LAPORAN TUGAS KECIL 2 IF2211

Disusun Oleh:

Raden Haryosatyo Wisjnunandono

13520070

Program Studi Teknik Informatika

Sekolah Teknik Elektro dan Informatika

Institut Teknologi Bandung

2021/2022

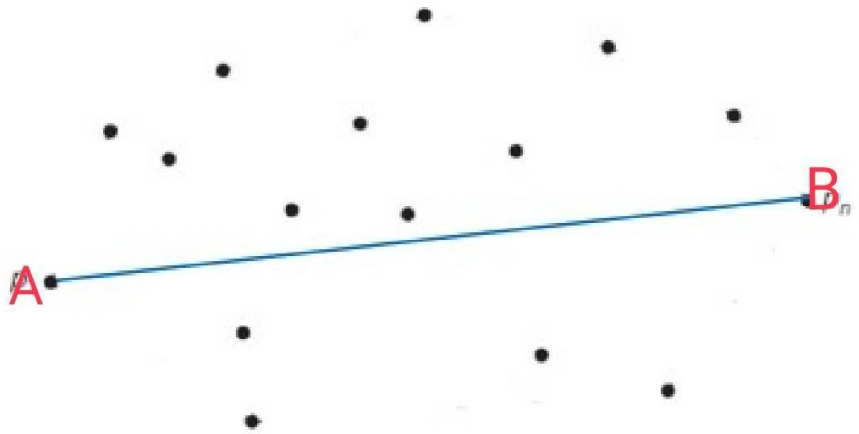
BAB I

ALGORITMA DIVIDE AND CONQUER DALAM PERSOALAN CONVEX HULL

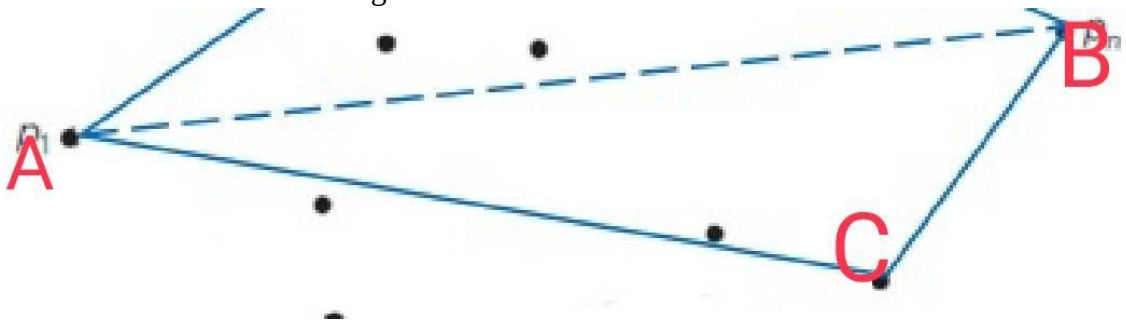
1.1 Algoritma Divide and Conquer

Dalam persoalan convex hull algoritma divide and coquer yang digunakan dalam program ini secara garis besar adalah sebagai berikut

1. Urutkan itik-titik yang ingin dicari convex hullnya menaik berdasarkan nilai absis dan jika ada yang nilai abisisnya sama maka diurutkan ordinatnya.
2. Pilih titik yang absis dan ordinatnya terkecil sebagai titik A titik yang absis dan ordinatnya terbesar sebagai titik B.
3. Tarik garis dari titik A ke B. Garis yang ditarik dari titik A dan B ini kita sebut dengan garis AB.



4. Masukkan titik A dan B kedalam himpunan solusi (titik-titik yang membentuk convex hull).*
5. Kelompokkan titik-titik yang tersisa menjadi dua bagian, titik-titik yang terletak di kiri garis AB dan di kanan garis AB dengan menggunakan determinan (jika determinan < 0 maka terletak dikanan dan sebaliknya).
6. Cari titik terjauh dari garis AB pada himpunan titik-titik yang terletak di kanan garis AB, disini kita sebut titik ini sebagai titik C.



