

Laporan Tugas Kecil 2 IF2211 Strategi Algoritma
Semester II Tahun Akademik 2021/2022

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



Disusun oleh:
Ahmad Alfani Handoyo
13520023
K-02

Program Studi S1 Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung

A. Algoritma *Divide and Conquer* untuk menyelesaikan *Convex Hull*

Algoritma yang digunakan untuk menyelesaikan *convex hull* menggunakan pendekatan *divide and conquer*. Algoritma *divide and conquer* pada umumnya membagi suatu persoalan menjadi beberapa sub-persoalan yang mirip dengan persoalan awalnya namun pada skala yang lebih kecil. Masing-masing sub-persoalan tersebut diselesaikan. Bila sudah berukuran kecil diselesaikan langsung dan bila masih berukuran besar diselesaikan secara rekursif dengan basis sub-persoalan terkecil. Terakhir, solusi dari masing-masing sub-persoalan digabung agar membentuk solusi untuk persoalan awalnya.

Pendekatan algoritma *divide and conquer* yang digunakan dalam menyelesaikan masalah *convex hull* mirip dengan *gift wrapping algorithm* atau juga dikenal sebagai *Jarvis march*. Langkah-langkah algoritma adalah sebagai berikut. $p(x, y)$ adalah suatu titik, $p_m p_n$ adalah suatu garis yang dihubungkan oleh titik p_m dan p_n , S adalah suatu himpunan titik p , dan $p_m p_n p_o$ suatu sudut dengan garis $p_m p_n$ dan $p_n p_o$ yang mengapit titik p_n :

1. Cari titik p_l yaitu titik dengan nilai koordinat x terkecil dan p_n yaitu titik dengan nilai koordinat x terbesar. $p_l p_n$ menjadi suatu garis yang membelah S himpunan titik pada bidang 2 dimensi
2. Bagi titik-titik menjadi S_l himpunan titik di sebelah kiri $p_l p_n$ dan S_r himpunan titik di sebelah kanan $p_l p_n$. Penentuan kiri/kanan suatu titik p_i ke $p_l p_n$ dapat ditentukan dengan hasil determinan:

$$\begin{vmatrix} x_1 & y_1 & 1 \\ x_i & y_i & 1 \\ x_n & y_n & 1 \end{vmatrix} = x_1 y_i + x_n y_1 + x_i y_n - x_n y_i - x_i y_1 - x_1 y_n$$

Bila hasil determinan positif p_i terletak pada kiri garis $p_l p_n$ dan kanan bila negatif. Bila hasil determinan adalah 0 maka p_i terletak persis pada garis $p_l p_n$ sehingga tidak perlu dimasukkan ke S_l dan S_r .

3. Untuk sebuah himpunan titik (misal S_l), bila tidak ada titik pada S_l maka kembalikan garis $p_l p_n$ sebagai pembentuk *convex hull* pada S_l . Jika S_l tidak kosong, cari titik p_{max} dengan sudut $p_{max} p_l p_n$ terbesar.
4. Tentukan kumpulan titik di sebelah kiri $p_l p_{max}$ menjadi $S_{l,1}$ dan kumpulan titik di sebelah kiri $p_{max} p_n$ menjadi $S_{l,2}$.
5. Ulangi langkah 3 dan 4 untuk bagian S_r hingga bagian kirinya kosong tetapi dengan menukar p_l dengan p_n untuk segala perhitungan pada langkah-langkah sebelumnya.
6. Kembalikan pasangan titik yang dihasilkan

Algoritma tersebut dapat digolongkan sebagai algoritma *divide and conquer* karena terdapat repetisi pemecahan sub-persoalan pada langkah ke-3 dan ke-4 hingga mencapai basis yaitu himpunan S yang kosong. Dapat dilihat sub-persoalan mirip dengan persoalan awal yaitu dengan membagi suatu himpunan titik menjadi dua berdasarkan letaknya pada kiri/kanan garis. Hanya saja, langkah pertama sedikit berbeda karena dicari himpunan titik kiri dan kanan dari garis sedangkan pada langkah-langkah rekursif berikutnya hanya dicari himpunan titik kiri dari garis.

B. Source Code Program

Program ditulis pada bahasa Python secara modular, terdiri atas tiga modul serta satu file *main.py* pada folder *src*.

1. geometry.py

Berisi fungsi-fungsi untuk membantu perhitungan geometris pada algoritma *convex hull* yaitu perhitungan letak kiri/kanan titik dari garis (*kiriKananGaris*), sudut dari dua titik yang mengapit satu titik (*sudut*), dan titik-titik dengan koordinat-x minimal dan maksimal (*minMax*).

```
import numpy as np

def kiriKananGaris(arr, p1, pn, px):
    '''Memeriksa apakah sebuah titik berada di sebelah kiri atau
    kanan garis menggunakan penentuan determinan. Bila sebelah
    kiri maka hasil positif, kanan negatif, di garis 0.'''

    x1 = arr[p1][0]; y1 = arr[p1][1]; x2 = arr[pn][0]; y2 = arr[pn][1];
    x3 = arr[px][0]; y3 = arr[px][1]

    return x1*y2 + x3*y1 + x2*y3 - x3*y2 - x2*y1 - x1*y3

def sudut(a, b, c):
    '''Mengembalikan sudut dari segitiga ABC dengan B sudut yang
    terapan menggunakan definisi dot product.'''

    # cos theta = ba.bc / |ba|*|bc|
    ba = a - b; bc = c - b
    cosine = np.dot(ba, bc)/(np.linalg.norm(ba)*np.linalg.norm(bc))

    # Clipping cosine due to floating rounding inaccuracy
    cosine = np.clip(cosine, -1, 1)

    return np.degrees(np.arccos(cosine))
```

```

def minMax(arr_np):
    """Mengembalikan tuple indeks titik dengan nilai x minimal
    dan maksimal."""

    # Mencari nilai x minimal atau maksimal
    x_coordinates = []
    for i in range(len(arr_np)):
        x_coordinates.append(arr_np[i][0])
    x_min = min(x_coordinates)
    x_max = max(x_coordinates)

    # Menentukan indeks dengan x minimal
    min_index = 0
    found = True
    while min_index < len(arr_np) and found:
        if arr_np[min_index][0] == x_min:
            found = False
        else: min_index += 1

    # Menentukan indeks dengan x maksimal
    max_index = 0
    found = True
    while (max_index < len(arr_np) and found):
        if arr_np[max_index][0] == x_max:
            found = False
        else: max_index += 1

    return (min_index, max_index)

```

2. myConvexHull.py

Berisi library yang melakukan algoritma *divide and conquer* convex hull dengan iterasi pertama dilakukan pada fungsi *myConvexHull* dan tahap-tahap rekursif selanjutnya pada fungsi *myConvexHullRecursive*. Kedua fungsi mengembalikan *list* yang berisi *list* garis *convex hull* yang direpresentasikan dengan indeks dari kedua titik pembentuk garis.

```

from geometry import *

def myConvexHull(arr):
    """Mengembalikan hasil pasang titik garis pembentuk bidang
    convex hull. Fungsi juga langkah pertama algoritma divide
    and conquer convex hull dengan membagi titik-titik menjadi
    kiri/kanan garis minimal/maksimal x."""

