LAPORAN TUGAS KECIL 2 IF2211 STRATEGI ALGORITMA

IMPLEMENTAI CONVEX HULL UNTUK VISUALISASI TES LINEAR SEPARABILITY DATASET DENGAN ALGORITMA DIVIDE AND CONQUER

Oleh :

Ilham Prasetyo Wibowo 13520013

PROGRAM STUDI TEKNIK INFORMATIKA

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA

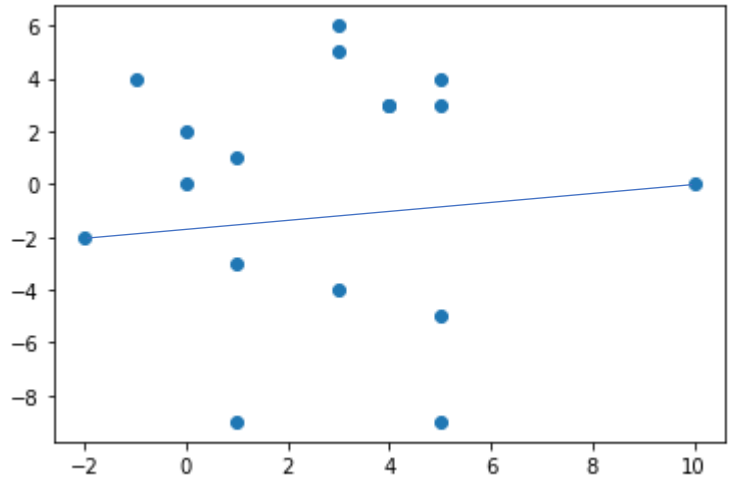
INSTITUT TEKNOLOGI BANDUNG

2022

A. DESKRIPSI ALGORITMA

Himpunan titik pada sebuah bidang planar disebut convex jika untuk sembarang dua titik pada bidang tersebut semua segmen garis yang berakhir di titik tersebut berada pada himpunan tersebut. Convex hull dari sekumpulan titik S adalah sebuah himpunan terkecil (convex terkecil) yang mengandung semua titik pada S.

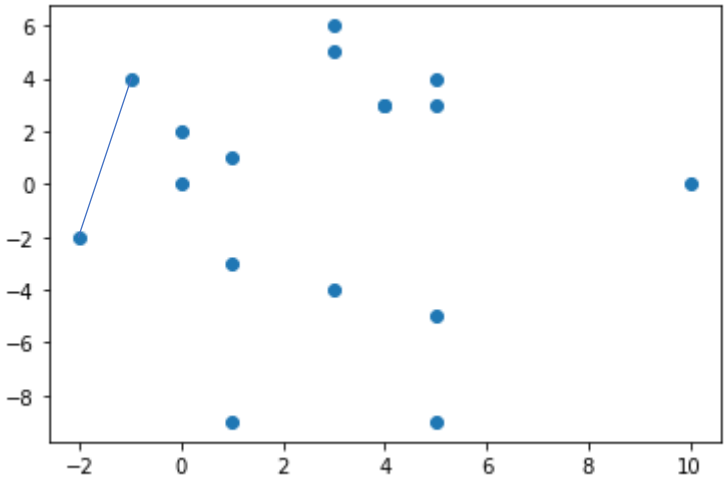
Program menerima sebuah larik (array) yang berisi titik-titik dalam ruang dua dimensi yang akan dicari convex hull-nya. Kemudian akan dicari garis yang membagi dua dari kumpulan titik. Garis tersebut dibentuk dari titik yang paling kiri (nilai x terkecil) dengan garis yang paling kanan(nilai x terbesar).



