



**LABORATORIUM PEMBELAJARAN ILMU KOMPUTER**  
**FAKULTAS ILMU KOMPUTER**  
**UNIVERSITAS BRAWIJAYA**

BAB : SELF-ORGANIZING MAPS  
NAMA : DIMAS TRI MUSTAKIM  
NIM : 205150200111049  
TANGGAL : 24/10/2022  
ASISTEN : ANDIKA IRZA PRADANA



## A. Praktikum

1. Buka Google Colaboratory melalui [tautan ini](#).
2. Tulis kode berikut ke dalam setiap *cell* pada *notebook* tersebut.
  - a. Fungsi self-organizing maps

```
import numpy as np

def som(X, lrate, b, max_epoch, n_cluster):
    centroids = np.random.uniform(size=(n_cluster, len(X[0])))
    epoch = 0

    labels = []
    while epoch < max_epoch:
        for x in X:
            d = [sum((w - x) ** 2) for w in centroids]
            min = np.argmin(d)
            centroids[min] += lrate * (x - centroids[min])
            lrate *= b
            epoch += 1

        for x in X:
            d = [sum((w - x) ** 2) for w in centroids]
            min = np.argmin(d)
            labels.append(min)

    return centroids, labels

def draw(X, target, centroids):
    colors = 'rgbcmyk'
    for x, label in zip(X, target):
        plt.plot(x[0], x[1], colors[label] + '.')
    plt.plot(centroids[:, 0], centroids[:, 1], 'kx')
```

### b. Klasterisasi

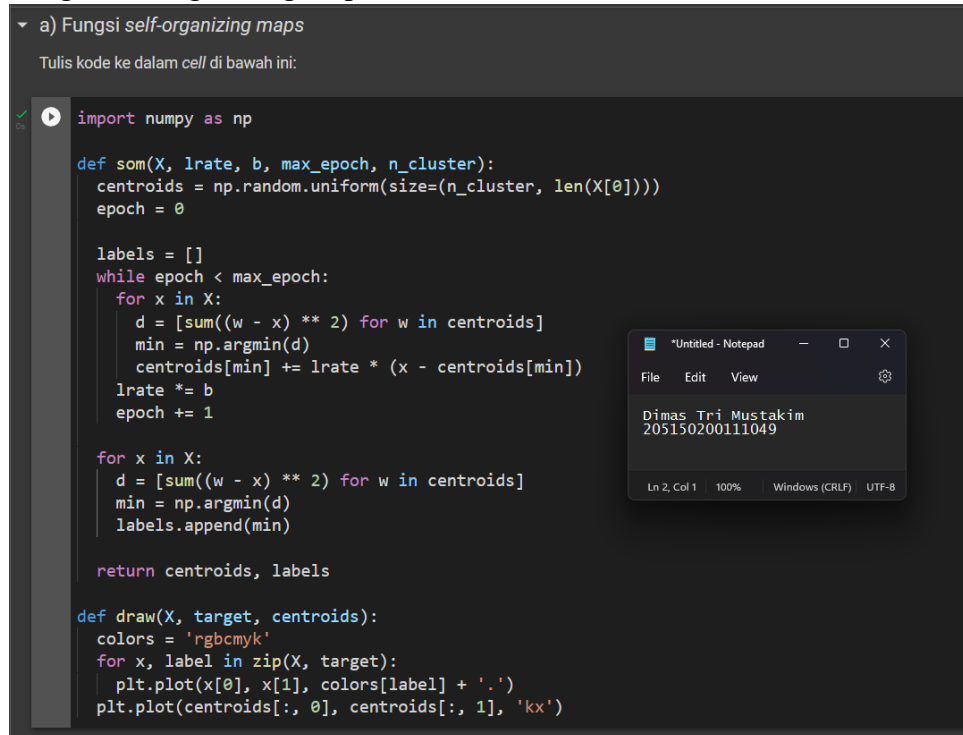
```
from sklearn.datasets import make_blobs
from sklearn.metrics import silhouette_score
import matplotlib.pyplot as plt

X, target = make_blobs(n_samples=30, n_features=2, centers=3,
                        random_state=3)
```

```
centroids, labels = som(X, lrate=.5, b=.5, max_epoch=100,
n_cluster=3)
silhouette = silhouette_score(X, labels)
print('Silhouette score:', silhouette)
draw(X, target, centroids)
```

