

# **LAPORAN TUGAS KECIL 2**

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

Mata Kuliah Strategi Algoritma (IF2211)

**KELAS 03**



**Dosen : Dr. Ir. Rinaldi, M.T.**

**DISUSUN OLEH:**

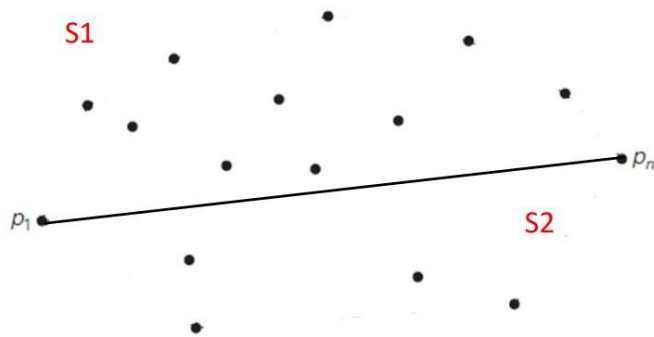
**Rheza Rizqullah Ecaldy (13520060)**

**PROGRAM STUDI TEKNIK INFORMATIKA  
SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA  
INSTITUT TEKNOLOGI BANDUNG  
SEMESTER GENAP TAHUN 2021-2022**

## A. Algoritma Divide and Conquer

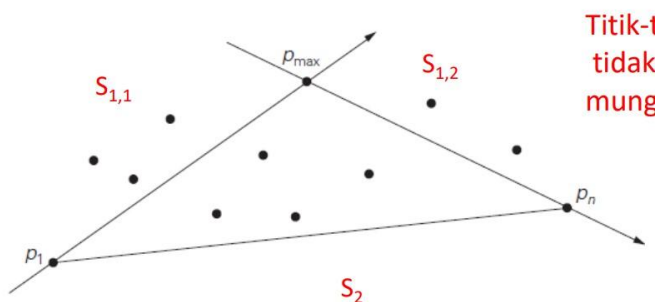
Algoritma yang saya gunakan dalam tucil ini merupakan variasi dari Quick Hull / Chan's Algorithm yang merupakan algoritma divide and conquer dalam problem convex hull. Proses dari algoritma divide and conquer yang saya buat yang terdapat di dalam convexHull.py secara umum adalah sebagai berikut :

1. Pemanggilan fungsi `myConvexHull(points)` dengan `points` merupakan himpunan semua titik akan menyebabkan program menyiapkan suatu array of points bernama `solution` sebagai himpunan hasil `convexHull`. Kemudian, akan dipanggil fungsi `getExtremes(points)` untuk memperoleh 2 titik ekstrim dengan nilai  $x$  terendah dan tertinggi. Sebuah garis ditarik antara kedua titik ini sehingga membagi himpunan titik menjadi 2 bagian, yaitu di atas ( $S1$ ) dan di bawah ( $S2$ ) garis.



2. Akan dipanggil prosedur `quickHull(solution, p1, p2, points, position)` untuk himpunan  $S1$  dan  $S2$  dengan `solution` merupakan himpunan hasil,  $p1$  dan  $p2$  merupakan titik ekstrim, `points` merupakan himpunan titik, dan `position` merupakan posisi dari himpunan (di bawah/di atas garis). Pada prosedur ini, akan dicari titik pada masing-masing himpunan dengan jarak terjauh dari garis  $p1$ - $p2$ . Proses ini akan memanfaatkan fungsi `getFarthestPoint(p1, p2, points, position)`.

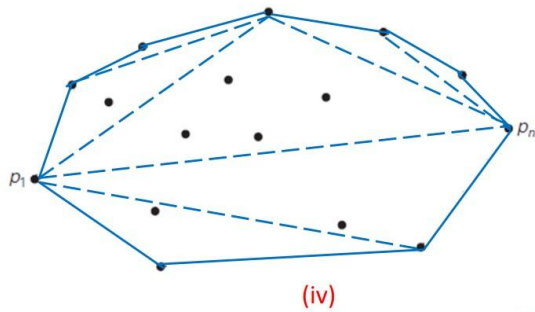
3. Setelah didapatkan titik terjauh, dibentuk segitiga dari kedua titik ekstrim dan titik terjauh yang akan membagi lagi himpunan titik-titik.



