# Laboratorio 03: kMeans: algoritmo de clustering

**Por:**
**Ian Gabriel Cañas Fernández, 1092228**

.

**Profesor: Juan S. Pérez R.,**
**Asignatura: INL367L, Secc 01**

## Resumen:

El presente laboratorio estará apreciando el método por agrupamiento, que busca la partición por grupos de una cantidad dada de observaciones, cada una perteneciendo a un valor medio más cercano. Es decir, se procurará encontrar patrones e identificarlos en un dataset. Además, se procurará aplicar este método, llamado k-means, para realizar una segmentación por colores de una imagen.

## Ejercicios previos:

### P3.1. Algoritmo k-Means.

Para la búsqueda de sentido a un conjunto de datos sin etiquetas pero con distinción aparente, se reconoce el algoritmo k-Means, que procura la agrupación de los valores observados a una cantidad K de centroides que conforman buenas representaciones de datos. El funcionamiento de este algoritmo se ve plasmado por Alpaydin en la Ilustración 1:

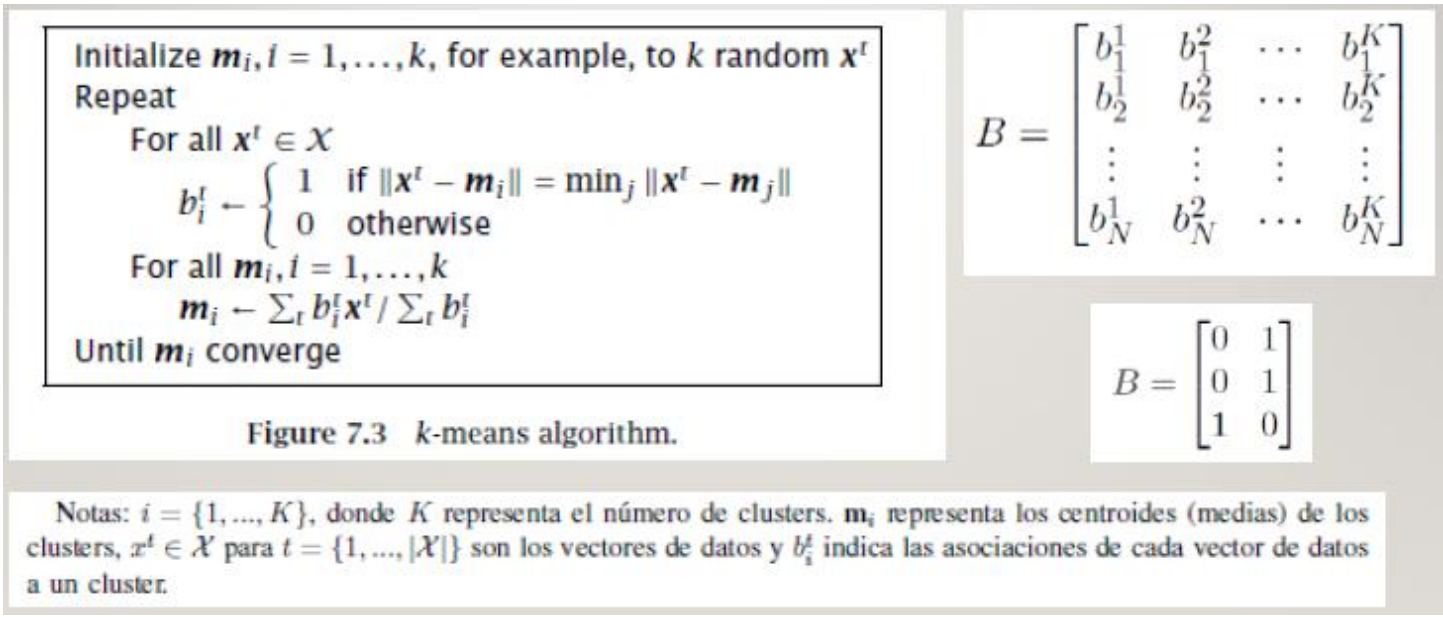


*Ilustración 1. Psudocódigo de K-Means.*

Este algoritmo da a entender que se ubican varios centros en posiciones al azar en el dataset y a partir de ello se va reajustando su posición en base a los grupos de datos cercanos a este. Dándole importancia a ciertas mediciones para cada centroide.

### P3.2. Empleo de K-Means para realizar una segmentación por colores de una imagen.

Para esta sección, busca aplicar el algorimo implementado en la sección anterior en la segmentación por colores de una imagen sustituyendo los píxeles de la imagen original por los centroides de los racimos (clústers) encontrados a los que pertenecen.

