LAPORAN TUGAS KECIL 2

MEMBUAT PUSTAKA IMPLEMENTASI CONVEX HULL UNTUK VISUALISASI TES LINEAR SEPARABILITY DATASER DENGAN ALGORITMA DIVIDE AND CONQUER

IF2211 Strategi Algoritma

Disusun Oleh:

Ikmal Alfaozi 13520125

Program Studi Teknik Informatika

Sekolah Teknik Elektro dan Informatika

Institut Teknologi Bandung

2021/2022

Daftar Isi

A. Algoritma Divide and Conquer

a. Algoritma Convex Hull

1. Kumpulan titik diurutkan berdasarkan nilai absis yang menaik, dan jika ada nilai absis yang sama, maka diurutkan dengan nilai ordinat yang menaik.
2. Ambil titik p_1 (titik paling kiri) dan titik p_n (titik paling kanan).
3. Garis p_1 dan p_n membagi kumpulan titik menjadi dua bagian yaitu S_1 (titik di bagian kiri garis p_1p_n) dan S_2 (titik di bagian kanan garis p_1p_n). Untuk membagi kumpulan titik tersebut menjadi dua bagian, penulis menggunakan penentuan determinan berikut:

$$\begin{vmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{vmatrix}$$

   Titik (x_3, y_3) berada di sebelah kiri dari garis $\big((x_1, y_1), (x_2, y_2\big)$ jika hasil determinan positif, begitu juga sebaliknya.

   Kumpulan titik pada S_1 membentuk *convex hull* bagian atas, dan kumpulan titik pada S_2 membentuk *convex hull* bagian bawah.
4. Untuk bagian S_1, terdapat dua kemungkinan:
   i. Jika S_1 kosong, maka titik p_1 dan p_n menjadi pembentuk convex hull bagian atas.
   ii. Jika S_1 tidak kosong, pilih sebuah titik yang memiliki jarak terjauh dari garis p_1p_n, misalkan p_{max}. Jika terdapat beberapa titik terjauh yang sama, pilihlah titik dengan dengan absis terkecil.
   Semua titik yang berada di dalam segitiga $p_1p_{max}p_n$ diabaikan.
5. Tentukan kumpulan titik yang berada di kiri garis p_1p_{max}, misalkan $S_{1,1}$, dan di kiri $p_{max}p_n$, misalkan $S_{1,2}$.
6. Ulangi langkah 4 dan 5 untuk kumpulan titik pada $S_{1,1}$ dan $S_{1,2}$ hingga tidak ada titik terluar lagi.
7. Lakukan hal yang sama (langkah 4, 5, dan 6) untuk bagian S_2.
8. Gabungkan pasangan titik yang dihasilkan.

