Tugas Kecil IF2211 Strategi Algoritma

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

Disusun Oleh :

Ilham Pratama

13520041

Kelas K-02

PROGRAM STUDI SARJANA TEKNIK INFORMATIKA
SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

2022

Daftar Isi

Algoritma Divide And Conquer

Algoritma Divide and Conquer adalah sebuah algoritma yang terdiri dari 3 bagia, yaitu divide, conquer, dan combine. Pada tugas kecil ini, secara garis besar penyelesaian algoritma divide and conquer yang digunakan mengikuti langkah – langkah berikut :

1. Langkah pertama adalah mengurutkan semua himpunan titik. Setelah Himpunan titik diurutkan selanjutnya himpunan titik akan dibagi menjadi 2 bagian. Pembagian titik ini dilakukan dengan menggunakan fungsi det(A,B,C). fungsi det tersebut akan membuat suatu garis yang membagi semua himpunan titik menjadi 2 bagian. Lalu titik yang berada di sebelah kiri/atas dari garis akan dimasukkan ke dalam suatu list yang bernama Bagiankiri dan titik yang berada pada kanan/bawah dari garis akan dimasukkan ke dalam sebuah list yang bernama BagianKanan.
2. Setelah himpunan titik dibagi menjadi 2 bagian, kedua bagian tersebut akan dimasukkan ke dalam fungsi rekursiv yang berisi algoritma divide dan conquer untuk menyelesaikan permasalahan ini. Fungsi rekursiv akan terus membagi himpunan titik tersebut hingga mencapai basis. Jika sudah mencapai basis fungsi akan me-return nilai dari jawaban dari subproblem bagian kanan, hal ini dilakukan supaya tidak ada pencatatan titik yang terjadi 2 kali. Jika belum mencapai basis maka fungsi akan terus memanggil fungsi rekursiv untuk membagi subproblem menjadi 2 subproblem berikutnya. Pemanggilan fungsi akan mengembalikan sebuah titik yang mana titik ini menjadi bagian convex hull (*implementasi bisa dilihat pada bagian source code*).
3. Setelah semua himpunan titik dibagi dan telah mencapai basis untuk masing-masing mereka. Selanjutnya fungsi rekursiv akan mengembalikan gabungan dari semua titik yang menjadi bagian dari convex hull sehingga diperoleh convex hull dari himpunan titik tersebut.

Source Code Program

Source Code myConvexHull.py

```
def Det(A,B,C):
    #menghitung determinan / garis yang digunakan untuk membagi
titik menjadi 2 bagian
    det = A[0]*B[1] + B[0]*C[1] + C[0]*A[1] - A[0]*C[1] -
B[0]*A[1] - C[0]*B[1]
    return det

def rekursiv(Points, P, Q):
    #fungsi rekursive untuk convexHull
    if(len(Points) == 0):
        #basis
        return [Q]
    else:
        hull = []
        detMaximum = 0 # inisiasi determinan
        for i in Points:
            determinant = Det(P, Q, i)
            if(determinant > detMaximum):
                detMaximum = determinant
                hull = i

        #bagi jadi 2 bagian, kanan dan kiri
        Bagiankanan = []
        BagianKiri = []
        for j in Points:
            detKanan = Det(hull, Q, j)
            detKiri = Det(P, hull, j)
            if(detKanan > 0.000000001):
                Bagiankanan.append(j)
            if(detKiri > 0.000000001):
                BagianKiri.append(j)

        #ulangi terus sampai menemui basis
        BagianKanan = rekursiv(Bagiankanan, hull, Q)
        BagianKiri = rekursiv(BagianKiri, P, hull)
        return BagianKanan + BagianKiri

#fungsi utama
def ConvexHull(Points):
    if(len(Points) < 2): #minimal harus 2 titik untuk membentuk
baris
        return Points
    else:
        Points.sort()
        P = Points[0] # ambil bagian paling kiri
        Q = Points[len(Points)-1] #ambil bagian paling kanan
        BagianKanan = []
        BagianKiri = []
        for i in Points:
            determinant = Det(P, Q, i)
```

```
            if(determinant > 0.000000001):
                #jika positif masukkan ke dalam bagian kiri
                BagianKiri.append(i)
            if(determinant < -0.000000001):
                #Jika negatif masukkan ke dalam bagian kanan
                BagianKanan.append(i)

        #ulangi terus sampai ketemu basis, isi elemen dari points
kurang dari 2
        BagianKanan = rekursiv(BagianKanan, Q, P)
        BagianKiri = rekursiv(BagianKiri, P, Q)
        return BagianKanan+BagianKiri
```