7. Masukkan titik C kedalam himpunan solusi .*
8. Tarik garis dari titik A ke C, kita sebut garis ini sebagai garis AC. Tarik garis juga dari titik C ke B, kita sebut garis ini sebagai garis CB.
9. Garis AC dan CB membagi himpunan titik-titik yang berada di kanan garis ini menjadi 3 wilayah, kita sebut wilayah-wlayah ini dengan X0, X1, dan X2. X1 adalah wilayah di sebelah kiri dari garis AC dan X2 adalah wilayah di sebelah kanan garis CB.
10. Ulangi langkah pada butir 6 sampai 9 sampai tidak ada lagi titik yang berada pada kiri garis AC dan kanan garis CB.
11. Ulangi langkah 6 sampai 10 untuk wilayah bagian atas garis AB.

**) note: Untuk prosedur memasukkan titik kedalam himpunan solusi sebenarnya tidak persis seperti yang disampaikan disini. Prosedur tersebut akan dijelaskan pada subbab selanjutnya.*

1.2 Algoritma Penyusunan Himpunan Titik-Titik Solusi

Algoritma penyusunan ini diperlukan supaya visualisasi convex hull yang dihasilkan sesuai. Hal ini karena pada program ini digunakan library matplotlib untuk visualisasi dan urutan titik dalam himpunan solusi berpengaruh saat plotting menggunakan library matplotlib. Algoritma penyusunannya adalah sebagai berikut.

1. Inisialisasi array himpunan titik-titik solusi global, bagian kanan, dan bagian kiri.
2. Masukkan titik A (titik paling kecil absis dan ordinatnya) kedalam himpunan solusi global.
3. Cari titik-titik pembentuk convex hull pada bagian kiri garis AB dengan algoritma yang telah dijelaskan pada subbab 1.1. Masukkan titik-titik solusi ini pada array himpunan solusi bagian kiri.
4. Urutkan array himpunan solusi bagian kiri berdasarkan absisnya secara menaik.
5. Masukkan array himpunan solusi bagian kiri ke dalam himpunan solusi global.
6. Masukkan titik B (titik paling besar absis dan ordinatnya) kedalam himpunan solusi global.
7. Cari titik-titik pembentuk convex hull pada bagian kiri kanan AB dengan algoritma yang telah dijelaskan pada subbab 1.1. Masukkan titik-titik solusi ini pada array himpunan solusi bagian kanan.
8. Urutkan array himpunan solusi bagian kanan berdasarkan absisnya secara menurun
9. Masukkan array himpunan solusi bagian kanan ke dalam himpunan solusi global.
10. Masukkan titik A (titik paling kecil absis dan ordinatnya) kedalam himpunan solusi global.

BAB II

SOURCE CODE

2.1 Import Modul yang Diperlukan

```
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import matplotlib.pyplot as plt
from sklearn import datasets
```

2.2 Fungsi Convex Hull Utama

```
def myConvexHull(points):
    solutionleft = []
    solutionright = []
    solution = []
    sortedPoints = sorted(points, key=lambda row: (row[0], row[1]))
    leftpoints = []
    rightpoints = []
    a = sortedPoints[0]
    b = sortedPoints[-1]

    for i in range(len(sortedPoints)):
        det = determinan(sortedPoints[i], a, b)
        if (det > 0):
            leftpoints.append(sortedPoints[i])
        elif (det < 0):
            rightpoints.append(sortedPoints[i])

    #titik paling kiri sebagai solusi
    solution.append(a)
    convex_hull(leftpoints, b, a, solutionleft) #divide and conquer bagian kiri dari garis utama
    solutionleft.sort(key=lambda row: (row[0], row[1])) #susun titik-titik pembentuk convexhull atas untuk memudahkan
    visualisasi
    solution += solutionleft #masukkan titik2 solusi bagian atas ke himpunan solusi utama

    solution.append(b)

    convex_hull(rightpoints, a, b, solutionright)
    solutionright.sort(key=lambda row: (row[0], row[1]), reverse=True)#susun titik-titik pembentuk convexhull bawah untuk
    memudahkan visualisasi
    solution += solutionright #masukkan titik2 solusi bagian atas ke himpunan solusi utama
    solution.append(a) #append titik awal lagi supaya memudahkan visualisasi

    return solution
```