Además, se solicita comparar los resultados obtenidos mediante este método y mediante el uso de la función modefilt a un tamaño de filtro de 9x9, y así comparar los resultados que este obtenga.

## E3.1 Implementación en Matlab de K-Means.

En la **Erreur ! Source du renvoi introuvable.** se puede observar el algoritmo que genera la base de datos de las "observaciones" como creando una gráfica con grupos seccionados por categorías. En la Ilustración 3 se aprecia el algoritmo encargado de crear 5 centroides al azar a partir de cuyas coordenadas iniciará le reubicación de sus posiciones. En la Ilustración 4 se presenta el proceso utilizado para graficar los datos por colores según las distinciones que se tuvieron en

```
clc, clear
hold off
tic

n = 300;
k=5;

for i = 1:10
    mu(i) = rand*100;
    desvEst(i) = rand*15;
    varianza(i) = desvEst(i)^2;
    X(:,i) = normrnd(mu(i), desvEst(i), n, 1);
end

Datos = [X(:,1) X(:,2); X(:,3) X(:,4); X(:,5) X(:,6); X(:,7) X(:,8); X(:,9) X(:,10)];

plot(X(:,1),X(:,2), '.k',X(:,3),X(:,4), '.k',X(:,5),X(:,6), '.k',X(:,7),X(:,8), '.k',X(:,9),X(:,10), '.k')
```

*Ilustración 2. Adaptación k-NN a varios tipos de distancias.*

```
figure
m = rand(k,2)*100;


hold on
plot(m(:, 1), m(:, 2), '^b')

dentro = true;
iteraciones = 0;
while dentro == true
% for z=1:100
    iteraciones = iteraciones +1;
    r = m;
    B = zeros(n, k);
    for i = 1:length(Datos(:,1))

            d = sqrt((sum(transpose(Datos(i, :) - m).^2)));

        [dis, pos] = mink(d, 1);

        B(i, pos) = 1;
    end

    for l = 1:k
        m(l, :) = sum(B(:,l).*Datos)/sum(B(:,l));
    end
    %z
    %plot(m(:, 1), m(:, 2), 'o')\
%     e = sqrt(sum((nansum(abs(m-r))/k).^2));
    e = nansum(abs(m-r))/k;
    if e < 0.001
        dentro = false;
        iteraciones
    end
end
```

*Ilustración 3. Seteo y reagrupación de centroides.*

```
b2 = reshape(B, [], 1);
C = nan(n*k, k*2);

for a = 1:k*2
% A = 1:10
% [round(A/2); ~(mod(A, 2))+1]

    newB = (Datos(logical(reshape(B(:,round(a/2)), [],1)), ~(mod(a, 2))+1)); % round(k/2) representa cada va
    newA = [newB; transpose(nan(1, n*k-length(newB)))];
    C(:, a) = newA;

end

% plot(m(:, 1), m(:, 2), 'o')
hold off

plot(C(:,1), C(:,2), '.m',C(:,3),C(:,4), '.b',C(:,5),C(:,6), '.r',C(:,7),C(:,8), '.g',C(:,9),C(:,10), '.c');
hold on
%plot(C(:,1), C(:,2), '.k')
plot(m(:, 1), m(:, 2), '.k')

% fprintf ('Centroides en:\n')
x = m(:,1);
y = m(:,2);

Centroides = table(x, y)

toc
```

*Ilustración 4. Graficado de datos.*

## R3.1 Resultados:

En la Ilustración 5se puede apreciar el dataset generado para trabajar con este método y en la Ilustración 6 se aprecia en grandes rasgos varios resultados obtenidos volviendo a correr el mismo código para varios valores iniciales de centroides, todos generados al azar.

