Laporan

Tugas Kecil Strategi Algoritma

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



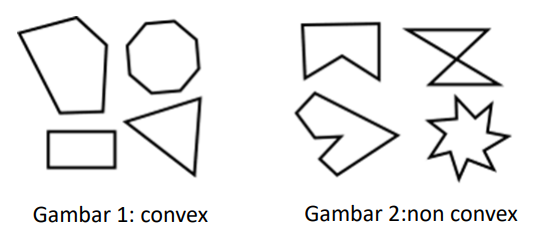
Disusun Oleh:  
Fransiskus Davin Anwari/13520025

**PROGRAM STUDI TEKNIK INFORMATIKA  
SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA  
INSTITUT TEKNOLOGI BANDUNG  
2021**

**ALGORITMA DIVIDE AND CONQUER**

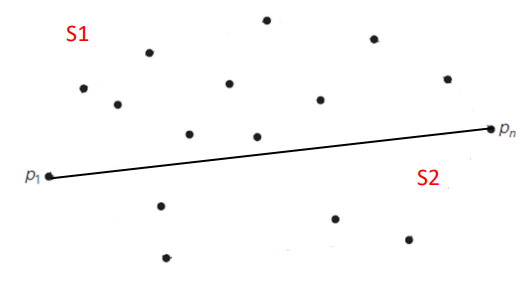
Algoritma Divide and Conquer, adalah suatu algoritma yang membantu menyelesaikan suatu masalah dengan kecepatan yang lebih tinggi dibandingkan algoritma seperti greedy dan brute force. Algoritma ini dibagi menjadi 3 tahap yaitu divide, conquer, dan combine. Pada bagian divide algoritma akan membagi persoalan menjadi beberapa upa-persoalan yang memiliki struktur sama dengan persoalan pertama hanya saja berukuran lebih kecil. Setelah dibagi menjadi ukuran sekecil mungkin, algoritma akan menjalankan bagian conquer yaitu menyelesaikan masing-masing upa-persoalan yang telah terbuat. Lalu setelah semua upa-persoalan diselesaikan baru mulai bagian combine yang menggabungkan semua jawaban upa-persoalan hingga dapat menjawab persoalan awalnya.

Untuk kali ini algoritma divide and conquer digunakan untuk menemukan suatu convex hull dari kumpulan data 2 dimensi. Himpunan titik pada bidang planar disebut convex jika untuk sembarang dua titik pada bidang tersebut (misal p dan q), seluruh segmen garis yang berakhir di p dan q berada pada himpunan tersebut.

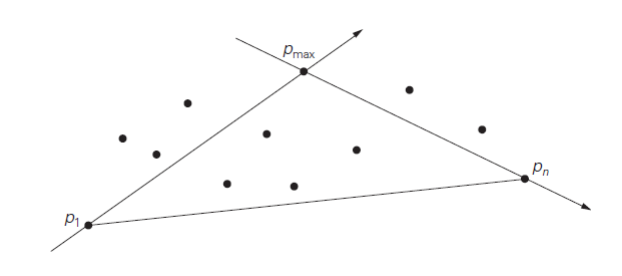


Dari contoh gambar diatas dapat dilihat bahwa gambar 1 adalah contoh poligon yang convex sedangkan gambar 2 adalah poligon yang non-convex. Convex hull dari kumpulan titik adalah himpunan convex terkecil yang mengandung kumpulan titik tersebut.

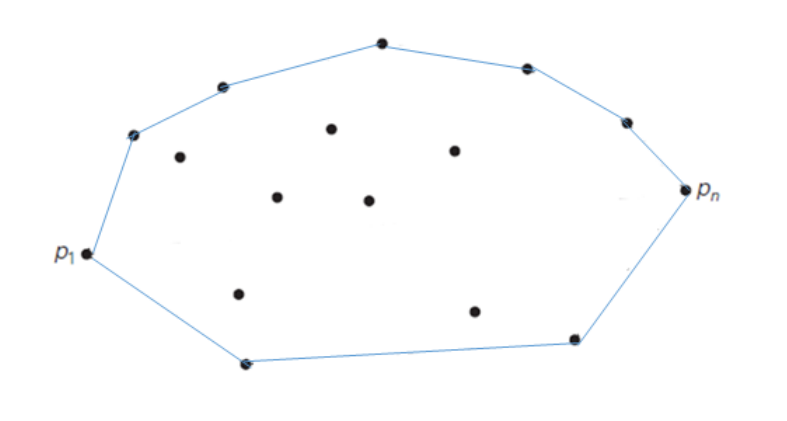
Dengan menggunakan algoritma divide and conquer, kita dapat menemukan suatu convex hull himpunan titik dengan berbagai langkah. Langkah pertama adalah dengan membagi himpunan titik tersebut menjadi 2 bagian, biasanya dilakukan dengan menarik garis dari 2 titik terjauh pada himpunan tersebut, contohnya seperti berikut.