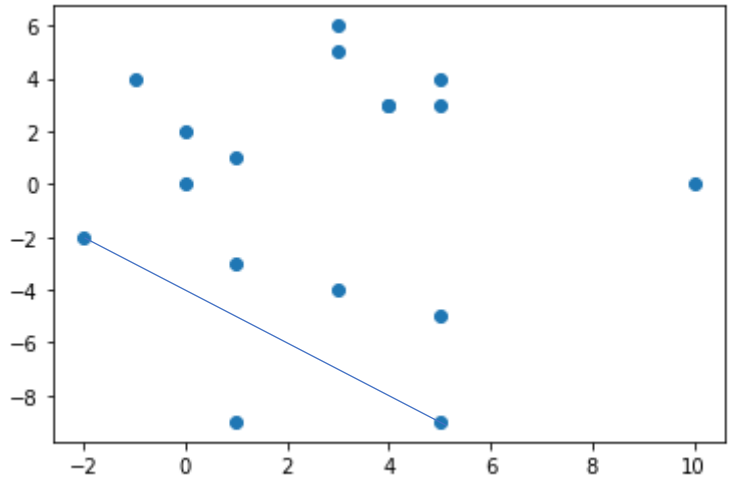
Ilustrasi mencari garis yang akan membagi titik menjadi dua bagian

Setelah garis ditemukan, kumpulan titik akan dibagi menjadi dua yaitu titik yang berada di atas garis, dan titik yang berada dibawah garis. Setelah itu akan dicari masing masing convex hull dari bagian dan akan digabungkan.

Pencarian convex hull dilakukan secara rekursif. Basis rekursif adalah jika tidak ada lagi titik di atas garis. Atau titik di atas garis hanya tersisa satu.



Ilustrasi ketika sudah mencapai basis rekursi pertama



Ilustrasi Ketika mencapai basis rekursi kedua

Ketika sudah mencapai basis, program akan mengembalikan sebuah tuple dengan dua elemen yaitu index titik yang membentuk garis . Ketika program mencapai basis rekursi kedua, program akan mengembalikan kedua garis misal AB, dan BC.

Jika masih terdapat lebih dari satu titik yang ada diatas garis, program akan mencari titik yang paling jauh dari garis. Misal garis AB, dan terdapat titik terjauh X. Program akan mencari titik titik yang berada di atas garis AX dan XB. kemudian secara rekursif akan mencari convex hull masing masing titik dan kedua hasil akan digabung.

B. SOURCE CODE

Berikut kode program yang telah ditulis.

```
#file myConvexHull.py

import numpy as np
from numpy.linalg import norm

def myConvexHull(points):
    #get index of max value in points
    max_idx = np.argmax(points, axis=0)[0]
    #get index of min value in points
    min_idx = np.argmin(points, axis=0)[0]

    #divide points into two part with a line shaped by point in min_idx and
    max_idx
    #get index of values above the line
```