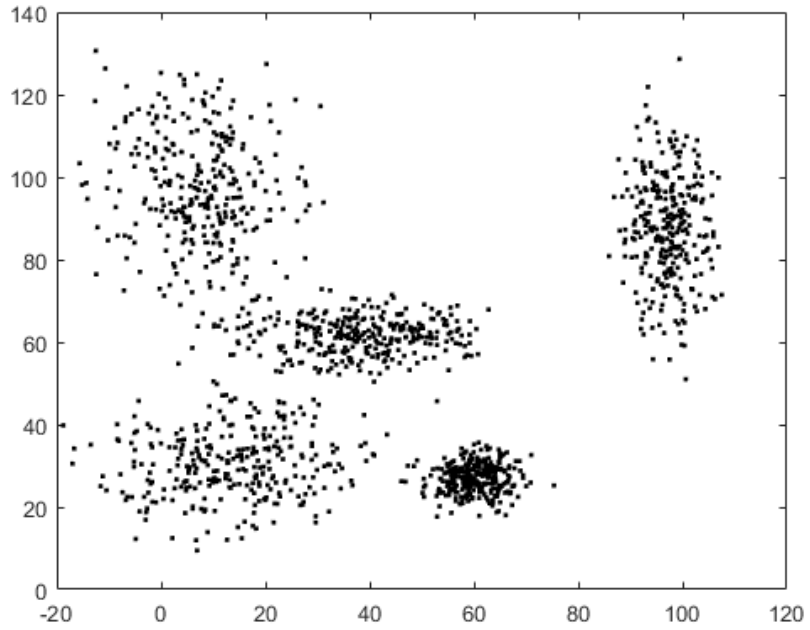


*Ilustración 5. Dataset generado.*

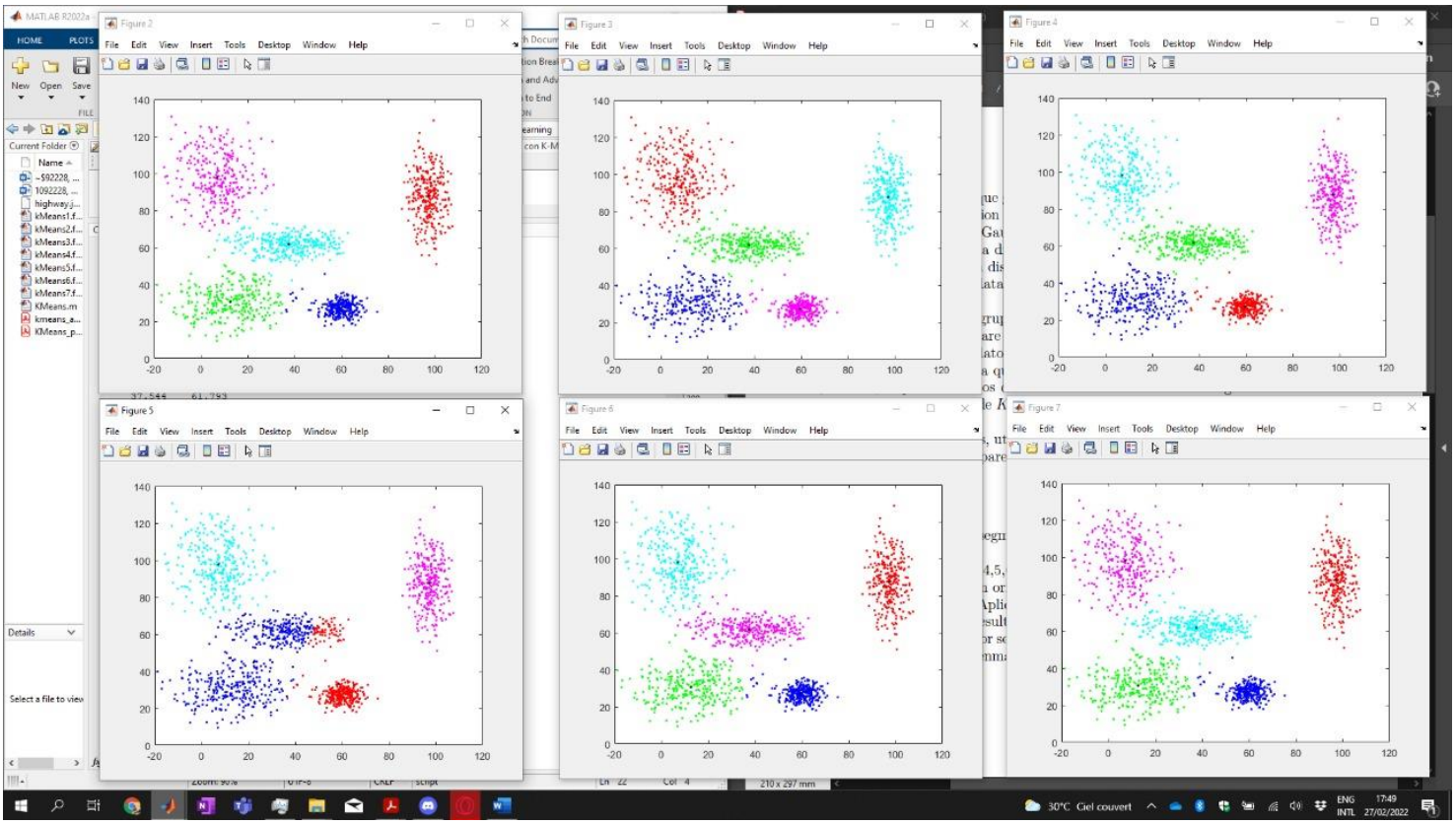

*Ilustración 6. Implementación de K-Means para diferentes valores de centroides.*

