobre clustering con el paquete 'Clustering'

Seminario Permanente de Formación en Inteligencia Artificial Aplicada a la Defensa

Ángel Miguel García Vico agvico@ujaen.es Universidad de Jaén - DaSCI

23 junio 2020

Contents

Este es un cuaderno o Notebook de R que incluye un tutorial práctico sobre clustering mediante el paquete *Clustering* dentro del "Seminario Permanente de Formación en Inteligencia Artificial Aplicada a la Defensa" conjunto entre el Mando de Adiestramiento y Doctrina y el Instituto Andaluz Interuniversitario DaSCI (Universidad de Granada y Universidad de Jaén).

Existen bloques de texto y bloques de código, sombreados en gris, con instrucciones R. Estas instrucciones, si se ejecutan, el resultado de las mismas aparecerá debajo.

Introducción

El agrupamiento o clustering es un tipo de técnica de aprendizaje no supervisado cuyo objetivo es la formación de subconjuntos de la población donde se maximice la similitud entre instancias pertenencientes a un mismo grupo o *cluster* y se minimice entre instancias que no pertencen al mismo cluster. El objetivo de este tutorial es proporcionar una guía paso a paso para la realización de un estudio de clustering mediante el empleo del paquete *Clustering*. Para ello, se seguirán los siguientes pasos:

1. Instalar y cargar los paquetes necesarios.
2. Cargar un fichero de datos y hacer un análisis exploratorio inicial.
3. Realizar una comparativa de algoritmos de clustering mediante el paquete *Clustering*.
4. Procesar e interpretar los resultados.

Instalar los paquetes requeridos

El primer paso de este tutorial es instalar los paquetes que vamos a usar. Los paquetes son complementos o bibliotecas de terceros que amplian las utilidades de R.

El paquete que vamos a usar se llama **Clustering**, el cual permite realizar de manera sencilla comparativas entre métodos de clustering para seleccionar a continuación el que nos interese. Actualmente se encuentra en fase *alpha*, por lo que tiene un funcionalidad limitada, y está en continuo desarrollo. Por esta razón, es recomendable instalar la versión disponible en el repositorio GitHub del mismo. Para instalarlo, ejecutaremos la siguiente orden (elimine el carácter de comentario #):

```
#install.packages("devtools")
#library(devtools)
#devtools::install_github(repo = "https://github.com/laperez/clustering",
#                         build = T, build_vignettes = T)
```

A continuación, cargue el paquete **Clustering**:

```
library(Clustering)
```

Es importante remarcar que el paquete Clustering posee una interfaz gráfica desde la cual se puede realizar todo este proceso.

Paso Uno. Carga de datos.

En este ejemplo bamos a utilizar el conjunto de datos `basketball` disponible en el paquete **Clustering**. Para cargar este conjunto de datos y poder trabajar con él, se puede ejecutar el siguiente comando:

```
data("basketball")
datos <- basketball
```

Sin embargo, en un trabajo real sería necesario cargar los datos desde un fichero, habitualmente en formato csv. Este mismo ejemplo se proporciona en el fichero *basketball.csv*. Para cargarlo, se ejecutaría el siguente código:

```
datos <- read.csv("basketball.csv", header = T)
```

A continuación, podemos realizar un análisis exploratorio inicial de los mismos mediante la función `summary`:

