LAPORAN TUGAS KECIL II

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

Laporan dibuat untuk memenuhi salah satu tugas mata kuliah

IF2211 Strategi Algoritma

Disusun oleh:

Fayza Nadia 13520001

PROGRAM STUDI TEKNIK INFORMATIKA

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

2022

DAFTAR ISI

Algoritma Divide and Conquer

Program implementasi *convex hull* merupakan salah satu permasalahan yang dapat diselesaikan dengan algoritma *divide and conquer.* Algoritma *divide and conquer* merupakan algoritma yang bekerja dengan membagi suatu permasalahan menjadi lebih kecil. Algoritma ini erat pula digunakan dengan skema rekursif.

Dalam menentukan *convex hull* atau titik-titik terluar, pertama-tama program akan mengurutkan kumpulan titik-titik yang dimiliki secara menaik pada absis. Setelah mengurutkan titik tersebut, didapatkan dua titik ekstrim, yaitu titik paling kiri serta titik paling kanan yang menjadi garis pivot dasar dalam algoritma *divide and conquer*. Garis yang terbentuk tersebut kemudian akan membagi sekumpulan titik tersebut menjadi dua bagian. Tahap ini disebut juga *divide*, yaitu membagi permasalahan menjadi subpermasalahan yang serupa dengan permasalahan awal tetapi hanya berukuran lebih kecil.

Pada masing-masing bagian, program akan menentukan titik dengan jarak terjauh dari garis pivot tersebut. Setelah titik tersebut didapatkan, garis pivot akan menjadi sisi dari sebuah area segitiga. Titik-titik yang terdapat di dalam area segitiga tersebut akan diabaikan. Garis pivot lama pun diabaikan dan digantikan dengan dua garis baru yang terbentuk. Kemudian, hal ini akan diulangi terus-menerus dengan kembali melakukan *divide and conquer* pada titik-titik yang terletak di atas garis pivot. Algoritma akan terus berjalan secara rekursif hingga sampai pada kasus basis, yaitu kasus di mana tidak ditemukan lagi adanya titik di atas garis pivot, dan akan mengembalikan dua titik yang membentuk garis pivot tersebut sebagai titik dari *convex hull*. Tahap ini disebut juga sebagai *conquer*, yaitu menyelesaikan subpermasalahan masing-masing, dalam hal ini secara rekursif, hingga subpermasalahan menjadi sangat kecil dan tidak dapat dibagi lagi.

Terakhir, hasil yang didapatkan dari penyelesaian masing-masing subpermasalahan digabung untuk membentuk sekumpulan resolusi yang menjadi akhir pencarian dari program ini. Dalam program ini, maka akan dihasilkan sekumpulan titik-titik yang membentuk *convex hull* atau titik-titik terluar. Tahap ini merupakan tahap terakhir dan disebut juga sebagai *merge*, yaitu penggabungan solusi dari masing-masing subpermasalahan sehingga membentuk solusi permasalahan utama.

Source Program

Program dibuat menggunakan Bahasa Python dan didekomposisi dalam tiga bagian besar.