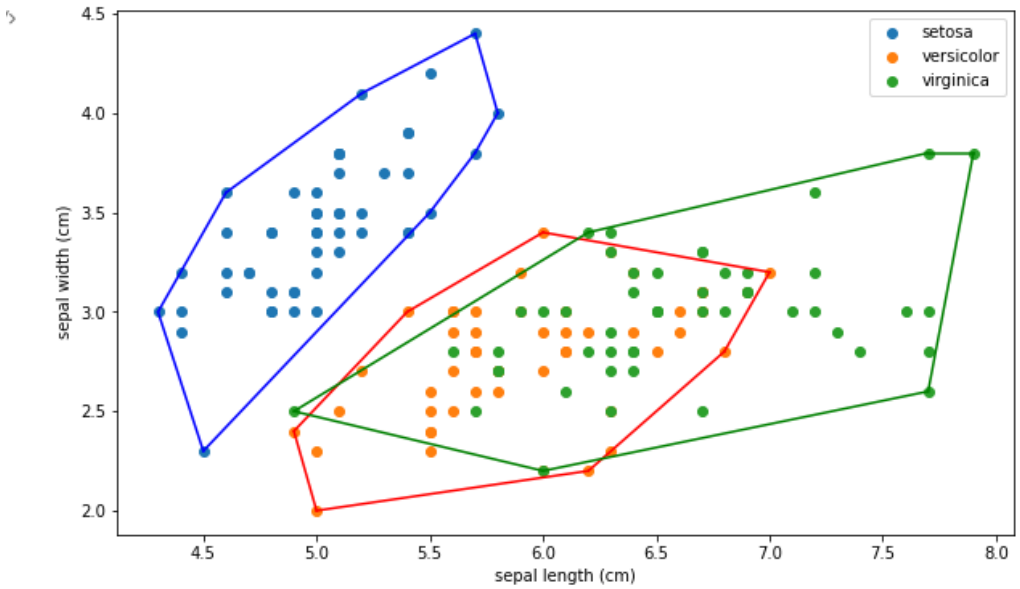
```
    index_above = partition_above(points,points[min_idx],points[max_idx])
    #get index of values above the line
    index_below = partition_above(points,points[max_idx],points[min_idx])

    #search convex hull above line
    above_convex = get_convex_hull(points,index_above,points[min_idx],
      points[max_idx],[min_idx,max_idx])
    #search convex hull below line
    below_convex = get_convex_hull(points,index_below,points[max_idx],
      points[min_idx],[max_idx,min_idx])

    #combine both convex hulls
    trueConvex = above_convex + below_convex

    return np.array(trueConvex)

def get_convex_hull(points,index_A,pA,pB,pLine):
    #there isnt any points on left/right side of pLine
    if (len(index_A) == 0):
        return [pLine]
    #only one point on left/right side of pLine
    #return line ab, bc
    elif (len(index_A) == 1) :
        return [[pLine[0],index_A[0]],[index_A[0],pLine[1]]]
    #still more than one points
    #search for a points farthest from line pApB
    else :
        farthest_idx = get_farthest_point(points,pA,pB,index_A)
        farthest = points[farthest_idx]


    #index of points on left side of line pA-farthestpoint
    convex_left = partition_above_2(points,pA,farthest,index_A)
    #index of points on left side of line farthestpoint,pB
    convex_right = partition_above_2(points,farthest,pB,index_A)

    #search convex hull left side of line pA-farthest recursively
    left_hull = get_convex_hull(points,convex_left,pA,
      farthest,[pLine[0],farthest_idx])
    #search convex hull left side of line farthest-pB recursively
    right_hull = get_convex_hull(points,convex_right,farthest,pB,
      [farthest_idx,pLine[1]])

