Laporan

Tugas Kecil 2 IF2211 Strategi Algoritma

Implementasi Convex Hull untuk Visualisasi Tes *Linear Separability Dataset* dengan Algoritma *Divide and Conquer*

Disusun Oleh:

Suryanto 13520059

PROGRAM STUDI TEKNIK INFORMATIKA

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

2022

DAFTAR ISI

DAFTAR ISI 2
ALGORITMA DIVIDE AND CONQUER 3
SOURCE CODE 4
1. convexHull.py 4
2. main.py 6
HASIL EKSEKUSI PROGRAM 8
1. Dataset Iris 8
2. Dataset Wine 9
3. Dataset Breast_cancer 10
SOURCE CODE 11
LAMPIRAN 11

ALGORITMA DIVIDE AND CONQUER

Divide and Conquer adalah salah satu strategi dalam pemecahan suata persoalan. *Divide* yang berarti membagi suatu persoalan menjadi beberapa upa-persoalan yang memiliki kemiripan dengan persoalan semula dengan ukuran yang lebih kecil. Sedangkan *conquer* sebagai tahapan menyelesaikan masing-masing upa-persoalan tersebut. Setelahnya, tiap solusi dari upa-persoalan akan digabung (*combine)* untuk membentuk solusi atas persoalan semula.

Salah satu persoalan yang dapat diselesaikan dengan strategi ini adalah penentuan *convex hull* dari kumpulan titik. Penentuan *convex hull* pada program ini mengikuti tahapan-tahapan berikut ini :

1. Program mengurutkan kumpulan titik berdasarkan nilai absis secara menaik. Apabila terdapat lebih dari satu titik dengan absis yang sama, maka akan dilakukan pengurutan terhadap ordinat secara menaik.
2. Selanjutnya akan dipilih dua titik yaitu titik pada indeks pertama (misal P dengan absis terkecil) dan titik pada indeks terakhir (Q dengan absis terbesar),
3. Kedua titik ini akan membentuk suatu garis lurus yang membagi bidang menjadi dua bagian, yaitu bagian S1 yang memuat kumpulan titik di atas garis dan bagian S2 yang memuat kumpulan titik di bawah garis,
4. Pada setiap bagian, akan dicari titik terjauh dari garis yang dibentuk sebelumnya. Jika terdapat beberapa titik dengan jarak yang sama, maka dipilih titik yang membentuk sudut terbesar,
5. Jika tidak ada titik lain, maka titik P dan Q akan disimpan sebagai himpunan titik yang membentuk *convex hull*,
6. Tahapan 3,4 dan 5 akan terus dilakukan ke tiap upa-bagian hingga kedua bagian tidak mengandung titik lagi,
7. Himpunan titik yang dihasilkan adalah titik-titik yang membangun convex hull.

