LAPORAN

TUGAS KECIL IF2211 STRATEGI ALGORITMA

SEMESTER II TAHUN 2021/2022

DISUSUN OLEH:
Yohana Golkaria Nainggolan 13520053

PROGRAM STUDI TEKNIK INFORMATIKA

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

2022

BAB I ALGORITMA DIVIDE AND CONQUER

Algoritma *convexhull* dibuat dengan menggunakan algoritma *divide and conquer*. Masalah akan dipecah menjadi sub-sub persoalan yang lebih besar. Pertama-tama akan dicari titik terkecil dan terbesar. Kemudian, akan dibentuk garis yang akan membagi dua titik ke sebelah kanan dan kiri. Penentuan titik di sebelah kanan ataupun kiri menggunakan determinan. Titik akan berada di sebelah kiri garis pembagi bila nilai determinan negative dan di sebelah kanan bila determinan positif. Bila tidak ada titik di sebuah himpunan titik, *convex hull* adalah garis pembagi yang telah ditentukan sebelumnya pada himpunan titik tersebut. Jika himpunan tidak kosong, carilah titik maks dimana sudut yang terbentuk antara ketiga titik menghasilkan sudut terbesar. Tentukan kumpulan titik berdasarkan titik maks dan titik awal hingga terbentuk *convex hull.*

BAB II SOURCE PROGRAM

Program ditulis dalam bahasa Python dengan isi program sebagai berikut:

1. convex_hull.py

```
from math import degrees
import numpy as np

def quickhull(array, p1, p2, point): ##fungsi untuk menentukan apakah sebuah
titik berada di sebelah kiri atau kanan garis pembagi
    x1 = array[p1][0]
    y1 = array[p1][1]
    x2 = array[p2][0]
    y2 = array[p2][1]
    x3 = array[point][0]
    y3 = array[point][1]
    return x1*y2 + x3*y1 + x2*y3 - x3*y2 - x2*y1 - x1*y3

def besar_sudut(A, B, C):
    sudut_ba = A - B
    sudut_bc = C - B
    cos_theta = np.dot(sudut_ba,
sudut_bc)/(np.linalg.norm(sudut_ba)*np.linalg.norm(sudut_bc))

    cos_theta = np.clip(cos_theta, -1, 1)

    return np.degrees(np.arccos(cos_theta))


def finding_maxmin(array):
    x = []
    for i in range(len(array)):
        x.append(array[i][0])
    Min = min(x)
    Max = max(x)

    idxMin = 0
```

```
    flag = True
    while idxMin < len(array) and flag:
        if array[idxMin][0] == Min:
            flag = False
        else: idxMin += 1

    idxMax = 0
    flag = True
    while (idxMax < len(array) and flag):
        if array[idxMax][0] == Max:
            flag = False
        else: idxMax += 1

    return (idxMin, idxMax)

def convexhull(vertices):
    arr = np.array(vertices).astype(float)

    p1, p2 = finding_maxmin(arr)

    left = []
    right = []

    for i in range(len(arr)):
        if quickhull(arr, p1, p2, i) > 0 and i != p1 and i != p2:
            left.append(i)
        elif quickhull(arr, p1, p2, i) < 0 and i != p1 and i != p2:
            right.append(i)

    left_recursion = convexhull2(arr, left, p1, p2)
    right_recursion = convexhull2(arr, right, p2, p1)
    return left_recursion + right_recursion


def convexhull2(arr, array, p1, p2): #menentukan pasangan indeks titik pada
garis pembentuk bidang
    if len(array) == 0:
        if p1 != p2:
            return [[p1, p2]]
        else:
            return []
    else:   # kondisi dimana ada titik di sisi garis
        degrees = []
        for i in range(len(array)):
            if p1 != p2 and p1 != array[i] and p2 != array[i]:
                derajat2 = besar_sudut(arr[p2], arr[p1], arr[array[i]])
            else:
                derajat2 = 0
            degrees.append(derajat2)
```

```
        point = array[degrees.index(max(degrees))]

        point1 = []
        for i in range(len(array)):
            if quickhull(arr, p1, point, array[i]) > 0 and array[i] != p1 and
array[i] != p2:
                point1.append(array[i])
        point2 = convexhull2(arr, point1, p1, point)

        point3 = []
        for i in range(len(array)):
            if quickhull(arr, point, p2, array[i]) > 0 and array[i] != p1 and
array[i] != p2:
                point3.append(array[i])
        point4 = convexhull2(arr, point3, point, p2)

        return point2 + point4
```