a. Bagian *Helper*

Bagian ini berisi sekumpulan fungsi yang digunakan saat mencari *convex hull*.

```python
# Helper
# Below listed functions used on the main convex hull program

import numpy as np

def sorter(data):
    # Key used for array sorting
    x = data[0]
    y = data[1]
    return (x, y)

def checkPosition(p1, p2, px):
    # Checks the position of a point based on a given line consisted of two points
    # (+) means point is located on the top/leftside of the line
    # (-) means point is located on the bottom/rightside of the line
    det = p1[0]*p2[1] + px[0]*p1[1] + p2[0]*px[1] - px[0]*p2[1] - p2[0]*p1[1] - p1[0]*px[1]
    return det

def pointDistance(p1, p2):
    # Returns the distance value between two points
    P1 = np.array([p1[0], p1[1]])
    P2 = np.array([p2[0], p2[1]])
    distance = np.linalg.norm(P2 - P1)
    return distance
```

```python
def pointDistanceMax(sortedData, pLeft, pRight):
    # Returns the maximum distance of a set of points to line made by two points
    maxDistance = 0
    point = []
    for i in range(len(sortedData)):
        distance = pointToLine(pLeft, pRight, sortedData[i])
        if (distance > maxDistance):
            maxDistance = distance
            point = sortedData[i]
    return point

def pointToLine(p1, p2, p3):
    # Returns the distance value of a point p3 to line
    P1 = np.array([p1[0], p1[1]])
    P2 = np.array([p2[0], p2[1]])
    P3 = np.array([p3[0], p3[1]])
    distance = np.abs(np.cross(P2 - P1, P3 - P1) / np.linalg.norm(P2 - P1))
    return distance

def divideTop(p1, pn, data):
    # Returns an array of points lie above/leftside the pivot line
    left = []
    for i in range(len(data)):
        det = checkPosition(p1, pn, data[i])
        if (det > 0) and (data[i] != p1).any() and (data[i] != pn).any():
            left.append(data[i])
    return left
```

```python
def divideBot(p1, pn, data):
    # Returns an array of points lie below/rightside the pivot line
    right = []
    for i in range(len(data)):
        det = checkPosition(p1, pn, data[i])
        if (det < 0) and (data[i] != p1).any() and (data[i] != pn).any():
            right.append(data[i])
    return right

def getIndex(value, data):
    # Returns index of point value based on index from data
    for i in range(len(data)):
        if (value[0] == data[i][0]) and (value[1] == data[i][1]):
            return i
```

b. Bagian *My Convex Hull*

Bagian ini merupakan pustaka dari *convex hull* yang dibuat. Pada bagian ini, fungsi utamanya merupakan ConvexHull yang di dalamnya akan memanggil fungsi FindHull secara rekursif yang kemudian akan digabung hasilnya sebagai bentuk penerapan dari algoritma *divide and conquer.*

```python
# My Convex Hull
# Below listed main function of my own convex hull program

import numpy as np
import pandas as pd

def ConvexHull(data):
    # Returns array of set of indexes from the sorted data
    #       Example: [[0, 2], [2, 3], [3, 0]]
    # The sets of indexes will be plotted as lines shaped as the edge of the convex hull
    # This main function calls the recursive FindHull function in which the divide and conquer algorithm is applied
    hull = []

    # Find the extreme points
    dataSorted = sorted(data, key=sorter)
    p1 = dataSorted[0]
    pn = dataSorted[len(dataSorted) - 1]

    # Divide points based on pivot line
    left = divideTop(p1, pn, data)
    right = divideBot(p1, pn, data)
    leftHull = FindHull(dataSorted, left, p1, pn)
    rightHull = FindHull(dataSorted, right, pn, p1)

    # Conquer
    hull = leftHull + rightHull

    return hull
```

```python
def FindHull(data, filteredData, a, b):
    # Returns array of set of indexes from a small part of the whole data
    #       Example: [[0, 2], [2, 3], [3, 0]]
    # Output will be merged with another FindHull output in the main ConvexHull function
    if (len(filteredData) == 0):        # Base Case
        aIndex = getIndex(a, data)
        bIndex = getIndex(b, data)
        return [[aIndex, bIndex]]
    else:                               # Recursive
        # Farthest point
        c = pointDistanceMax(filteredData, a, b)

        # Divide points based on pivot line
        left1 = divideTop(a, c, filteredData)
        left2 = divideTop(c, b, filteredData)
        S1 = FindHull(data, left1, a, c)
        S2 = FindHull(data, left2, c, b)

        # Conquer
        S = S1 + S2
        return S
```