    #combine the results from left_hull and right_hull
    ultimate_hull = left_hull + right_hull

    #return
    return ultimate_hull
```

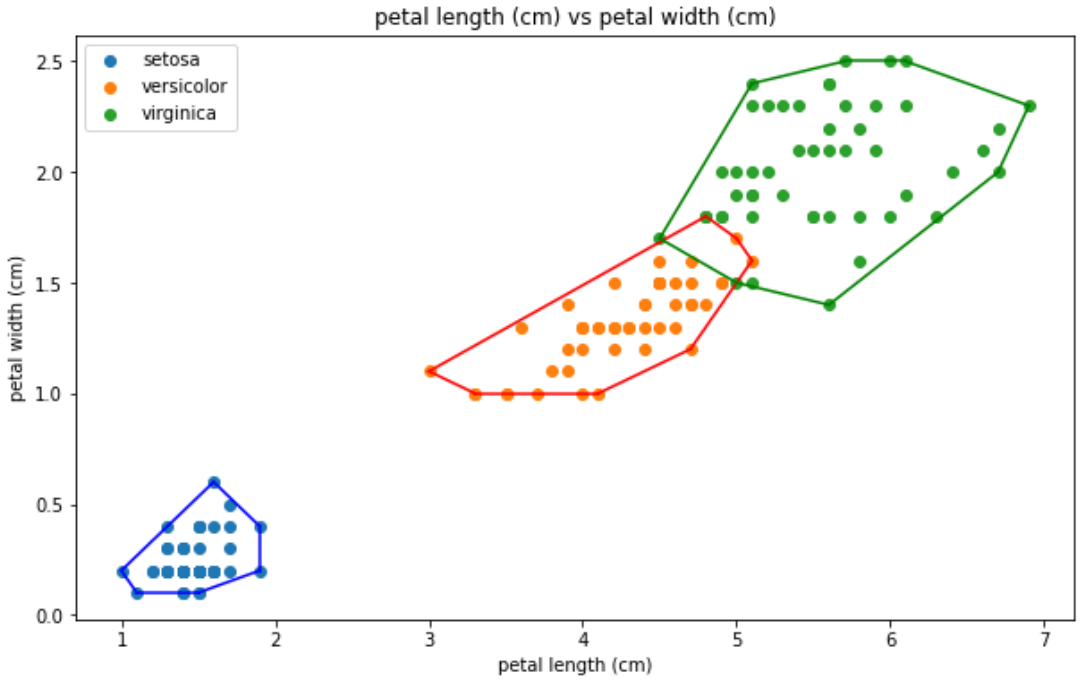
```
#search points by ALL points in point
def partition_above(points,pA,pB):
    points_result = []
    for i in range(len(points)):

        #Determinant
        x = pA[0]*pB[1] + points[i][0]*pA[1] + pB[0]*points[i][1] -
      points[i][0]*pB[1] - pB[0]*pA[1] - pA[0]*points[i][1]
        if (x > 0):
            #point[i] is above line pApB
            points_result.append(i)

    #returns indices of points in list
    return points_result

#search points only by array of index ind
def partition_above_2(points,pA,pB,ind):
    points_result = []
    for i in ind:
        if (points_equal(points[i],pA)):
            continue
        elif (points_equal(points[i],pB)):
            continue
        else :
            #determine whether the point is above or below line pA-pB
            x = (pA[0]*pB[1]) + (points[i][0]*pA[1]) +
             (pB[0]*points[i][1]) - (points[i][0]*pB[1]) - (pB[0]*pA[1]) -
             (pA[0]*points[i][1])
            if (x > 0):
                #point[i] is above line pApB
                points_result.append(i)

    #returns indices of points in list
    return points_result

def points_equal(pA,pB):
    if (pA[0] == pB[0]):
        if (pA[1] == pB[1]):
            return True

    return False

#get the farthest point from line pA-pB
def get_farthest_point(points,pA,pB,idx):
    ires = 0
    dres = 0
    for i in idx:
```

```
        d = abs(np.cross(pB-pA,points[i]-pA)/norm(pB-pA))
        if (dres < d):
            ires = i
            dres = d
    return ires
```