```

```

arr_used = np.array(arr).astype(float)

# Cari p1 dan pn terluar
p1, pn = minMax(arr_used)

# Bagi menjadi kiri dan kanan garis p1, pn
left_arr = []; right_arr = []
for i in range(len(arr_used)):
    if kiriKananGaris(arr_used, p1, pn, i) > 0 and i != p1 and i !=
pn:
        left_arr.append(i)
    elif kiriKananGaris(arr_used, p1, pn, i) < 0 and i != p1 and i
!= pn:
        right_arr.append(i)

# Masuk fungsi rekursi convex hull kiri garis
kiri = myConvexHullRecursive(arr_used, left_arr, p1, pn)
# Masuk fungsi rekursi convex hull kanan garis
kanan = myConvexHullRecursive(arr_used, right_arr, pn, p1)
return kiri + kanan

def myConvexHullRecursive(arr_used, arr, p1, pn):
    """Fungsi rekursif penentu pasangan indeks titik garis
    pembentuk bidang convex hull pada sisi garis dengan arr
    array titik pada sisi garis."""

    # Basis: tidak ada titik di sisi garis
    if len(arr) == 0:
        # Memastikan pasangan titik tidak sama
        if p1 != pn:
            return [[p1, pn]]
        else:
            return []
    else: # Rekursi: terdapat titik di sisi garis
        # Menentukan besar sudut p1-titik-pn untuk setiap titik
        degrees = []
        for i in range(len(arr)):
            # Memastikan ketiga titik sudut tidak sama agar
            # tidak error pembagian dibagi 0
            if p1 != pn and p1 != arr[i] and pn != arr[i]:
                tempdeg = sudut(arr_used[pn], arr_used[p1],
arr_used[arr[i]])
            else:
                tempdeg = 0
            degrees.append(tempdeg)
        # Menentukan indeks titik dengan derajat terbesar
        px = arr[degrees.index(max(degrees))]

```

```

        # Rekursi titik-titik pada kiri garis p1-titik
        p1px = []
        for i in range(len(arr)):
            if kiriKananGaris(arr_used, p1, px, arr[i]) > 0 and arr[i]
!= p1 and arr[i] != pn:
                p1px.append(arr[i])
            p1titik = myConvexHullRecursive(arr_used, p1px, p1, px)

        # Rekursi titik-titik pada kiri garis titik-pn
        pxpn = []
        for i in range(len(arr)):
            if kiriKananGaris(arr_used, px, pn, arr[i]) > 0 and arr[i]
!= p1 and arr[i] != pn:
                pxpn.append(arr[i])
            titikpn = myConvexHullRecursive(arr_used, pxpn, px, pn)

        return p1titik + titikpn

```

3. graph.py

Berisi prosedur *graph* yang menggambarkan, menampilkan, dan menyimpan grafik dari *convex hull*. Terdapat pula fungsi *plotColor* untuk menggenerasikan warna dari titik dan garis grafik sesuai dengan jumlah label yang ada di data.

```

import matplotlib.pyplot as plt
from myConvexHull import myConvexHull
import random

def plotColor(n):
    '''Generasi warna sebanyak n'''
    colors = ['b', 'r', 'g', 'c', 'm', 'y', 'k']
    if n > len(colors):
        for i in (range(n-len(colors))):
            r = random.random()
            g = random.random()
            b = random.random()
            colors.append((r, g, b))
    return colors

def graph(df, data, graphtitle, xlabel, ylabel, xrow, yrow,
labelnames):
    '''Memvisualisasikan grafik hasil convex hull dengan memplot
titik-titik dan garis convex hull dan menyimpannya pada folder
test'''

    plt.figure(figsize = (10, 6))
    labelsize = len(df['label'].unique())

```

```

colors = plotColor(labelsize)

plt.title(graphtitle)
plt.xlabel(xlabel)
plt.ylabel(ylabel)

for i in range(labelsize):
    bucket = df[df['label'] == i]
    bucket = bucket.iloc[:,[xrow,yrow]].values

    # Implementasi algoritma divide and conquer Convex Hull
    hull = myConvexHull(bucket)

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

plt.legend()
print(f"\nShowing convex hull graph of {xlabel} vs. {ylabel}")
print("Make sure to close the graph pop-up so the program can
continue!")
fig1 = plt.gcf()
plt.show()

outputcondition = input("\nWould you like to save the convex hull
graph? (Y/N): ")

if outputcondition.upper() == "Y":
    output = input("\nOutput file name: ") + ".png"
    fig1.savefig('test/' + output)
    print(f"Your convex hull graph of {data.feature_names[xrow]}
vs. {data.feature_names[yrow]} at {output} has successfully been made
at folder test!")
plt.close('all')

```

4. main.py

File utama pada program memiliki urutan prosedur:

1. Membentuk folder *test* bila belum ada
2. Memilih *dataset* (iris/wine/digits/breast cancer wisconsin) yang digunakan
3. Memilih pasangan atribut *dataset* yang digunakan sebagai x dan y *convex hull*
4. Menerapkan algoritma *divide and conquer* convex hull pada pasangan atribut
5. Menggambar dan menampilkan grafik convex hull yang kemudian dapat disimpan pada suatu file pada folder *test*.

Terdapat prosedur pembantu *attributePair* untuk memproses pilihan dataset dan pasangan attribute user pada fungsi *graph*.

```
import os
import pandas as pd
from graph import graph

def attributePair(df, data, x, y):
    '''Prosedur perantara pembantu memilih pasangan attribute'''

    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)

if __name__ == "__main__":
    # Make test folder for graph output
    if not os.path.exists('test'):
        os.makedirs('test')

    print("=====")
    print("    WELCOME TO AFAN'S CONVEX HULL GENERATOR!")
    print("=====\\n")
    print("Made by:\\n13520023\\nAhmad Alfani Handoyo")

    condition = True
    while condition:
        print("\\nChoose dataset to use:")
        print("1. Iris dataset")
        print("2. Wine dataset")
        print("3. Digits dataset")
        print("4. Breast cancer wisconsin dataset")
        dataset_choose = int(input("Your input (1-4): "))

        if 1 <= dataset_choose <= 4:
            from sklearn import datasets
            if dataset_choose == 1:
                # Load dataset iris
                data = datasets.load_iris()
            elif dataset_choose == 2:
                # Load dataset wine
                data = datasets.load_wine()
            elif dataset_choose == 3:
                # Load dataset digits
                data = datasets.load_digits()
            elif dataset_choose == 4:
                # Load dataset breast cancer wisconsin
                data = datasets.load_breast_cancer()
```



```

df = pd.DataFrame(data.data, columns=data.feature_names)
df['label'] = pd.DataFrame(data.target)
attributeLen = len(data.feature_names)

# Pilih pasangan atribut untuk dijadikan koordinat x,y
print("\nChoose attribute for x:")
for i in range(attributeLen):
    print(f"{i+1}. {data.feature_names[i]}")
x = int(input(f"Your input (1-{attributeLen}): "))

    print(f"\nChoose attribute for y (make sure it is different
from {x}. {data.feature_names[x-1]}):")
    for i in range(attributeLen):
        print(f"{i+1}. {data.feature_names[i]}")
    y = int(input(f"Your input (1-{attributeLen}): "))

    if 1 <= x <= attributeLen and 1 <= y <= attributeLen and x
!= y:
        attributePair(df, data, x-1, y-1)
    else:
        print(f"Invalid attribute pair input x: {x} y: {y}!")
else:
    print("\nInvalid dataset input (1-4)!")

    loop = input("\nDo you want to try another dataset or pair of
attributes? (Y/N): ")

    if loop.upper() != "Y":
        condition = False
        print("\n=====
=====")
        print("    THANK YOU FOR USING AFAN'S CONVEX HULL
GENERATOR!")
        print("=====
=====\\n")