B. Kode Program

a. Modul myConvexHull.py

Modul ini berisi implementasi algoritma convexHull dengan *divide and conquer*.

```
from asyncio.windows_events import NULL
import numpy as np

def convexHull(arrayOfPoint):
    # Mengurutkan point berdasarkan absis kemudian ordinat secara menaik
    arrayOfPoint = np.array(sorted(arrayOfPoint, key=lambda k: [k[0], k[1]]))

    # Mengambil P1 dan Pn
    p1 = arrayOfPoint[0]
    pn = arrayOfPoint[-1]

    # Divide: Membagi titik ke sebelah kiri (s1) dan kanan (s2) berdasarkan garis P1-Pn
    arrayOfPointLeft = np.array(left(arrayOfPoint,p1, pn))
    arrayOfPointRight = np.array(left(arrayOfPoint, pn, p1))

    # Divide and Conquer: Memproses titik-titik di s1 dan di s2
    leftVertices = np.array(process(arrayOfPointLeft,p1,pn))
    rightVertices = np.array(process(arrayOfPointRight,pn,p1))

    # Merge
    vertices = np.array([p1])
    if (len(leftVertices) != 0):
        vertices = np.concatenate((vertices, leftVertices), axis=0)
    vertices = np.append(vertices, [pn], axis=0)
    if (len(rightVertices) != 0):
        vertices = np.concatenate((vertices, rightVertices), axis=0)
    vertices = np.append(vertices, [p1], axis=0)

    return np.array(vertices)

def left(array,p1,p2):
    arrayOfPointLeft = []
    for i in range(len(array)):
     # x1y2 + x3y1 + x2y3 - x3y2- x2y1 - x1y3 = (y3-y1)(x2-x1) – (y2-y1)(x3-x1)
        if ((array[i][1] - p1[1]) * (p2[0] - p1[0]) - (p2[1] - p1[1]) * (array[i][0] - p1[0])) > 0:
            arrayOfPointLeft.append(array[i])
    return arrayOfPointLeft

def getFarthestPoint(array,p1,pn):
    # Mengambil titik dengan jarak terjauh
    if len(array) == 0:
        return NULL
```

```
    else:
        m = (pn[1]-p1[1])/(pn[0]-p1[0])
        # y - y1 = m(x-x1) <--> y - p1[1] = m*(x-p1[0]) <--> y - p1[1] = m*x - m*p1[0] <--> m*x - y + (p1[1] - m*p1[0]) = 0
        # ax + by + c = 0
        arraydistance = []
        for p in array:
            arraydistance.append(abs(m*p[0] - p[1] + p1[1] - m*p1[0]))
        maxDistance = max(arraydistance)
        arrayIdxMaxDistance = [x for x, y in enumerate(arraydistance) if y == maxDistance]
        return array[arrayIdxMaxDistance[0]]

def process(array,p1,pn):
    vertices = []
    farthestPoint = getFarthestPoint(array,p1,pn)
    if (farthestPoint is not NULL):
        # print(farthestPoint)
        vertices.append(farthestPoint)
        array1 = np.array(left(array,p1,farthestPoint))
        array2 = np.array(left(array,farthestPoint,pn))
        # print(array1,array2)
        vertices1 = process(array1,p1,farthestPoint)
        vertices2 = process(array2,farthestPoint,pn)
        vertices = vertices1 + vertices + vertices2
    return vertices
```

b. File main.ipynb

File ini berisi contoh penggunaan modul convexHull.py.

i. Memanfaatkan library nump untuk membuat data dumy

```
import matplotlib.pyplot as plt
from myConvexHull import convexHull
import numpy as np

plt.figure(figsize = (11, 6))
plt.xlabel('Data X')
plt.ylabel('Data Y')

array = np.random.rand(100,2)
array *= 100
plt.scatter(array[:,0],array[:,1])
vertices = convexHull(array)
plt.plot(vertices[:,0], vertices[:,1])

array = np.random.rand(100,2)
array *= 100
plt.scatter(array[:,0],array[:,1])
vertices = convexHull(array)
plt.plot(vertices[:,0], vertices[:,1])
```

```
array = np.random.rand(100,2)
array *= 100
plt.scatter(array[:,0],array[:,1])
vertices = convexHull(array)
plt.plot(vertices[:,0], vertices[:,1])

plt.show()
```

ii. Dataset Iris : Sepal Width vs Sepal Length

```
from sklearn import datasets
import matplotlib.pyplot as plt
import pandas as pd
from myConvexHull import convexHull
data = datasets.load_iris()
#create a DataFrame
df = pd.DataFrame(data.data, columns=data.feature_names)
df['Target'] = pd.DataFrame(data.target)
plt.figure(figsize = (10, 6))
colors = ['b','r','g']
plt.title('Sepal Width vs Sepal Length')
plt.xlabel(data.feature_names[0])
plt.ylabel(data.feature_names[1])
for i in range(len(data.target_names)):
    bucket = df[df['Target'] == i]
    bucket = bucket.iloc[:,[0,1]].values
    hull = convexHull(bucket)
    plt.scatter(bucket[:, 0], bucket[:, 1], label=data.target_names[i])
    plt.plot(hull[:,0], hull[:,1], colors[i])
plt.legend()
plt.show()
```