SOURCE CODE

1. myConvexHull.py

```
import numpy as np

def angle(p1,p2,p3):
    '''
        Fungsi untuk mencari sudut yang p213
    '''
    p1 = np.array(p1)
    p2 = np.array(p2)
    p3 = np.array(p3)

    p12 = p1 - p2
    p13 = p1 - p3

    cosine_angle = np.dot(p12, p13) / (np.linalg.norm(p12) *
np.linalg.norm(p13))
    angle = np.arccos(cosine_angle)
    return angle

def distpoint(p1,p2):
    '''
        Fungsi untuk mencari jarak antara 2 titik
    '''
    x = p2[0] - p1[0]
    y = p2[1] - p1[1]
    return (x*x + y*y)**0.5

def dist(p1,p2,p3):
    '''
        Fungsi untuk mencari jarak suatu titik (p3) terhadap
        garis yang dibentuk oleh titik p1 dan p2
    '''
    a = p2[0] - p1[0]           #x2 - x1
    b = p1[1] - p3[1]           #y1 - y3
    c = p1[0] - p3[0]           #x1 - x3
    d = p2[1] - p1[1]           #y2 - y1
    return abs(a*b - c*d) / distpoint(p1,p2)

def determinan(p1, p2, p3):
    '''
        Fungsi untuk penentuan determinan. Digunakan untuk mengecek
        posisi titik p3 terhadap garis yang dibentuk titik p1 dan p2.
    '''
    return (p1[0] - p3[0]) * (p2[1] - p3[1]) - (p2[0] - p3[0]) * (p1[1] -
p3[1])
```

```
#
def quickhull(arr):
    '''
        Fungsi utama untuk mencari convex hull.
    '''
    hull = []

    # Mencari titik dengan absis minimum dan maksimum.
    sort=sorted(arr,key=lambda x:x[0])
    p1=sort[0]
    p2=sort[-1]

    # Memanggil fungsi quickhullPartition
    # Dipanggil dua kali untuk mencari solusi dibawah dan diatas garis.
    hull += quickhullPartition(arr, p1, p2)
    hull += quickhullPartition(arr, p2, p1)

    return hull

def quickhullPartition(arr, p1, p2):
    '''
        Implementasi Algoritma Divide and Conquer
        untuk menghasilkan convex hull
    '''

    # Divide, dengan hanya mengambil titik di sebelah kiri garis
    # Jika absis p2 > p1, maka bagian sebelah kiri == bagian atas garis
    # Jika absis p1 > p2, maka bagian sebelah kiri == bagian bawah garis
    # Titik di sebelah kanan tidak diproses karena tidak mungkin membentuk
convex hull
    El_point = []
    for point in arr:
        if (determinan(p1, p2, point) >0):
            El_point.append(point)
    # Points = [point for point in arr if determinan(p1, p2, point) >0]

    # Mencari titik terjauh terhadap garis yang dibentuk p1 dan p2
    pmax = None
    maxDist = 0
    maxAngle = 0
    for point in El_point:
        tempDist = dist(p1, p2,point)
        if  tempDist >= maxDist:
            maxDist = tempDist
            pmax = point
        elif tempDist == maxDist:
            tempAngle = angle(p1,p2,point)
```

```
            if (tempAngle > maxAngle):
                maxAngle = tempAngle
                pmax = point

    if pmax == None:
        return [p2]
    hull = []

    # Bagian Conquer, Rekursi
    # Dipanggil dua kali untuk mencari solusi di sebelah kiri p1-pmax dan
    # di sebelah kiri pmax-p2
    hull += quickhullPartition(El_point, p1, pmax)
    hull += quickhullPartition(El_point, pmax, p2)
    return hull
```

2. main.py

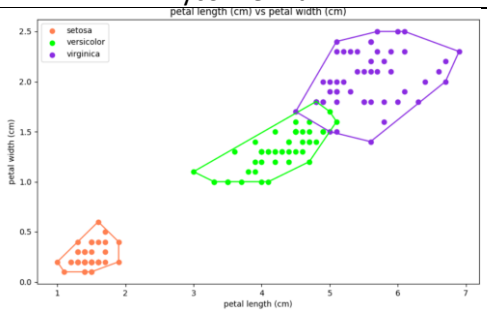
```
import myConvexHull
import pandas as pd
import matplotlib.pyplot as plt
from sklearn import datasets

print("Daftar Dataset")
print("1. Dataset iris")
print("2. Dataset wine")
print("3. Dataset breast cancer")

#pemilihan dataset
option = int(input("Pilih Dataset yang ingin digunakan : "))
while (option not in range(1,4)):
    option = int(input("Pilih Dataset yang ingin digunakan : "))

if ( option == 1 ):
    data = datasets.load_iris()
elif ( option == 2):
    data = datasets.load_wine()
elif ( option == 3):
    data = datasets.load_breast_cancer()


#create a DataFrame
df = pd.DataFrame(data.data, columns=data.feature_names)
df['Target'] = pd.DataFrame(data.target)

#pemilihan atribut yang ingin digunakan
print("\nDaftar atribut:")
for i in range(len(data.feature_names)):
    print(i+1,  data.feature_names[i])
```

```
ATx = int(input("atribut-x yang ingin digunakan : "))
ATy = int(input("atribut-y yang ingin digunakan : "))


#visualisasi hasil ConvexHull
plt.figure(figsize = (10, 6))
colors = ['coral','lime','blueviolet', 'magenta','blue', 'red', 'green']
plt.title(data.feature_names[ATx-1] + ' vs ' + data.feature_names[ATy-1])
plt.xlabel(data.feature_names[ATx-1])
plt.ylabel(data.feature_names[ATy-1])

for i in range(len(data.target_names)):
    bucket = df[df['Target'] == i]
    bucket = bucket.iloc[:,[ATx-1,ATy-1]].values

    # Copy titik2 pada bucket ke arr
    arr=[]
    for x in bucket:
        arr +=[[x[0], x[1]]]
    hulls = myConvexHull.quickhull(arr)

    # Digunakan untuk keperluan plot
    x_points = []
    y_points = []
    for point in hulls:
        x_points.append(point[0])
        y_points.append(point[1])

    x_points.append(hulls[0][0])
    y_points.append(hulls[0][1])


    plt.plot(x_points, y_points, color = colors[i])
    plt.scatter(bucket[:, 0], bucket[:, 1], label=data.target_names[i],
color = colors[i])

plt.legend()
plt.show()
3.
```