```

C. Screenshot Hasil Input dan Output Program

Diambil 8 kasus dari keempat dataset *toy* (iris, wine, digits, dan breast cancer wisconsin) yang tersedia dari *library* scikit-learn. Program dijalankan dengan *command* “*python src/main.py*” pada folder *root* dari *repository*.

1. Dataset iris, petal length (cm) vs. petal width (cm)

```
=====
WELCOME TO AFAN'S CONVEX HULL GENERATOR!
=====

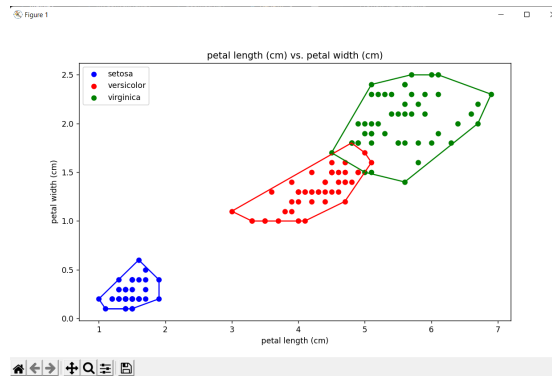
Made by:
13520023
Ahmad Alfani Handoyo

Choose dataset to use:
1. Iris dataset
2. Wine dataset
3. Digits dataset
4. Breast cancer wisconsin dataset
Your input (1-4): 1

Choose attribute for x:
1. sepal length (cm)
2. sepal width (cm)
3. petal length (cm)
4. petal width (cm)
Your input (1-4): 3

Choose attribute for y (make sure it is different from 3. petal length (cm)):
1. sepal length (cm)
2. sepal width (cm)
3. petal length (cm)
4. petal width (cm)
Your input (1-4): 4

Showing convex hull graph of petal length (cm) vs. petal width (cm)
Make sure to close the graph pop-up so the program can continue!
```



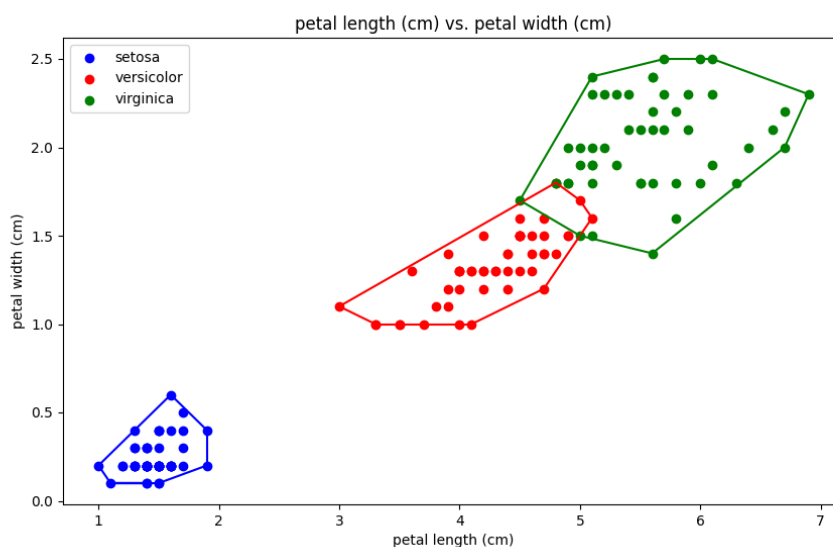
```
Would you like to save the convex hull graph? (Y/N): y

Output file name: testcase1
Your convex hull graph of petal length (cm) vs. petal width (cm) at testcase1.png has suc
cessfully been made at folder test!

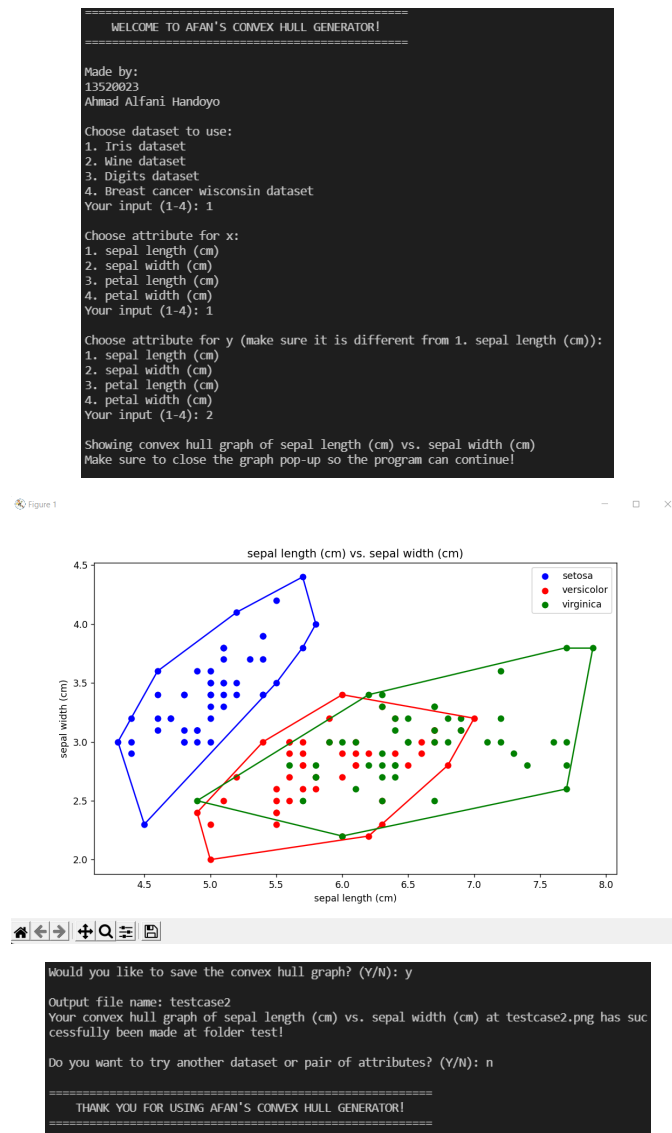
Do you want to try another dataset or pair of attributes? (Y/N): n

=====
THANK YOU FOR USING AFAN'S CONVEX HULL GENERATOR!
=====
```