En la ilustración vista más atrás se puede apreciar que los valores de los centroides tienden a ubicarse en valores parecidos según sus posiciones iniciales, estas coordenadas y la cantidad de iteraciones que hayan tomado se presentan en Ilustración 7:

```
iteraciones =

    10


Centroides =

  5×2 table

      x        y
    _____    _____

    7.0339   98.074
    59.582   27.172
    97.236   87.587
    12.562   30.549
    37.544   61.793

Elapsed time is 0.238080 seconds.
```

```
iteraciones =

    11


Centroides =

  5×2 table

      x        y
    _____    _____

    97.236   87.587
    12.562   30.549
    59.582   27.172
    37.544   61.793
    7.0339   98.074

Elapsed time is 0.205621 seconds.
```

```
iteraciones =

     6


Centroides =

  5×2 table

      x        y
    _____    _____

    37.544   61.793
    59.582   27.172
    97.236   87.587
    12.562   30.549
    7.0339   98.074

Elapsed time is 0.165064 seconds.
```

```
iteraciones =

    10


Centroides =

  5×2 table

      x        y
    _____    _____

    59.582   27.172
    12.562   30.549
    7.0339   98.074
    37.544   61.793
    97.236   87.587

Elapsed time is 0.194205 seconds.
```

```
iteraciones =

     8


Centroides =

  5×2 table

      x        y
    _____    _____

    97.236   87.587
    22.229   44.703
    58.629    33.31
       NaN      NaN
    7.2428   97.792

Elapsed time is 0.207885 seconds.
```

```
iteraciones =

    10


Centroides =

  5×2 table

      x        y
    _____    _____

    7.0339   98.074
    59.582   27.172
    97.236   87.587
    12.562   30.549
    37.544   61.793

Elapsed time is 0.199725 seconds.
```

*Ilustración 7. Centroides y cantidad de iteraciones para K-Means*

### A.3.1 Análisis:

Se ha podido observar que los centroides en casi todas las iteraciones se presentan prácticamente en los mismos lugares, a excepción de ciertos casos en que un centroide se pierda por no ser el más cercano a ningún dato, pero al ser un caso poco probable podemos tener garantía o suficiente seguridad de que el algoritmo es suficientemente útil para reconocer la presencia de patrones y clasificarlos como tal. Se observa que este método tiene cierto comportamiento inverso a los kNN, pues este último crea patrones a partir de etiquetas y k-Means crea una etiqueta a partir de patrones.

## E3.2 Implementación en Matlab de K-Means.

En la Ilustración 8, Ilustración 9 y Ilustración 10 se puede observar el algoritmo implementado para el filtrado de la imagen, retomando el funcionamiento del algoritmo implementado en kMeans, siendo adaptado a tres dimensiones, generalizándose así a un espacio n-dimensional con k-clústers. Nótese que los datos han sido cambiados de uint8 a int32; 32 porque se llega a trabajar con valores mayores a 16 bits y con signos por los valores negativos que surgen en los cálculos en la línea 37.

```
1   clc
2   clear
3
4   I = imread('highway.jpg');
5   I = int32(I);
6
7   %size('highway.jpg')
8   Ir = I(:, :, 1);
9   Ig = I(:, :, 2);
10  Ib = I(:, :, 3);
11
12  Irmoda = modefilt(I(:, :, 1), [9,9]);
13  Igmoda = modefilt(I(:, :, 2), [9,9]);
14  Ibmoda = modefilt(I(:, :, 3), [9,9]);
15  %colormap([Ir Ig Ib])
16
17  %k = input('Introduzca la cantidad de colores: ');
18  k=6;
19  m = rand(k,3)*256;
20  m = int32(m);
21  [filas, columnas, capas] = size (I);
22  IkMeans = int32(nan(filas, columnas, capas));
23
24  IModa = int32(nan(filas, columnas, capas));
25
26  dentro = true;
27  iteraciones = 0;
28
```

