National Technology of Mexico
Technological Institute of Tijuana

ACADEMIC SUBDIRECTION
Systems and Computing Department

SEMESTER
February - June 2021

RACE
Computer Systems Engineer
and
Information and Communication Technologies Engineer

Subject and Key
Minería de Datos BDD-1703TI9A

STUDENT'S NAME AND REGISTRATION
Camacho Manabe Juan Daniel 17210534
Velázquez Farrera César Alejandro 17212937

NAME OF THE JOB
Práctica Evaluatoria #4

UNIT TO BE EVALUATED
Unidad IV

TEACHER'S NAME
M.C. José Christian Romero Hernández

Tijuana, Baja California **Due date:** June 18, 2021

Evaluative Practice #4

Instructions

Implement the K-Means grouping model with the Iris.csv dataset found at https://github.com/jcromerohdz/iris using the kmeans () method in R. Once the grouping model is obtained do the corresponding data visualization analysis.
At the end of the development, explain in detail what the K-Means grouping model consists of and what were your observations in the data visualization analysis.

Código fuente

To start the practice, we must find our data file in the system. Once the file has been found, the data can be loaded in the following different ways:

1. The establishment of the working directory.
2. Through the file.choose () method

```
//Primer metodo
setwd('C:\\Users\\vcesa\\Documents')
Iris = read.csv('Iris.csv')

//Segundo método
Iris = read.csv(file.choose())
```

```
C:/Users/vcesa/Desktop/Tareas/8vo Semestre/Minería de Datos/source/Unit_3/
> iris = read.csv(file.choose())
> iris
    i..sepal.length sepal.width petal.length petal.width    variety
1               5.1         3.5          1.4         0.2     Setosa
2               4.9         3.0          1.4         0.2     Setosa
3               4.7         3.2          1.3         0.2     Setosa
4               4.6         3.1          1.5         0.2     Setosa
5               5.0         3.6          1.4         0.2     Setosa
6               5.4         3.9          1.7         0.4     Setosa
7               4.6         3.4          1.4         0.3     Setosa
8               5.0         3.4          1.5         0.2     Setosa
9               4.4         2.9          1.4         0.2     Setosa
10              4.9         3.1          1.5         0.1     Setosa
11              5.4         3.7          1.5         0.2     Setosa
12              4.8         3.4          1.6         0.2     Setosa
13              4.8         3.0          1.4         0.1     Setosa
14              4.3         3.0          1.1         0.1     Setosa
15              5.8         4.0          1.2         0.2     Setosa
16              5.7         4.4          1.5         0.4     Setosa
17              5.4         3.9          1.3         0.4     Setosa
```

Then, to better understand the characteristics of the data set, the following functions are used.

```
str(Iris)          #Visualizar la estructura del conjunto de datos
summary(Iris)      #Resumen estadístico del conjunto de datos
head(Iris)         #Visualización de las primeras tuplas del conjunto
```

```
C:/Users/vcesa/Desktop/Tareas/8vo Semestre/Minería de Datos/source/Unit_3/
> str(iris) #view structure of dataset
'data.frame':   150 obs. of  5 variables:
 $ ï..sepal.length: num  5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
 $ sepal.width    : num  3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
 $ petal.length   : num  1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
 $ petal.width    : num  0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
 $ variety        : Factor w/ 3 levels "Setosa","Versicolor",..: 1 1 1 1 1 1 1 1 1 1 ...
>
```

```
> summary(iris) #view statistical summary of dataset
 ï..sepal.length  sepal.width     petal.length    petal.width          variety
 Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100   Setosa    :50
 1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300   Versicolor:50
 Median :5.800   Median :3.000   Median :4.350   Median :1.300   Virginica :50
 Mean   :5.843   Mean   :3.057   Mean   :3.758   Mean   :1.199
 3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100   3rd Qu.:1.800
 Max.   :7.900   Max.   :4.400   Max.   :6.900   Max.   :2.500
>
```

```
> head(iris) #view top  rows of dataset
  ï..sepal.length sepal.width petal.length petal.width variety
1             5.1         3.5          1.4         0.2  Setosa
2             4.9         3.0          1.4         0.2  Setosa
3             4.7         3.2          1.3         0.2  Setosa
4             4.6         3.1          1.5         0.2  Setosa
5             5.0         3.6          1.4         0.2  Setosa
6             5.4         3.9          1.7         0.4  Setosa
>
```

To prepare the data set for data normalization, the following methods are used:

1. A new dataset called “Iris.new” is created and the first four columns of data are inserted.
2. Another new set "Iris.class" is created where the species belonging to each tuple are housed.

```
#Preprocess the dataset
Iris.new <- Iris[, c(1, 2, 3, 4)]
Iris.class <- Iris[, "Species"]
```

```
> head(iris.new)
  Sepal.Length Sepal.Width Petal.Length Petal.Width
1          5.1         3.5          1.4         0.2
2          4.9         3.0          1.4         0.2
3          4.7         3.2          1.3         0.2
4          4.6         3.1          1.5         0.2
5          5.0         3.6          1.4         0.2
6          5.4         3.9          1.7         0.4
>
```

```
> head(iris.class)
[1] setosa setosa setosa setosa setosa setosa
Levels: setosa versicolor virginica
>
```

For the normalization of the data, a function is used that returns a column of normalized data. As parameter it takes the column to be normalized. To obtain the normalized data set, the function is used in each column. At the end of the function execution, the result of the normalization will be displayed.

But what is normalization?
Data normalization is essentially a type of process in which the data within a database is reorganized in such a way that users can appropriately use that database for further queries and analysis.

There are some goals in mind when undertaking the data normalization process. The first is to get rid of any duplicate data that may appear within the dataset. This basically goes through the database and eliminates any redundancy that may occur. Redundancies can negatively affect data analysis, as they are values that are not exactly needed. Deleting them from the database helps clean up the data, making it easier to analyze. The other goal is to logically group the data. You want the data that relates to each other to be stored together. This will occur in a database that has been normalized. If the data depend on each other, they must be very close within the data set.

```
normalize <- function(x){
  return ((x-min(x))/(max(x)-min(x)))
}

iris.new$Sepal.Length<- normalize(iris.new$Sepal.Length)
iris.new$Sepal.Width<- normalize(iris.new$Sepal.Width)
iris.new$Petal.Length<- normalize(iris.new$Petal.Length)
iris.new$Petal.Width<- normalize(iris.new$Petal.Width)
head(iris.new)
```

```
> head(iris.new)
  Sepal.Length Sepal.Width Petal.Length Petal.Width
1    0.22222222   0.6250000   0.06779661  0.04166667
2    0.16666667   0.4166667   0.06779661  0.04166667
3    0.11111111   0.5000000   0.05084746  0.04166667
4    0.08333333   0.4583333   0.08474576  0.04166667
5    0.19444444   0.6666667   0.06779661  0.04166667
6    0.30555556   0.7916667   0.11864407  0.12500000
>
```

Once the data is normalized, it is necessary to obtain the appropriate number of clusters to invoke the K-Means method.

Elbow Method

To obtain the adequate number of clusters it is necessary to apply the elbow method. It consists of the sum of total squares within the data set as a function of the number of clusters. The location of the elbow is often considered a good indicator of clusters, because it means that adding another cluster does not improve the partition much.

```
#Using the elbow method to find the optimal number of clusters
set.seed(6)
wcss = vector()
for (i in 1:10) wcss[i] = sum(kmeans(iris.new, i)$withinss)
plot(1:10,
     wcss,
     type = 'b',
     main = paste('The Elbow Method'),
     xlab = 'Number of clusters',
     ylab = 'WCSS')
```

What is K-Means?

The K Means algorithm is an iterative algorithm that attempts to divide the data set into distinct non-overlapping subgroups (clusters) previously defined by K where each data point belongs to a single group. You try to make the data points within the cluster as similar as possible while keeping the clusters as different (far away) as possible. Assign data points to a group so that the sum of the squared distance between the data points and the centroid of the group (arithmetic mean of all data points that belong to that group) is minimal. The less variation we have within the clusters, the more homogeneous (similar) the data points will be within the same cluster.

To obtain the result of the K-Means algorithm, you need to use the kmeans () function
To invoke the method, you need to send some parameters to the function, they are the following:

- X: Numeric array of data, or an object that can be forced into such a array (such as a numeric vector or a data frame with all numeric columns).
- Centers: Either the number of clusters, say k, or a set of initial (distinct) cluster centers. If it is a number, a random set of (distinct) rows at x is chosen as the initial centers.

```
result<- kmeans(iris.new,3)   #apply k-means algorithm with no. of
centroids(k)=3
result$size  # returns the number of records in each cluster
```

```
> result$size # gives no. of records in each cluster
[1] 39 61 50
>
```

To get the value of the center of the cluster, in other words, give the approximate values of each column of the three centers.

```
result$centers # gives value of cluster center datapoint value(3 centers for k=3)
```

```
> result$centers # gives value of cluster center datapoint value(3
 centers for k=3)
  Sepal.Length Sepal.Width Petal.Length Petal.Width
1    0.7072650   0.4508547    0.79704476  0.82478632
2    0.4412568   0.3073770    0.57571548  0.54918033
3    0.1961111   0.5950000    0.07830508  0.06083333
>
```