c. Bagian *Main Program*

Bagian ini merupakan program utama yang meminta *user* untuk memilih *sample dataset* yang ingin ditampilkan visualisasinya.

```
# Main Program

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

# ############
# Interface
print("============ CONVEX HULL ============")
print("Which dataset do you want to use?")
print("1. Iris Plants")
print("2. Breast Cancer")
print("3. Wine Recognition")
dataOption = int(input(">> "))
```

```
print("Which data visualization do you want to see?")
if (dataOption == 1):
    data = datasets.load_iris()
    print("1. Petal Width vs Petal Length")
    print("2. Sepal Width vs Sepal Length")
    setOption = int(input(">> "))
    if (setOption == 1):
        columnIndex = [0, 1]
    elif (setOption == 2):
        columnIndex = [2, 3]
    else:
        print("Input invalid!")
elif (dataOption == 2):
    data = datasets.load_breast_cancer()
    print("1. Radius vs Concavity")
    print("2. Area vs Fractal Dimension")
    setOption = int(input(">> "))
    if (setOption == 1):
        columnIndex = [0, 6]
    elif (setOption == 2):
        columnIndex = [3, 9]
    else:
        print("Input invalid!")
elif (dataOption == 3):
    data = datasets.load_wine()
    print("1. Alcohol vs Proanthocyanins")
    print("2. Malic Acid vs Color Intensity")
    setOption = int(input(">> "))
    if (setOption == 1):
        columnIndex = [0, 8]
    elif (setOption == 2):
        columnIndex = [1, 9]
    else:
        print("Input invalid!")
else:
    print("Input invalid!")
```

```
# ############
# Data Setting
df = pd.DataFrame(data.data, columns=data.feature_names)
df['Target'] = pd.DataFrame(data.target)
```

```
# ############
# Visualization
plt.figure(figsize = (10, 6))
colors = ['b','r','g']

if (dataOption == 1):
    if (setOption == 1):
        plt.title('Petal Width vs Petal Length')
    elif (setOption == 2):
        plt.title('Sepal Width vs Sepal Length')
elif (dataOption == 2):
    if (setOption == 1):
        plt.title('Radius vs Concavity')
    elif (setOption == 2):
        plt.title('Area vs Fractal Dimension')
elif (dataOption == 3):
    if (setOption == 1):
        plt.title('Alcohol vs Proanthocyanins')
    elif (setOption == 2):
        plt.title('Malic Acid vs Color Intensity')

plt.xlabel(data.feature_names[columnIndex[0]])
plt.ylabel(data.feature_names[columnIndex[1]])
for i in range(len(data.target_names)):
    bucket = df[df['Target'] == i]
    bucket = bucket.iloc[:,[columnIndex[0],columnIndex[1]]].values
    hull = ConvexHull(bucket)
    bucket = bucket[np.lexsort((bucket[:, 1], (bucket[:,0])))]
    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()
```

