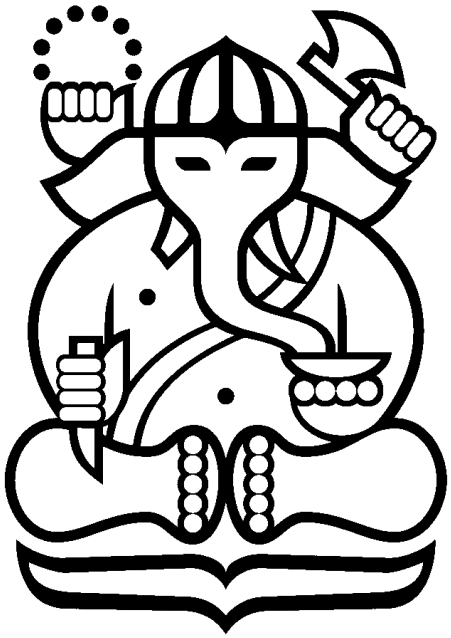
Laporan Tugas Kecil 2

IF2211 Strategi Algoritma

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

Disusun oleh:

Diky Restu Maulana 13520017



PROGRAM STUDI TEKNIK INFORMATIKA

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

BANDUNG

2022

1. Algoritma Divide and Conquer

Divide and Conquer adalah salah satu strategi penyelesaian suatu persoalan. *Divide* berarti membagi persoalan menjadi beberapa upa-persoalan yang lebih sederhana dengan tipe persoalan yang masih identik dengan persoalan awal. *Conquer* berarti menyelesaikan masing-masing upa-persoalan, lalu digabungkan hasilnya menjadi solusi persoalan yang lebih besar hingga didapatkan solusi persoalan awal. Dalam implementasinya, strategi ini dilakukan secara rekursif hingga didapatkan basis yang merupakan upa-persoalan yang paling kecil.

Langkah-langkah yang dilakukan untuk menemukan kumpulan titik yang membentuk *convex hull* berdasarkan algoritma *Divide and Conquer* sebagai berikut

1. *S*: himpunan titik sebanyak *n*, dengan $n>1$, yaitu titik $p_1(x_1,y_1)$ hingga $p_n(x_n,y_n)$ pada bidang kartesian dua dimensi. Dalam implementasinya, S dibuat dalam sebuah array berdimensi $n \times 2$.
2. Urutkan array *S* berdasarkan nilai absis menaik. Jika nilai absisnya sama, urutkan berdasarkan nilai ordinat menaik.
3. Ada dua kemungkinan pada array *S*, yaitu
   - Jika hanya terdapat dua titik pada *S*, maka garis yang menghubungkan kedua titik adalah pembentuk *convex hull*. Kembalikan kedua titik tersebut.
   - Jika terdapat lebih dari dua titik, selanjutnya ambil elemen pertama p_1 (absis terkecil/titik paling kiri) dan elemen terakhir p_n (absis terbesar/titik paling kanan) dari array *S* yang sudah diurutkan.
4. Garis yang menghubungkan p_1 dan p_n (p_1p_n) membagi *S* menjadi dua segmen, yaitu *above* (kumpulan titik di sebelah kiri atau atas garis p_1p_n) dan *below* (kumpulan titik di sebelah kanan atau bawah garis p_1p_n).
5. Semua titik pada *S* yang berada pada garis p_1p_n (selain titik p_1 dan p_n) tidak mungkin membentuk *convex hull* sehingga bisa diabaikan.
6. Kumpulan titik pada *above* bisa membentuk *convex hull* bagian atas dan kumpulan titik pada *below* dapat membentuk *convex hull* bagian bawah.
7. Untuk sebuah segmen (misalnya *above*), terdapat dua kemungkinan:
   - Jika tidak ada titik di segmen *above*, maka titik p_1 dan p_n menjadi pembentuk *convex hull* bagian *above*.
   - Jika *above* tidak kosong, pilih sebuah titik yang memiliki jarak terjauh dari garis p_1p_n (misalnya p_{max}). Jika terdapat beberapa titik dengan jarak yang sama, pilih sebuah titik yang memaksimalkan sudut $p_{max}p_1p_n$.
8. Semua titik yang berada di dalam segitiga $p_{max}p_1p_n$ dapat diabaikan pada pemeriksaan selanjutnya karena tidak mungkin membentuk *convex hull*.
9. Tarik garis p_1p_{max} dan $p_{max}p_n$. Lalu, tentukan *above1*, yaitu kumpulan titik pada *above* yang berada di sebelah kiri atau atas garis p_1p_{max} dan *above2*, yaitu kumpulan titik pada *above* yang berada di sebelah kiri atau atas garis $p_{max}p_n$.
10. Lakukan hal yang sama (poin 4 dan 5) untuk segmen *below*, hingga tidak ada lagi titik yang berada di luar segitiga.
11. Kembalikan pasangan titik yang dihasilkan.