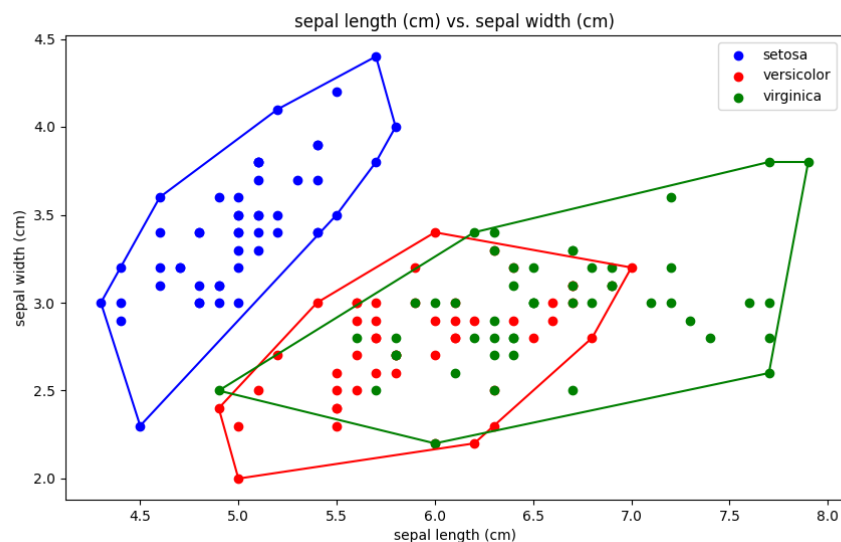
Output graf pada gambar “testcase1.png” pada folder test:



2. Dataset iris, sepal-length vs. sepal-width



Output graf pada gambar “testcase2.png” pada folder test:



3. Dataset wine, magnesium vs. proanthocyanins

```
=====
WELCOME TO AFAN'S CONVEX HULL GENERATOR!
=====

Made by:
13520023
Ahmad Alfani Handoyo

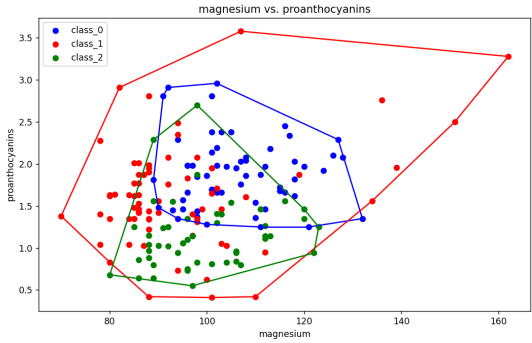
Choose dataset to use:
1. Iris dataset
2. Wine dataset
3. Digits dataset
4. Breast cancer wisconsin dataset
Your input (1-4): 2

Choose attribute for x:
1. alcohol
2. malic_acid
3. ash
4. alcalinity_of_ash
5. magnesium
6. total_phenols
7. flavanoids
8. nonflavanoid_phenols
9. proanthocyanins
10. color_intensity
11. hue
12. od280/od315_of_diluted_wines
13. proline
Your input (1-13): 5

Choose attribute for y (make sure it is different from 5. magnesium):
1. alcohol
2. malic_acid
3. ash
4. alcalinity_of_ash
5. magnesium
6. total_phenols
7. flavanoids
8. nonflavanoid_phenols
9. proanthocyanins
10. color_intensity
11. hue
12. od280/od315_of_diluted_wines
13. proline
Your input (1-13): 9

Showing convex hull graph of magnesium vs. proanthocyanins
Make sure to close the graph pop-up so the program can continue!
```

Figure 1

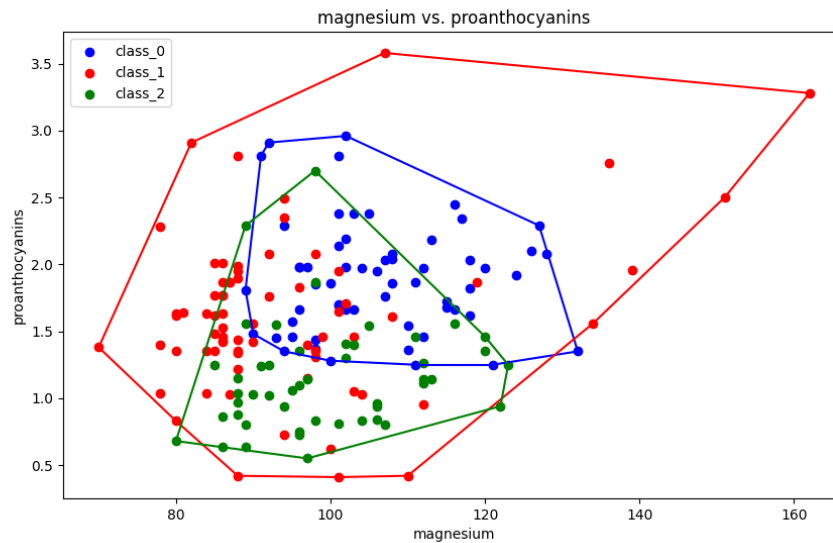


```
Would you like to save the convex hull graph? (Y/N): y
Output file name: testcase3
Your convex hull graph of magnesium vs. proanthocyanins at testcase3.png has successfully
been made at folder test!

Do you want to try another dataset or pair of attributes? (Y/N): n

=====
THANK YOU FOR USING AFAN'S CONVEX HULL GENERATOR!
=====
```

Output graf pada gambar “testcase3.png” pada folder test:



