TUGAS KECIL 2 IF 2211

STRATEGI ALGORITMA

Oleh

Mohamad Hilmi Rinaldi 13520149

PROGRAM STUDI TEKNIK INFORMATIKA

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

2022

DAFTAR ISI

BAB 1

ALGORITMA *DIVIDE AND CONQUER*

Di dalam implementasi Convex Hull untuk visualisasi tes *linear separability* dataset dengan algoritma *divide and conquer*, program diawali dengan menerima masukan array dataset berisi kumpulan titik. Dari array data tersebut akan dicari dua titik ekstrim (memiliki nilai x minimum dan maksimum) yang akan membentuk convex hull untuk kumpulan titik tersebut.

Setelah mendapatkan kedua titik ekstrim yaitu p1 dan pn, lalu partisi kumpulan titik yang berada di sebelah kiri garis p1pn menjadi himpunan titik s1 dan sebelah kanan garis p1pn menjadi himpunan titik s2 dengan mengecek determinan antara p1,pn, dan titik yang ingin diuji. Lalu, himpunan s1 dan s2 masing-masing akan dimasukkan ke dalam fungsi divide and conquer bersama kedua titik ekstrim tersebut untuk mempartisi kembali setiap titik yang berada di himpunan s1 dan s2.

Di dalam fungsi divide and conquer tersebut terdapat dua kemungkinan kondisi. Kondisi yang pertama yaitu jika himpunan titik kosong, maka p1 dan pn merupakan pembentuk Convex Hull. Lalu, kondisi yang kedua yaitu jika himpunan titik tidak kosong, maka akan dipiilih terlebih dahulu sebuah titik px yang merupakan titik terjauh dengan garis p1pn. Jika terdapat titik lain yang memiliki jarak yang sama, maka akan dipilih titik yang memaksimalkan sudut pxp1pn.

Jika px sudah ditemukan, selanjutnya akan dicari kumpulan titik yang berada di bagian luar garis p1px dan garis pxpn. Kedua kumpulan titik tersebut selanjutkan akan diproses ulang dengan fungsi divide and conquer sebelumnya hingga didapatkan kedua kumpulan titik yang kosong. Setelah pemanggilan fungsi divide and conquer dari himpunan s1 dan s2 selesai dilakukan, hasil dari fungsi divide and conquer himpunan s1 dan juga hasil divide and conquer himpunan s2 akan digabung sehingga didapatkan hasil akhir Convex Hull.

BAB 2

KODE PROGRAM

```
#Modul myConvexHull

import math

def indeks_ekstrim(array):
    # Digunakan untuk mencari indeks dari suatu array yang memiliki nilai x
minimum dan maksimum dari kumpulan titik
    indeks_x_min = 0
    indeks_x_max = 0
    for i in range(len(array)):
        if(array[i][0] < array[indeks_x_min][0]):
            indeks_x_min = i
        if(array[i][0] > array[indeks_x_max][0]):
            indeks_x_max = i

    return indeks_x_min, indeks_x_max

def determinan(p1, pn, px, array):
    # Digunakan untuk mencari determinan
    return array[p1][0]*array[pn][1]+ array[px][0]*array[p1][1] +
array[pn][0]*array[px][1] - array[px][0]*array[pn][1] -
array[pn][0]*array[p1][1] - array[p1][0]*array[px][1]

def partisi(p1, pn, array):
    # Digunakan untuk mempartisi dua bagian yang berada di kiri garis p1pn
(s1, det > 0) dan kanan garis p1pn (s2, det < 0)
    s1 = []; s2 = []
    for i in range(len(array)):
        if(i != p1 and i != pn and determinan(p1, pn, i, array) > 0):
            s1.append(i)
        elif(i != p1 and i != pn and determinan(p1, pn, i, array) < 0):
            s2.append(i)
    return s1, s2

def partisi_satu_sisi(p1, pn, px, array, array_partisi):
    # Digunakan untuk mempartisi bagian yang berada di luar dua bagian garis
yang dihubungkan oleh tiga titik
    p1px = [] ; pxpn = []
    for i in range(len(array_partisi)):
        if(array_partisi[i] != p1 and array_partisi[i] != pn and
determinan(p1, px, array_partisi[i], array) > 0):
            p1px.append(array_partisi[i])
        if(array_partisi[i] != p1 and array_partisi[i] != pn and
determinan(px, pn, array_partisi[i], array) > 0):
```