iii. Dataset Iris: Petal Width vs Petal Length

```
from sklearn import datasets
import matplotlib.pyplot as plt
import pandas as pd
from myConvexHull import convexHull
data = datasets.load_iris()
#create a DataFrame
df = pd.DataFrame(data.data, columns=data.feature_names)
df['Target'] = pd.DataFrame(data.target)
plt.figure(figsize = (10, 6))
colors = ['b','r','g']
plt.title('Petal Width vs Petal Length')
plt.xlabel(data.feature_names[2])
plt.ylabel(data.feature_names[3])
for i in range(len(data.target_names)):
    bucket = df[df['Target'] == i]
    bucket = bucket.iloc[:,[2,3]].values
```

```
    hull = convexHull(bucket)
    plt.scatter(bucket[:, 0], bucket[:, 1], label=data.target_names[i])
    plt.plot(hull[:,0], hull[:,1], colors[i])
plt.legend()
plt.show()
```

iv. Dataset Wine : Alcohol vs Malic acid

```
from sklearn import datasets
import matplotlib.pyplot as plt
import pandas as pd
from myConvexHull import convexHull
data = datasets.load_wine()
#create a DataFrame
df = pd.DataFrame(data.data, columns=data.feature_names)
df['Target'] = pd.DataFrame(data.target)
plt.figure(figsize = (10, 6))
colors = ['b','r','g']
plt.title('Alcohol vs Malic acid')
plt.xlabel(data.feature_names[0])
plt.ylabel(data.feature_names[1])
for i in range(len(data.target_names)):
    bucket = df[df['Target'] == i]
    bucket = bucket.iloc[:,[0,1]].values
    hull = convexHull(bucket)
    plt.scatter(bucket[:, 0], bucket[:, 1], label=data.target_names[i])
    plt.plot(hull[:,0], hull[:,1])
plt.legend()
plt.show()
```

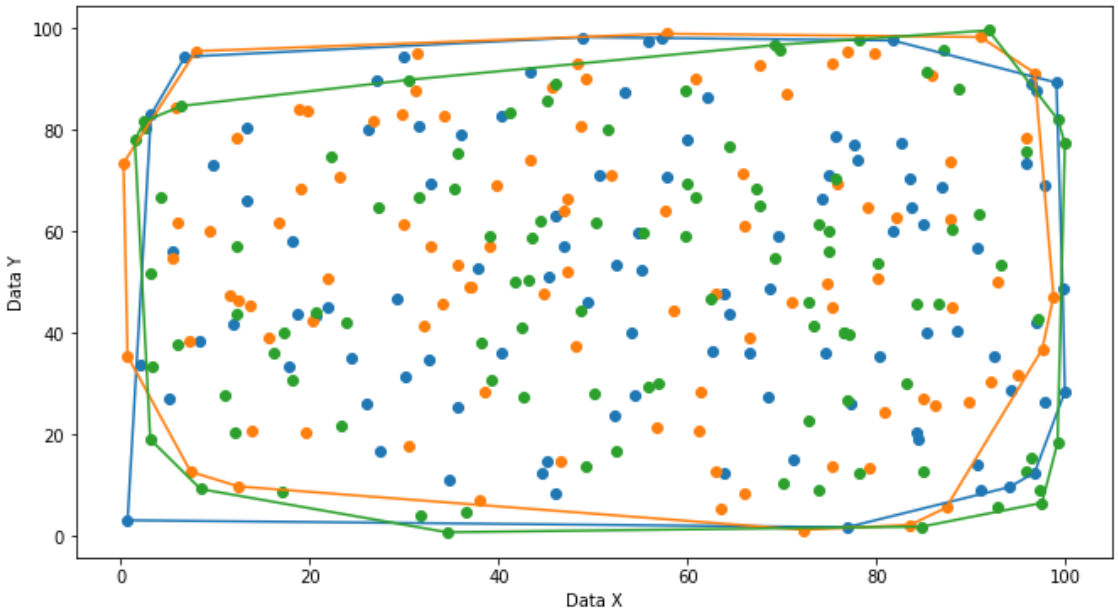
v. Dataset Breast cancer wisconsin (diagnostic) : Radius (Mean) vs Texture (Mean)

```
from sklearn import datasets
import matplotlib.pyplot as plt
import pandas as pd
from myConvexHull import convexHull
data = datasets.load_breast_cancer()
#create a DataFrame
df = pd.DataFrame(data.data, columns=data.feature_names)
df['Target'] = pd.DataFrame(data.target)
plt.figure(figsize = (10, 6))
colors = ['b','r']
plt.title('Radius (Mean) vs Texture (Mean)')
plt.xlabel(data.feature_names[0])
plt.ylabel(data.feature_names[1])
for i in range(len(data.target_names)):
    bucket = df[df['Target'] == i]
    bucket = bucket.iloc[:,[0,1]].values
    hull = convexHull(bucket)
    plt.scatter(bucket[:, 0], bucket[:, 1], label=data.target_names[i])
```