Source Code Main.py

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

print("masukkan data yang ingin anda gunakan: \n1. iris \n2.
wine\n")
pilihanData = int(input("masukkan input : "))

if(pilihanData == 1):
    data = datasets.load_iris()
    print("1. Sepal Width VS Sepal Length \n2. Petal Width VS
Petal Length")
    pilihan = int(input("masukkan perbandingan : "))
    if(pilihan == 1):
        t = 0
        u = 1
    elif(pilihan == 2):
        t = 2
        u = 3

if(pilihanData == 2):
    data = datasets.load_wine()
    print("1. Malic Acid VS Color Intensity \n2. Alcohol VS
Flavanoids")
    pilihan = int(input("masukkan perbandungan: "))
    if(pilihan == 1):
        t = 1
        u = 9
    elif(pilihan == 2):
        t = 0
        u = 6

df = pd.DataFrame(data.data, columns=data.feature_names)
df['Target'] = pd.DataFrame(data.target)
plt.figure(figsize = (10, 6))
colors = ['b','r','g']
```

```
plt.title(f'{data.feature_names[t].title()} vs
{data.feature_names[u].title()}')
plt.xlabel(data.feature_names[t])
plt.ylabel(data.feature_names[u])
for i in range(len(data.target_names)):
    bucket = df[df['Target'] == i]
    bucket = bucket.iloc[:,[t,u]].values
    bucket = bucket.tolist()
    hull = ConvexHull(bucket) #hasil implementasi convexHull
    bucket = np.array(bucket)
    plt.scatter(bucket[:, 0], bucket[:, 1],
label=data.target_names[i])
    x = [hull[len(hull)-1][0]]
    y = [hull[len(hull)-1][1]]
    for p in hull:
        x.append(p[0])
        y.append(p[1])
    plt.plot(x, y, colors[i])
    plt.legend()
plt.show()
```

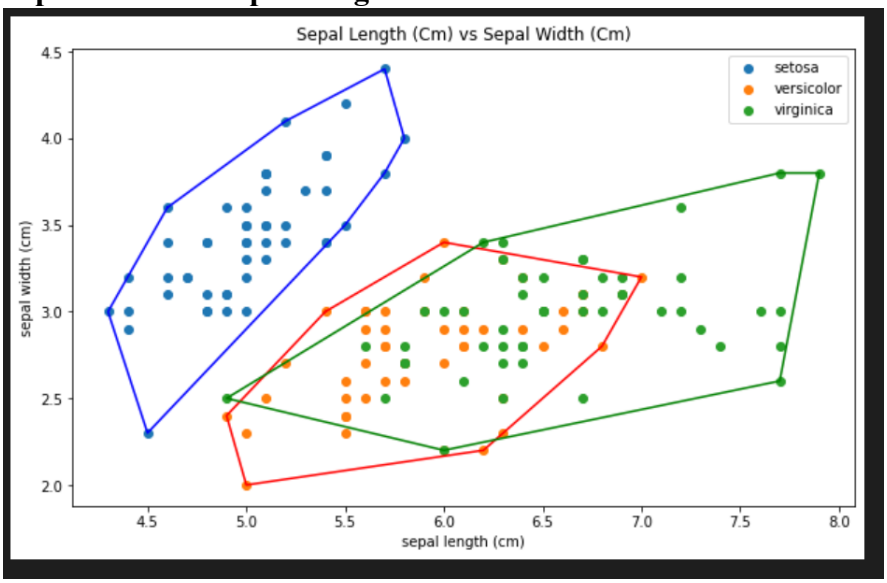
Screenshots Input Dan Output

Input Data iris

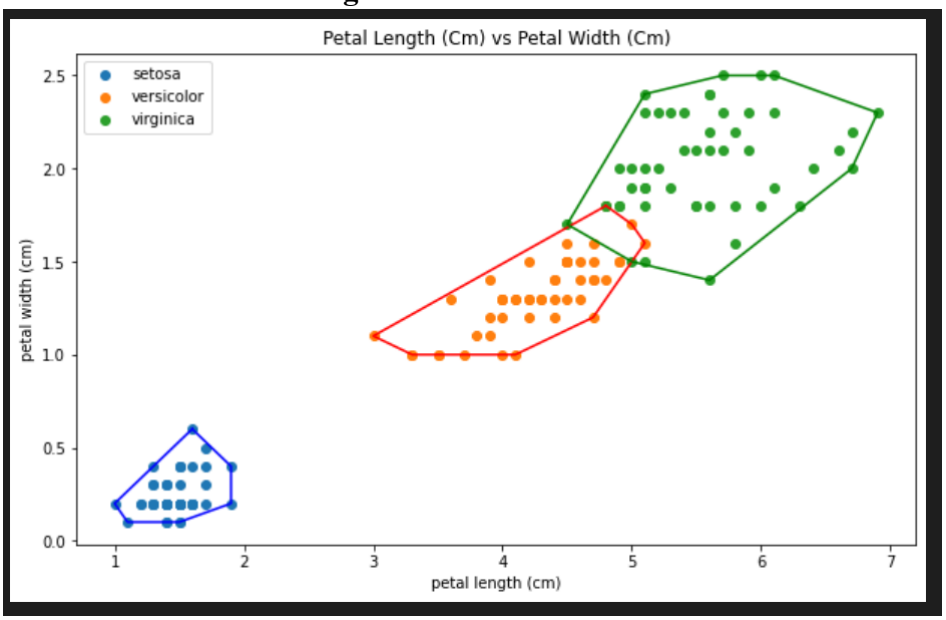
	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)	Target
0	5.1	3.5	1.4	0.2	0
1	4.9	3.0	1.4	0.2	0
2	4.7	3.2	1.3	0.2	0
3	4.6	3.1	1.5	0.2	0
4	5.0	3.6	1.4	0.2	0

Output:

- **Sepal Widht vs Sepal Length**



- **Petal Width vs Petal Length**

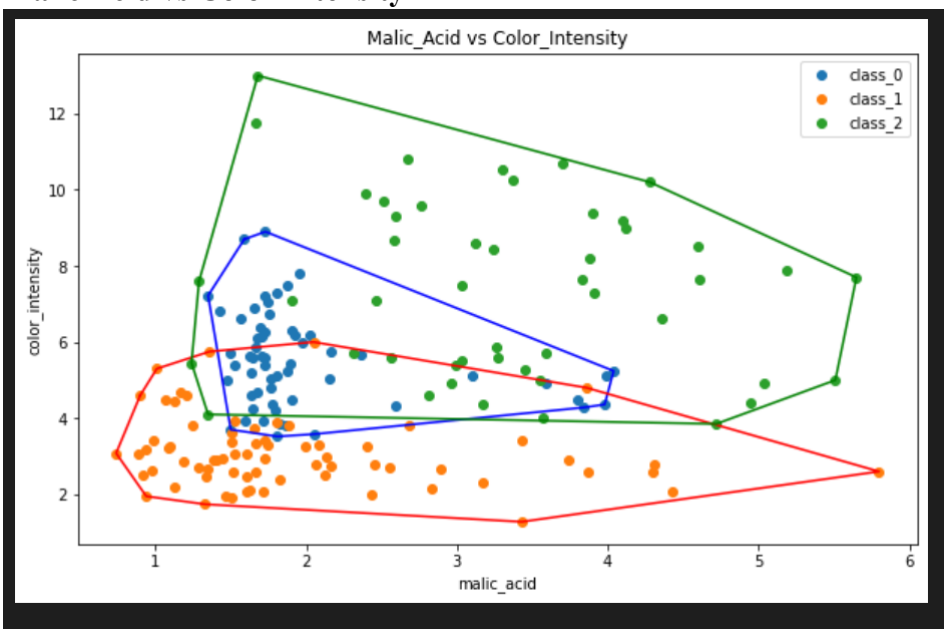


Input Data Wine

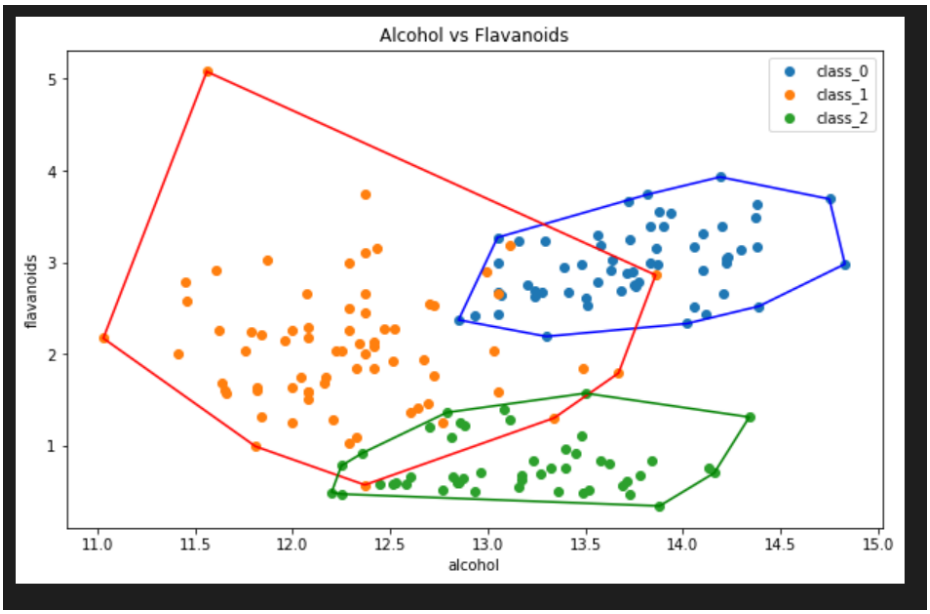