Pada tugas kecil kali ini, algoritma *Divide and Conquer* diimplementasikan dalam pembuatan suatu *library* bernama myConvexHull. *Library* tersebut terdapat pada file ‘myConvexHull.py’ dan berfungsi untuk mencari kumpulan titik yang membentuk *convex*

hull dari pasangan nilai yang didapatkan dari sebuah dataset. Terdapat 7 fungsi dalam file tersebut, yaitu

No.	Fungsi	Keterangan
1.	ConvexHull(points)	Mengembalikan hasil akhir berupa kumpulan titik yang membentuk *convex hull*
2.	findHull(p1, p2, segment, flag)	Mengembalikan kumpulan titik yang membentuk *convex hull* di segmen atas dan bawah.
3.	gradien(p1, p2)	Menghitung kemiringan garis yang dibentuk oleh dua titik.
4.	constant(p1, p2)	Menghitung konstanta c pada persamaan garis.
5.	findDistance(p1, p2, point)	Menghitung jarak terpendek antara titik dan garis.
6.	createSegment(p1, p2, points)	Membagi kumpulan titik menjadi dua segmen, yaitu atas dan bawah yang dipisahkan oleh garis yang dibentuk oleh dua titik.
7.	sortHull(hull)	Mengurutkan kumpulan titik secara melingkar berlawanan arah jarum jam dari titik paling kiri ke titik untuk keperluan *plotting*.

2. Source Code Program dalam Bahasa Python

2.1. myConvexHull.py

```python
# Fungsi utama berparameter points (array of point) dan mengembalikan titik-titik pembentuk Convex Hull
def ConvexHull(points):
    # BASIS: Jika hanya ada dua titik, kembalikan keduanya
    if (len(points) <= 2):
        return points

    # Lakukan pengurutan membesar berdasarkan nilai x
    # Jika nilai x sama, urutkan nilai y membesar
    sort = sorted(points, key=lambda x:(x[0], x[1]))

    # Ambil titik paling kiri dan paling kanan
    p1 = sort[0]
    p2 = sort[-1]

    # Kedua titik menjadi pembentuk Convex Hull
    hull = [p1, p2]

    # Hapus kedua titik dari kumpulan titik agar tidak diperiksa lagi
    sort.pop(0)
    sort.pop(-1)

    # DIVIDE
    # Titik p1 dan p2 membentuk garis
    # Pisahkan titik-titik yang berada di atas dan bawah garis tersebut
    above, below = createSegment(p1, p2, sort)
```

```python
    # CONQUER
    # Gabungkan hasil pemeriksaan segmen atas dan bawah
    hull += findHull(p1, p2, above, "above")
    hull += findHull(p1, p2, below, "below")

    # Menghapus titik yang muncul dua kali/duplikat
    hull = (np.unique(np.asarray(hull), axis=0)).tolist()

    # Mengurutkan titik untuk keperluan plotting
    hull = sortHull(hull)

    return hull
```

```python
# Fungsi sampingan untuk memeriksa setiap segmen
# p1p2 adalah garis yang akan menjadi acuan
# segment adalah kumpulan titik yang akan diperiksa
# flag bernilai "above" atau "below"
def findHull(p1, p2, segment, flag):
    # Jika tidak ada titik di dalam segmen, kembalikan array kosong
    if (len(segment) == 0):
        return []

    # Inisialisasi
    hull = []
    farthest_distance = -1
    farthest_point = None

    # Mencari titik dengan jarak terjauh terhadap garis p1p2
    for point in segment:
        distance = findDistance(p1, p2, point)
        if(distance > farthest_distance):
            farthest_distance = distance
            farthest_point = point

    # Titik terjauh pasti akan membentuk Convex Hull
    hull += [farthest_point]

    # Menghapus titik dari segment agar tidak diperiksa lagi
    segment.remove(farthest_point)
    # Membuat segmen baru berdasarkan garis p1-farthest_point dan farthest_point-p2
    p1a, p1b = createSegment(p1, farthest_point, segment)
    p2a, p2b = createSegment(p2, farthest_point, segment)

    # Jika segmen yang diperiksa adalah segmen atas, terus menerus akan diperiksa segmen bagian atas
    # Begitupun sebaliknya
    if flag == "above":
        hull += findHull(p1, farthest_point, p1a, "above")
        hull += findHull(farthest_point, p2, p2a, "above")
    else:
        hull += findHull(p1, farthest_point, p1b, "below")
        hull += findHull(farthest_point, p2, p2b, "below")

    return hull
# Fungsi untuk menghitung gradien garis p1p2
def gradien(p1, p2):
    return (p1[1] - p2[1]) / (p1[0] - p2[0])
# Fungsi untuk menghitung konstanta pada persamaan garis p1p2
def constant(p1, p2):
    m = gradien(p1, p2)
    return (p1[1] - m*p1[0])
# Fungsi untuk menghitung jarak terpendek antara garis p1p2 dan titik point
def findDistance(p1, p2, point):
    a = p1[1]-p2[1]
    b = p2[0]-p1[0]
    c = p1[0]*p2[1]-p2[0]*p1[1]
    return abs((a*point[0] + b*point[1] + c) / ((a*a + b*b)**0.5))
# Fungsi untuk membagi kumpulan titik menjadi dua segmen (atas dan bawah) berdasarkan garis p1p2
def createSegment(p1, p2, points):
    above = []
    below = []

    # Jika garis yang dibentuk vertikal, tidak ada atas dan bawah
    # Sekaligus menghindari gradien yang tidak terdefinisi (pembagian dengan nol)
    if(p1[0] - p2[0] == 0):
        return above, below

    m = gradien(p1, p2)
    c = constant(p1, p2)

    for p in points:
        if (p[1] > m*p[0] + c):
            above.append(p)
        elif (p[1] < m*p[0] + c):
            below.append(p)
    return above,below
# Fungsi untuk mengurutkan titik untuk keperluan plotting
def sortHull(hull):
    above, below = createSegment(hull[0], hull[-1], hull)
    newHull = []
    newHull.append(hull[0])
    newHull += below
    newHull.append(hull[-1])
    above.reverse()
    newHull += above
    return newHull
```