4. Dataset wine, color_intensity vs. alcohol

```
=====
WELCOME TO AFAN'S CONVEX HULL GENERATOR!
=====

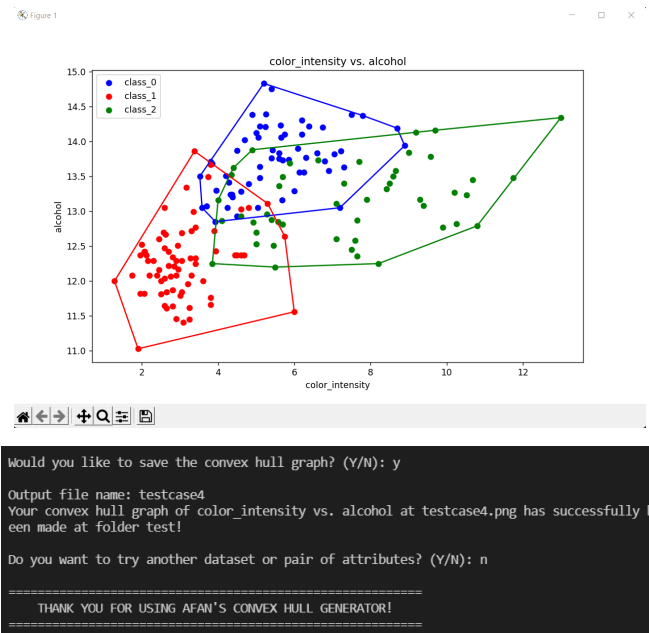
Made by:
13520023
Ahmad Alfani Handoyo

Choose dataset to use:
1. Iris dataset
2. Wine dataset
3. Digits dataset
4. Breast cancer wisconsin dataset
Your input (1-4): 2

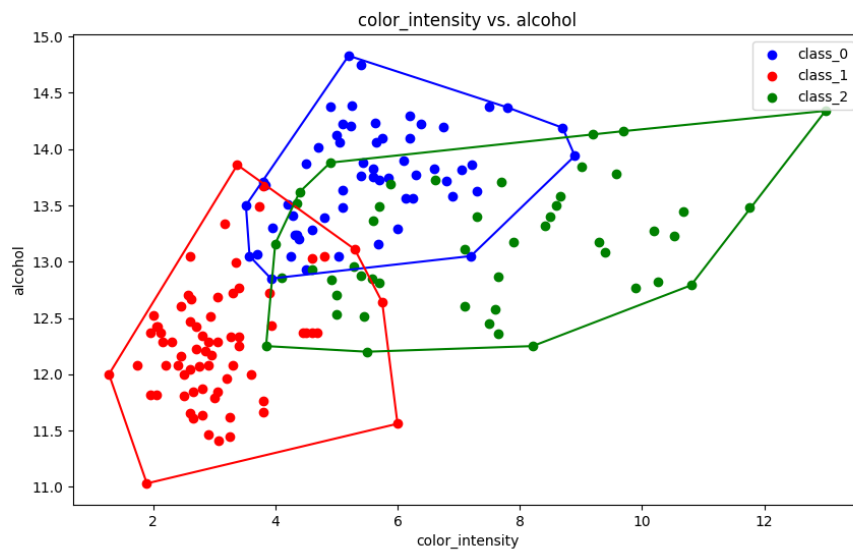
Choose attribute for x:
1. alcohol
2. malic_acid
3. ash
4. alcalinity_of_ash
5. magnesium
6. total_phenols
7. flavanoids
8. nonflavanoid_phenols
9. proanthocyanins
10. color_intensity
11. hue
12. od280/od315_of_diluted_wines
13. proline
Your input (1-13): 10

Choose attribute for y (make sure it is different from 10. color_intensity):
1. alcohol
2. malic_acid
3. ash
4. alcalinity_of_ash
5. magnesium
6. total_phenols
7. flavanoids
8. nonflavanoid_phenols
9. proanthocyanins
10. color_intensity
11. hue
12. od280/od315_of_diluted_wines
13. proline
Your input (1-13): 1

Showing convex hull graph of color_intensity vs. alcohol
Make sure to close the graph pop-up so the program can continue!
```



Output graf pada gambar “testcase4.png” pada folder test:



5. Dataset digits, pixel_2_6 vs. pixel_6_4

```
=====
WELCOME TO AFAN'S CONVEX HULL GENERATOR!
=====

Made by:
13520023
Ahmad Alfani Handoyo

Choose dataset to use:
1. Iris dataset
2. Wine dataset
3. Digits dataset
4. Breast cancer wisconsin dataset
Your input (1-4): 3
```

```

Choose attribute for x:
1. pixel_0_0
2. pixel_0_1
3. pixel_0_2
4. pixel_0_3
5. pixel_0_4
6. pixel_0_5
7. pixel_0_6
8. pixel_0_7
9. pixel_1_0
10. pixel_1_1
11. pixel_1_2
12. pixel_1_3
13. pixel_1_4
14. pixel_1_5
15. pixel_1_6
16. pixel_1_7
17. pixel_2_0
18. pixel_2_1
19. pixel_2_2
20. pixel_2_3
21. pixel_2_4
22. pixel_2_5
23. pixel_2_6
24. pixel_2_7
25. pixel_3_0
26. pixel_3_1
27. pixel_3_2
28. pixel_3_3
29. pixel_3_4
30. pixel_3_5
31. pixel_3_6
32. pixel_3_7
33. pixel_4_0
34. pixel_4_1
35. pixel_4_2
36. pixel_4_3
37. pixel_4_4
38. pixel_4_5
39. pixel_4_6
40. pixel_4_7
41. pixel_5_0
42. pixel_5_1
43. pixel_5_2
44. pixel_5_3
45. pixel_5_4
46. pixel_5_5
47. pixel_5_6
48. pixel_5_7
49. pixel_6_0
50. pixel_6_1
51. pixel_6_2
52. pixel_6_3
53. pixel_6_4
54. pixel_6_5
55. pixel_6_6
56. pixel_6_7
57. pixel_7_0
58. pixel_7_1
59. pixel_7_2
60. pixel_7_3
61. pixel_7_4
62. pixel_7_5
63. pixel_7_6
64. pixel_7_7
Your input (1-64): 23

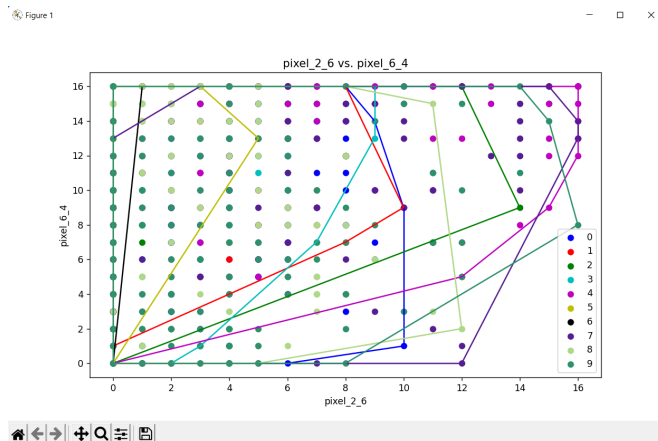
```

```

Choose attribute for y (make sure it is different from 23, pixel_2_6):
1. pixel_0_0
2. pixel_0_1
3. pixel_0_2
4. pixel_0_3
5. pixel_0_4
6. pixel_0_5
7. pixel_0_6
8. pixel_0_7
9. pixel_1_0
10. pixel_1_1
11. pixel_1_2
12. pixel_1_3
13. pixel_1_4
14. pixel_1_5
15. pixel_1_6
16. pixel_1_7
17. pixel_2_0
18. pixel_2_1
19. pixel_2_2
20. pixel_2_3
21. pixel_2_4
22. pixel_2_5
23. pixel_2_6
24. pixel_2_7
25. pixel_3_0
26. pixel_3_1
27. pixel_3_2
28. pixel_3_3
29. pixel_3_4
30. pixel_3_5
31. pixel_3_6
32. pixel_3_7
33. pixel_4_0
34. pixel_4_1
35. pixel_4_2
36. pixel_4_3
37. pixel_4_4
38. pixel_4_5
39. pixel_4_6
40. pixel_4_7
41. pixel_5_0
42. pixel_5_1
43. pixel_5_2
44. pixel_5_3
45. pixel_5_4
46. pixel_5_5
47. pixel_5_6
48. pixel_5_7
49. pixel_6_0
50. pixel_6_1
51. pixel_6_2
52. pixel_6_3
53. pixel_6_4
54. pixel_6_5
55. pixel_6_6
56. pixel_6_7
57. pixel_7_0
58. pixel_7_1
59. pixel_7_2
60. pixel_7_3
61. pixel_7_4
62. pixel_7_5
63. pixel_7_6
64. pixel_7_7
Your input (1-64): 53

