

Laporan
Tugas Kecil 2 IF2211 Strategi Algoritma
Implementasi Convex Hull untuk Visualisasi Tes *Linear*
Separability Dataset* dengan Algoritma *Divide and Conquer



Disusun Oleh:

Aldwin Hardi Swastia 13520167

PROGRAM STUDI TEKNIK INFORMATIKA
SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA INSTITUT
TEKNOLOGI BANDUNG

2022

DAFTAR ISI

Contents

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

ALGORITMA DIVIDE AND CONQUER

Divide and Conquer merupakan strategi untuk memecahkan masalah dengan membagi persoalan menjadi upa-persoalan yang memiliki kemiripan dengan persoalan semula dengan ukuran yang lebih kecil. *Divide* berarti membagi dan *Conquer* berarti menggabungkan. Jadi strategi ini dimulai dengan membaginya menjadi persoalan yang lebih kecil kemudian menggabungkannya kembali .

Salah satu persoalan yang dapat diselesaikan dengan strategi *Divide and Conquer* 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 kemudian berdasarkan ordinat apabila ditemukan nilai absis yang sama secara menaik.
2. Kemudian program akan memilih titik dengan nilai absis terkecil(p_1) dan absis terbesar(p_2),
3. Kedua titik ini akan membentuk suatu garis lurus yang membagi bidang menjadi dua bagian, yaitu partisi S_1 yang memuat kumpulan titik di kiri/atas garis dan partisi S_2 yang memuat kumpulan titik di kanan garis,
4. Pada setiap bagian, akan dicari titik terjauh(p_{max}) dari garis yang dibentuk sebelumnya,
5. Lalu akan dibentuk dua garis, yaitu antara titik p_1 dengan p_{max} dan p_{max} dengan p_2 ,
6. Setelah itu, program akan memvalidasi apabila terdapat titik yang berada pada bagian kiri garis p_1 , p_{max} dan pada bagian kanan garis p_{max} , p_2 ,
7. Lakukan langkah 5 dan 6 hingga tidak terdapat titik yang di luar lingkup segitiga yang terbentuk dari p_1 , p_{max} , dan p_2 ,
8. Jika tidak ada titik lain, maka titik p_1 dan p_2 akan disimpan sebagai himpunan titik yang membentuk *convex hull*,
9. Kemudian gabungkan titik-titik yang menjadi titik terjauh tersebut menjadi himpunan titik yang membentuk *convex hull*,
10. Himpunan titik yang dihasilkan adalah titik-titik yang membangun *convex hull*.

SOURCE CODE

1. myConvexHull.py

```
# library of myConvexHull

def leftRight(p1,p2,p3): #Return boolean True apabila ada di left
    x = p1[0]*p2[1]+p3[0]*p1[1]+p2[0]*p3[1]-p3[0]*p2[1]-p2[0]*p1[1]-
    p1[0]*p3[1]
    return x>0

def distance(p1,p2,p3): #Return jarak dua titik terhadap satu titik
    return ((p1[0]-p3[0])**2+(p1[1]-p3[1])**2)**0.5+((p2[0]-p3[0])**2+(p2[1]-
    p3[1])**2)**0.5

def setOfPoint(points, p1, p2): #Return seluruh koordinat titik yang terdapat
pada suatu partisi
    # Inisiasi nilai dan point max
    maxDistance = 0
    dist = 0
    farthest = []

    # Iterasi mencari nilai terbesar
    for i in range(len(points)):
        dist = distance(p1,p2,points[i])
        if(maxDistance<dist):
            maxDistance = dist
            farthest = points[i]

    # Apabila tidak ada yang menjadi poin terjauh langsung kembalikan list
    kosong
    if farthest == []:
        return []

    # Inisiasi bagian kiri dari garis antara p1 dan pmax serta bagian kanan
    dari garis antara pmax dan p2
    left = []
    right = []

    # check left and right dan bagi ke dalam list
    for point in points:
        if(leftRight(p1,farthest,point) and point != farthest):
            left.append(point)
        if(leftRight(farthest,p2,point) and point != farthest):
            right.append(point)

    # Inisiasi koordinat hull yang terdapat pada kiri p1,pmax dan koordinat
    hull pada kanan pmax,p2
```

```

leftPoint = []
rightPoint = []

# Lakukan rekursi apabila masih terdapat titik yang diluar garis
if(len(left) != 0):
    leftPoint = setOfPoint(left,p1,farthest)
if(len(right) != 0):
    rightPoint = setOfPoint(right,farthest,p2)

return leftPoint+farthest+rightPoint

def myConvexHull(bucket): #Return Simplices yang merupakan hull
    # Melakukan sort pada list dari data
    sortedBucket = sorted(bucket, key=lambda x:[x[0],x[1]])

    # Inisiasi nilai list yang terdapat pada kiri dan kanan garis
    left = []
    right = []

    # Membagi koordinat titik yang ada menjadi dua oleh garis
    for i in range(1,len(sortedBucket)-1):
        if(leftRight(sortedBucket[0],sortedBucket[-1],sortedBucket[i])):
            left.append(sortedBucket[i])
        else:
            right.append(sortedBucket[i])

    # Melakukan rekursi partisi pada bagian kiri dan kanan garis hingga
    # didapatkan koordinat hull
    leftPoint = setOfPoint(left,sortedBucket[0],sortedBucket[-1])
    rightPoint = setOfPoint(right,sortedBucket[-1],sortedBucket[0])

    # Gabungkan data koordinat titik sehingga terbentuk vertex-vertex searah
    # jarum jam
    coordinatesOfVertices = []+sortedBucket[0]+leftPoint+sortedBucket[-
1]+rightPoint

    # Membuat data menjadi point untuk setiap koordinatnya
    pointOfVertices = []
    for i in range(0,len(coordinatesOfVertices)-1,2):
        temp = []
        temp.append(coordinatesOfVertices[i])
        temp.append(coordinatesOfVertices[i+1])
        pointOfVertices.append(temp)

    # Membentuk vertex dari data koordinat point
    vertices = []
    for point in pointOfVertices:

```

```

        vertices.append(bucket.index(point))

# Membentuk Simplices dari Vertices
simplices = []
simplices.append([vertices[0],vertices[-1]])
for i in range(1,len(vertices)):
    simplices.append([vertices[i],vertices[i-1]])

return simplices

```

2. main.py

```

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

# MAIN
print(
    """
    Visualizer myConvexHull
    Datasets:
    1. iris
    2. wine
    3. breast cancer
    """
)

opsi = int(input("Pilihan: "))
while (not(opsi == 1 or opsi == 2 or opsi == 3)):
    print("Masukan salah")
    opsi = int(input("Pilihan: "))

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

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

print("Atributes:")
for i in range(len(data.feature_names)):
    print(i+1, data.feature_names[i])

```

```

print()

inputAbsis = int(input("Pilihan X: "))
inputOrdinat = int(input("Pilihan Y: "))

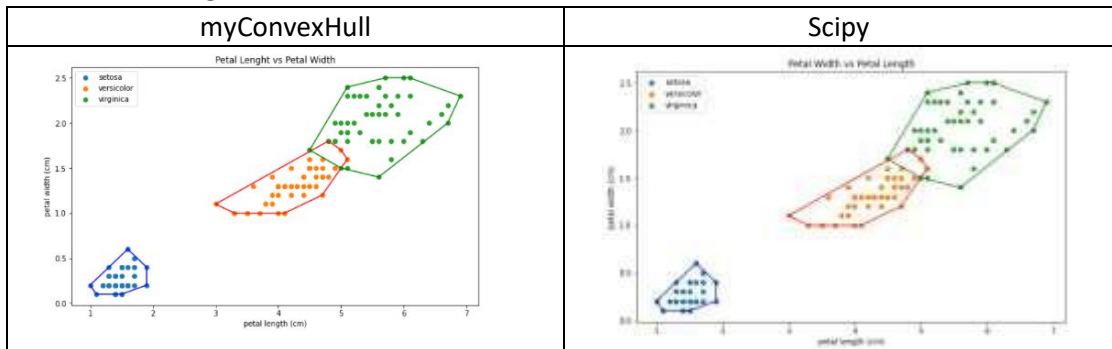
plt.figure(figsize = (10, 6))
colors = ['b','r','g']
plt.title(data.feature_names[inputAbsis-1]+" vs "+data.feature_names[inputOrdinat-1])
plt.xlabel(data.feature_names[inputAbsis-1])
plt.ylabel(data.feature_names[inputOrdinat-1])
for i in range(len(data.target_names)):
    bucket = df[df['Target'] == i]
    bucket = bucket.iloc[:,[inputAbsis-1,inputOrdinat-1]].values
    bucket = bucket.tolist() #membuat numpy array menjadi list
    hull = myConvexHull.myConvexHull(bucket)
    hull = np.asarray(hull) #membuat list menjadi numpy array
    bucket = np.asarray(bucket)
    plt.scatter(bucket[:, 0], bucket[:, 1], label=data.target_names[i])
    for simplex in hull:
        plt.plot(bucket[simplex, 0], bucket[simplex, 1], colors[i])
plt.legend()
plt.show()