2.2. main.py

```python
# File: main.py
# Berisi program utama untuk visualisasi kumpulan titik dan Convex Hull yang terbentuk

import myConvexHull as mc
import pandas as pd
import matplotlib.pyplot as plt
from sklearn import datasets

# Fungsi untuk menerima input pilihan dataset
def pilihDataset():
    print("Daftar dataset:")
    print("1. iris")
    print("2. wine")
    print("3. breast cancer")
    opsi = int(input("Masukkan pilihan dataset: "))
    data = None
    if (opsi == 1):
        data = datasets.load_iris()
    elif (opsi == 2):
        data = datasets.load_wine()
    elif (opsi == 3):
        data = datasets.load_breast_cancer()
    else:
        print("Pilihan tidak valid!\n")
        return pilihDataset()
    return data
```

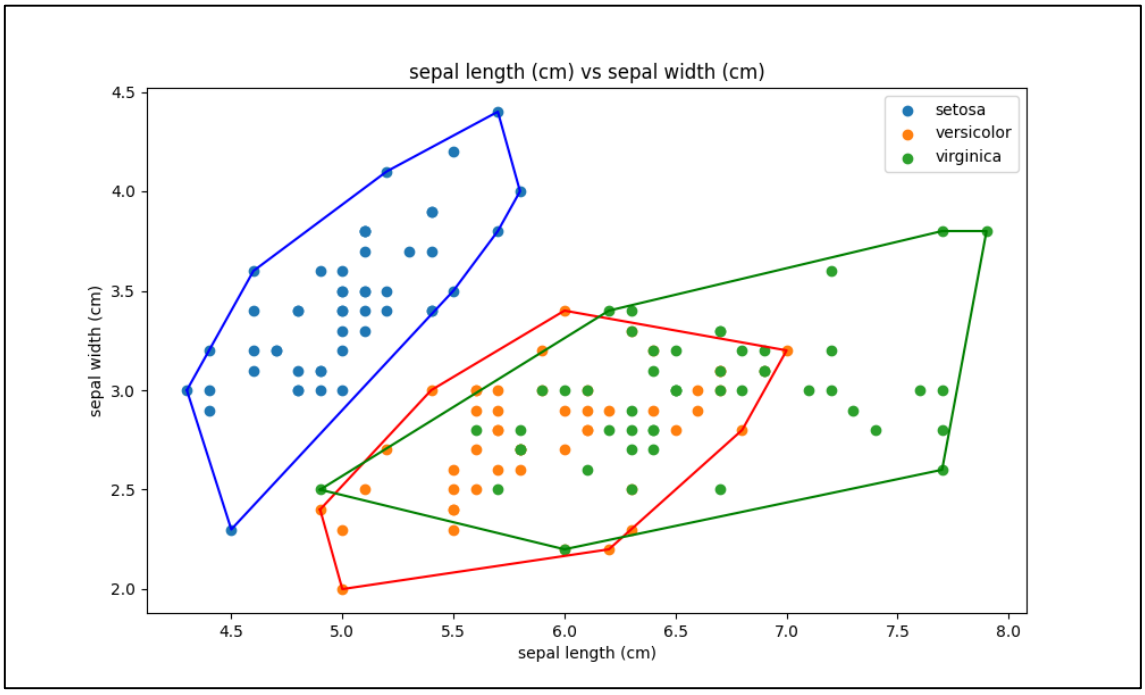
```python
# Fungsi untuk melakukan plotting kumpulan titik dan Convex Hull yang dihasilkan
def visualisasi(data):
    colors = ['b','r','g','c','m','y','k']

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

    nkolom = len(data.feature_names) # jumlah kolom
    ntarget = len(data.target_names) # jumlah target

    # Cetak pilihan kolom dan meminta masukan sepasang kolom
    for i in range(nkolom):
        print(f"{i+1}. {data.feature_names[i]}")
    opsi = input("Masukkan dua pilihan atribut berbeda (co. 1 2): ")
    xidx = int(opsi.split()[0]) - 1
    yidx = int(opsi.split()[1]) - 1
    try:
        x, y = data.feature_names[xidx], data.feature_names[yidx]
    except:
        print("Pilihan tidak valid! Program dihentikan.")
        exit(0)
```

```python
    # Menampilkan hasil dalam bentuk gambar
    plt.figure(figsize = (10, 6))
    plt.title(f"{x} vs {y}")
    plt.xlabel(x)
    plt.ylabel(y)
    for i in range(ntarget):
        bucket = df[df['Target'] == i]
        bucket = bucket.iloc[:,[xidx,yidx]].values
        listbucket = bucket.tolist()
        hull = mc.ConvexHull(listbucket)
        hull.append(hull[0]) # Menambahkan titik pertama supaya terbentuk poligon tertutup
        xs, ys = zip(*hull)
        plt.scatter(bucket[:, 0], bucket[:, 1], label=data.target_names[i]) # plot titik
        plt.plot(xs, ys, colors[i]) # plot garis
        plt.legend()
    plt.show()

# Fungsi utama
def main():
    print("Hai kamu... Selamat datang di program visualisasi Convex Hull ini.")
    print("Silakan ikuti perintah yang muncul setelah ini, ya!\n")
    data = pilihDataset()
    visualisasi(data)

if (__name__ == "__main__"):
    main()
```