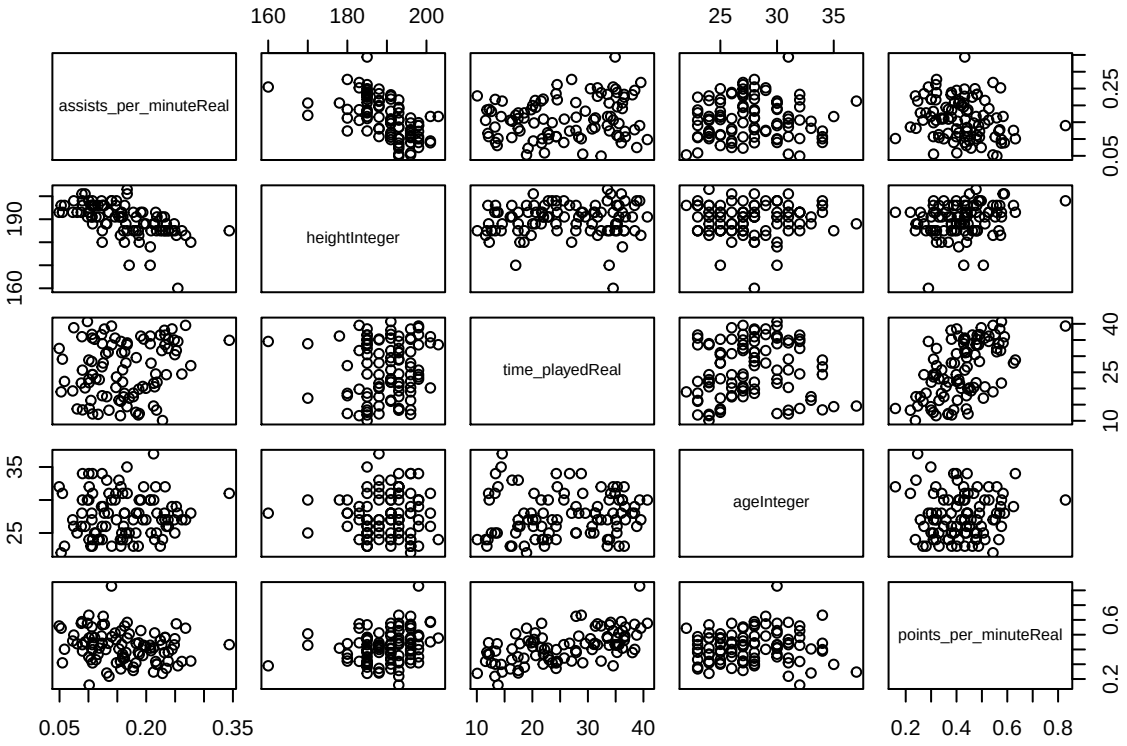
```
head(datos)
```

```
##   assists_per_minuteReal heightInteger time_playedReal ageInteger
## 1                 0.0888           201           36.02         28
## 2                 0.1399           198           39.32         30
## 3                 0.0747           198           38.80         26
## 4                 0.0983           191           40.71         30
## 5                 0.1276           196           38.40         28
## 6                 0.1671           201           34.10         31
##   points_per_minuteReal
## 1                0.5885
## 2                0.8291
## 3                0.4974
## 4                0.5772
## 5                0.5703
## 6                0.5835
```

```
# Resumen de datos
summary(datos)
```

```
##  assists_per_minuteReal heightInteger   time_playedReal   ageInteger
##  Min.   :0.0494         Min.   :160.0   Min.   :10.08   Min.   :22.00
##  1st Qu.:0.1098         1st Qu.:185.0   1st Qu.:18.85   1st Qu.:25.00
##  Median :0.1584         Median :191.0   Median :25.71   Median :27.00
##  Mean   :0.1613         Mean   :189.9   Mean   :25.94   Mean   :27.74
##  3rd Qu.:0.2095         3rd Qu.:196.0   3rd Qu.:33.94   3rd Qu.:30.00
##  Max.   :0.3437         Max.   :203.0   Max.   :40.71   Max.   :37.00
##  points_per_minuteReal
##  Min.   :0.1593
##  1st Qu.:0.3384
##  Median :0.4283
##  Mean   :0.4203
##  3rd Qu.:0.4831
##  Max.   :0.8291
```

```r
# Visualización gráfica por pares de variables:
plot(datos)
```