2.3 Fungsi mencari determinan dan titik terjauh

```python
def determinan(p3,p1, p2):
    #mencari determinan untuk menentukan apakah p3 berada di atas garis atau di bawah garis
    x1, y1 = p1
    x2, y2 = p2
    #ax + by + c
    x3, y3 = p3
    return (x2*y3-x3*y2)-(x1*y3-x3*y1)+(x1*y2-x2*y1)



def get_farthest_point(area, p1, p2):
    #mencari titik terjauh dari di area "area" dari garis ab
    farthest_dist = -1
    C = None
    for p3 in area:
        # print(p3)
        f = abs((p2[0]-p1[0])*(p1[1]-p3[1])-(p1[0]-p3[0])*(p2[1]-p1[1]))
        # print(f)
        if f>farthest_dist:
            farthest_dist = f
            C = p3
    # print(C)
    # print()
    return C
```

2.4 Fungsi Convex Hull rekursif divide and conquer

```python
#fungsi convexhull rekursif
def convex_hull(area, a , b, solution):

    #jika tidak ada titik yang merupakan area maka
    if len(area) == 0:
        return
    else:
        #cari titik terjauh dari garis ab

        c = get_farthest_point(area, a, b)
        # print(c)
        # print(determinan(c, a, b))

        #masukkan titik c ke himpunan solusi

        solution.append(c)


        #divide masalah dengan garis ac dan cb
        ac_right = []
        cb_right = []
        for point in area:
            if determinan(point, a, c) < 0:
                ac_right.append(point)
            if determinan(point, c, b) < 0:
                cb_right.append(point)

        #Proses rekursif
        convex_hull(ac_right, a, c, solution)
        convex_hull(cb_right, c, b, solution)
```

2.5 Untuk dataset Iris

```
data = datasets.load_iris()
#create a DataFrame
df = pd.DataFrame(data.data, columns=data.feature_names)
df['Target'] = pd.DataFrame(data.target)
print(df.shape)
df.head()
plt.figure(figsize = (10, 6))
colors = ['b','r','g', 'c', 'm', 'y']
plt.title('Petal Width vs Petal 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
    sol = myConvexHull(bucket)

    plt.scatter(bucket[:, 0], bucket[:, 1], label=data.target_names[i])
    xpoint = []
    ypoint = []
    for points in sol:
        xpoint.append(points[0])
        ypoint.append(points[1])

    plt.plot(xpoint, ypoint, colors[i])
plt.legend()
```

2.6 Untuk dataset Wine

```
data = datasets.load_wine()
#create a DataFrame
df = pd.DataFrame(data.data, columns=data.feature_names)
df['Target'] = pd.DataFrame(data.target)
print(df.shape)
df.head()
plt.figure(figsize = (10, 6))
colors = ['b','r','g', 'c', 'm', 'y']
plt.title('Dataset Wine')
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
    sol = myConvexHull(bucket)

    plt.scatter(bucket[:, 0], bucket[:, 1], label=data.target_names[i])
    xpoint = []
    ypoint = []
    for points in sol:
        xpoint.append(points[0])
        ypoint.append(points[1])

    plt.plot(xpoint, ypoint, colors[i])
plt.legend()
```