4. Kemudian, akan dipanggil prosedur `quickHull(solution, p1, pmax, points, position)` untuk himpunan titik di sebelah kiri segitiga ( $S1,1$ ) dan prosedur `quickHull(solution, pmax, p2, points, position)` untuk himpunan titik di sebelah kanan segitiga ( $S1,2$ ).

5. Proses 2-4 akan terus berulang hingga tidak terdapat lagi titik di sebelah kanan atau kiri dari segitiga yang terbentuk. Apabila kondisi ini terjadi,  $p_1$  dan  $p_2$  dari `quickHull(solution, p1, p2, points, position)` terakhir akan dimasukkan ke dalam himpunan solusi.

6. Himpunan solusi ini akan berisi titik-titik yang merupakan `convexHull` dari himpunan semua titik.



14

## B. Source Code dalam python

### convexHull.py

```
import numpy as np
import math

# mencari distance titik p3 ke garis p1-p2
def getDistance(p1, p2, p3, points) :

    a = points[p1]
    b = points[p2]
    c = points[p3]

    # Rumus yang diturunkan dari rumus jarak titik ke garis
    return (c[1] - a[1])*(b[0] - a[0]) - (c[0] - a[0])*(b[1] - a[1])

# mencari posisi dari titik terhadap garis p1-p2
def positionToLine(p1, p2, p3, points) :
    dist = getDistance(p1, p2, p3, points)

    if dist > 0 : # berada di atas garis
        return 1
    elif dist < 0: # berada di bawah garis
        return -1
    else : # berada di garis
        return 0

# mencari titik dengan distance terjauh dari garis p1-p2
def getFarthestPoint(p1, p2, points, position) :
    idx = -1
    maxDist = 0

    for i in range(len(points)) : # perbandingan secara bruteforce
        dist = getDistance(p1, p2, i, points)

        if (abs(dist) > maxDist) and (positionToLine(p1, p2, i, points) == position) :
            idx = i
            maxDist = abs(dist)
```

```

        return idx

# mencari nilai ekstrim(min, max) berdasarkan nilai x dari himpunan
def getExtremes(points) :
    min = 0
    max = 0

    for i in range(len(points)) : # perbandingan secara bruteforce
        if points[i][0] > points[max][0] :
            max = i
        if points[i][0] < points[min][0] :
            min = i

    return max, min

# fungsi divide and conquer dari
def quickHull(solution, p1, p2, points, position) :
    idxFarthest = getFarthestPoint(p1, p2, points, position)

    if (idxFarthest == -1) : # tidak ada lagi titik di luar convexhull yang menghadap garis p1-p2
        solution.append([p1, p2])
        return

    # membagi himpunan menjadi 2 bagian sesuai sisi segitiga yang terbentuk dari p1, p2, dan idxFarthest
    quickHull(solution, p1, idxFarthest, points, position)
    quickHull(solution, idxFarthest, p2, points, position)

# fungsi utama dari convexhull
def myConvexHull(points) :
    solution = [] # himpunan hasil

    max, min = getExtremes(points)

    # membagi himpunan jadi dua bagian
    quickHull(solution, max, min, points, 1)
    quickHull(solution, max, min, points, -1)

    return solution

```

## main.py

```

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

print("-----")
print("CONVEX HULL 13520060")
print("-----")

# memilih dataset
print()
print("Dataset yang bisa digunakan :")
print("1. Iris")
print("2. Breast Cancer")
print("3. Wine")
print()

datasetsChoice = int(input("Dataset pilihan : "))

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