Langkah selanjutnya adalah menyambungkan suatu titik dengan kedua titik tersebut, dengan kondisi bahwa titik tersebut adalah titik yang jaraknya terjauh dari garis yang dibuat kedua titik awal, contoh ilustrasi sebagai berikut.



Kemudian, dilakukan lagi langkah sebelumnya tetapi kali ini dengan titik awal yaitu titik p1 dan pmax pada ilustrasi diatas sehingga kita mendapatkan titik baru lagi, lakukan hal yang sama dengan titik pmax dan titik pn. Ketika tidak terdapat titik diluar garis akhir, maka didapatkan jawaban untuk upa-persoalan bagian atas himpunan, maka dilakukan kembali hal yang sama tetapi kali ini untuk bagian bawah himpunan. Ketika selesai, maka akhirnya akan mendapatkan convex hull yang diilustrasikan seperti berikut.



Algoritma ini diimplementasi ke python dengan membuat berbagai fungsi untuk membagi himpunan, kemudian membusts fungsi untuk mencari titik terjauh dari garis bentukan 2 titik. Hal ini dilakukan berkali-kali hingga mendapat sebuah array yang terdiri dari kumpulan titik terjauh yang kemudian divisualisasikan dengan bantuan matplotlib.

**SOURCE CODE CONVEX HULL PADA PYTHON**

**Kode MyConvexHull.py ( algoritma divide and conquer )**

from cmath import inf

# Mencari titik-titik ekstreme pada suatu himpunan titik

def titikExtreme(bucket):

    nilaimin = inf

    nilaimaks = -inf

    ttkmin = 0

    ttkmaks = 0

    for i in range(len(bucket)):

        if bucket[i][1] < nilaimin:

            nilaimin = bucket[i][1]

            ttkmin = i

        if bucket[i][1] > nilaimaks:

            nilaimaks = bucket[i][1]

            ttkmaks = i

    return (ttkmin,ttkmaks)

# Boolean jika suatu titik di sebelah kiri garis atau tidak

def inLeft(p1,p2,p0):

    result = ( (p2.item(1) - p1.item(1)) \* (p0.item(0) - p2.item(0)) ) - ( (p2.item(0) - p1.item(0)) \* (p0.item(1) - p2.item(1)) )

    return (result < 0)

# Menghitung jarak 1 titik ke garis yang dibentuk oleh 2 titik berbeda

def Jarak(p1,p2,p0):

    return (1/2) \* abs((p1.item(0) - p0.item(0)) \* (p2.item(1)-p1.item(1)) - (p1.item(0) - p2.item(0)) \* (p0.item(1) - p1.item(1)))

# Mencari titik paling jauh pada himpunan titik dari garis bentukan 2 titik

def TitikTerjauh(awal,akhir,bucket):

    jarak = 0

    titik = -1

    notFound = True

    if len(bucket) == 1:

        if bucket == [[]]:

            return -1

        else:

            return 0

    for i in range(len(bucket)):

        if (inLeft(awal,akhir,bucket[i])):

            if jarak < Jarak(awal,akhir,bucket[i]):

                notFound = False

                jarak = Jarak(awal,akhir,bucket[i])

                titik = i

    if notFound:

        return -1

    else:

        return titik

# Memisah Himpunan titik pada 2 titik, untuk yang kiri tinggal dibalik awal dan akhirnya

def splitBucketLeft(awal,akhir,bucket):

    finbucket = [[]]

    for i in range(len(bucket)):

        if inLeft(awal,akhir,bucket[i]):

            if finbucket == [[]]:

                finbucket[0] = bucket[i]

            else:

                finbucket.append(bucket[i])

    return finbucket

# Penyelesaian convex hull setelah di divide

def subConvexHull(awal,akhir,bucket,hull):

    titik = TitikTerjauh(awal,akhir,bucket)

    if titik == -1:

        return

    else:

        subConvexHull(awal,bucket[titik],splitBucketLeft(awal,bucket[titik],bucket),hull)

        hull.append(bucket[titik])

        subConvexHull(bucket[titik],akhir,splitBucketLeft(bucket[titik],akhir,bucket),hull)