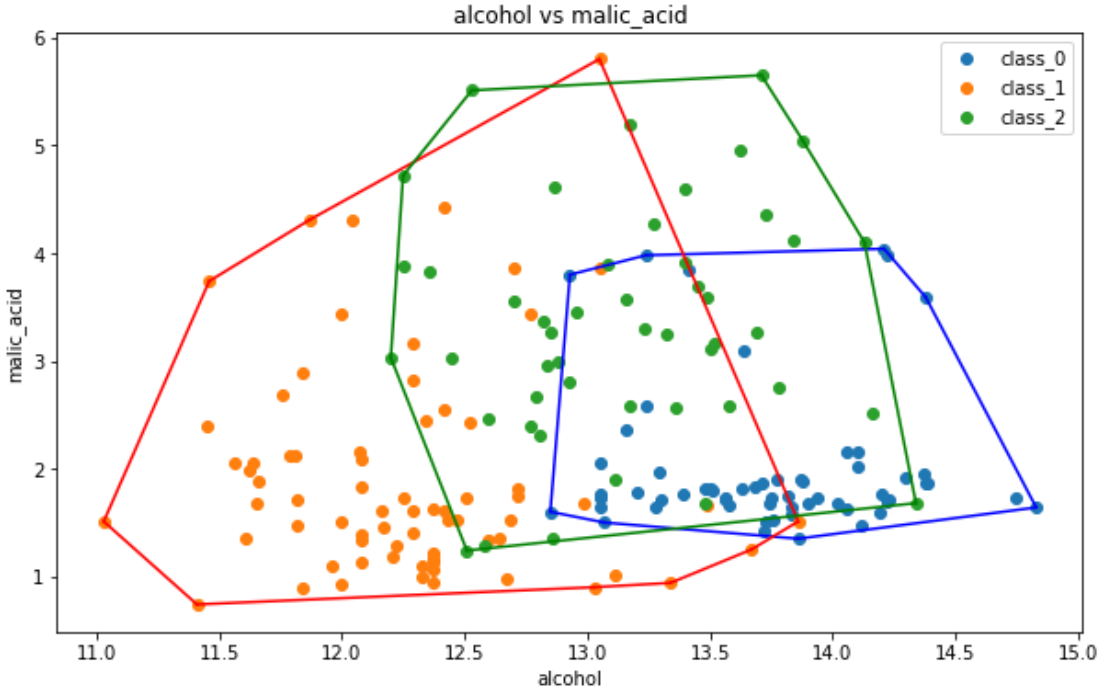
C. HASIL PENGUJIAN

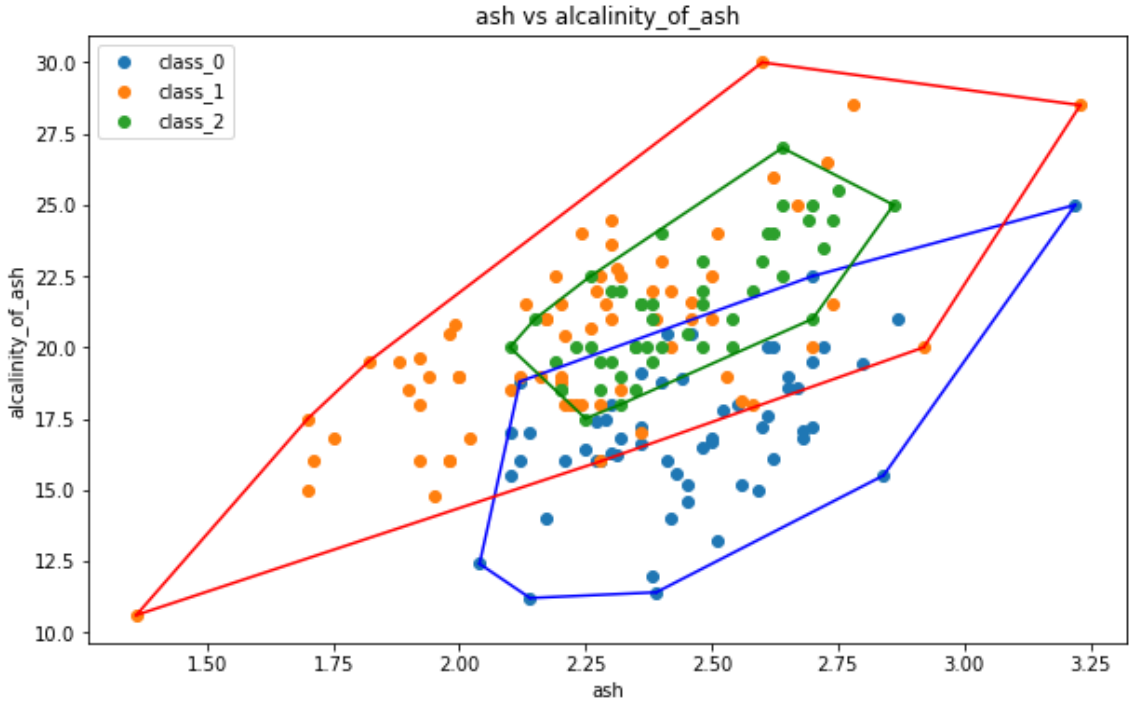
IRIS DATASET



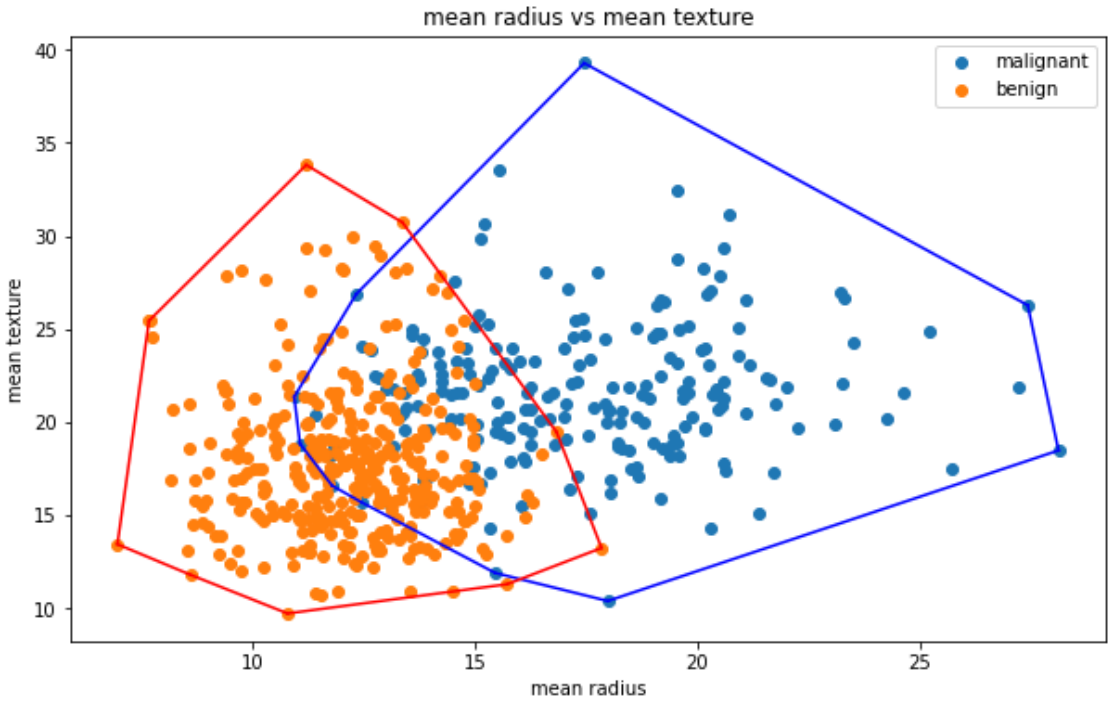


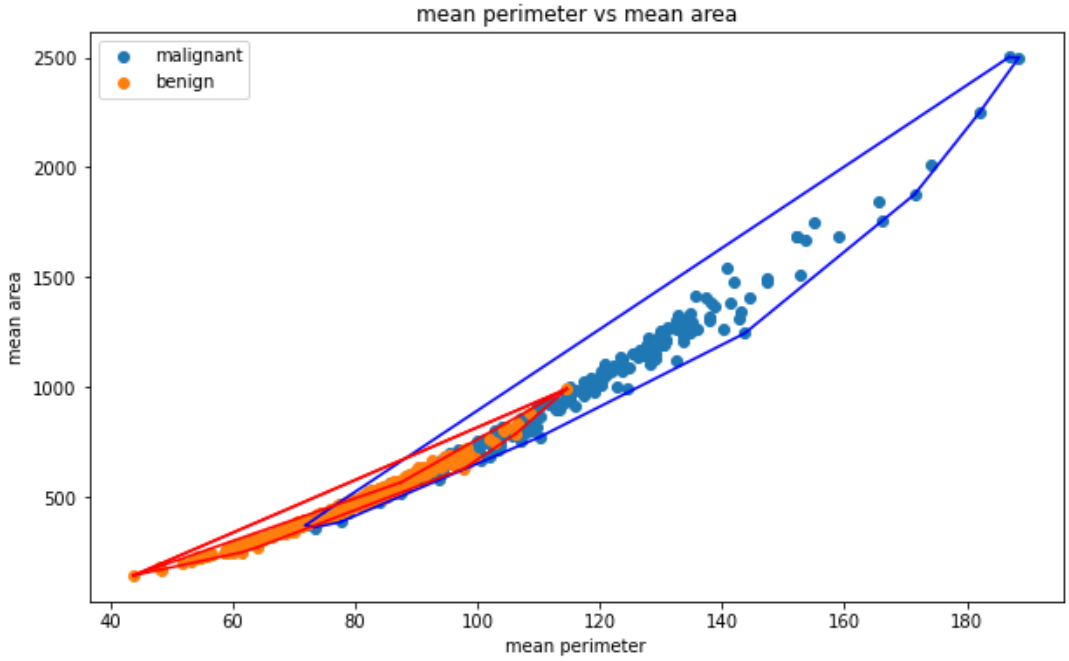
WINE DATASET





BREAST CANCER DATASET





D. TAUTAN DRIVE PROGRAM

https://github.com/ilhamwibowo/Tucil-2-STIMA

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.	✓	