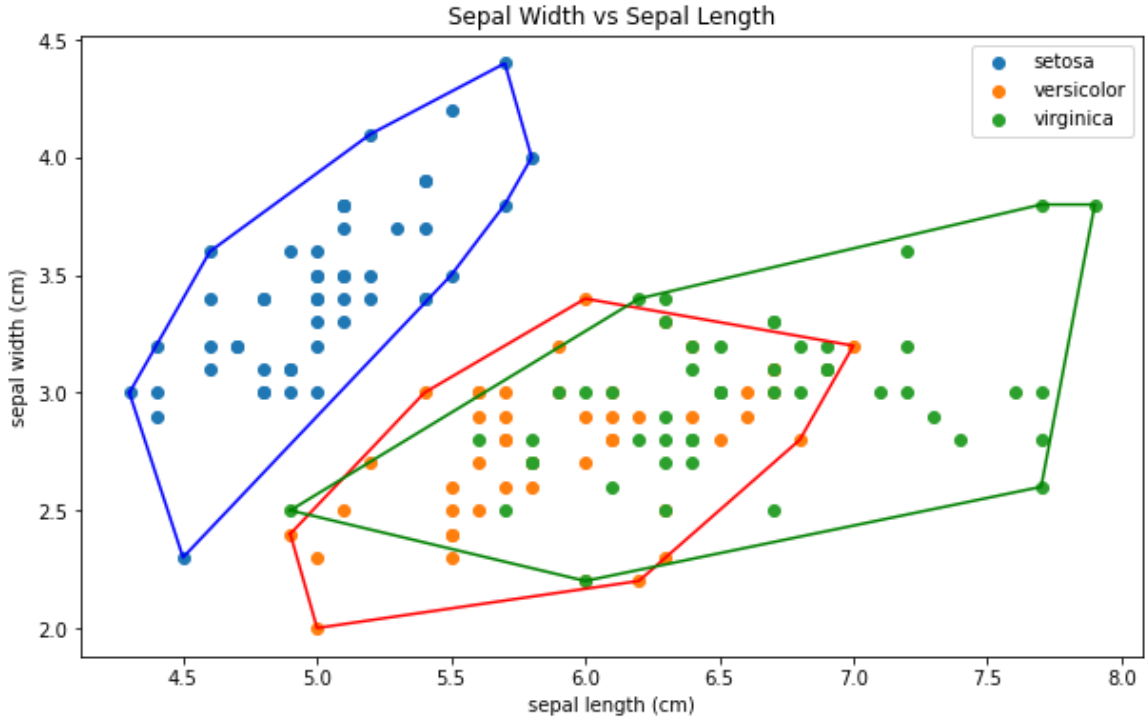
2.7 Untuk dataset Breast Cancer

```
data = datasets.load_breast_cancer()
#create a DataFrame
df = pd.DataFrame(data.data, columns=data.feature_names)
df['Target'] = pd.DataFrame(data.target)
print(df.shape)
df.head()
plt.figure(figsize = (10, 6))
colors = ['b','r','g', 'c', 'm', 'y']
plt.title('Dataset Breast Cancer')
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
    sol = myConvexHull(bucket)
    print(np.array(sol))
    print()

    plt.scatter(bucket[:, 0], bucket[:, 1], label=data.target_names[i])
    xpoint = []
    ypoint = []
    for points in sol:
        xpoint.append(points[0])
        ypoint.append(points[1])

    plt.plot(xpoint, ypoint, colors[i])
plt.legend()
```