# Main Function Convex Hull

def ConvexHull(bucket):

    (titikMin,titikMaks) = titikExtreme(bucket)

    hull = [bucket[titikMin]]

    subConvexHull(bucket[titikMin],bucket[titikMaks],splitBucketLeft(bucket[titikMin],bucket[titikMaks],bucket),hull)

    hull.append(bucket[titikMaks])

    subConvexHull(bucket[titikMaks],bucket[titikMin],splitBucketLeft(bucket[titikMaks],bucket[titikMin],bucket),hull)

    hull.append(bucket[titikMin])

    return hull

**Kode main.py ( visualisasi convex hull untuk 3 dataset )**

#%%

from scipy.spatial import ConvexHull

import pandas as pd

import matplotlib.pyplot as plt

import MyConvexHull as mch

from sklearn import datasets

while (True):

    x = input("Pilih Dataset yang diinginkan (iris, wine, breast\_cancer) :")

    if x == "iris":

        data = datasets.load\_iris()

        #create a DataFrame

        df = pd.DataFrame(data.data, columns=data.feature\_names)

        df['Target'] = pd.DataFrame(data.target)

        print("ATTRIBUTES\n 1. Sepal Length\n 2. Sepal Width\n 3. Petal Length\n 4. Petal Width")

        Attributes= ["Sepal Length","Sepal Width","Petal Length","Petal Width"]

        df.head()

        idx\_x = int(input("Attribute pertama : "))

        idx\_y = int(input("Attribute kedua : "))

        plt.figure(figsize = (10, 6))

        colors = ['b','r','g','p']

        title = Attributes[idx\_x-1] + " vs " + Attributes[idx\_y-1]

        plt.title(title)

        plt.xlabel(data.feature\_names[idx\_x-1])

        plt.ylabel(data.feature\_names[idx\_y-1])

        for i in range(len(data.target\_names)):

            bucket = df[df['Target'] == i]

            bucket = bucket.iloc[:,[idx\_x-1,idx\_y-1]].values

            hull = mch.ConvexHull(bucket)

            # print("HASIL Koordinat : ",hull)

            x, y = zip(\*hull)

            plt.plot(x,y,c=colors[i])

            plt.scatter(bucket[:, 0], bucket[:, 1], label=data.target\_names[i])

            plt.legend()

        plt.figure(figsize = (10, 6))

        colors = ['b','r','g','p']

        title = Attributes[idx\_x-1] + " vs " + Attributes[idx\_y-1] + " (Library Scipy)"

        plt.title(title)

        plt.xlabel(data.feature\_names[idx\_x-1])

        plt.ylabel(data.feature\_names[idx\_y-1])

        for j in range(len(data.target\_names)):

            bucket = df[df['Target'] == j]

            bucket = bucket.iloc[:,[idx\_x-1,idx\_y-1]].values

            hull = ConvexHull(bucket)

            plt.scatter(bucket[:, 0], bucket[:, 1], label=data.target\_names[j])

            for simplex in hull.simplices:

                plt.plot(bucket[simplex, 0], bucket[simplex, 1], colors[j])

            plt.legend()

        break

    elif x == "wine":

        data = datasets.load\_wine()

        #create a DataFrame

        df = pd.DataFrame(data.data, columns=data.feature\_names)

        df['Target'] = pd.DataFrame(data.target)

        print("ATTRIBUTES\n 1. Alcohol\n 2. Malic Acid\n 3. Ash\n 4. Alcalinity of ash\n 5. Magnesium\n 6. Total phenols\n 7. Vlavanoids")

        Attributes= ["Alcohol","Malic Acid","Ash","Alcalinity of ash","Magnesium","Total phenols","Flavanoids"]

        df.head()

        idx\_x = int(input("Attribute pertama : "))

        idx\_y = int(input("Attribute kedua : "))

        plt.figure(figsize = (10, 6))

        colors = ['b','r','g','p']

        title = Attributes[idx\_x-1] + " vs " + Attributes[idx\_y-1]

        plt.title(title)

        plt.xlabel(data.feature\_names[idx\_x-1])

        plt.ylabel(data.feature\_names[idx\_y-1])

        for i in range(len(data.target\_names)):

            bucket = df[df['Target'] == i]

            bucket = bucket.iloc[:,[idx\_x-1,idx\_y-1]].values

            hull = mch.ConvexHull(bucket)