To visualize which species each tuple in the data set fell into.

```
result$cluster #gives cluster vector showing the custer where each record falls
```

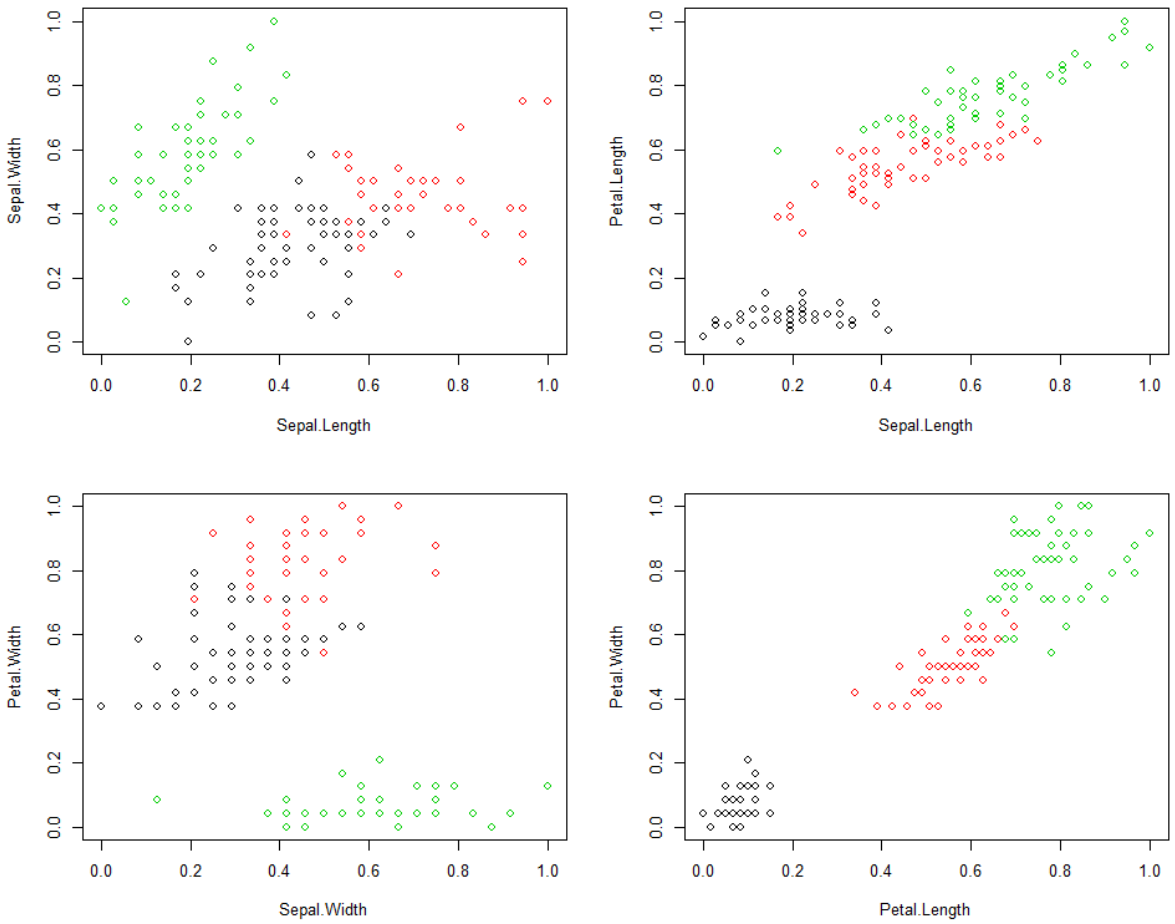
```
> result$cluster #gives cluster vector showing the custer where each record falls
  [1] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
 [30] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 1 2 1 2 2 2 2 2
 [59] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2
 [88] 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 1 1 1 1 2 1 1 1 1 1 1 2 1 1
[117] 1 1 1 2 1 2 1 2 1 1 2 2 1 1 1 1 1 2 2 1 1 1 2 1 1 1 2 1 1
[146] 1 2 1 1 2
```