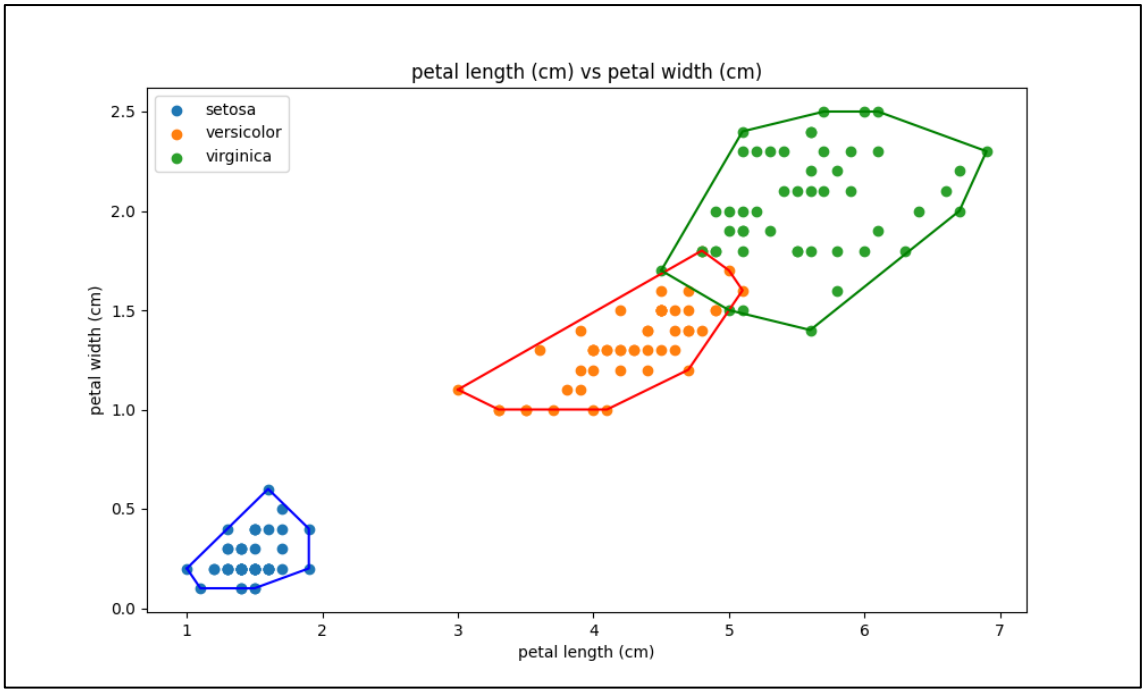
3. Input-Output Program

3.1. Dataset Iris

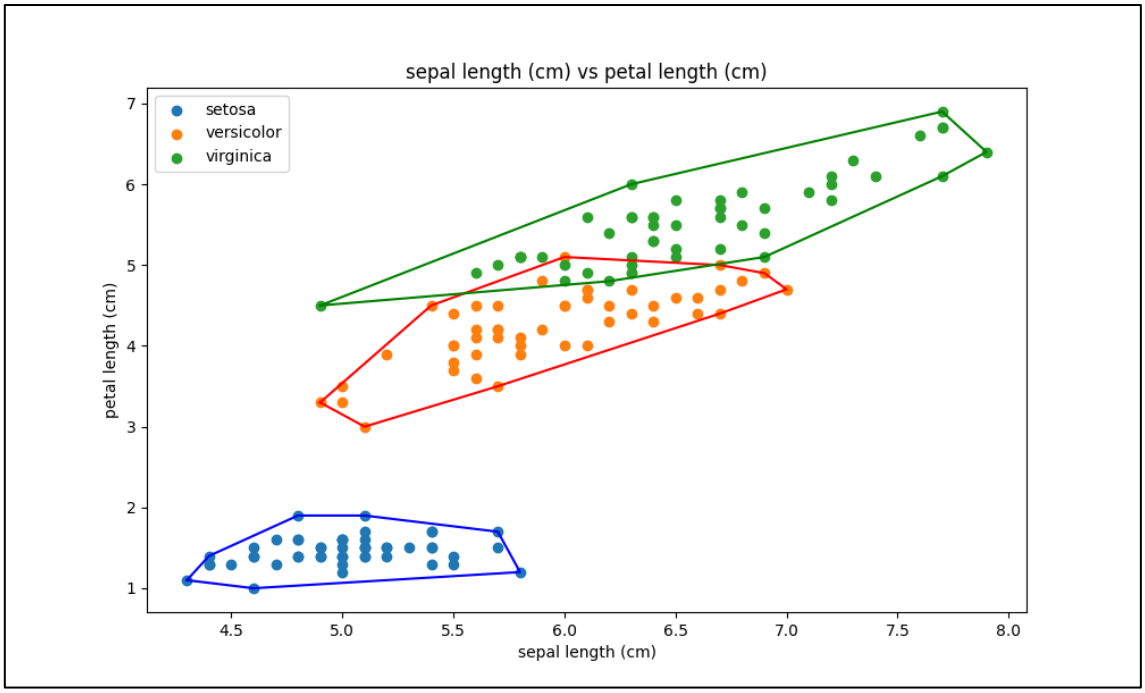
```
Hai kamu... Selamat datang di program visualisasi Convex Hull ini.
Silakan ikuti perintah yang muncul setelah ini, ya!

Daftar dataset:
1. iris
2. wine
3. breast cancer
Masukkan pilihan dataset: 1
1. sepal length (cm)
2. sepal width (cm)
3. petal length (cm)
4. petal width (cm)
Masukkan dua pilihan atribut berbeda (co. 1 2): 1 2
```