Realizar una comparativa de algoritmos mediante el paquete Clustering

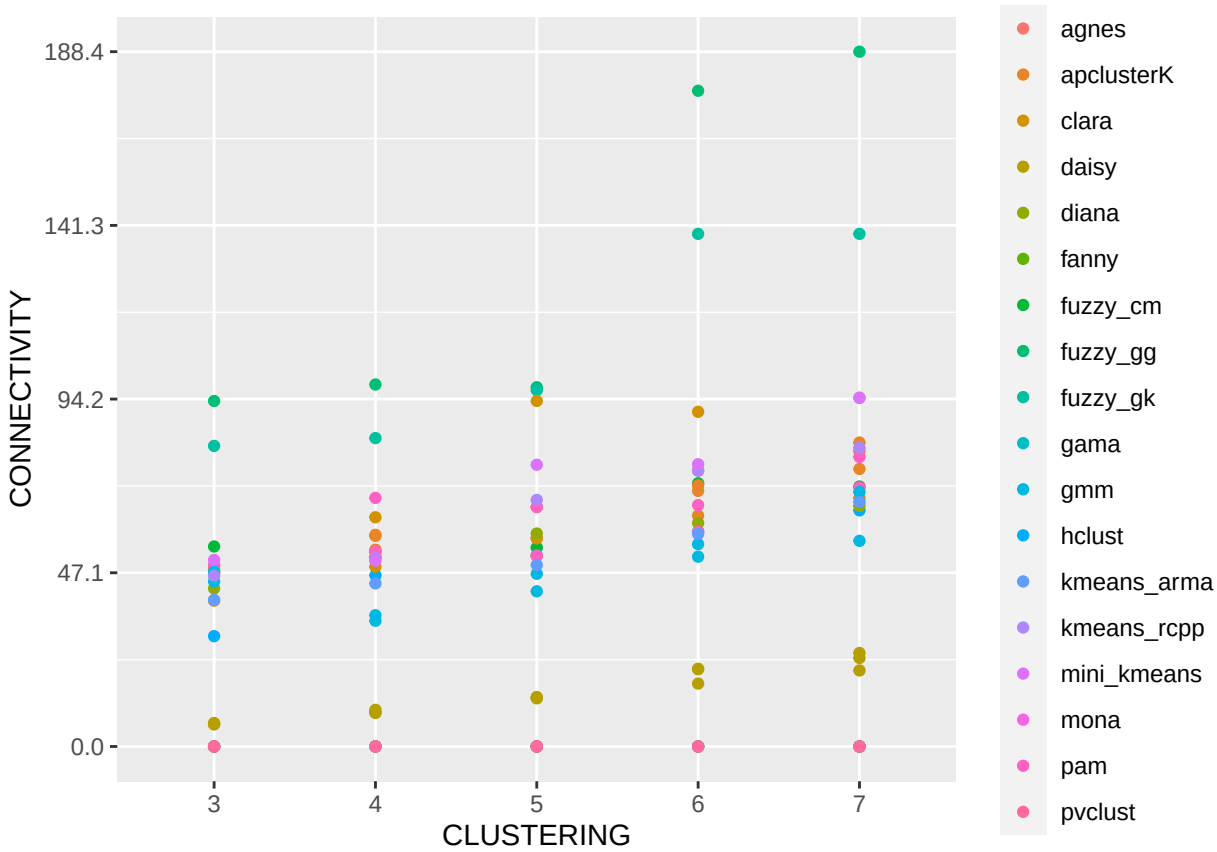
El paquete **Clustering** proporciona el método `clustering` que permite la ejecución simultánea de un amplio conjunto de algoritmos disponibles en varios paquetes de R. Para más información sobre los parámetros de esta función y detalles acerca de su implementación, revise la página de ayuda mediante el comando `help`:

```r
help("clustering", package = "Clustering")
```

En la mayoría de métodos de clustering es necesario establecer el número de clusters que se van a encontrar de antemano. Una de los métodos más utilizados para determinar el número óptimo de clusters es mediante la "regla del codo", que consiste en mostrar en un gráfico la calidad del método con diferentes número de clusters. Para llevarlo a cabo, en primer lugar ejecutaremos el método `clustering`con todos los algoritmos disponibles, con un número de cluster mínimo de 3 y máximo de 7. **Nota:** esta ejecución puede tardar unos minutos en completarse.

Tras la ejecución de este algoritmo, podemos visualizar de manera sencilla en un gráfico una métrica de calidad de las muchas que proporciona este paquete mediante la función `plot_clustering`, de modo que podamos aplicar la regla del codo según nuestras necesidades. Por ejemplo, podemos visualizar el comportamiento de los métodos por un métrica interna, es decir, que `connectivity`, que hay que minimizar :

```r
plot_clustering(resultado, "connectivity")
```



Atendiendo a estos resultados, el mejor número de clusters en todos los algoritmos analizados se sitúa en 3, y los mejores resultados los presentan los algoritmo `agnes` y `daisy`, los cuales podemos ejecutar en sus respectivos paquetes.