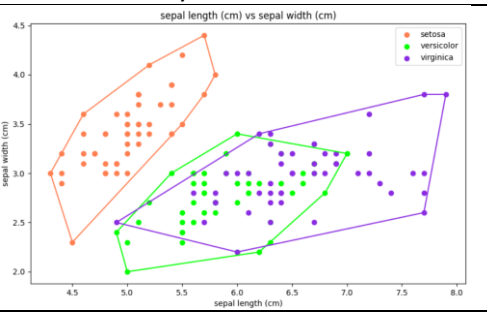
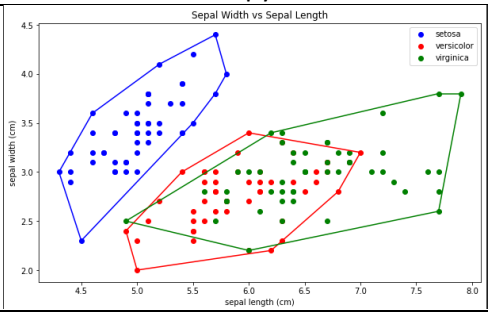
HASIL EKSEKUSI PROGRAM

1. Dataset Iris

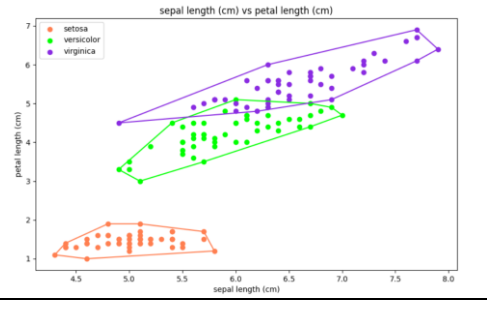
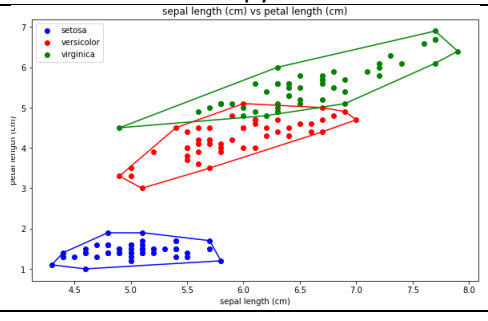
- Petal length – Petal Width

MyconvexHull	Scipy
	

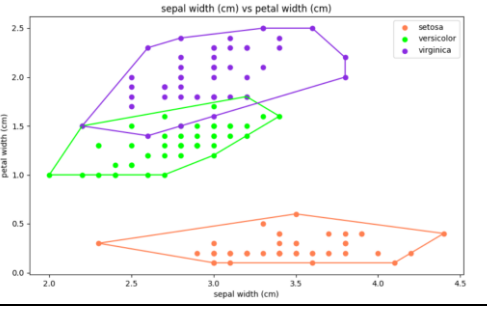
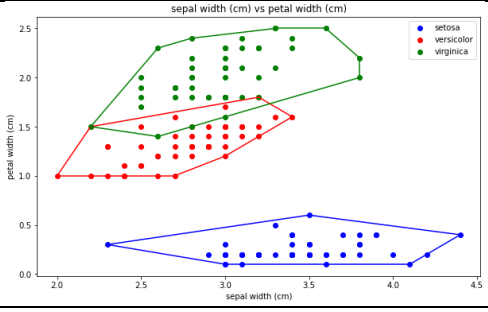
- Sepal length – Sepal Width

MyconvexHull	Scipy
	

- Sepal length – Petal Length

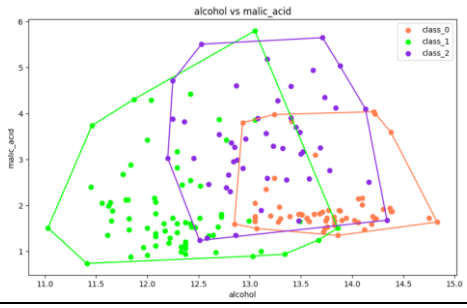
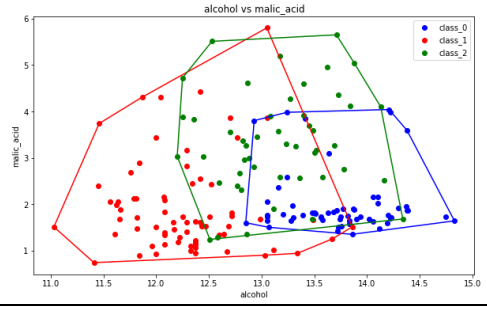
MyconvexHull	Scipy
	