## B. Screenshot

### a. Fungsi self-organizing maps



The screenshot shows a Jupyter Notebook interface with a dark theme. The active cell is titled "a) Fungsi self-organizing maps" and contains Python code for a Self-Organizing Map (SOM) algorithm. The code defines a function `som` that takes input data `X`, learning rate `lrate`, bias `b`, maximum epochs `max_epoch`, and number of clusters `n_cluster`. It initializes centroids randomly and iteratively updates them based on the input data points. A second function `draw` is also defined, which plots the input data points and the final centroids on a 2D plane. The code is well-formatted with syntax highlighting and includes comments in Indonesian.

```
import numpy as np

def som(X, lrate, b, max_epoch, n_cluster):
    centroids = np.random.uniform(size=(n_cluster, len(X[0])))
    epoch = 0

    labels = []
    while epoch < max_epoch:
        for x in X:
            d = [sum((w - x) ** 2) for w in centroids]
            min = np.argmin(d)
            centroids[min] += lrate * (x - centroids[min])
        lrate *= b
        epoch += 1

    for x in X:
        d = [sum((w - x) ** 2) for w in centroids]
        min = np.argmin(d)
        labels.append(min)

    return centroids, labels

def draw(X, target, centroids):
    colors = 'rgbcmyk'
    for x, label in zip(X, target):
        plt.plot(x[0], x[1], colors[label] + '.')
    plt.plot(centroids[:, 0], centroids[:, 1], 'kx')
```

### b. Klasterisasi

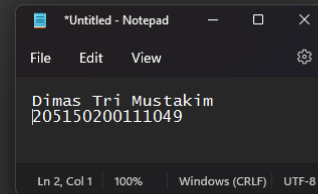
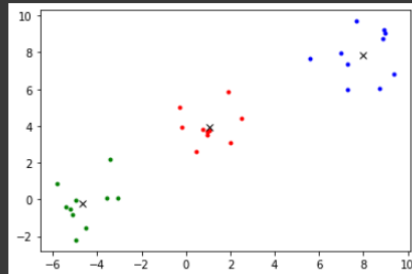
## ▼ b) Klasterisasi

Tulis kode ke dalam cell di bawah ini:

```
from sklearn.datasets import make_blobs
from sklearn.metrics import silhouette_score
import matplotlib.pyplot as plt

X, target = make_blobs(n_samples=30, n_features=2, centers=3, random_state=3)
centroids, labels = som(X, lrate=.5, b=.5, max_epoch=100, n_cluster=3)
silhouette = silhouette_score(X, labels)
print('Silhouette score:', silhouette)
draw(X, target, centroids)
```