Screenshot Input dan Output

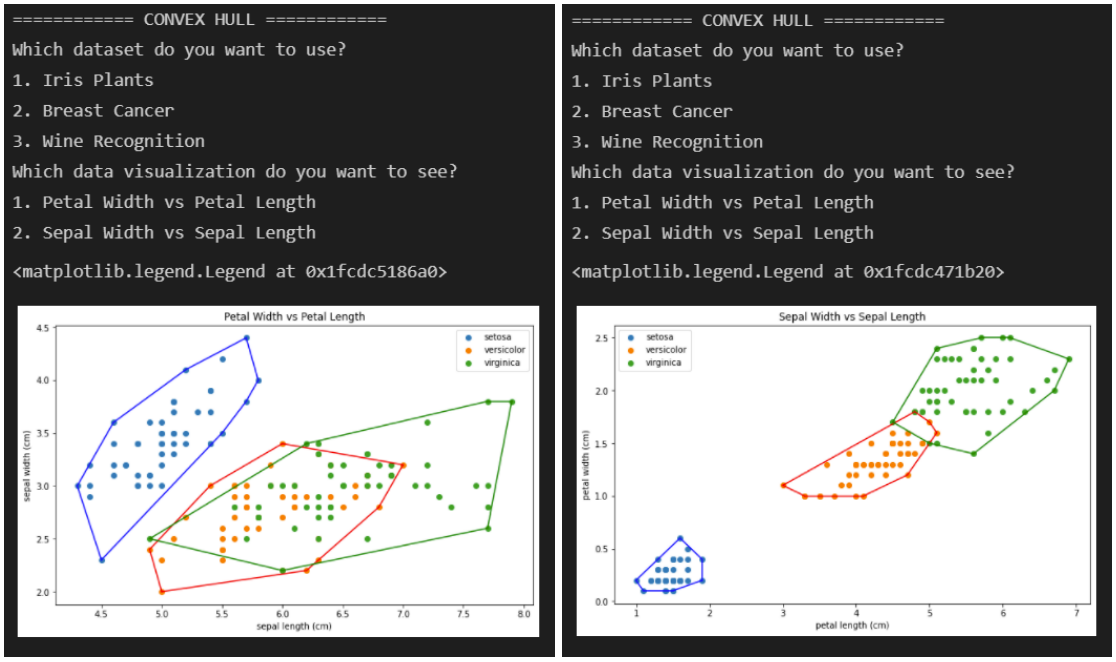
a. Data Iris Plant

Input

	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

Gambar 1
Data Iris Plant

Output



Gambar 2
Visualisasi Data Iris Plant:
Petal Width vs Petal Length (kiri) dan Sepal Width vs Sepal length (kanan)