Otra opción que proporciona este paquete es ordenar los métodos en función de un ranking de métricas tanto internas, como externas, y seleccionar el método que convenga. Para ello, ejecutamos el siguiente código:

```
#Obtener ranking por métricas internas:
rankInternas <- best_ranked_internal_metrics(resultado)

#Obtener ranking por métricas externas:
rankExternas <- best_ranked_external_metrics(resultado)

print(rankInternas)
```

```
## Result:
##       Algorithm            Distance Clusters   Dataset Ranking timeInternal
## 1      fuzzy_cm            fuzzy_cm        3 dataframe       1       0.0046
## 2      fuzzy_cm            fuzzy_cm        4 dataframe       1       0.0045
## 3      fuzzy_cm            fuzzy_cm        5 dataframe       1       0.0045
## 4      fuzzy_cm            fuzzy_cm        6 dataframe       1       0.0045
## 5      fuzzy_cm            fuzzy_cm        7 dataframe       1       0.0045
## 6      fuzzy_gg            fuzzy_gg        3 dataframe       1       0.0045
## 7      fuzzy_gg            fuzzy_gg        4 dataframe       1       0.0044
## 8      fuzzy_gg            fuzzy_gg        5 dataframe       1       0.0044
## 9      fuzzy_gg            fuzzy_gg        6 dataframe       1       0.0045
## 10     fuzzy_gg            fuzzy_gg        7 dataframe       1       0.0045
## 11     fuzzy_gk            fuzzy_gk        3 dataframe       1       0.0054
## 12     fuzzy_gk            fuzzy_gk        4 dataframe       1       0.0054
```

```
## 13     fuzzy_gk             fuzzy_gk       5 dataframe       1       0.0054
## 14     fuzzy_gk             fuzzy_gk       6 dataframe       1       0.0054
## 15     fuzzy_gk             fuzzy_gk       7 dataframe       1       0.0053
## 16       hclust     hclust_euclidean       3 dataframe       1       0.0047
## 17       hclust     hclust_euclidean       4 dataframe       1       0.0052
## 18       hclust     hclust_euclidean       5 dataframe       1       0.0047
## 19       hclust     hclust_euclidean       6 dataframe       1       0.0052
## 20       hclust     hclust_euclidean       7 dataframe       1       0.0048
## 21   apclusterK apclusterK_euclidean       3 dataframe       1       0.0044
## 22   apclusterK apclusterK_euclidean       4 dataframe       1       0.0044
## 23   apclusterK apclusterK_euclidean       5 dataframe       1       0.0045
## 24   apclusterK apclusterK_euclidean       6 dataframe       1       0.0044
## 25   apclusterK apclusterK_euclidean       7 dataframe       1       0.0047
## 26   apclusterK apclusterK_manhattan       3 dataframe       1       0.0045
## 27   apclusterK apclusterK_manhattan       4 dataframe       1       0.0046
## 28   apclusterK apclusterK_manhattan       5 dataframe       1       0.0045
## 29   apclusterK apclusterK_manhattan       6 dataframe       1       0.0047
## 30   apclusterK apclusterK_manhattan       7 dataframe       1       0.0046
## 31   apclusterK apclusterK_minkowski       3 dataframe       1       0.0054
## 32   apclusterK apclusterK_minkowski       4 dataframe       1       0.0057
## 33   apclusterK apclusterK_minkowski       5 dataframe       1       0.0053
## 34   apclusterK apclusterK_minkowski       6 dataframe       1       0.0054
## 35   apclusterK apclusterK_minkowski       7 dataframe       1       0.0057
## 36       agnes      agnes_euclidean       3 dataframe       1       0.0000
## 37       agnes      agnes_euclidean       4 dataframe       1       0.0000
## 38       agnes      agnes_euclidean       5 dataframe       1       0.0000
## 39       agnes      agnes_euclidean       6 dataframe       1       0.0000
## 40       agnes      agnes_euclidean       7 dataframe       1       0.0000
## 41       agnes      agnes_manhattan       3 dataframe       1       0.0000
## 42       agnes      agnes_manhattan       4 dataframe       1       0.0000
## 43       agnes      agnes_manhattan       5 dataframe       1       0.0000
## 44       agnes      agnes_manhattan       6 dataframe       1       0.0000
## 45       agnes      agnes_manhattan       7 dataframe       1       0.0000
## 46       clara      clara_euclidean       3 dataframe       1       0.0045
## 47       clara      clara_euclidean       4 dataframe       1       0.0046
## 48       clara      clara_euclidean       5 dataframe       