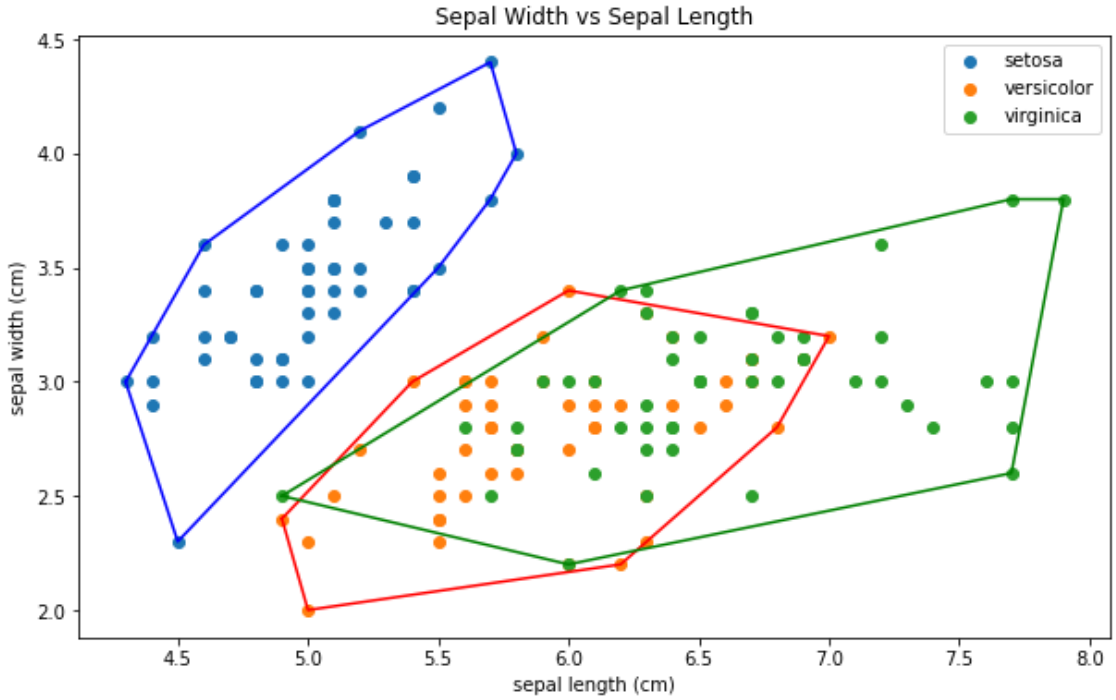
```
    plt.plot(hull[:,0], hull[:,1])
plt.legend()
plt.show()
```