```

Showing convex hull graph of pixel_2_6 vs. pixel_6_4
Make sure to close the graph pop-up so the program can continue!



```

Would you like to save the convex hull graph? (Y/N): y

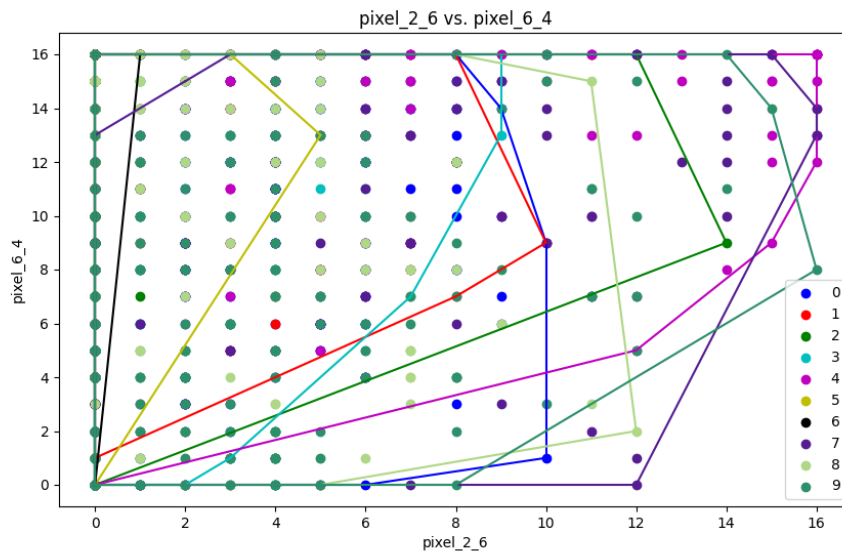
Output file name: testcase5
Your convex hull graph of pixel_2_6 vs. pixel_6_4 at testcase5.png has successfully been
made at folder test!

Do you want to try another dataset or pair of attributes? (Y/N): n

=====
THANK YOU FOR USING AFAN'S CONVEX HULL GENERATOR!
=====

```

Output graf pada gambar “testcase5.png” pada folder test:



6. Dataset digits, pixel_0_2 vs. pixel_5_1

```

=====
WELCOME TO AFAN'S CONVEX HULL GENERATOR!
=====

Made by:
13520023
Ahmad Alfani Handoyo

Choose dataset to use:
1. Iris dataset
2. Wine dataset
3. Digits dataset
4. Breast cancer wisconsin dataset
Your input (1-4): 3

Choose attribute for x:
1. pixel_0_0
2. pixel_0_1
3. pixel_0_2
4. pixel_0_3
5. pixel_0_4
6. pixel_0_5
7. pixel_0_6
8. pixel_0_7
9. pixel_1_0
10. pixel_1_1
11. pixel_1_2
12. pixel_1_3
13. pixel_1_4
14. pixel_1_5
15. pixel_1_6
16. pixel_1_7
17. pixel_2_0
18. pixel_2_1
19. pixel_2_2
20. pixel_2_3
21. pixel_2_4
22. pixel_2_5
23. pixel_2_6
24. pixel_2_7
25. pixel_3_0
26. pixel_3_1
27. pixel_3_2
28. pixel_3_3
29. pixel_3_4
30. pixel_3_5
31. pixel_3_6
32. pixel_3_7
33. pixel_4_0
34. pixel_4_1
35. pixel_4_2
36. pixel_4_3
37. pixel_4_4
38. pixel_4_5
39. pixel_4_6
40. pixel_4_7
41. pixel_5_0
42. pixel_5_1
43. pixel_5_2
44. pixel_5_3
45. pixel_5_4
46. pixel_5_5
47. pixel_5_6
48. pixel_5_7
49. pixel_6_0
50. pixel_6_1
51. pixel_6_2
52. pixel_6_3
53. pixel_6_4
54. pixel_6_5
55. pixel_6_6
56. pixel_6_7
57. pixel_7_0
58. pixel_7_1
59. pixel_7_2
60. pixel_7_3
61. pixel_7_4
62. pixel_7_5
63. pixel_7_6
64. pixel_7_7
Your input (1-64): 3

```

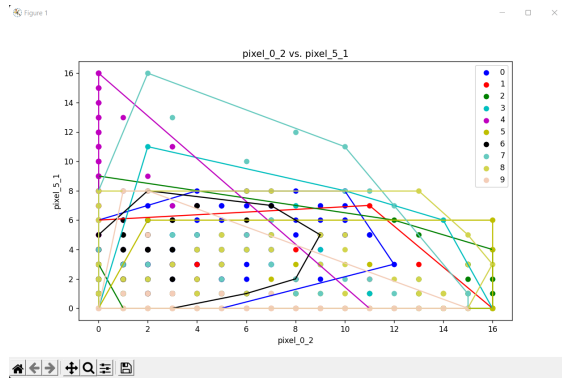


```

Choose attribute for y (make sure it is different from 3, pixel_0_2):
1. pixel_0_0
2. pixel_0_1
3. pixel_0_2
4. pixel_0_3
5. pixel_0_4
6. pixel_0_5
7. pixel_0_6
8. pixel_0_7
9. pixel_1_0
10. pixel_1_1
11. pixel_1_2
12. pixel_1_3
13. pixel_1_4
14. pixel_1_5
15. pixel_1_6
16. pixel_1_7
17. pixel_2_0
18. pixel_2_1
19. pixel_2_2
20. pixel_2_3
21. pixel_2_4
22. pixel_2_5
23. pixel_2_6
24. pixel_2_7
25. pixel_3_0
26. pixel_3_1
27. pixel_3_2
28. pixel_3_3
29. pixel_3_4
30. pixel_3_5
31. pixel_3_6
32. pixel_3_7
33. pixel_4_0
34. pixel_4_1
35. pixel_4_2
36. pixel_4_3
37. pixel_4_4
38. pixel_4_5
39. pixel_4_6
40. pixel_4_7
41. pixel_5_0
42. pixel_5_1
43. pixel_5_2
44. pixel_5_3
45. pixel_5_4
46. pixel_5_5
47. pixel_5_6
48. pixel_5_7
49. pixel_6_0
50. pixel_6_1
51. pixel_6_2
52. pixel_6_3
53. pixel_6_4
54. pixel_6_5
55. pixel_6_6
56. pixel_6_7
57. pixel_7_0
58. pixel_7_1
59. pixel_7_2
60. pixel_7_3
61. pixel_7_4
62. pixel_7_5
63. pixel_7_6
64. pixel_7_7
Your input (1-64): 42