1       0.0045
## 49       clara      clara_euclidean       6 dataframe       1       0.0045
## 50       clara      clara_euclidean       7 dataframe       1       0.0046
## 51       clara      clara_manhattan       3 dataframe       1       0.0046
## 52       clara      clara_manhattan       4 dataframe       1       0.0046
## 53       clara      clara_manhattan       5 dataframe       1       0.0047
## 54       clara      clara_manhattan       6 dataframe       1       0.0046
## 55       clara      clara_manhattan       7 dataframe       1       0.0047
## 56       daisy      daisy_manhattan       3 dataframe       1       0.0048
## 57       daisy      daisy_manhattan       4 dataframe       1       0.0048
## 58       daisy      daisy_manhattan       5 dataframe       1       0.0047
## 59       daisy      daisy_manhattan       6 dataframe       1       0.0047
## 60       daisy      daisy_manhattan       7 dataframe       1       0.0047
## 61       daisy          daisy_gower       3 dataframe       1       0.0048
## 62       daisy          daisy_gower       4 dataframe       1       0.0049
## 63       daisy          daisy_gower       5 dataframe       1       0.0050
## 64       daisy          daisy_gower       6 dataframe       1       0.0047
## 65       daisy          daisy_gower       7 dataframe       1       0.0049
## 66       daisy      daisy_euclidean       3 dataframe       1       0.0046
```

```
## 67        daisy      daisy_euclidean        4 dataframe       1       0.0045
## 68        daisy      daisy_euclidean        5 dataframe       1       0.0045
## 69        daisy      daisy_euclidean        6 dataframe       1       0.0045
## 70        daisy      daisy_euclidean        7 dataframe       1       0.0045
## 71        diana      diana_euclidean        3 dataframe       1       0.0057
## 72        diana      diana_euclidean        4 dataframe       1       0.0049
## 73        diana      diana_euclidean        5 dataframe       1       0.0048
## 74        diana      diana_euclidean        6 dataframe       1       0.0048
## 75        diana      diana_euclidean        7 dataframe       1       0.0046
## 76        fanny      fanny_euclidean        3 dataframe       1       0.0000
## 77        fanny      fanny_euclidean        4 dataframe       1       0.0000
## 78        fanny      fanny_euclidean        5 dataframe       1       0.0000
## 79        fanny      fanny_euclidean        6 dataframe       1       0.0000
## 80        fanny      fanny_euclidean        7 dataframe       1       0.0000
## 81        fanny      fanny_manhattan        3 dataframe       1       0.0000
## 82        fanny      fanny_manhattan        4 dataframe       1       0.0000
## 83        fanny      fanny_manhattan        5 dataframe       1       0.0000
## 84        fanny      fanny_manhattan        6 dataframe       1       0.0000
## 85        fanny      fanny_manhattan        7 dataframe       1       0.0000
## 86         mona                 mona        3 dataframe       1       0.0000
## 87         mona                 mona        4 dataframe       1       0.0000
## 88         mona                 mona        5 dataframe       1       0.0000
## 89         mona                 mona        6 dataframe       1       0.0000
## 90         mona                 mona        7 dataframe       1       0.0000
## 91          pam       pam_euclidean        3 dataframe       1       0.0046
## 92          pam       pam_euclidean        4 dataframe       1       0.0045
## 93          pam       pam_euclidean        5 dataframe       1       0.0046
## 94          pam       pam_euclidean        6 dataframe       1       0.0045
## 95          pam       pam_euclidean        7 dataframe       1       0.0045
## 96          pam       pam_manhattan        3 dataframe       1       0.0045
## 97          pam       pam_manhattan        4 dataframe       1       0.0047
## 98          pam       pam_manhattan        5 dataframe       1       0.0047
## 99          pam       pam_manhattan        6 dataframe       1       0.0047
## 100         pam       pam_manhattan        7 dataframe       1       0.0048
## 101         gmm       gmm_euclidean        3 dataframe       1       0.0045
## 102         gmm       gmm_euclidean        4 dataframe       1       