C. Screenshot Input dan Output Program

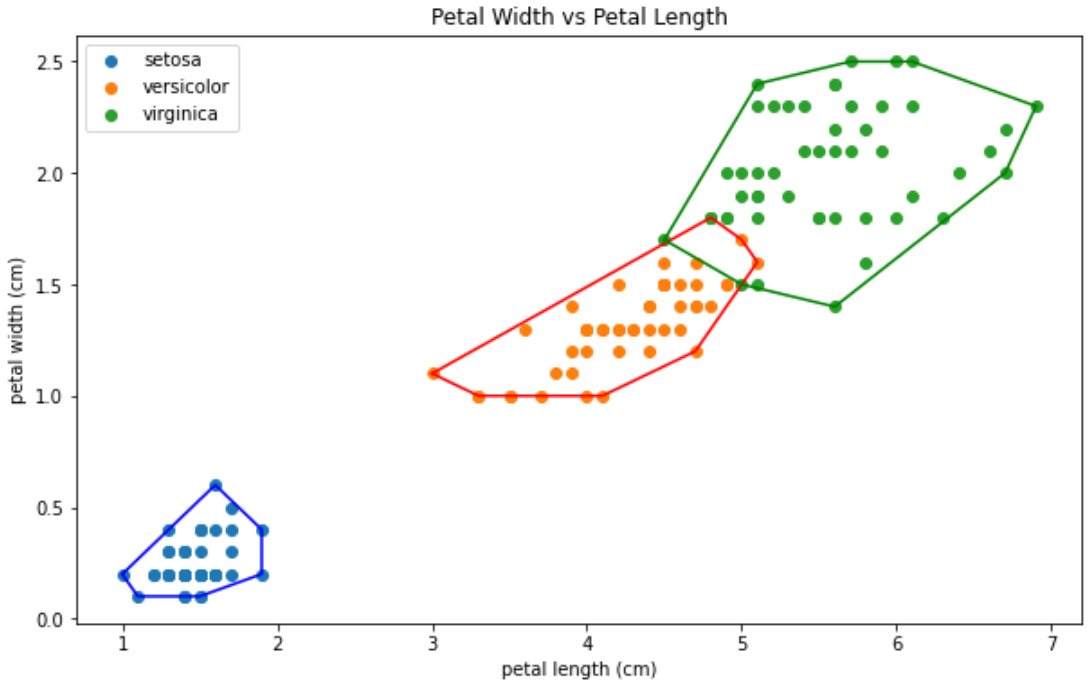
a. Memanfaatkan library nump untuk membuat data dumy



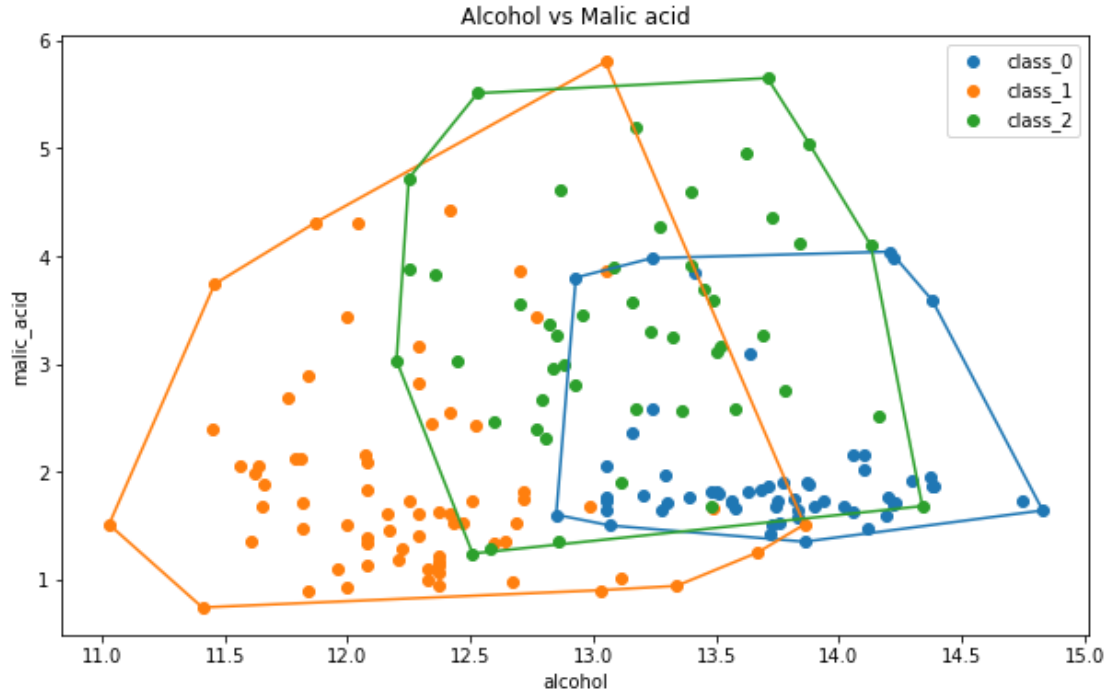
b. Dataset Iris : Sepal Width vs Sepal Length