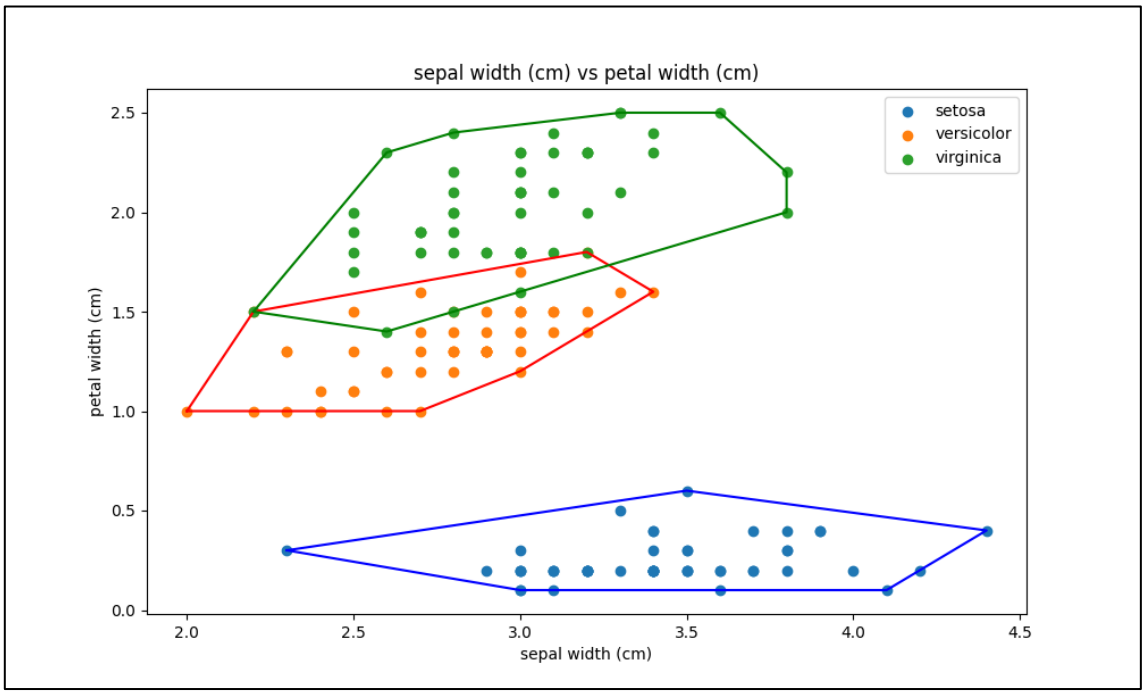

```
Hai kamu... Selamat datang di program visualisasi Convex Hull ini.
Silakan ikuti perintah yang muncul setelah ini, ya!

Daftar dataset:
1. iris
2. wine
3. breast cancer
Masukkan pilihan dataset: 1
1. sepal length (cm)
2. sepal width (cm)
3. petal length (cm)
4. petal width (cm)
Masukkan dua pilihan atribut berbeda (co. 1 2): 3 4
```




```
Hai kamu... Selamat datang di program visualisasi Convex Hull ini.
Silakan ikuti perintah yang muncul setelah ini, ya!

Daftar dataset:
1. iris
2. wine
3. breast cancer