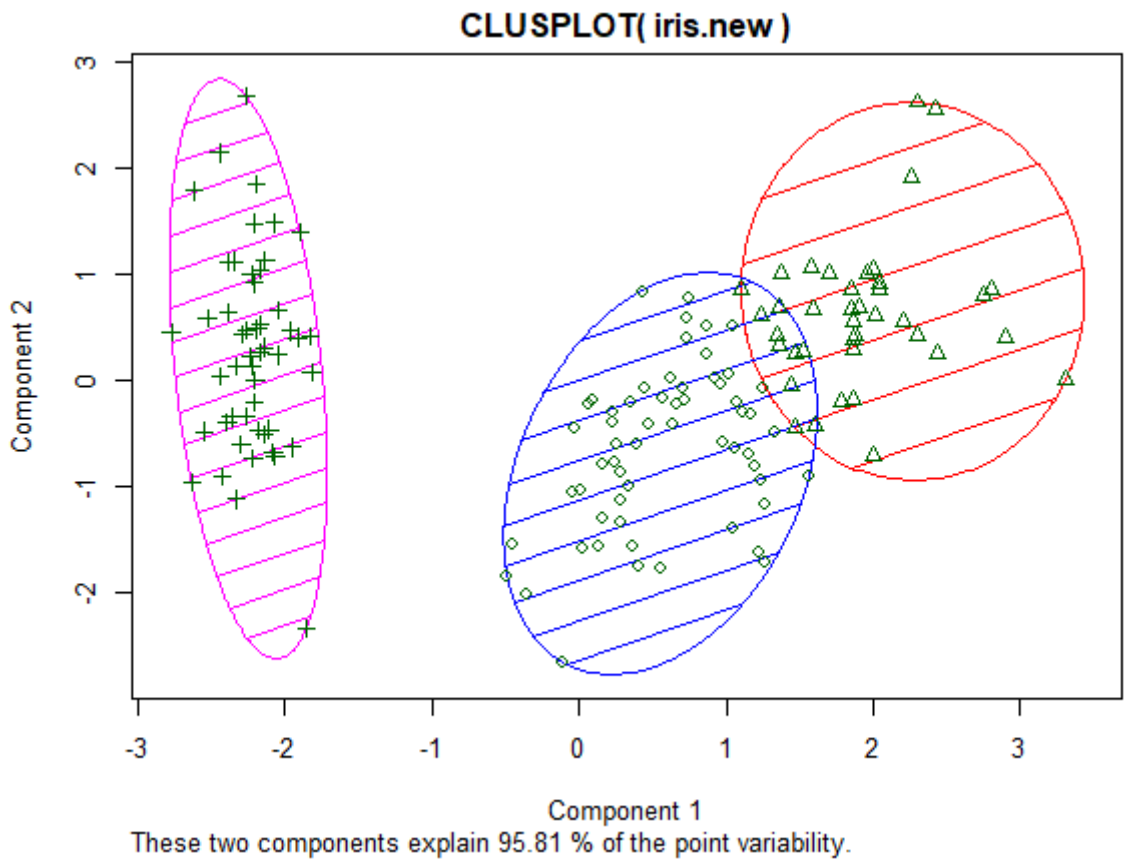
The 'par' function can be to integrate parameters for graphs. Parameters can be specified by means of arguments. By means of this, the graphs are arranged in a 4x4 grid.

```
#Verify results of clustering with graphics
par(mfrow=c(2,2), mar=c(5,4,2,2))
plot(iris.new[c(1,2)], col=result$cluster)# Plot to see how Sepal.Length and Sepal.Width data points have been distributed in clusters
plot(iris.new[c(1,2)], col=iris.class)# Plot to see how Sepal.Length and Sepal.Width data points have been distributed originally as per "class" attribute in dataset
plot(iris.new[c(3,4)], col=result$cluster)# Plot to see how Petal.Length and Petal.Width data points have been distributed in clusters
plot(iris.new[c(3,4)], col=iris.class)
```

Each of the graphs shows the comparisons between the different characteristics of the flowers. In the first graph, the width of a flower and the length of one are compared; the second, the length of the petal and the length of the sepal; the third, the width of the sepal and the petal; the fourth, the width of the petal and its length.

Finally, a two-dimensional graph is shown where it is shown where the different types of flowers are.




CLUSPLOT(iris.new)
Component 2
Component 1
3
2
1
0
-1
-2
-3
-2
-1
0
1
2
3
These two components explain 95.81 % of the point variability.

Referencias

1. Sunny Anand. (2017). Finding Optimal Number of Clusters. Junio 18, 2021. R Bloggers. Sitio web: https://www.r-bloggers.com/2017/02/finding-optimal-number-of-clusters/
2. Imad Dabbura. (2018). K-Means Clustering: Algorithm, Applications, Evaluation Methods and Drawbacks. Junio 18, 2021. Towards Data Science. Sitio web: https://towardsdatascience.com/k-means-clustering-algorithm-applications-evaluation-methods-and-drawbacks-aa03e644b48a
3. CTi Soluciones. Normalizacion de Base de Datos. Junio 28, 2021. Sitio web: https://www.ctisoluciones.com/blog/normalizacion-base-de-datos