```

Showing convex hull graph of pixel_0_2 vs. pixel_5_1
Make sure to close the graph pop-up so the program can continue!



```

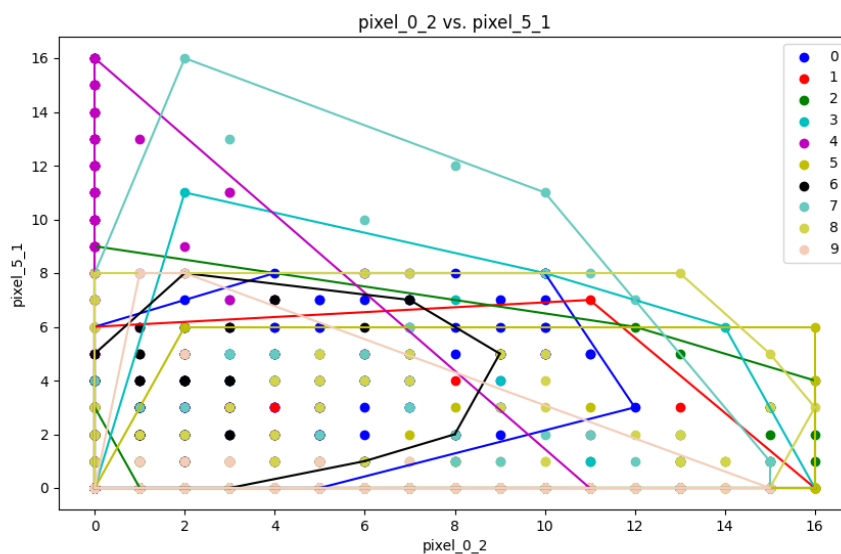
Would you like to save the convex hull graph? (Y/N): y
Output file name: testcase6
Your convex hull graph of pixel_0_2 vs. pixel_5_1 at testcase6.png has successfully been
made at folder test!

Do you want to try another dataset or pair of attributes? (Y/N): n

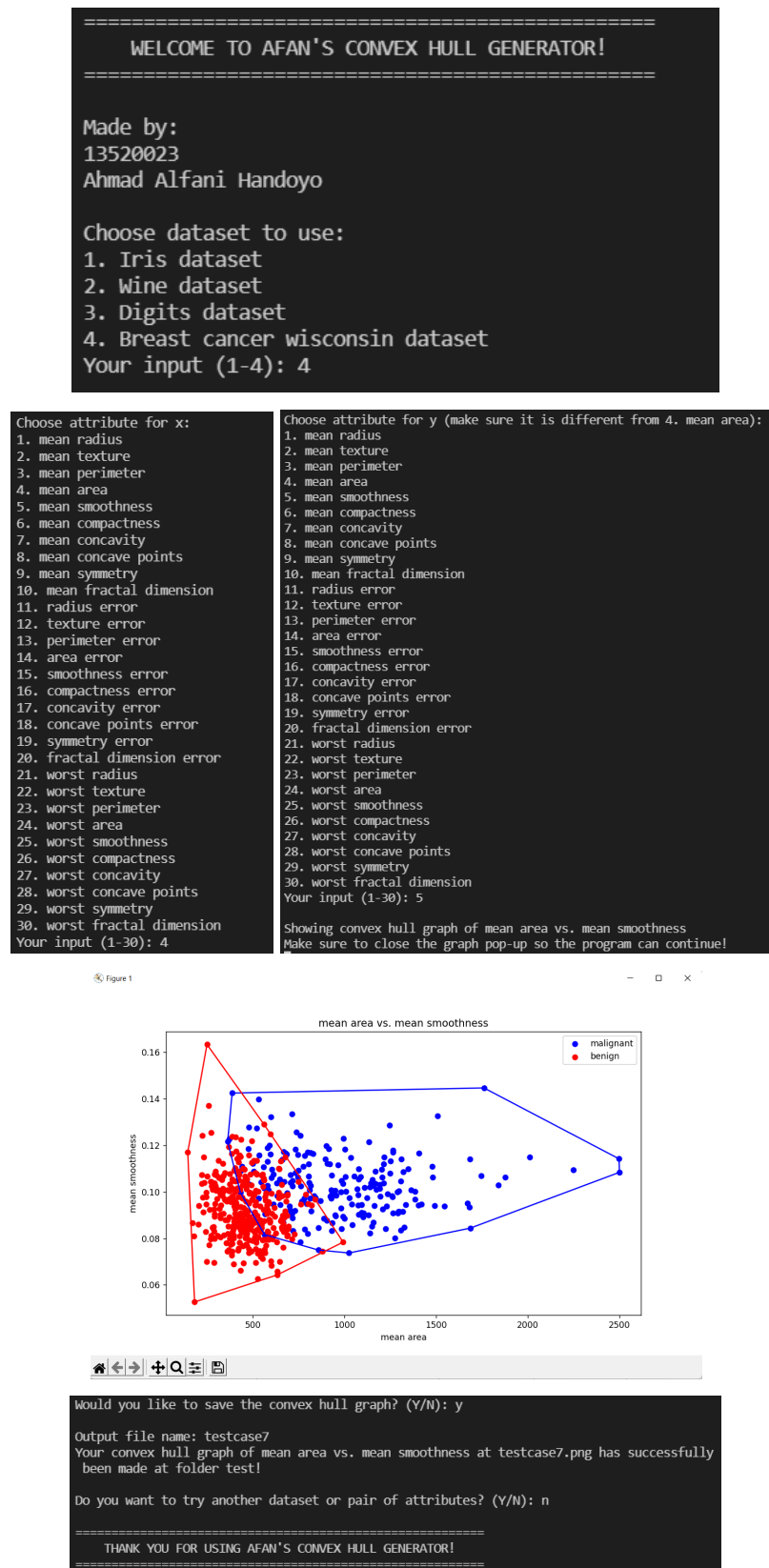
=====
THANK YOU FOR USING AFAN'S CONVEX HULL GENERATOR!
=====