```

```

# membuat dataframe
df = pd.DataFrame(data.data, columns=data.feature_names)
df['Target'] = pd.DataFrame(data.target)
df.head()

# memilih input x dan y
print("Attributes:")
for i in range (len(data.feature_names)) :
    print(str(i)+". "+str(data.feature_names[i]))
print()

# diasumsikan input x dan y selalu benar
x = int(input("Nilai x pilihan : "))
y = int(input("Nilai y pilihan : "))

# visualisasi hasil ConvexHull
plt.figure(figsize = (10, 6))
colors = ['blue','red','green', 'purple', 'pink', 'brown', 'orange', 'black', 'beige', 'yellow']

title = str(data.feature_names[x]) + " vs " + str(data.feature_names[y])
plt.title(title)
plt.xlabel(data.feature_names[x])
plt.ylabel(data.feature_names[y])

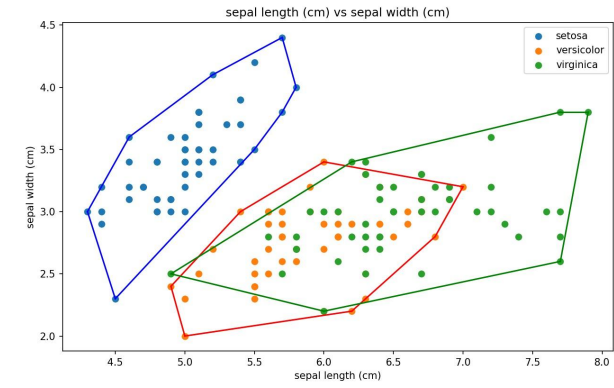
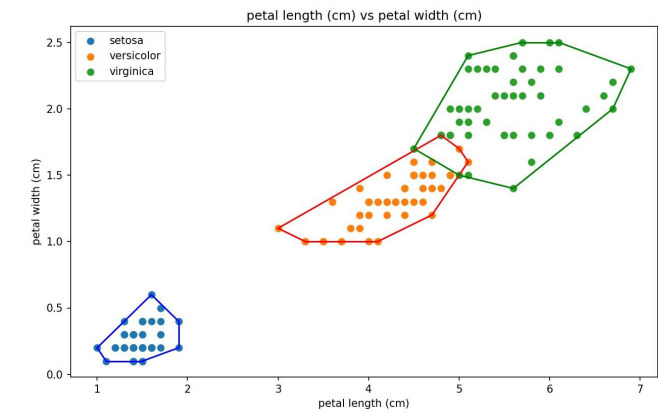
for i in range(len(data.target_names)):
    bucket = df[df['Target'] == i]
    bucket = bucket.iloc[:,[x,y]].values
    hull = myConvexHull(bucket) #fungsi myConvexHull hasil implementasi
    plt.scatter(bucket[:, 0], bucket[:, 1], label=data.target_names[i])
    for simplex in hull:
        plt.plot(bucket[simplex, 0], bucket[simplex, 1], colors[i])

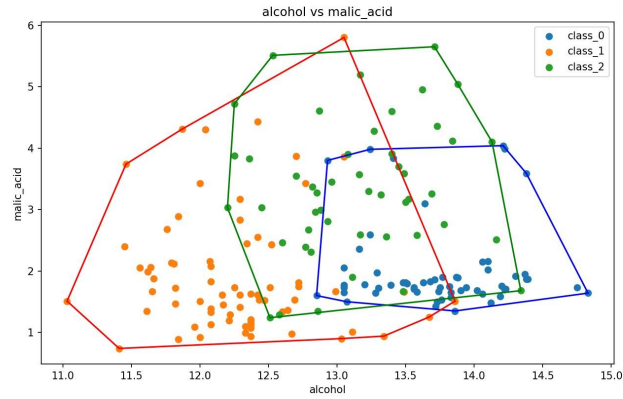
plt.legend()
plt.show()

```

## C. Input dan Output

Nomor	Input/Output
1	 <pre> D:\KULIAH\Sem 4\StimaTucil\Y2&gt;python main.py CONVEX HULL 13520060 ----- Dataset yang bisa digunakan : 1. Iris 2. Breast Cancer 3. Wine  Dataset pilihan : 1 Attributes: 0. sepal length (cm) 1. sepal width (cm) 2. petal length (cm) 3. petal width (cm)  Nilai x pilihan : 0 Nilai y pilihan : 1 </pre>