*Ilustración 8. Preparación de datos.*

```
29  while dentro == true
30      iteraciones = iteraciones +1;
31  %     dentro = false;
32      B = zeros(filas, columnas, k);
33      B = int32(B);
34      r = m;
35      for i = 1:filas
36          for j = 1:columnas
37              d = sqrt(sum(transpose((reshape(I(i, j, :), 1,3) - m).^2)));
38              [dis, pos] = mink(d, 1);
39              B(i, j, pos) = 1;
40          end
41      end
42      for l = 1:k
43          m(l, :, :) = sum(B(:,:,l).*I, [1 2])/sum(B(:,:,l), [1 2]);
44      end
45      e = sqrt(sum((nansum(abs(m-r))/k).^2));
46      if e < 0.01
47          dentro = false;
48          iteraciones
49          for i = 1:filas
50              for j = 1:columnas
51                  mi = m(logical(reshape(B(i, j,:),k,1)), :);
52                  IkMeans(i,j,:) = mi;
53              end
54          end
55      end
56  end
```

*Ilustración 9. Reasignación de color según centroides.*

```
58    I = uint8(I);
59    IkMeans = uint8(IkMeans);
60    IModa(:,:,1) = Irmoda;
61    IModa(:,:,2) = Igmoda;
62    IModa(:,:,3) = Ibmoda;
63    IModa = uint8(IModa);
64
65    figure
66    imshow(I)
67    title('Imagen original')
68
69    figure
70    imshow(IkMeans)
71    title('kMeans aplicado')
72
73    figure
74    imshow(IModa)
75    title('Filtro de moda')
76
```

*Ilustración 10. Muestreo de resultado.*

## R3.2 Resultados:

En la Ilustración 11 se observa la imagen original; en la Ilustración 12 se observa el filtro de moda aplicado y en la Ilustración 13 a la Ilustración 17 se presentan los resultados aplicando kMeans para los valores de k = 3, 4, 5, 6.