	alcohol	malic_acid	ash	alcalinity_of_ash	magnesium	total_phenols	flavanoids	nonflavanoid_phenols	proanthocyanins	color_intensity	hue	od280/od315
0	14.23	1.71	2.43	15.6	127.0	2.80	3.06	0.28	2.29	5.64	1.04	
1	13.20	1.78	2.14	11.2	100.0	2.65	2.76	0.26	1.28	4.38	1.05	
2	13.16	2.36	2.67	18.6	101.0	2.80	3.24	0.30	2.81	5.68	1.03	
3	14.37	1.95	2.50	16.8	113.0	3.85	3.49	0.24	2.18	7.80	0.86	
4	13.24	2.59	2.87	21.0	118.0	2.80	2.69	0.39	1.82	4.32	1.04	

Output

- **Malic Acid vs Color Intensity**



- **Alcohol vs Flavanoids**



Checklist

No	Spesifikasi	Iya	Tidak
1	Pustaka myConvexHull berhasil dibuat dan tidak ada kesalahan	v	
2	Convex hull yang dihasilkan sudah benar	v	
3	Pustaka myConvexHull dapat digunakan untuk menampilkan convex hull setiap label dengan warna yang berbeda.	v	
4	program dapat menerima input dan menuliskan output untuk dataset lainnya.	v	

Lokasi program dan Referensi

Program bisa di temukan pada : https://github.com/ilhampratama2109/Tucil-2-IF2211

Referensi :

- https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2021-2022/Algoritma-Divide-and-Conquer-(2022)-Bagian4.pdf