b. Data Breast Cancer

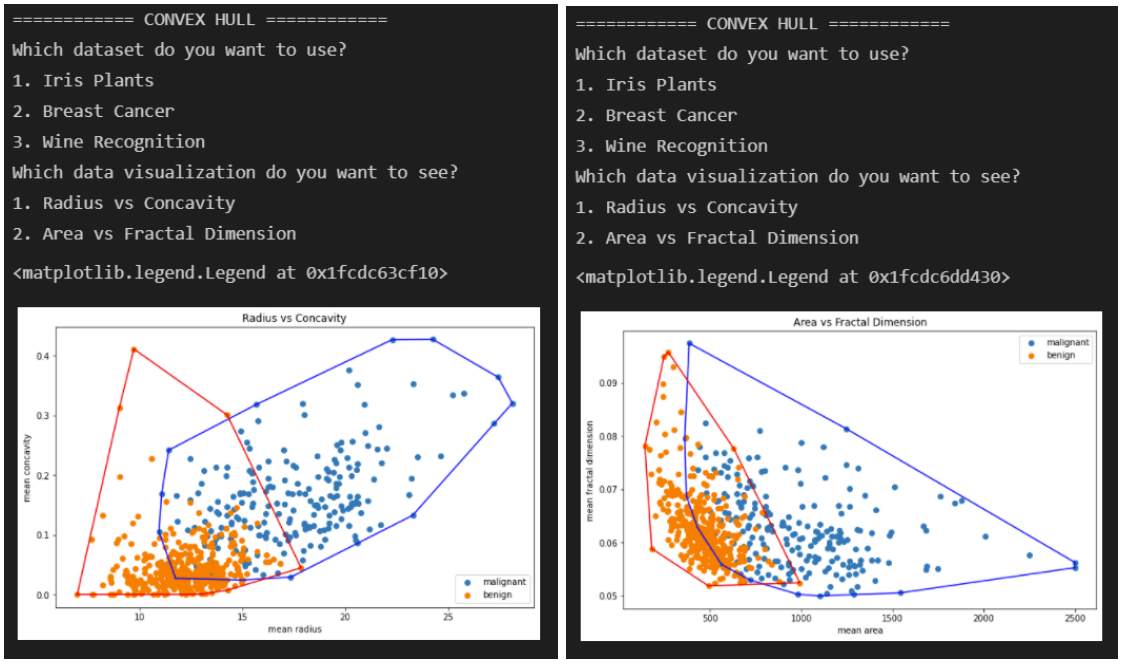
Input

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	mean concavity	mean concave points	mean symmetry	mean fractal dimension	...	worst texture	worst perimeter	worst area	worst smoothness	compa
0	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001	0.14710	0.2419	0.07871	...	17.33	184.60	2019.0	0.1622	
1	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869	0.07017	0.1812	0.05667	...	23.41	158.80	1956.0	0.1238	
2	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1974	0.12790	0.2069	0.05999	...	25.53	152.50	1709.0	0.1444	
3	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2414	0.10520	0.2597	0.09744	...	26.50	98.87	567.7	0.2098	
4	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1980	0.10430	0.1809	0.05883	...	16.67	152.20	1575.0	0.1374	

Gambar 3

Data Breast Cancer

Output



Gambar 4

Visualisasi Data Breast Cancer:

Radius vs Concavity (kiri) dan Area vs Fractal Dimension (kanan)

c. Data Wine Recognition

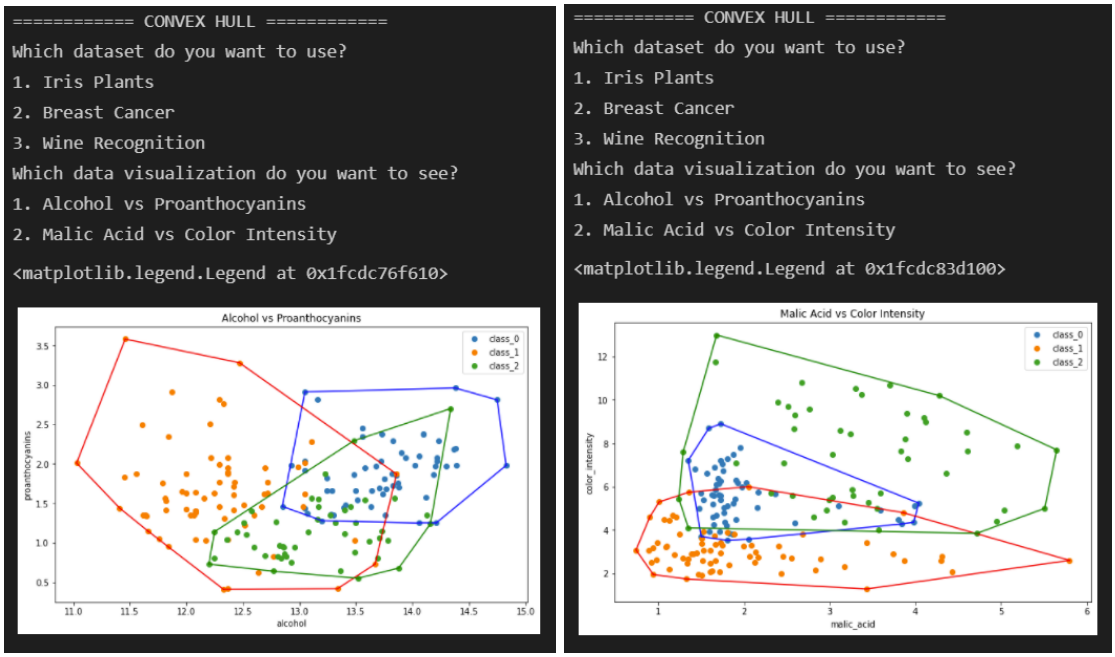
Input

	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	

Gambar 5

Data Wine Recognition

Output



Gambar 6

Visualisasi Data Wine Recognition:

Alcohol vs Proanthocyanins (kiri) dan Malic Acid vs Color Intensity (kanan)

Link Kode Program

https://github.com/fayzanadia/Tucil2_13520001

Checklist

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	✔	

Daftar Referensi

https://informatika.stei.itb.ac.id/~rinaldi.munir/