```

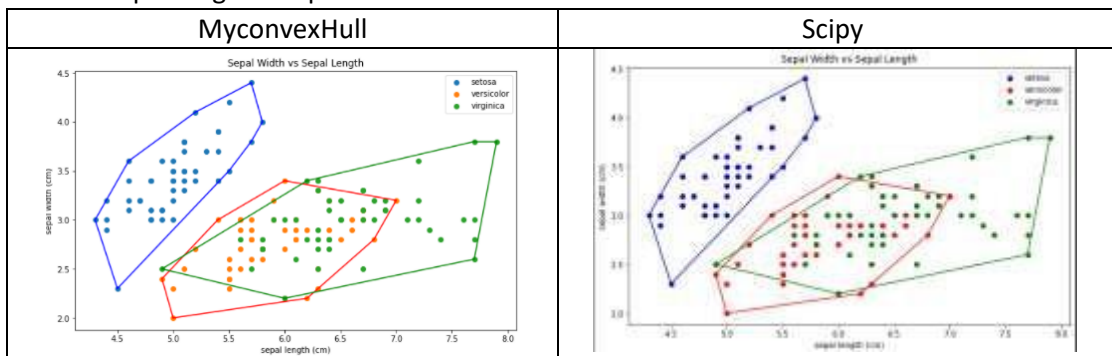
HASIL EKSEKUSI PROGRAM

Dataset Iris

- Petal length – Petal Width

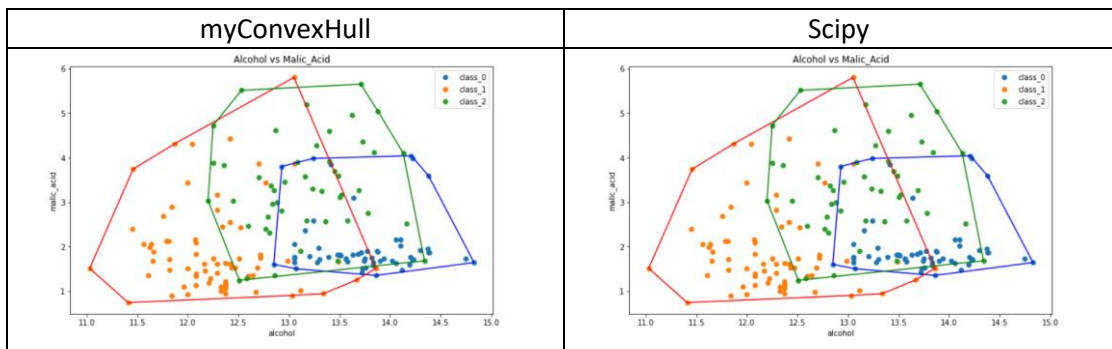


- Sepal length – Sepal Width



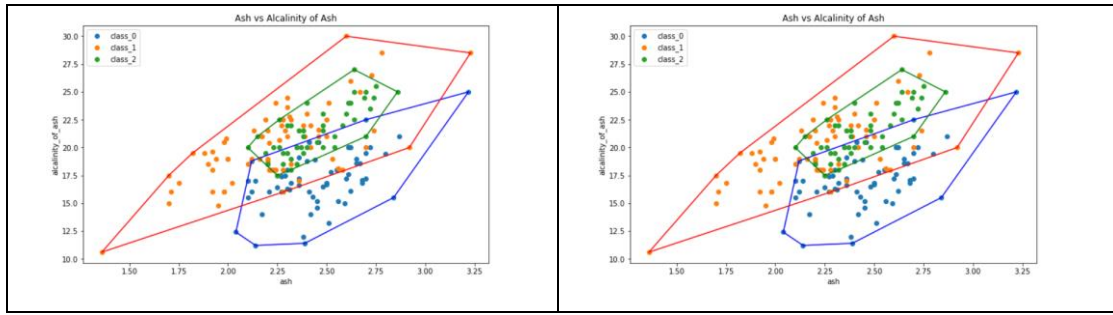
Dataset Wine

- Alcohol – Malic Acid



- Sepal length – Sepal Width





SOURCE CODE

<https://github.com/aldwinhs/myConvexHull>

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.	✓	