K-Means Clustering

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

Input Libraries

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans
from sklearn.preprocessing import MinMaxScaler
```

Menginput Data

```
driver = pd.read_csv("/content/drive/MyDrive/MATERI/Pembelajaran Mesin/Praktikum Genap 20212022/go_track_tracks.csv")
driver.head()
```

	id	id_android	speed	time	distance	rating	rating_bus	rating_weather	car_or_bus	linha
0	1	0	19.210586	0.138049	2.652	3	0	0	1	NaN
1	2	0	30.848229	0.171485	5.290	3	0	0	1	NaN
2	3	1	13.560101	0.067699	0.918	3	0	0	2	NaN
3	4	1	19.766679	0.389544	7.700	3	0	0	2	NaN
4	8	0	25.807401	0.154801	3.995	2	0	0	1	NaN

Menghilangkan kolom yang tidak diperlukan

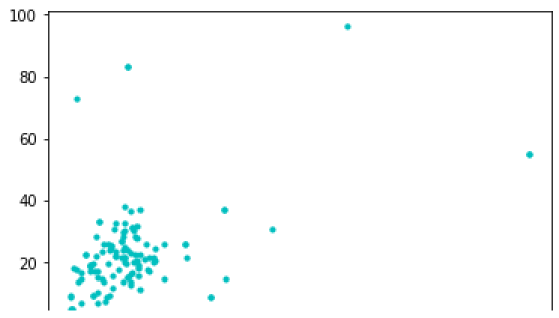
```
# Menghilangkan Kolom Yang Tidak Perlu
driver = driver.drop(["linha", "car_or_bus","rating_weather", "rating_bus","rating","time"], axis = 1)
driver.head()
```

	id	id_android	speed	distance
0	1	0	19.210586	2.652
1	2	0	30.848229	5.290
2	3	1	13.560101	0.918
3	4	1	19.766679	7.700

```
#Menentukan variabel yang akan di klusterkan ---
driver_x = driver.iloc[:, 1:3]
driver_x.head()
```

```
        id_android      speed
```

```
#Memvisualkan persebaran data ---
plt.scatter(driver.distance, driver.speed, s =10, c = "c", marker = "o", alpha = 1)
plt.show()
```



Menentukan Nilai K

```
#Mengubah Variabel Data Frame Menjadi Array ---
x_array =  np.array(driver_x)
print(x_array)
```

```
[[0.00000000e+00 1.92105856e+01]
 [0.00000000e+00 3.08482291e+01]
 [1.00000000e+00 1.35601009e+01]
 [1.00000000e+00 1.97666790e+01]
 [0.00000000e+00 2.58074009e+01]
 [2.00000000e+00 1.34691332e+00]
 [3.00000000e+00 3.68507874e+01]
 [1.00000000e+00 1.74051313e+01]
 [1.00000000e+00 1.53954361e+01]
 [1.00000000e+00 8.90272944e+00]
 [3.00000000e+00 1.50413480e+01]
 [3.00000000e+00 1.44400981e+01]
 [1.00000000e+00 1.63567325e+01]
 [1.00000000e+00 1.75427999e+01]
 [4.00000000e+00 9.45181557e+00]
 [4.00000000e+00 9.45181557e+00]
 [4.00000000e+00 1.62635039e+01]
 [4.00000000e+00 2.12235944e+01]
 [4.00000000e+00 1.94236545e+01]
 [4.00000000e+00 2.07996291e+01]
 [4.00000000e+00 8.72437242e+00]
 [4.00000000e+00 8.72437242e+00]
 [3.00000000e+00 8.68613764e+00]
 [3.00000000e+00 5.49959473e+01]
 [3.00000000e+00 5.49959473e+01]
 [1.00000000e+00 1.26110448e+01]
 [3.00000000e+00 1.45342872e+01]
 [3.00000000e+00 1.02882267e+01]
 [3.00000000e+00 1.83281891e+01]
 [1.00000000e+00 1.71776350e+01]
 [1.00000000e+00 1.70978234e+01]
 [1.00000000e+00 3.25207021e+01]
 [1.00000000e+00 1.99348099e+01]
 [1.00000000e+00 2.15138018e+01]
 [1.00000000e+00 2.75257703e+01]
 [1.00000000e+00 2.81045207e+01]
 [3.00000000e+00 2.23224625e+01]
 [3.00000000e+00 2.23224625e+01]
 [3.00000000e+00 3.23773040e+01]
 [1.00000000e+00 2.23779245e+01]
 [1.00000000e+00 2.49082559e+01]
 [5.00000000e+00 3.71409017e+01]
 [5.00000000e+00 3.71409017e+01]
 [2.00000000e+00 7.29267545e+01]
 [1.00000000e+00 6.63753526e-02]
 [1.00000000e+00 8.33281351e+01]
 [1.00000000e+00 8.33281351e+01]
 [6.00000000e+00 9.62060288e+01]
 [1.00000000e+00 3.23767476e+00]
 [1.00000000e+00 5.82100412e-01]
 [1.00000000e+00 1.38683788e+00]
 [1.00000000e+00 1.80865138e-01]
 [1.00000000e+00 1.87306515e-01]
 [1.00000000e+00 2.74035168e+00]
 [1.00000000e+00 1.55177191e+00]
 [1.00000000e+00 8.91614215e-01]
 [1.00000000e+00 1.26197446e+00]
 [7.00000000e+00 1.98334721e+01]
```