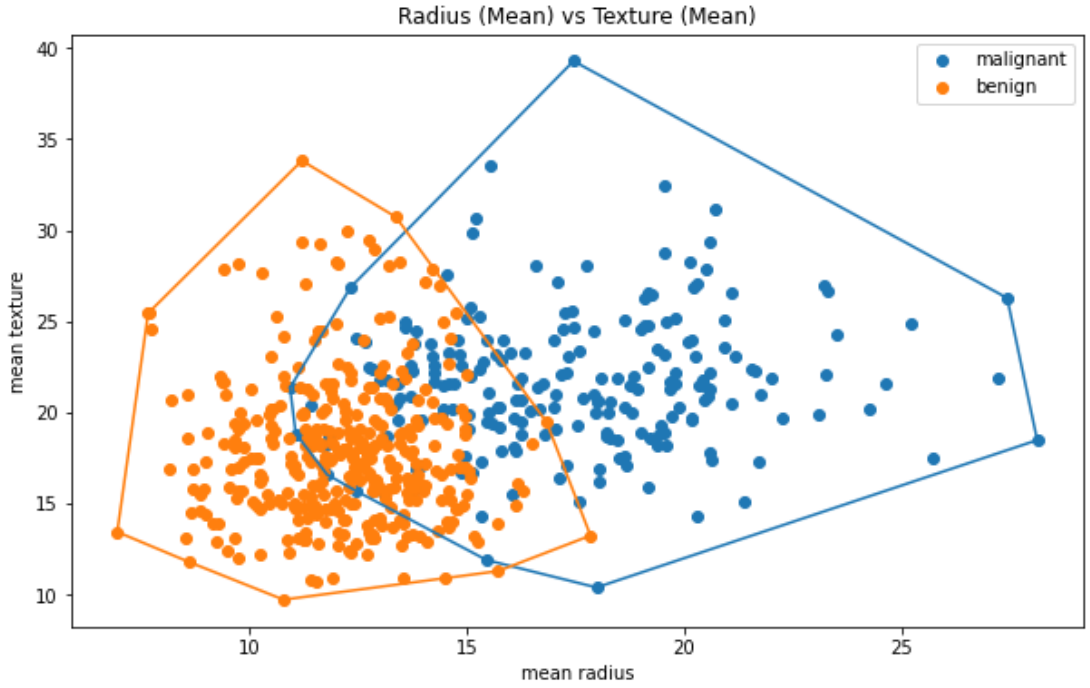
c. Dataset Iris: Petal Width vs Petal Length



d. Dataset Wine : Alcohol vs Malic acid



e. Dataset Breast cancer wisconsin (diagnostic) : Radius (Mean) vs Texture (Mean)


Radius (Mean) vs Texture (Mean)
malignant
benign
mean texture
mean radius
40
35
30
25
20
15
10
10
15
20
25

D. Alamat Drive Kode Program

IF2211-Strategi-Algoritma/Tucil2 at main · ikmalalfaozi/IF2211-Strategi-Algoritma (github.com)

E. Cek List

Poin	Ya	Tidak
1. Pustaka myConvexHull berhasil dibuat dan tidak ada kesalahan	✓	
2. Convex hull yang dihasilkan sudah benar	✓	
3. Pustaka myConvexHull dapat digunakan untuk menampilkan convex hull setiap label dengan warna yang berbeda.	✓	
4. Bonus: program dapat menerima input dan menuliskan output untuk dataset lainnya.	✓	