            # print("HASIL Koordinat : ",hull)

            x, y = zip(\*hull)

            plt.plot(x,y,c=colors[i])

            plt.scatter(bucket[:, 0], bucket[:, 1], label=data.target\_names[i])

            plt.legend()

        plt.figure(figsize = (10, 6))

        colors = ['b','r','g','p']

        title = Attributes[idx\_x-1] + " vs " + Attributes[idx\_y-1] + " (Library Scipy)"

        plt.title(title)

        plt.xlabel(data.feature\_names[idx\_x-1])

        plt.ylabel(data.feature\_names[idx\_y-1])

        for j in range(len(data.target\_names)):

            bucket = df[df['Target'] == j]

            bucket = bucket.iloc[:,[idx\_x-1,idx\_y-1]].values

            hull = ConvexHull(bucket)

            plt.scatter(bucket[:, 0], bucket[:, 1], label=data.target\_names[j])

            for simplex in hull.simplices:

                plt.plot(bucket[simplex, 0], bucket[simplex, 1], colors[j])

            plt.legend()

        break

    elif x == "breast\_cancer":

        data = datasets.load\_breast\_cancer()

        #create a DataFrame

        df = pd.DataFrame(data.data, columns=data.feature\_names)

        df['Target'] = pd.DataFrame(data.target)

        print("ATTRIBUTES\n 1. Radius\n 2. Texture\n 3. Perimeter\n 4. Area\n 5. Smoothness\n 6. Compactness\n 7. Concavity")

        Attributes= ["Radius","Texture","Perimeter","Area","Smoothness","Compactness","Concavity"]

        df.head()

        idx\_x = int(input("Attribute pertama : "))

        idx\_y = int(input("Attribute kedua : "))

        plt.figure(figsize = (10, 6))

        colors = ['b','r','g']

        title = Attributes[idx\_x-1] + " vs " + Attributes[idx\_y-1]

        plt.title(title)

        plt.xlabel(data.feature\_names[idx\_x-1])

        plt.ylabel(data.feature\_names[idx\_y-1])

        for i in range(len(data.target\_names)):

            bucket = df[df['Target'] == i]

            bucket = bucket.iloc[:,[idx\_x-1,idx\_y-1]].values

            hull = mch.ConvexHull(bucket)

            # print("HASIL Koordinat : ",hull)

            x, y = zip(\*hull)

            plt.plot(x,y,c=colors[i])

            plt.scatter(bucket[:, 0], bucket[:, 1], label=data.target\_names[i])

            plt.legend()

        plt.figure(figsize = (10, 6))

        colors = ['b','r','g','p']

        title = Attributes[idx\_x-1] + " vs " + Attributes[idx\_y-1] + " (Library Scipy)"

        plt.title(title)

        plt.xlabel(data.feature\_names[idx\_x-1])

        plt.ylabel(data.feature\_names[idx\_y-1])

        for j in range(len(data.target\_names)):

            bucket = df[df['Target'] == j]

            bucket = bucket.iloc[:,[idx\_x-1,idx\_y-1]].values

            hull = ConvexHull(bucket)

            plt.scatter(bucket[:, 0], bucket[:, 1], label=data.target\_names[j])

            for simplex in hull.simplices:

                plt.plot(bucket[simplex, 0], bucket[simplex, 1], colors[j])

            plt.legend()

        break

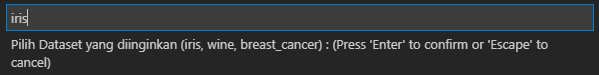
    else:

        print("Gunakan dataset sesuai pilihan!")

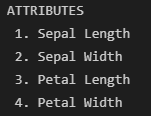
# %%

**SCREENSHOT INPUT DAN OUTPUT**

# **Dataset Iris**

****

Input dataset

****

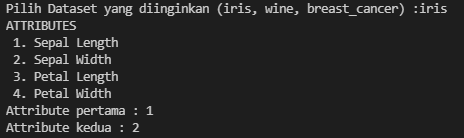
Tampilan Attributes

****

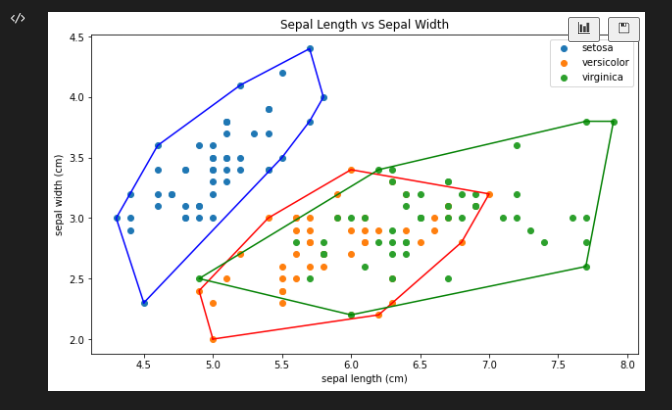
Input Attributes pertama yang ingin dibandingkan