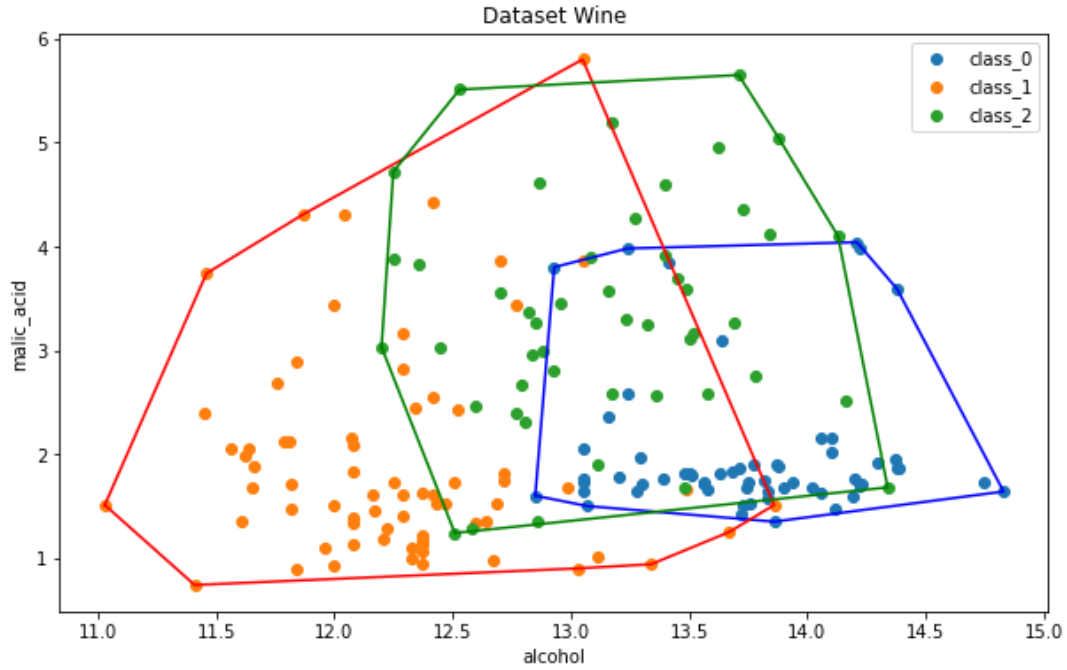
BAB III

INPUT DAN OUTPUT

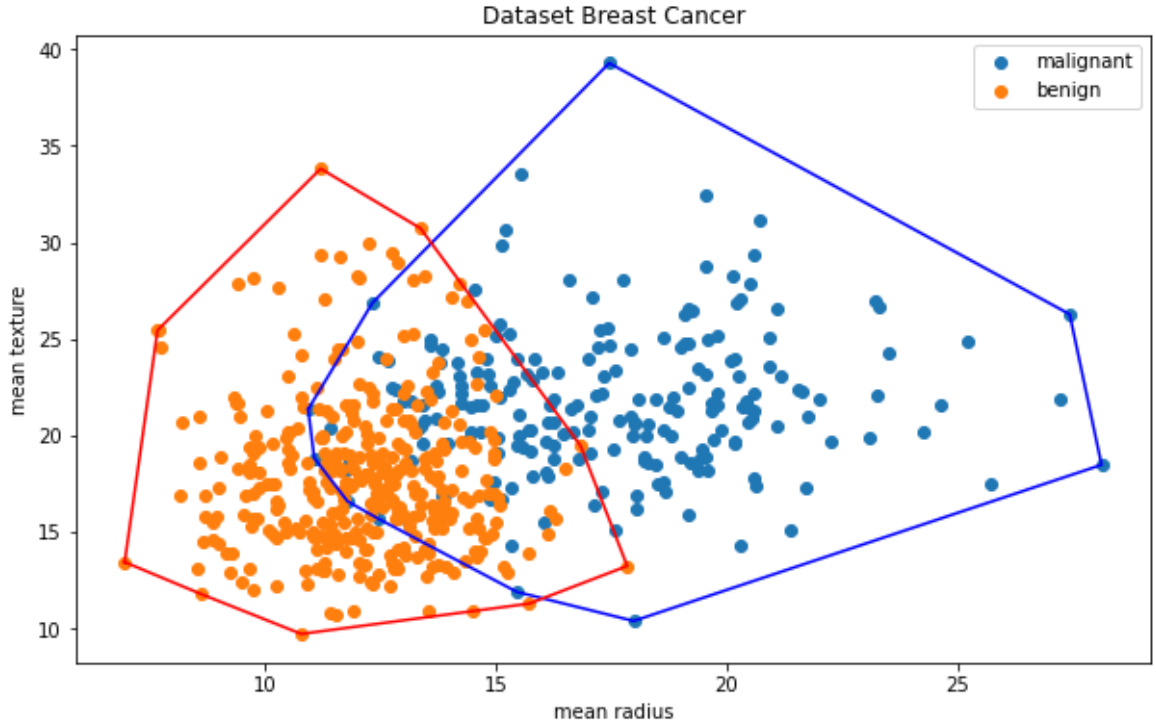
3.1 Dataset Iris (Perbadingan Sepal)