- Sepal Width – Petal Width

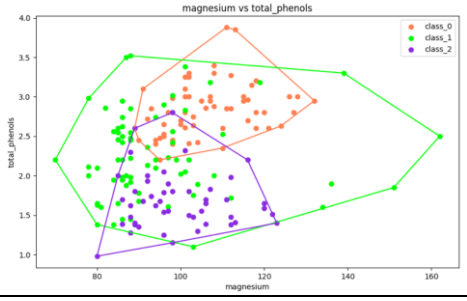
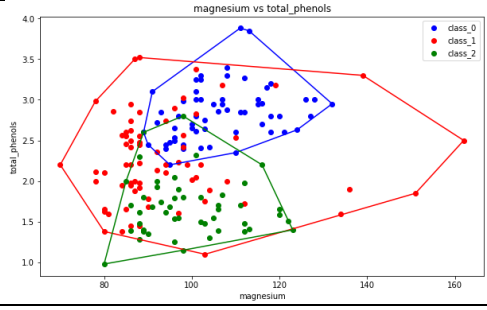
MyconvexHull	Scipy
	

2. Dataset Wine

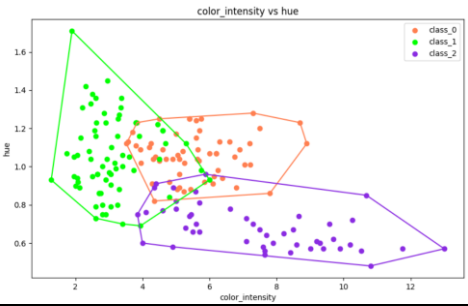
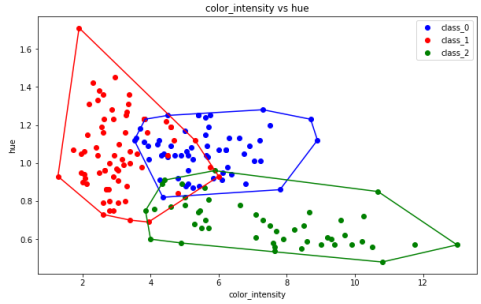
- alcohol – malic_acid

MyconvexHull	Scipy
	

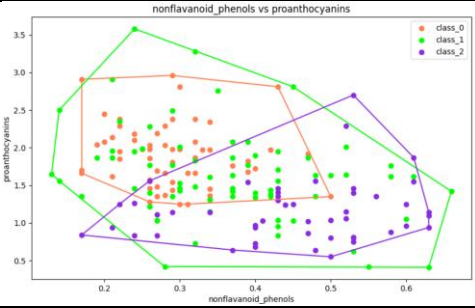
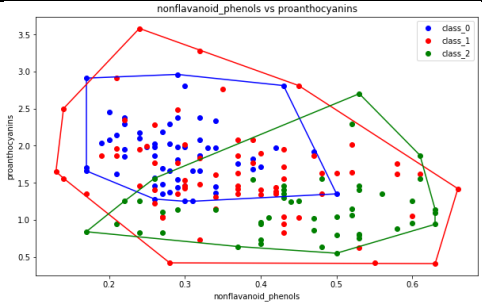
- magnesium – total_phenols

MyconvexHull	Scipy
	

- color_intensity – hue

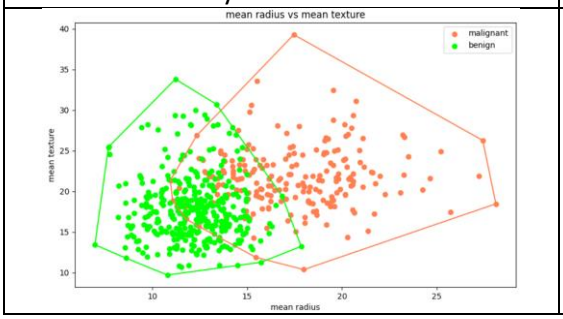
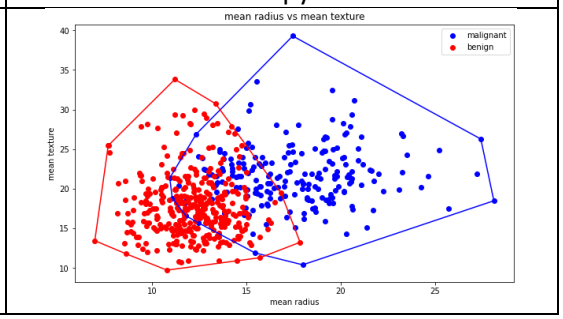
MyconvexHull	Scipy
	