Masukkan pilihan dataset: 1
1. sepal length (cm)
2. sepal width (cm)
3. petal length (cm)
4. petal width (cm)
Masukkan dua pilihan atribut berbeda (co. 1 2): 1 3
```

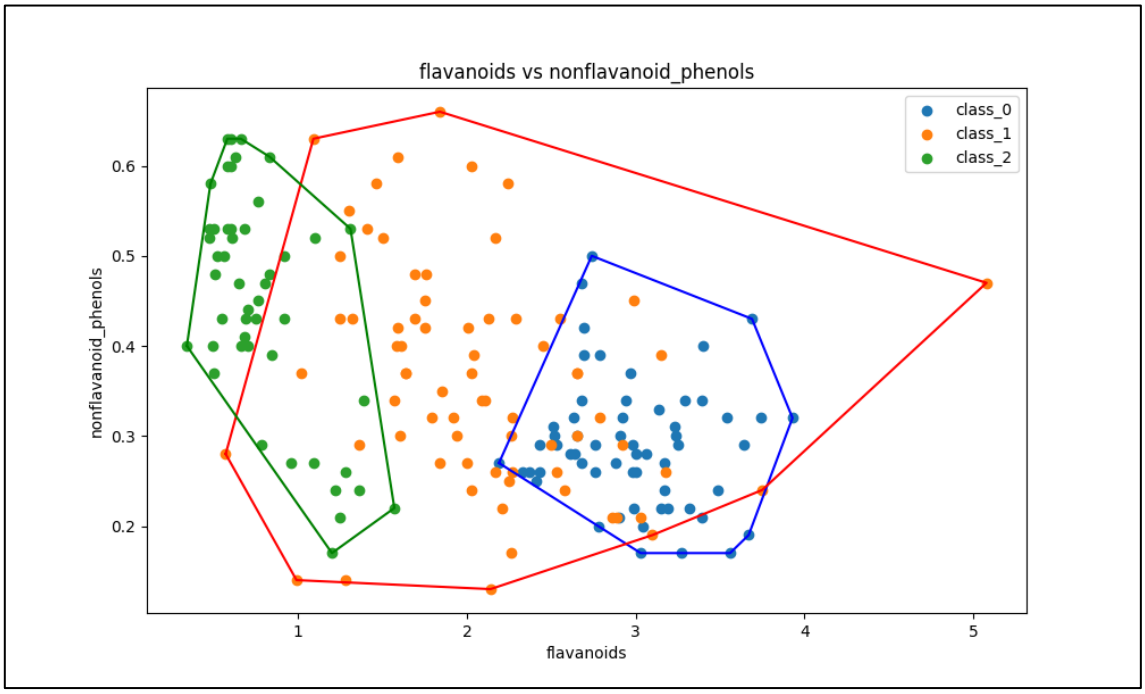


```
Hai kamu... Selamat datang di program visualisasi Convex Hull ini.
Silakan ikuti perintah yang muncul setelah ini, ya!

Daftar dataset:
1. iris
2. wine
3. breast cancer
Masukkan pilihan dataset: 1
1. sepal length (cm)
2. sepal width (cm)
3. petal length (cm)
4. petal width (cm)
Masukkan dua pilihan atribut berbeda (co. 1 2): 2 4
```