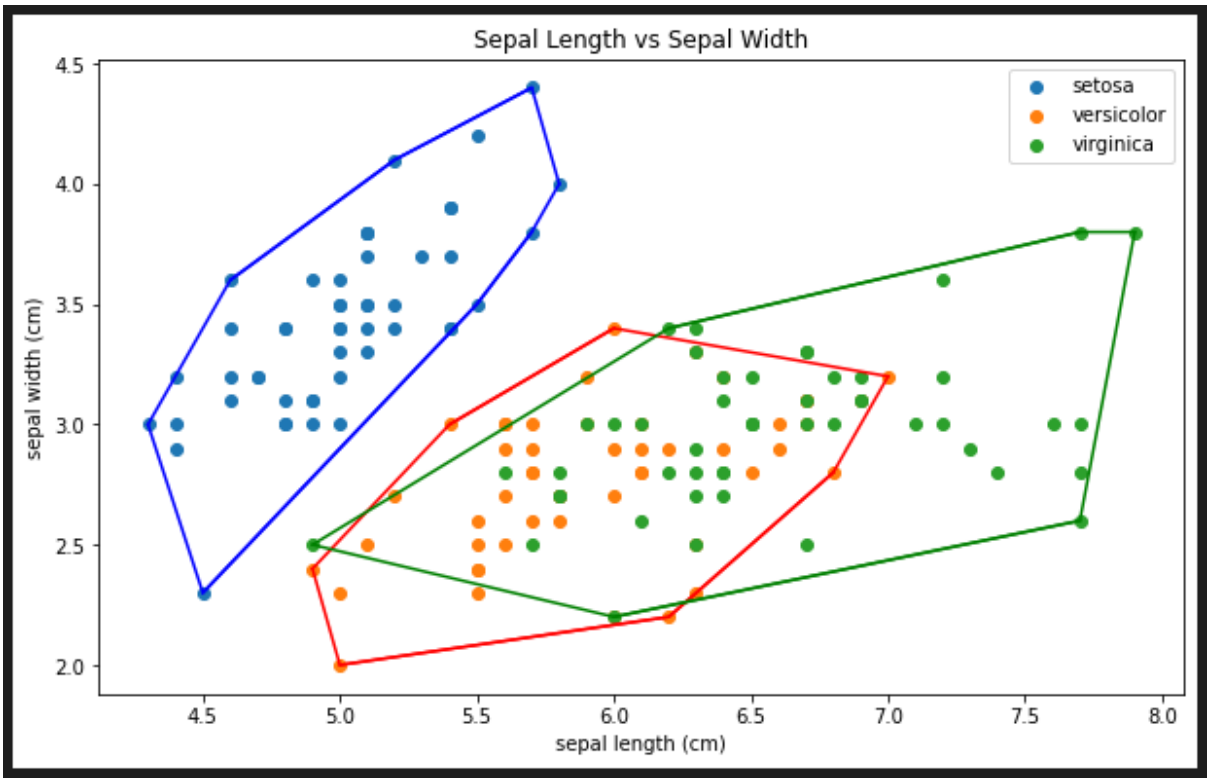
```
            pxpn.append(array_partisi[i])
    return p1px, pxpn

def jarak(p1, pn, px, array):
    # Digunakan untuk mencari jarak dari titik px ke garis p1pn
    return abs(((array[pn][0]-array[p1][0])*(array[p1][1]-array[px][1]) -
(array[p1][0]-array[px][0])*(array[pn][1]-array[p1][1])) /
math.sqrt((array[pn][0]-array[p1][0])**2 + (array[pn][1]-array[p1][1])**2))

def sudut_p1(px,p1,pn, array):
    # Digunakan untuk mencari sudut p1
    sudut = math.degrees(math.atan2(array[pn][1]-array[p1][1], array[pn][0]-
array[p1][0]) - math.atan2(array[px][1]-array[p1][1], array[px][0]-
array[p1][0]))

    if(sudut < 0):
        return 360 + sudut
    else:
        return sudut

def titik_terjauh(p1, pn, array, array_partisi):
    # Digunakan untuk mencari titik terjauh dari garis p1pn (px)
    temp = 0
    for i in range(len(array_partisi)):
        if(jarak(p1, pn, array_partisi[temp], array) < jarak(p1, pn,
array_partisi[i], array)):
            temp = i
        elif(jarak(p1, pn, array_partisi[temp], array) == jarak(p1, pn,
array_partisi[i], array)):
            if(sudut_p1(array_partisi[temp], p1, pn, array) <=
sudut_p1(array_partisi[i], p1, pn, array)):
                temp = i
    return array_partisi[temp]

def divide_and_conquer(p1, pn, array, array_partisi, ConvexHull):
    # Digunakan untuk mendapatkan titik pembentuk convex hull dari kumpulan
titik
    # Jika array yang ingin dipartisi kosong, berarti tidak ada titik yang
berada di kanan atau kiri garis
    if(len(array_partisi)) == 0:
        ConvexHull.append([p1, pn])
    else:
        # Mencari titik terjauh dari garis p1pn (px)
        px = titik_terjauh(p1, pn, array, array_partisi)

        # Mengecek titik yang berada di bagian luar garis p1px dan pxpn
        p1px, pxpn = partisi_satu_sisi(p1, pn, px, array, array_partisi)
```