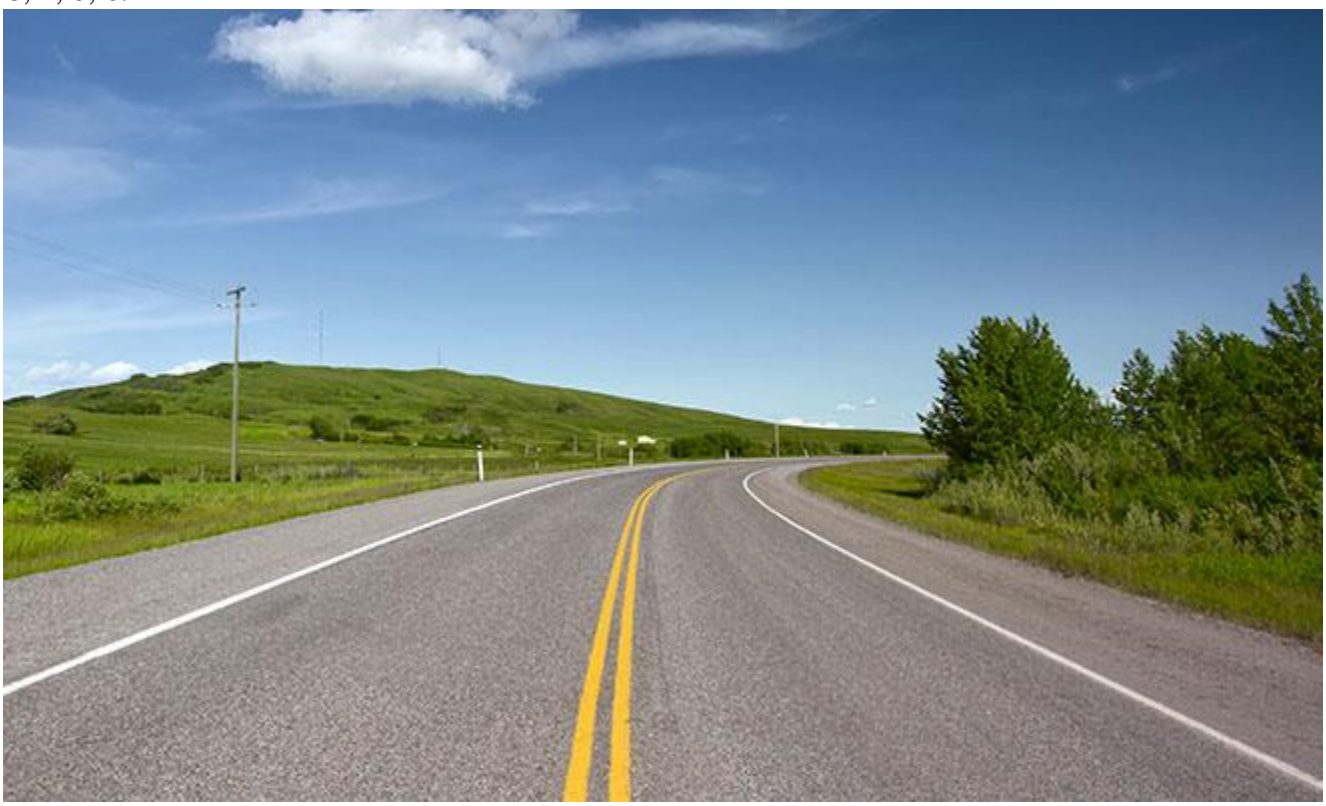

*Ilustración 11. Imagen original.*


*Ilustración 12. Filtro de moda aplicado a imagen.*


*Ilustración 13. Filtrado con* $k = 3$.


*Ilustración 14. Filtrado con $k = 4$.*

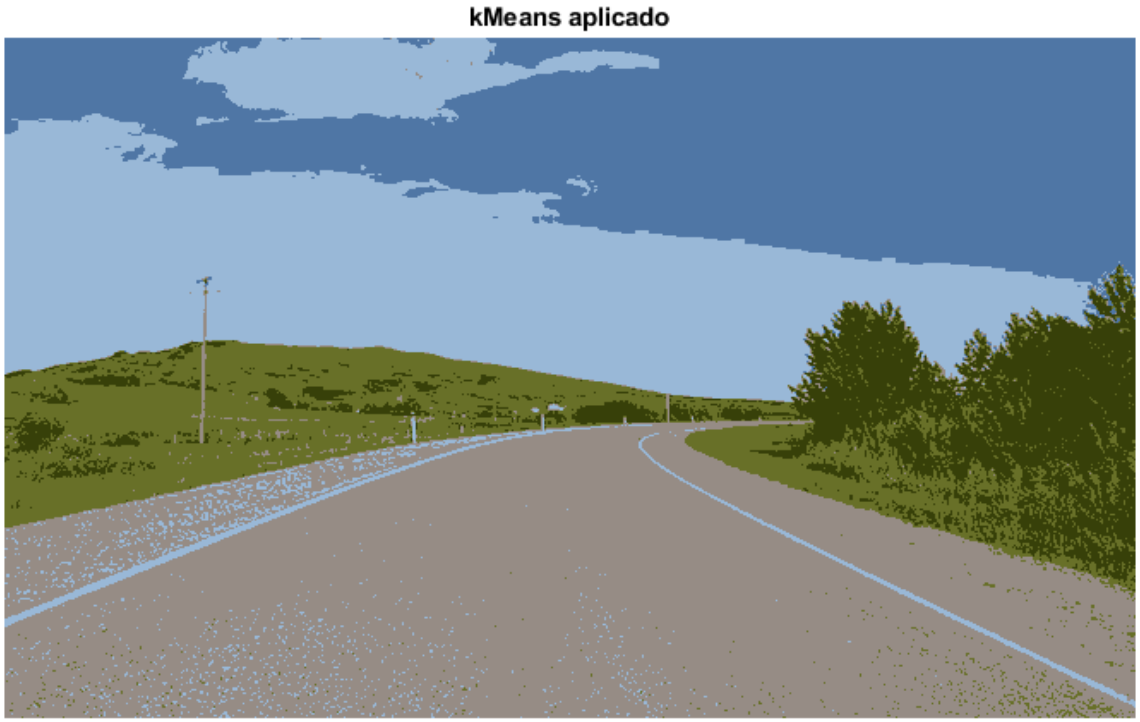


*Ilustración 15. Filtrado con $k = 5$, primera corrida.*


*Ilustración 16. Filtrado con $k = 5$, segunda corrida.*


*Ilustración 17. Filtrado con $k = 6$.*

### A.3.2 Análisis:

Con el uso de kMeans se ha podido simplificar la imagen original a varias imágenes filtradas, especialmente se ha podido observar que mientras mayor sea el valor de k, mayor es el detalle del resultado de la imagen, reconociendo asimismo que se parecería más a la imagen original y que, a la vez es menos comprimida. Finalmente, se pudo ver que el uso del filtrado por moda lo que afecta a la imagen original es en el de hacerla más borrosa.