📄 Silhouette score: 0.7215682462170806

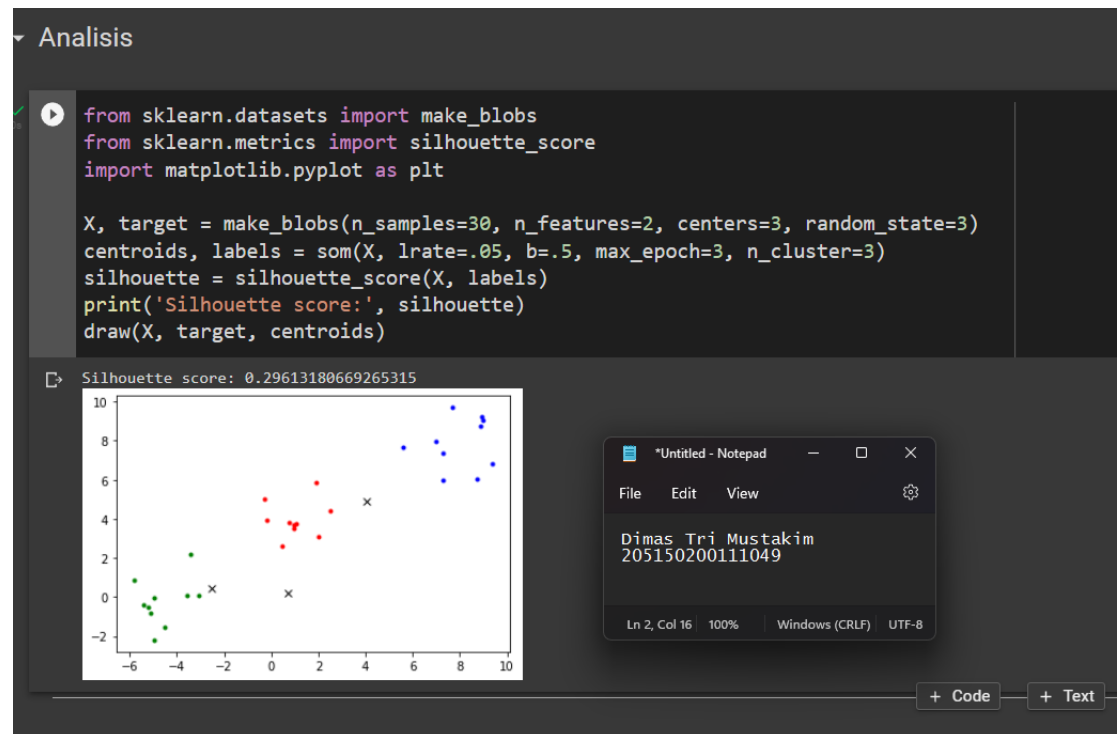


### C. Analisis

1. Ubah parameter pada kode b menjadi learning rate = 0,05 dan epoch maksimum = 3 lalu jalankan program. Amati gambar hasil klasterisasi dan nilai silhouette yang didapatkan.

**Jawab:**

Parameter Dengan mengganti parameter lr menjadi 0,05 dan epoch maksimum menjadi 3 didapatkan hasil yang lebih buruk dari sebelumnya. Nilai silhouette score menurun menjadi 0.296 dan dari gambar hasil klasterisasi dapat dilihat bahwa centroid tidak dalam posisi yang terlihat benar. Hal tersebut karena learning rate yang lebih dikecilkan sehingga pengupdatean bobot menjadi lebih sedikit dan lambat, kemudian ditambah jumlah maksimal epoch yang hanya 3 membuat algoritma tidak bisa konvergen.



#### **D. Kesimpulan**

Adaline SOM (Self Organizing Maps) adalah sebuah algoritma unsupervised learning yang dapat digunakan untuk kebutuhan klasterisasi data. Algoritma ini diusulkan oleh Kohonen pada tahun 1980an sehingga juga dikenal dengan nama Kohonen's Map. Pada algoritma ini, nilai bobot berfungsi sebagai centroid untuk setiap cluster. Centroid yang digunakan dapat diatur dalam bentuk grid dua dimensi yang dapat berbentuk persegi maupun heksagon. Banyaknya centroid juga akan ditentukan dahulu sebelum proses clustering dilakukan. Pada proses pelatihan, centroid terdekat dengan suatu data akan dipilih beserta dengan centroid pada suatu radius, kemudian akan diperbarui posisinya. Pergeseran akan diatur dengan learning rate yang akan secara perlahan diturunkan. Pada akhirnya bentuk grid dari centroid akan mewakili bentuk atau topologi dari data.

Pada kode program bagian "Fungsi self-organizing maps", tepatnya pada fungsi som, akan menerima parameter berupa data latih (X), learning rate (lrate), beta (b) atau nilai yang digunakan untuk mengupdate learning rate, max\_epoch, dan jumlah cluster (n\_cluster). Di awal dilakukan inisiasi centroid/bobot menggunakan fungsi numpy random uniform. Kemudian akan dilakukan perulangan menggunakan while loop sebanyak max\_epoch yang dimasukkan. Dalam perulangan tersebut, akan dihitung jarak (d) masing masing data input (x) dengan masing-masing centroid yang ada (w). Kemudian centroid yang memiliki jarak minimum dengan data akan diupdate. Proses perhitungan dan pengupdatean data X tersebut dilakukan di dalam for loop. Kemudian untuk setiap epoch (while loop) learning rate akan diupdate dengan dikalikan nilai beta (b) dan nilai epoch diupdate. Kemudian di akhir fungsi terdapat for loop yang akan menghitung label kelas dari setiap data dengan mencari centroid yang terdekat dan dimasukkan ke dalam List label. Di akhir nilai centroid/bobot dan label akan dikembalikan. Setelah itu juga ada fungsi draw yang akan menggambar diagram hasil dengan parameter masukan berupa data, target, dan centroid.