```

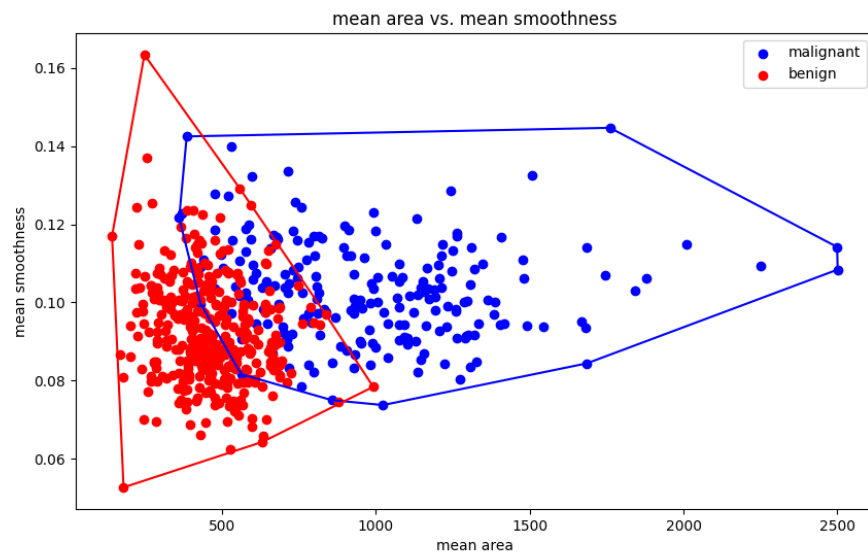
Output graf pada gambar “testcase6.png” pada folder test:



7. Dataset breast cancer wisconsin, mean area vs. mean smoothness



Output graf pada gambar “testcase7.png” pada folder test:



8. Dataset breast cancer wisconsin, mean radius vs. compactness error

```
=====
WELCOME TO AFAN'S CONVEX HULL GENERATOR!
=====

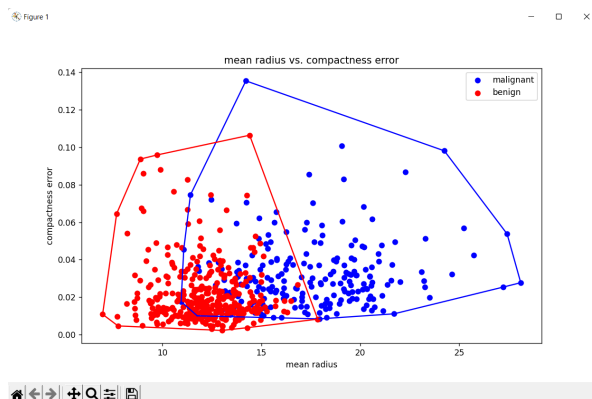
Made by:
13520023
Ahmad Alfani Handoyo

Choose dataset to use:
1. Iris dataset
2. Wine dataset
3. Digits dataset
4. Breast cancer wisconsin dataset
Your input (1-4): 4

Choose attribute for x:
1. mean radius
2. mean texture
3. mean perimeter
4. mean area
5. mean smoothness
6. mean compactness
7. mean concavity
8. mean concave points
9. mean symmetry
10. mean fractal dimension
11. radius error
12. texture error
13. perimeter error
14. area error
15. smoothness error
16. compactness error
17. concavity error
18. concave points error
19. symmetry error
20. fractal dimension error
21. worst radius
22. worst texture
23. worst perimeter
24. worst area
25. worst smoothness
26. worst compactness
27. worst concavity
28. worst concave points
29. worst symmetry
30. worst fractal dimension
Your input (1-30): 1

Choose attribute for y (make sure it is different from 1. mean radius):
1. mean radius
2. mean texture
3. mean perimeter
4. mean area
5. mean smoothness
6. mean compactness
7. mean concavity
8. mean concave points
9. mean symmetry
10. mean fractal dimension
11. radius error
12. texture error
13. perimeter error
14. area error
15. smoothness error
16. compactness error
17. concavity error
18. concave points error
19. symmetry error
20. fractal dimension error
21. worst radius
22. worst texture
23. worst perimeter
24. worst area
25. worst smoothness
26. worst compactness
27. worst concavity
28. worst concave points
29. worst symmetry
30. worst fractal dimension
Your input (1-30): 16

Showing convex hull graph of mean radius vs. compactness error
Make sure to close the graph pop-up so the program can continue!
```



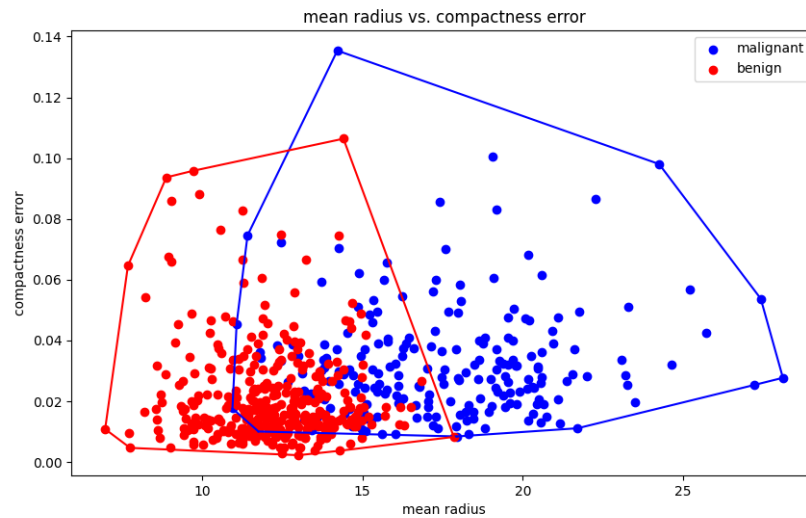
```

would you like to save the convex hull graph? (Y/N): y
Output file name: testcase8
Your convex hull graph of mean radius vs. compactness error at testcase8.png has successfully been made at folder test!
Do you want to try another dataset or pair of attributes? (Y/N): n

=====
THANK YOU FOR USING AFAN'S CONVEX HULL GENERATOR!
=====

```

Output graf pada gambar “testcase8.png” pada folder *test*:



D. Alamat Unduh Program

Program dapat diunduh pada repository berikut:

<https://github.com/blueguy42/Convex-Hull-Algorithm>

E. Tabel Keberjalanan Program

Poin	Ya	Tidak
1. Pustaka <i>myConvexHull</i> berhasil dibuat dan tidak ada kesalahan	✓	
2. <i>Convex hull</i> yang dihasilkan sudah benar	✓	
3. Pustaka <i>myConvexHull</i> dapat digunakan untuk menampilkan <i>convex hull</i> setiap label dengan warna yang berbeda	✓	
4. Bonus: program dapat menerima input dan menuliskan output untuk dataset lainnya.	✓	