0.0045
## 103         gmm       gmm_euclidean        5 dataframe       1       0.0045
## 104         gmm       gmm_euclidean        6 dataframe       1       0.0046
## 105         gmm       gmm_euclidean        7 dataframe       1       0.0046
## 106         gmm       gmm_manhattan        3 dataframe       1       0.0046
## 107         gmm       gmm_manhattan        4 dataframe       1       0.0046
## 108         gmm       gmm_manhattan        5 dataframe       1       0.0047
## 109         gmm       gmm_manhattan        6 dataframe       1       0.0046
## 110         gmm       gmm_manhattan        7 dataframe       1       0.0047
## 111 kmeans_arma         kmeans_arma        3 dataframe       1       0.0045
## 112 kmeans_arma         kmeans_arma        4 dataframe       1       0.0043
## 113 kmeans_arma         kmeans_arma        5 dataframe       1       0.0044
## 114 kmeans_arma         kmeans_arma        6 dataframe       1       0.0044
## 115 kmeans_arma         kmeans_arma        7 dataframe       1       0.0045
## 116 kmeans_rcpp         kmeans_rcpp        3 dataframe       1       0.0046
## 117 kmeans_rcpp         kmeans_rcpp        4 dataframe       1       0.0046
## 118 kmeans_rcpp         kmeans_rcpp        5 dataframe       1       0.0045
## 119 kmeans_rcpp         kmeans_rcpp        6 dataframe       1       0.0046
## 120 kmeans_rcpp         kmeans_rcpp        7 dataframe       1       0.0046
```

```
## 121 mini_kmeans          mini_kmeans       3 dataframe       1       0.0046
## 122 mini_kmeans          mini_kmeans       4 dataframe       1       0.0045
## 123 mini_kmeans          mini_kmeans       5 dataframe       1       0.0044
## 124 mini_kmeans          mini_kmeans       6 dataframe       1       0.0045
## 125 mini_kmeans          mini_kmeans       7 dataframe       1       0.0044
## 126        gama       gama_euclidean       3 dataframe       1       0.0000
## 127        gama       gama_euclidean       4 dataframe       1       0.0000
## 128        gama       gama_euclidean       5 dataframe       1       0.0000
## 129        gama       gama_euclidean       6 dataframe       1       0.0000
## 130        gama       gama_euclidean       7 dataframe       1       0.0000
## 131     pvclust    pvclust_euclidean       3 dataframe       1       0.0000
## 132     pvclust    pvclust_euclidean       4 dataframe       1       0.0000
## 133     pvclust    pvclust_euclidean       5 dataframe       1       0.0000
## 134     pvclust    pvclust_euclidean       6 dataframe       1       0.0000
## 135     pvclust    pvclust_euclidean       7 dataframe       1       0.0000
## 136     pvclust  pvclust_correlation       3 dataframe       1       0.0058
## 137     pvclust  pvclust_correlation       4 dataframe       1       0.0059
## 138     pvclust  pvclust_correlation       5 dataframe       1       0.0057
## 139     pvclust  pvclust_correlation       6 dataframe       1       0.0000
## 140     pvclust  pvclust_correlation       7 dataframe       1       0.0000
##     connectivity   dunn silhouette
## 1         54.230 0.1486       0.24
## 2         51.370 0.1682       0.27
## 3         53.950 0.1538       0.25
## 4         71.370 0.1741       0.23
## 5         70.490 0.1619       0.21
## 6         93.680 0.1016       0.12
## 7         98.160 0.1143       0.06
## 8         97.420 0.1029       0.10
## 9        177.800 0.0961       0.02
## 10       188.400 0.1016       0.00
## 11        81.510 0.1289       0.15
## 12        83.620 0.1393       0.15
## 13        96.610 0.1280       0.11
## 14       139.000 0.1172       0.09
## 15       139.000 0.1241       0.10
## 16        29.920 0.1465       0.22
## 17        46.450 0.1684       0.19
## 18        51.880 0.1799       0.19
## 19        58.310 0.2149       0.21
## 20        64.030 0.2218       0.21
## 21        49.040 0.1396       0.23
## 22        53.370 0.1413       0.26
## 23        51.690 0.1495       0.25
## 24        62.640 0.1648       0.24
## 25        75.260 0.1540       0.22
## 26        47.210 0.1122       0.24
## 27        57.100 0.1162       0.24
## 28        64.940 0.1490       0.24
## 29        70.810 0.1490       0.23
## 30        80.100 0.1515       0.21
## 31        46.590 0.1260       0.25
## 32        57.400 0.1295       0.26
## 33        56.430 0.1198       0.25
```