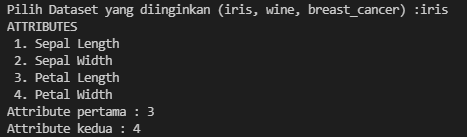
Input Attributes kedua yang ingin dibandingkan

****

Kumpulan input output Sepal Length vs Sepal Width dalam terminal python ( Untuk test case selanjutnya menggunakan tampilan ini agar lebih mudah dilihat )



Hasil Convex Hull Sepal Length vs Sepal Width

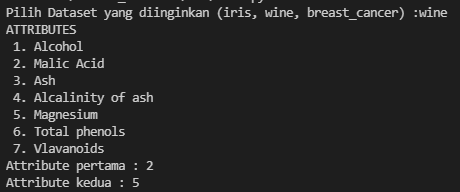


Kumpulan input output Petal Length vs Petal Width dalam terminal python

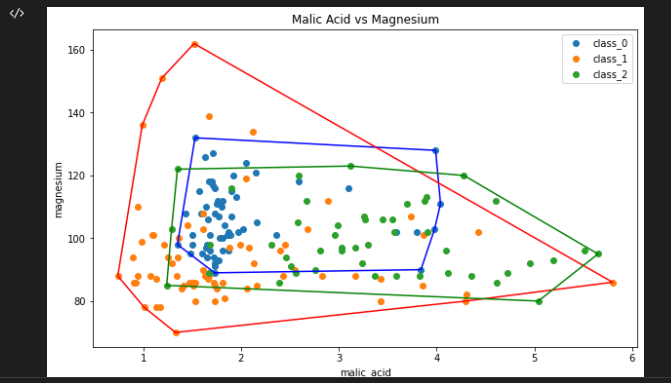


Hasil Convex Hull Petal Length vs Petal Width

# **Dataset wine**

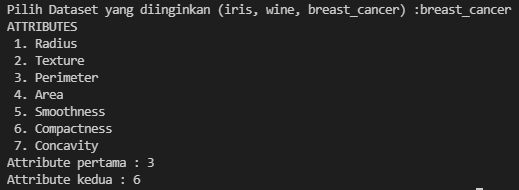


Kumpulan input output Malic Acid vs Magnesium dalam terminal python

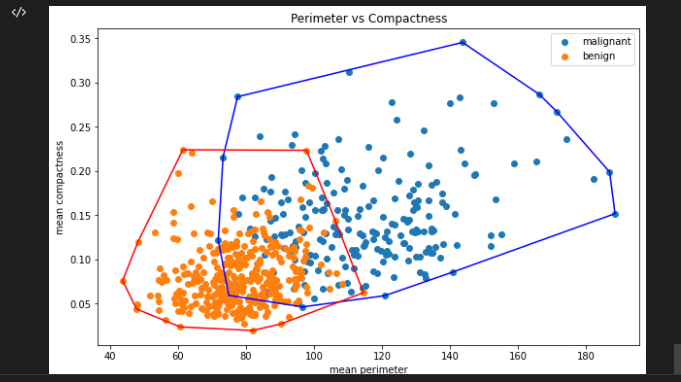


Hasil Convex Hull Malic Acid vs Magnesium

# **Dataset breast\_cancer**



Kumpulan input output Perimeter vs Compactness dalam terminal python



Hasil Convex Hull Perimeter vs Compactness

**Kode Program**

Github: <https://github.com/fdavin/Tucil2_13520025>  
Google Drive: https://drive.google.com/drive/folders/1BfJcotpy9Ah2veojikOLAFhON5dQRtwO?usp=sharing

Cara menjalankan Program dapat dibaca pada README.md

|  |  |  |
| --- | --- | --- |
| 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. | ✓  ( untuk dataset wine dan breast\_cancer saja ) |  |