```
#Menstandarkan Ukuran Variabel ---
scaler = MinMaxScaler()
x_scaled = scaler.fit_transform(x_array)
x_scaled
```

```
array([[0.00000000e+00, 1.99600362e-01],
       [0.00000000e+00, 3.20578504e-01],
       [3.70370370e-02, 1.40861225e-01],
       [3.70370370e-02, 2.05381184e-01],
       [0.00000000e+00, 2.68176999e-01],
       [7.40740741e-02, 1.39000630e-02],
       [1.11111111e-01, 3.82977592e-01],
       [3.70370370e-02, 1.80831914e-01],
       [3.70370370e-02, 1.59940297e-01],
       [3.70370370e-02, 9.24459124e-02],
       [1.11111111e-01, 1.56259404e-01],
       [1.11111111e-01, 1.50009162e-01],
       [3.70370370e-02, 1.69933373e-01],
       [3.70370370e-02, 1.82263037e-01],
       [1.48148148e-01, 9.81538910e-02],
       [1.48148148e-01, 9.81538910e-02],
       [1.48148148e-01, 1.68964223e-01],
       [1.48148148e-01, 2.20526426e-01],
       [1.48148148e-01, 2.01815302e-01],
       [1.48148148e-01, 2.16119130e-01],
       [1.48148148e-01, 9.05918170e-02],
       [1.48148148e-01, 9.05918170e-02],
       [1.11111111e-01, 9.01943506e-02],
       [1.11111111e-01, 5.71604072e-01],
       [1.11111111e-01, 5.71604072e-01],
       [3.70370370e-02, 1.30995392e-01],
       [1.11111111e-01, 1.50988296e-01],
       [1.11111111e-01, 1.06848732e-01],
       [1.11111111e-01, 1.90427484e-01],
       [3.70370370e-02, 1.78466996e-01],
       [3.70370370e-02, 1.77637320e-01],
       [3.70370370e-02, 3.37964556e-01],
       [3.70370370e-02, 2.07128974e-01],
       [3.70370370e-02, 2.23543252e-01],
       [3.70370370e-02, 2.86040163e-01],
       [3.70370370e-02, 2.92056514e-01],
       [1.11111111e-01, 2.31949616e-01],
       [1.11111111e-01, 2.31949616e-01],
       [1.11111111e-01, 3.36473873e-01],
       [3.70370370e-02, 2.32526167e-01],
       [3.70370370e-02, 2.58830013e-01],
       [1.85185185e-01, 3.85993452e-01],
       [1.85185185e-01, 3.85993452e-01],
       [7.40740741e-02, 7.58002267e-01],
       [3.70370370e-02, 5.88338524e-04],
       [3.70370370e-02, 8.66128941e-01],
       [3.70370370e-02, 8.66128941e-01],
       [2.22222222e-01, 1.00000000e+00],
       [3.70370370e-02, 3.35553141e-02],
       [3.70370370e-02, 5.94951490e-03],
       [3.70370370e-02, 1.43150954e-02],
       [3.70370370e-02, 1.77850744e-03],
       [3.70370370e-02, 1.84546823e-03],
       [3.70370370e-02, 2.83854340e-02],
       [3.70370370e-02, 1.60296532e-02],
       [3.70370370e-02, 9.16703953e-03],
       [3.70370370e-02, 1.30170883e-02],
       [2.59259259e-01, 2.06075526e-01],
```

```
#Menentukan dan mengkonfigurasi fungsi kmeans ---
kmeans = KMeans(n_clusters = 3, random_state=123)
```

```
#Menentukan kluster dari data ---
kmeans.fit(x_scaled)
```

```
KMeans(n_clusters=3, random_state=123)
```

Memvisualisasikan Kluster

```
#Menampilkan pusat cluster ---
print(kmeans.cluster_centers_)
```

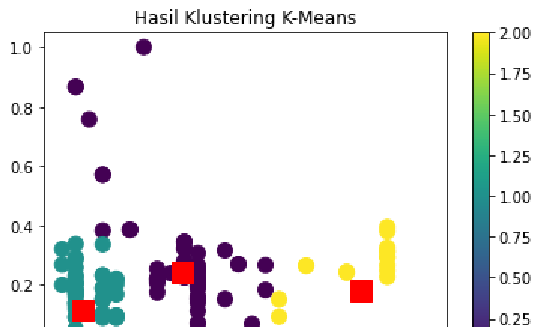
```
[[0.32745444 0.24013088]
 [0.05750487 0.11540774]
 [0.81635802 0.18289936]]
```

```
#Menampilkan Hasil Kluster ---
print(kmeans.labels_)