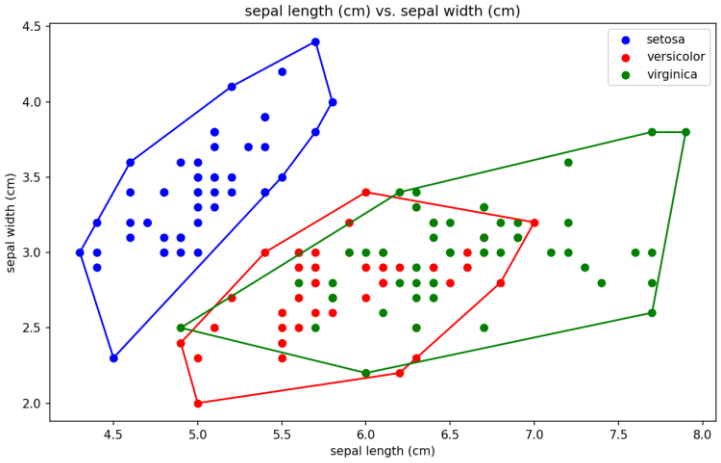
2. test.py

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

def graph(df, data, namagraph, label1, label2, x, y, namalabel):

    plt.figure(figsize = (10, 6))
    size = len(df['Target'].unique())
    colors = ['b','r','g']

    plt.title(namagraph)
    plt.xlabel(label1)
    plt.ylabel(label2)

    for i in range(size):
        bucket = df[df['Target'] == i]
        bucket = bucket.iloc[:,[x,y]].values
        hull = convexhull(bucket)

        plt.scatter(bucket[:, 0], bucket[:, 1], label=namalabel[i],
color=colors[i])
        for simplex in hull:
            plt.plot(bucket[simplex, 0], bucket[simplex, 1], color=colors[i])

    plt.legend()
    plt.show()

def pasangantitik(df,data,x,y):
```

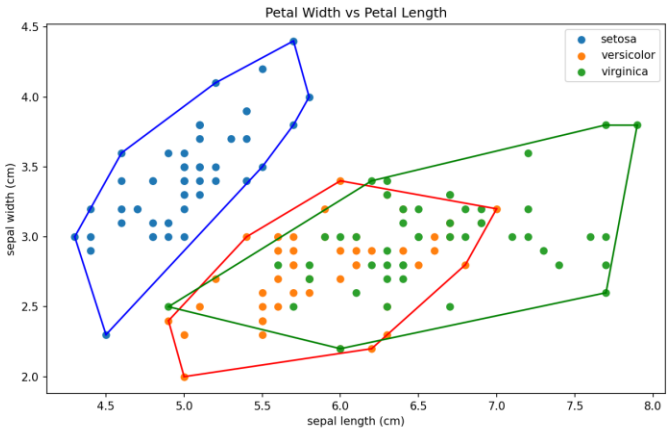
```
    graph(df, data,f"{data.feature_names[x]} vs. {data.feature_names[y]}",
data.feature_names[x],data.feature_names[y],x,y,data.target_names)

data = datasets.load_iris()
df = pd.DataFrame(data.data, columns=data.feature_names)
df['Target'] = pd.DataFrame(data.target)
pasangantitik(df, data, 0, 1)
```

BAB III HASIL UJI PROGRAM

Hasil convex hull yang dibuat



Hasil dari convex hull library python



Untuk pengujian dataset di atas, hasil yang ditunjukan oleh algoritma yang dibuat dan algoritma dari library python memiliki hasil yang sama.

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.	√	

BAB IV ALAMAT PROGRAM

https://github.com/Yohanagn/TucilStima2