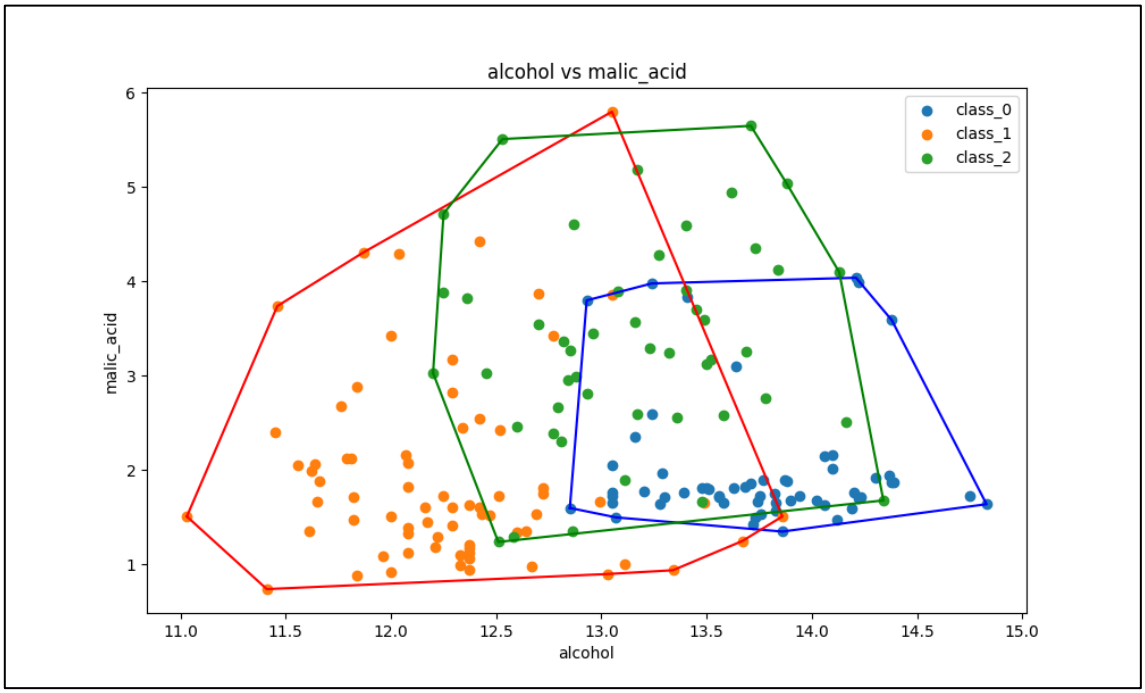


3.2. Dataset Wine

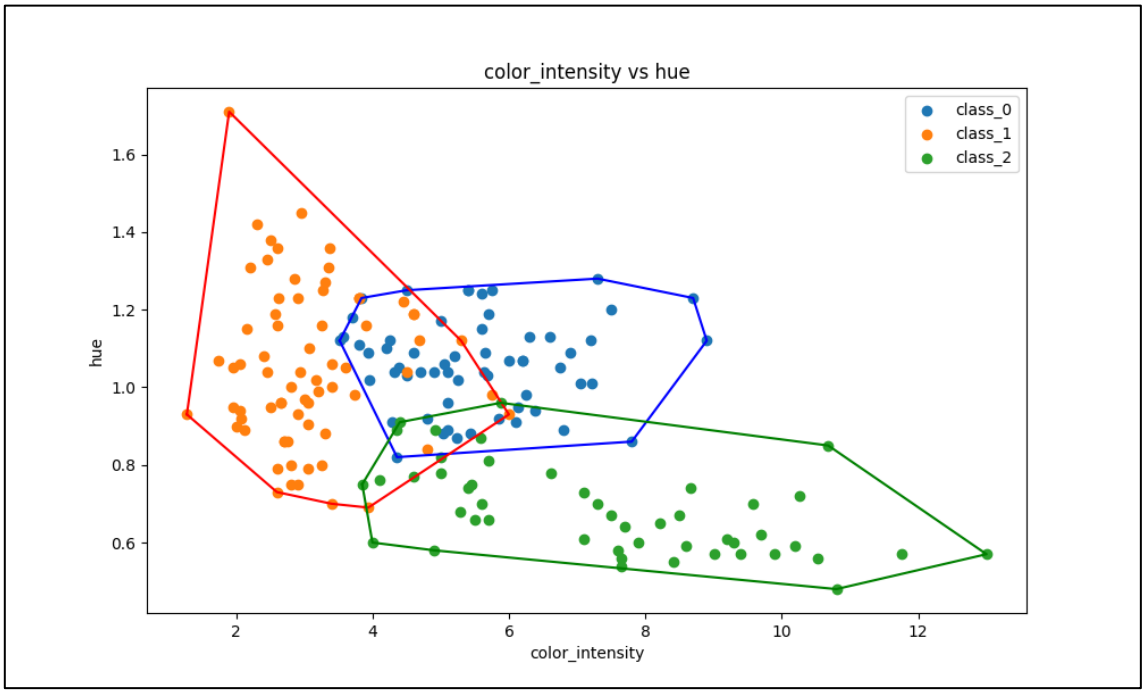
```
Hai kamu... Selamat datang di program visualisasi Convex Hull ini.
Silakan ikuti perintah yang muncul setelah ini, ya!

Daftar dataset:
1. iris
2. wine
3. breast cancer
1. iris
2. wine
3. breast cancer
Masukkan pilihan dataset: 2
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
Masukkan dua pilihan atribut berbeda (co. 1 2): 7 8
```



```