```
## 34        69.320 0.1322       0.24
## 35        82.430 0.1509       0.21
## 36         0.000 0.0000       0.00
## 37         0.000 0.0000       0.00
## 38         0.000 0.0000       0.00
## 39         0.000 0.0000       0.00
## 40         0.000 0.0000       0.00
## 41         0.000 0.0000       0.00
## 42         0.000 0.0000       0.00
## 43         0.000 0.0000       0.00
## 44         0.000 0.0000       0.00
## 45         0.000 0.0000       0.00
## 46        48.090 0.1413       0.23
## 47        48.760 0.1465       0.25
## 48        56.520 0.1495       0.24
## 49        90.760 0.1474       0.19
## 50        67.350 0.1838       0.24
## 51        39.550 0.1102       0.24
## 52        62.150 0.1122       0.22
## 53        93.700 0.1182       0.15
## 54        74.830 0.1500       0.22
## 55        94.550 0.1182       0.18
## 56         6.025 0.2916       0.20
## 57         9.096 0.2623       0.07
## 58        13.190 0.1695      -0.02
## 59        20.950 0.1722      -0.04
## 60        24.000 0.1722      -0.08
## 61         6.025 0.2454       0.14
## 62         9.204 0.2239      -0.02
## 63        13.400 0.1789      -0.11
## 64        21.080 0.1835      -0.06
## 65        25.360 0.1596      -0.11
## 66         6.358 0.3520       0.27
## 67         9.954 0.1895       0.15
## 68        13.050 0.1895       0.08
## 69        17.060 0.1245       0.00
## 70        20.620 0.1326      -0.09
## 71        42.850 0.1322       0.23
## 72        52.670 0.1165       0.21
## 73        57.760 0.1396       0.22
## 74        60.590 0.1408       0.21
## 75        65.240 0.1560       0.22
## 76         0.000 0.0000       0.00
## 77         0.000 0.0000       0.00
## 78         0.000 0.0000       0.00
## 79         0.000 0.0000       0.00
## 80         0.000 0.0000       0.00
## 81         0.000 0.0000       0.00
## 82         0.000 0.0000       0.00
## 83         0.000 0.0000       0.00
## 84         0.000 0.0000       0.00
## 85         0.000 0.0000       0.00
## 86         0.000 0.0000       0.00
## 87         0.000 0.0000       0.00
```

```
## 88         0.000 0.0000       0.00
## 89         0.000 0.0000       0.00
## 90         0.000 0.0000       0.00
## 91        49.040 0.1396       0.23
## 92        52.810 0.1396       0.25
## 93        51.690 0.1495       0.25
## 94        58.150 0.1657       0.23
## 95        70.200 0.1838       0.22
## 96        47.210 0.1122       0.24
## 97        67.410 0.1332       0.22
## 98        64.940 0.1490       0.24
## 99        65.460 0.1490       0.24
## 100       78.500 0.1500       0.22
## 101       47.390 0.1096       0.18
## 102       34.090 0.1646       0.23
## 103       42.080 0.1619       0.25
## 104       51.460 0.1619       0.23
## 105       55.770 0.1657       0.23
## 106       44.730 0.1183       0.19
## 107       35.590 0.1348       0.23
## 108       46.830 0.1322       0.26
## 109       54.870 0.1467       0.25
## 110       69.020 0.1500       0.24
## 111       39.770 0.1235       0.19
## 112       44.210 0.1495       0.23
## 113       49.220 0.1538       0.26
## 114       57.630 0.1619       0.24
## 115       66.320 0.1520       0.23
## 116       46.310 0.1465       0.25
## 117       51.040 0.1741       0.23
## 118       66.850 0.1520       0.19
## 119       74.780 0.1522       0.19
## 120       81.020 0.1522       0.18
## 121       50.590 0.1350       0.23
## 122       50.350 0.1571       0.21
## 123       76.400 0.1216       0.17
## 124       76.530 0.1500       0.17
## 125       94.570 0.1578       0.18
## 126        0.000 0.0000       0.00
## 127        0.000 0.0000       0.00
## 128        0.000 0.0000       0.00
## 129        0.000 0.0000       0.00
## 130        0.000 0.0000       0.00
## 131        0.000 0.0000       0.00
## 132        0.000 0.0000       0.00
## 133        0.000 0.0000       0.00
## 134        0.000 0.0000       0.00
## 135        0.000 0.0000       0.00
## 136        0.000 0.1686       0.00
## 137        0.000 1.1470       0.00
## 138        0.000 0.0000       0.00
## 139        0.000 0.0000       0.00
## 140        0.000 0.0000       0.00
```

Comentarios Finales

Si necesitan más detalles sobre como realizar otro tipo de tareas, no duden en poner en contacto conmigo a través de correo electrónico en la siguente dirección: agvico@ujaen.es.