#Menambahkan Kolom "kluster" Dalam Data Frame Driver ---
driver["kluster"] = kmeans.labels_
```

```
[1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 0 0 0 1 0 0 0 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 1 1 0 0 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0
 0 1 1 1 1 1 1 1 1 0 1 0 0 0 0 0 1 1 0 0 0 0 0 0 1 2 2 2 2 2 1 0 1 2 2 2 2
 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2]
```

```
#Memvisualkan hasil kluster ---
output = plt.scatter(x_scaled[:,0], x_scaled[:,1], s = 100, c = driver.kluster, marker = "o", alpha = 1, )
centers = kmeans.cluster_centers_
plt.scatter(centers[:,0], centers[:,1], c='red', s=200, alpha=1 , marker="s");
plt.title("Hasil Klustering K-Means")
plt.colorbar (output)
plt.show()
```



Penerapan Pada Image Segmentation

```
!pip install opencv-python numpy matplotlib
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: opencv-python in /usr/local/lib/python3.7/dist-packages (4.1.2.30)
Requirement already satisfied: numpy in /usr/local/lib/python3.7/dist-packages (1.21.6)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.7/dist-packages (3.2.2)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.7/dist-packages (from matplotlib) (0.11.0)
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.1 in /usr/local/lib/python3.7/dist-packages (f
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.7/dist-packages (from matplotlib) (1.4.2)
Requirement already satisfied: python-dateutil>=2.1 in /usr/local/lib/python3.7/dist-packages (from matplotlib) (2.8
Requirement already satisfied: typing-extensions in /usr/local/lib/python3.7/dist-packages (from kiwisolver>=1.0.1->
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.7/dist-packages (from python-dateutil>=2.1->matplo
import cv2
import numpy as np
import matplotlib.pyplot as plt
```

Load The Image

```
# read the image
image = cv2.imread("/content/drive/MyDrive/MATERI/Pembelajaran Mesin/Praktikum Genap 20212022/image.jpg")
```

Konversi citra ke RGB

```
# convert to RGB
image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
```

Kita akan menggunakan fungsi cv2.kmeans() yang mengambil array 2D sebagai input, dan karena gambar asli kita adalah 3D (lebar, tinggi dan kedalaman 3 nilai RGB), kita perlu meratakan tinggi dan lebar menjadi satu vektor piksel (3 nilai RGB):

```
# reshape the image to a 2D array of pixels and 3 color values (RGB)
pixel_values = image.reshape((-1, 3))
# convert to float
pixel_values = np.float32(pixel_values)
```

```
print(pixel_values.shape)
```

```
    (336592, 3)
```

kode di bawah ini menentukan kriteria penghentian ini di OpenCV:

```
# define stopping criteria
criteria = (cv2.TERM_CRITERIA_EPS + cv2.TERM_CRITERIA_MAX_ITER, 100, 0.2)
```

Jika dilihat dari gambar, ada tiga warna utama (hijau untuk pepohonan, biru untuk laut/danau, dan putih untuk oranye untuk langit). Sebagai hasilnya, kita akan menggunakan tiga cluster untuk gambar ini:

jumlah cluster (K)

```
# number of clusters (K)
k = 3
_, labels, (centers) = cv2.kmeans(pixel_values, k, None, criteria, 10, cv2.KMEANS_RANDOM_CENTERS)
```

```
# convert back to 8 bit values
centers = np.uint8(centers)

# flatten the labels array
labels = labels.flatten()
```

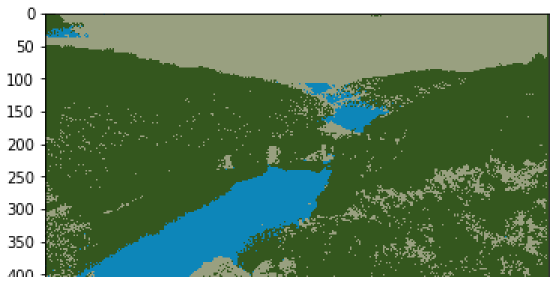
Sekarang mari kita buat gambar tersegmentasi:

```
# convert all pixels to the color of the centroids
segmented_image = centers[labels.flatten()]
```

Mengonversi kembali ke bentuk gambar asli dan menampilkannya:

```
# reshape back to the original image dimension
segmented_image = segmented_image.reshape(image.shape)
# show the image
```

```
plt.imshow(segmented_image)
plt.show()
```



Kita bisa menonaktifkan cluster. Misalnya, mari kita nonaktifkan cluster nomor 2 dan tampilkan gambar aslinya:

```
# disable only the cluster number 2 (turn the pixel into black)
masked_image = np.copy(image)
# convert to the shape of a vector of pixel values
masked_image = masked_image.reshape((-1, 3))
# color (i.e cluster) to disable
cluster = 2
masked_image[labels == cluster] = [0, 0, 0]
# convert back to original shape
masked_image = masked_image.reshape(image.shape)
# show the image
plt.imshow(masked_image)
plt.show()
```