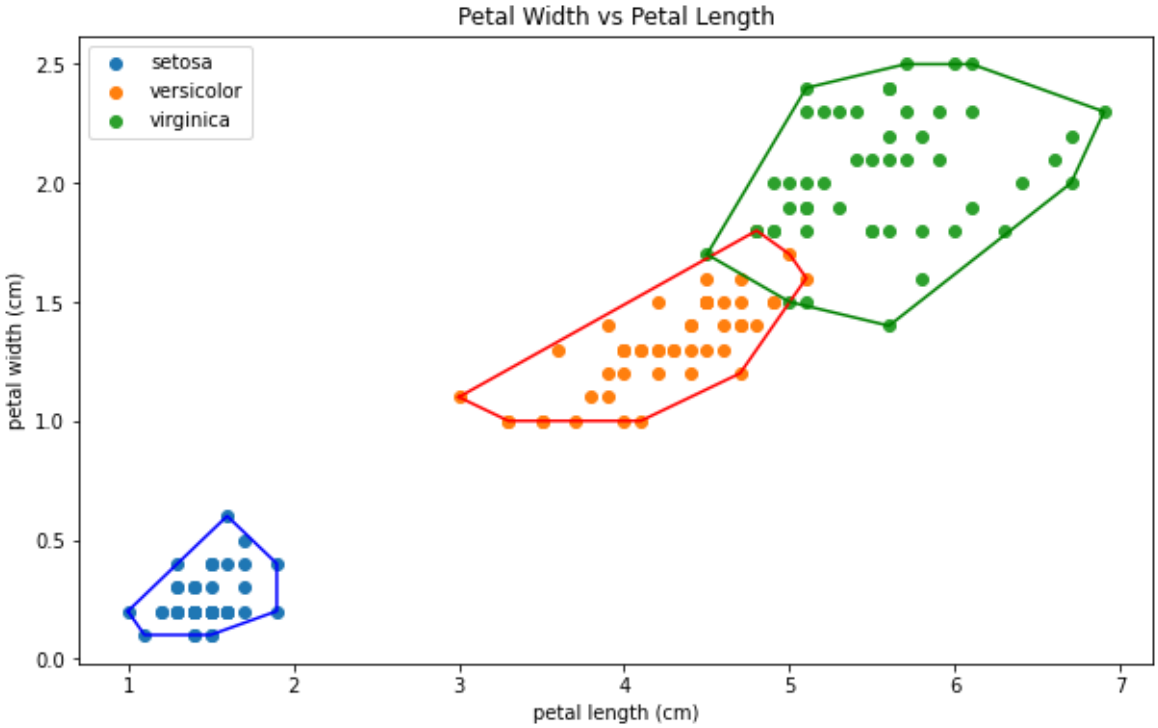
3.2 Dataset Wine (alcohol vs malic acid)



3.3 Dataset Breast Cancer (mean texture vs mean radius)



3.4 Dataset Iris (Petal)



BAB IV

KESIMPULAN

Permasalahan mencari convex hull dapat diselesaikan dengan menggunakan algoritma divide and conquer. Mula-mula tarik garis pembagi awal. Garis pembagi awal adalah garis yang menghubungkan titik-titik ekstrem pada himpunan titik tersebut. Lalu bagi dua berdasarkan titik-titik yang tersisa berdasarkan posisinya terhadap garis pembagi awal. Selanjutnya cari titik terjauh dari garis pembagi awal dan masukkan titik tersebut ke himpunan solusi. Kemudian divide wilayah yang sudah ada dengan menarik garis dari titik-titk ekstrem ke titik terjauh dari garis tersebut. Ulang dua langkah sebelumnya sampai titik-titik terluar sudah masuk ke himpunan solusi semua.

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..		

Link drive:

https://drive.google.com/drive/folders/1ZFjapJF4jwx_ZGl0JQaQB3WVuEv4Xd2P?usp=sharing

(buka dengan akun std)

Link GitHub

https://github.com/nandono206/convexHull-divide-and-conquer