	
2	<div><pre>CONVEX HULL 13520060 ----- Dataset yang bisa digunakan : 1. Iris 2. Breast Cancer 3. Wine  Dataset pilihan : 1 Attributes: 0. sepal length (cm) 1. sepal width (cm) 2. petal length (cm) 3. petal width (cm)  Nilai x pilihan : 2 Nilai y pilihan : 3 File: "C:\Users\user\Desktop\convex_hull_FUDS_Fortran\CHUD_FUDS"</pre></div> 
3	<div><pre>Dataset yang bisa digunakan : 1. Iris 2. Breast Cancer 3. Wine  Dataset pilihan : 3 Attributes: 0. alcohol 1. malic_acid 2. ash 3. alcalinity_of_ash 4. magnesium 5. total_phenols 6. flavanoids 7. nonflavanoid_phenols 8. proanthocyanins 9. color_intensity 10. hue 11. od280/od315_of_diluted_wines 12. proline  Nilai x pilihan : 0 Nilai y pilihan : 1</pre></div>



4

CONVEX HULL 13520060

Dataset yang bisa digunakan :

1. Iris
2. Breast Cancer
3. Wine

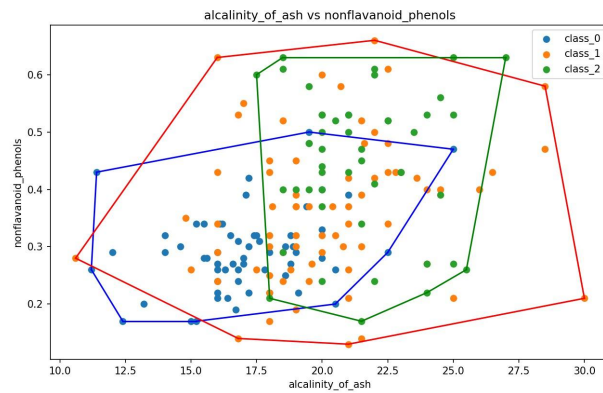
Dataset pilihan : 3

Attributes:

0. alcohol
1. malic\_acid
2. ash
3. alcalinity\_of\_ash
4. magnesium
5. total\_phenols
6. flavanoids
7. nonflavanoid\_phenols
8. proanthocyanins
9. color\_intensity
10. hue
11. od280/od315\_of\_diluted\_wines
12. proline

Nilai x pilihan : 3

Nilai y pilihan : 7



5

Dataset yang bisa digunakan :

1. Iris
2. Breast Cancer
3. Wine

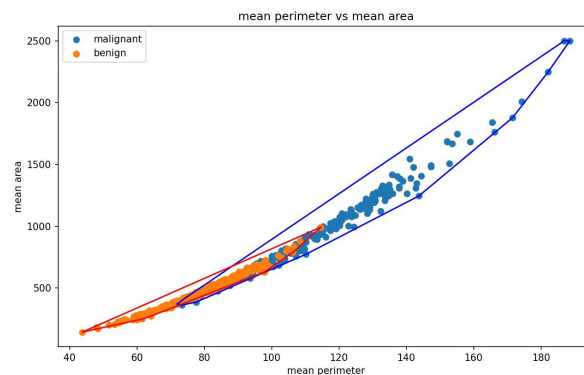
Dataset pilihan : 2

Attributes:

0. mean radius
1. mean texture
2. mean perimeter
3. mean area
4. mean smoothness
5. mean compactness
6. mean concavity
7. mean concave points
8. mean symmetry
9. mean fractal dimension
10. radius error
11. texture error
12. perimeter error
13. area error
14. smoothness error
15. compactness error
16. concavity error
17. concave points error
18. symmetry error
19. fractal dimension error
20. worst radius
21. worst texture
22. worst perimeter
23. worst area
24. worst smoothness
25. worst compactness
26. worst concavity
27. worst concave points
28. worst symmetry
29. worst fractal dimension

Nilai x pilihan : 2

Nilai y pilihan : 3



## D. Link Google Drive (Source Code)

[https://drive.google.com/drive/folders/1mIY215UW4W4jTgqjDb\\_Fd96e4u7vnrrj?usp=sharing](https://drive.google.com/drive/folders/1mIY215UW4W4jTgqjDb_Fd96e4u7vnrrj?usp=sharing)

## E. Link github

[EdgarAllanPoo/Tucil2\\_13520060: Program yang menghasilkan convex hull dari himpunan titik datasets iris, breast cancer, dan wine dari scikit \(github.com\)](#)



## F. Tabel Check List

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. <b>Bonus:</b> program dapat menerima input dan menuliskan output untuk dataset lainnya.	√	