- nonflavanoid_phenols – proanthocyanins

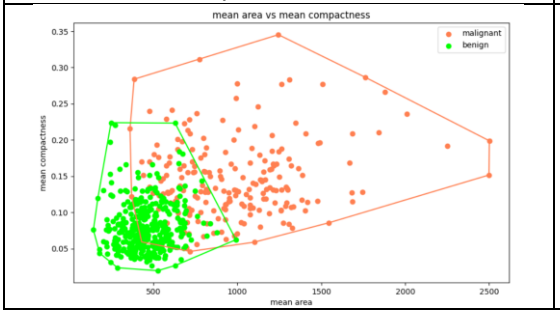
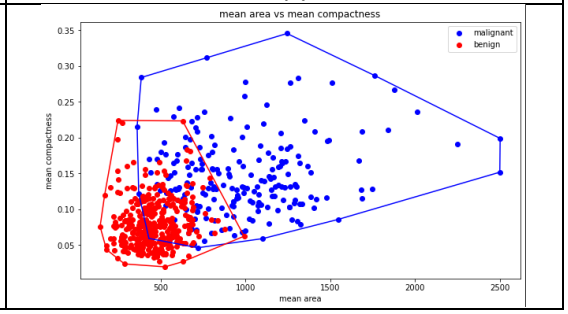
MyconvexHull	Scipy
	

3. Dataset Breast_cancer

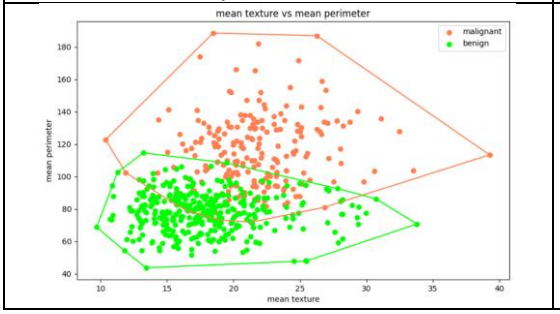
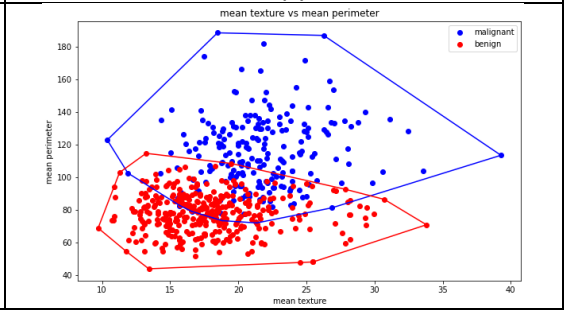
- mean_radius – mean_texture

MyconvexHull	Scipy
	

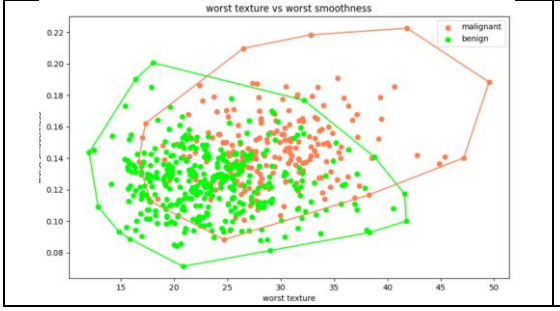
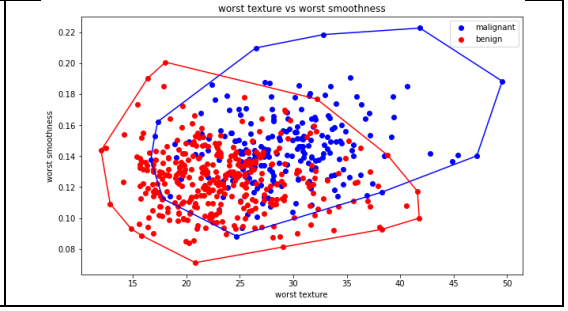
- mean_area – mean_compactness

MyconvexHull	Scipy
	

- mean_texture – mean_perimeter

MyconvexHull	Scipy
	

- worst_texture – worst_smoothness

MyconvexHull	Scipy
	

SOURCE CODE

https://github.com/SurTan02/Convex-Hull

LAMPIRAN

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.	√	