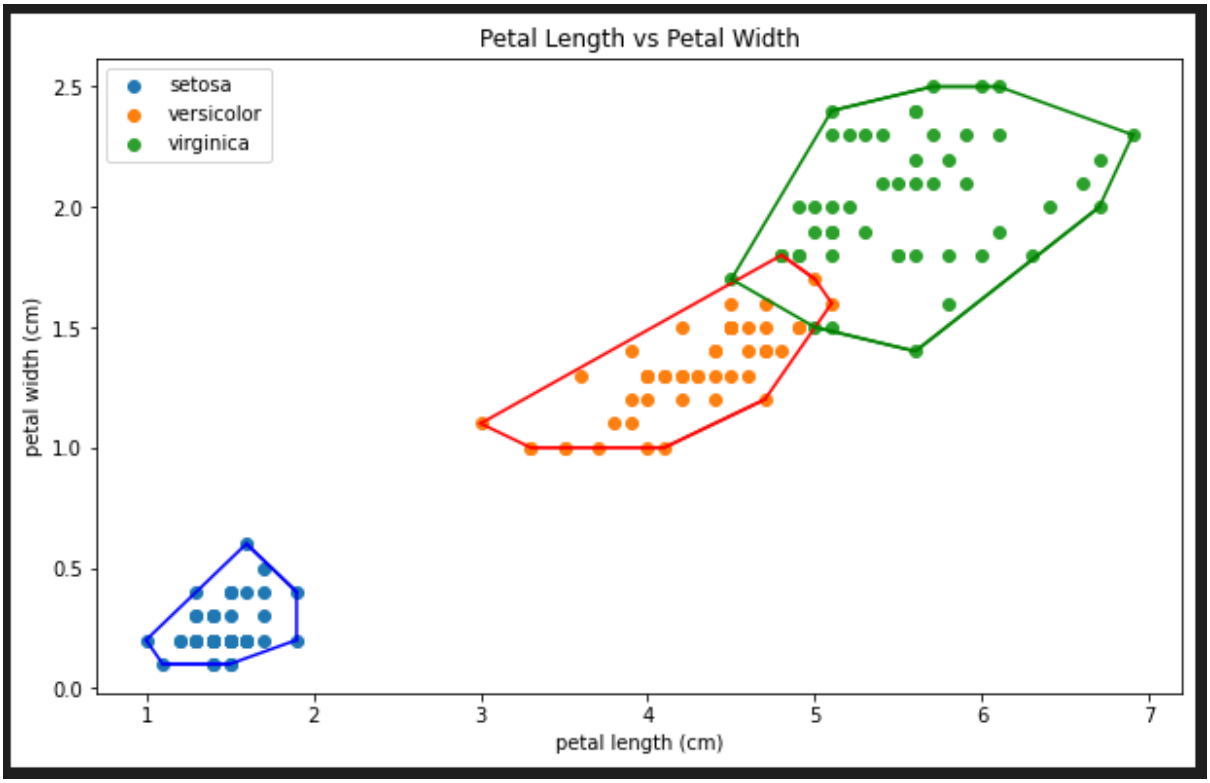
```
        # Lakukan partisi kembali hingga tidak ada titik yang berada di luar
convex hull
        divide_and_conquer(p1, px, array, p1px, ConvexHull)
        divide_and_conquer(px, pn, array, pxpn, ConvexHull)

# Implementasi dari myConvexHull

def myConvexHull(array):
    ConvexHull = []
    # Mencari indeks dari titik yang memiliki nilai x minimum dan maksimum
    p1, pn = indeks_ekstrim(array)

    # Membuat kumpulan titik dari sebelah kiri garis p1pn (s1) dan sebelah
kanan garis p1pn (s2)
    s1, s2 = partisi(p1, pn, array)

    # Mempartisi kembali setiap titik yang berada di s1 dan s2 sehingga
didapatkan kumpulan titik pembentuk convex hull
    divide_and_conquer(p1, pn, array, s1, ConvexHull)
    divide_and_conquer(pn, p1, array, s2, ConvexHull)

    return ConvexHull
```

BAB 3

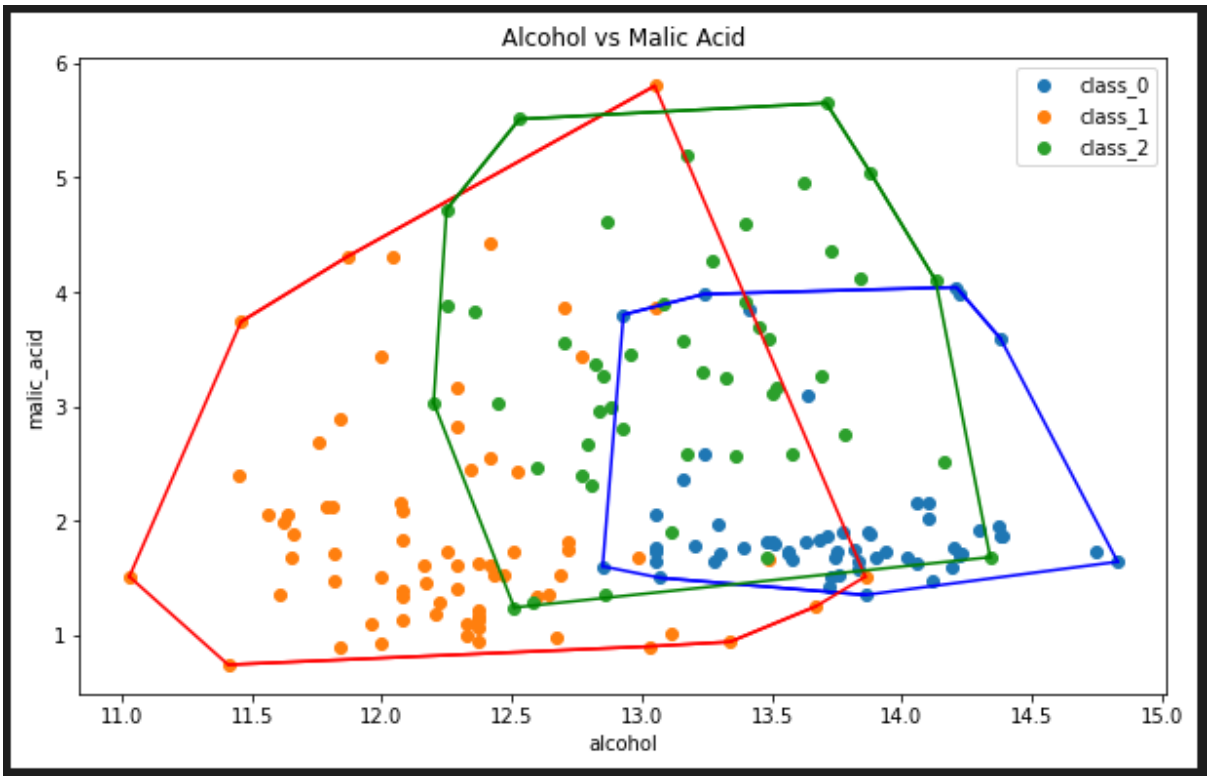
SCREENSHOT INPUT DAN *OUTPUT*

3.1.Dataset Iris

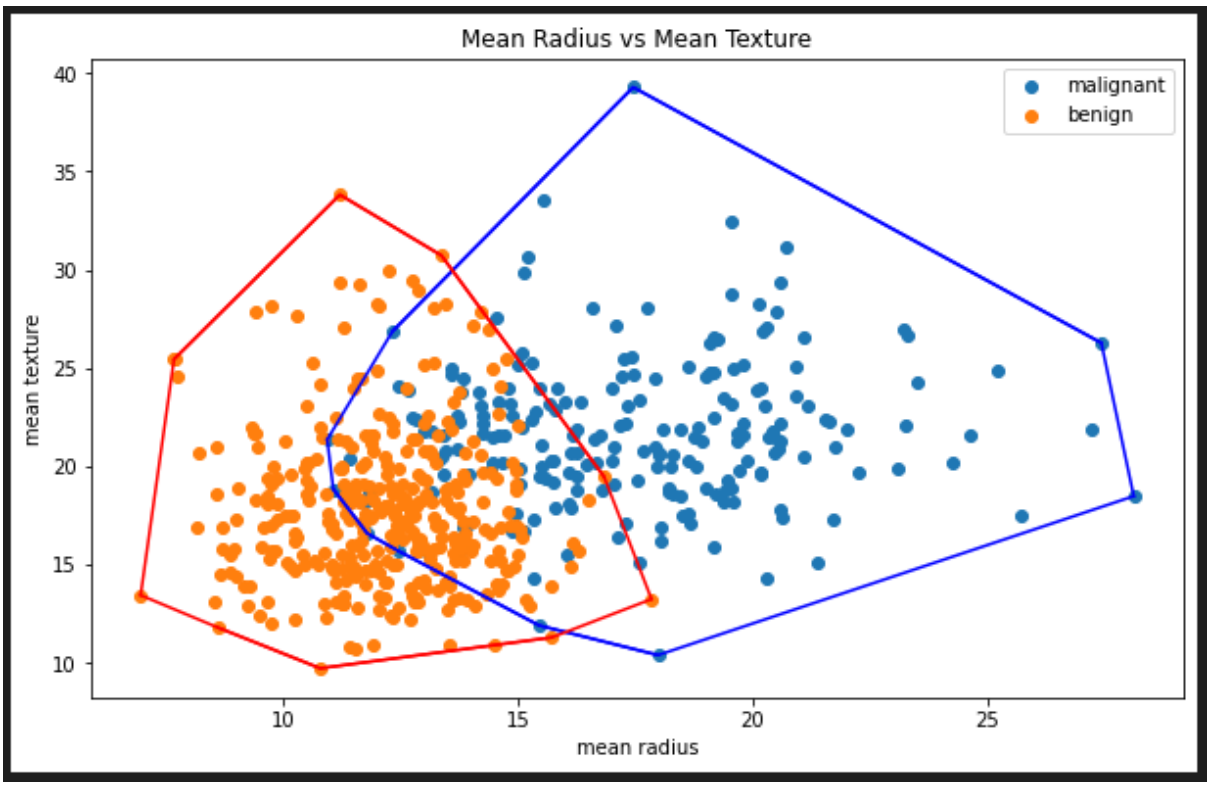




3.2.Dataset Wine



3.3.Dataset Breast Cancer



BAB 4

ALAMAT DRIVE

https://github.com/mhilmirinaldi/Convex-Hull-Python

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.	✓	