Hai kamu... Selamat datang di program visualisasi Convex Hull ini.
Silakan ikuti perintah yang muncul setelah ini, ya!

Daftar dataset:
1. iris
2. wine
3. breast cancer
Masukkan pilihan dataset: 2
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
Masukkan dua pilihan atribut berbeda (co. 1 2): 1 2
```

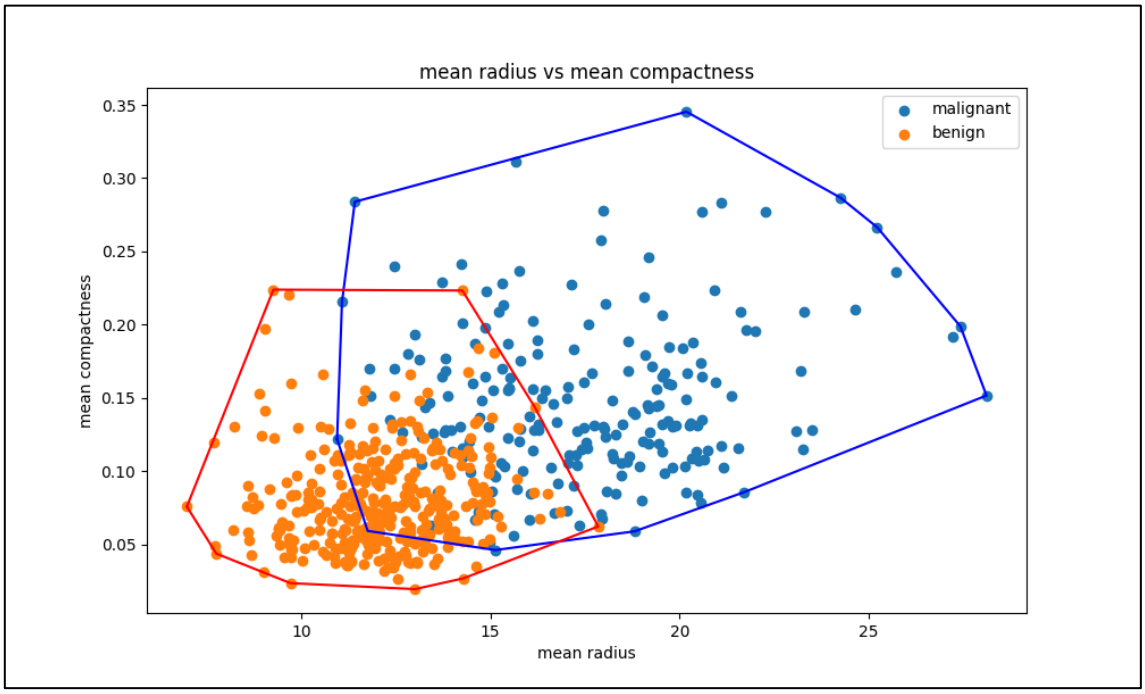


```
Hai kamu... Selamat datang di program visualisasi Convex Hull ini.
Silakan ikuti perintah yang muncul setelah ini, ya!

Daftar dataset:
1. iris
2. wine
3. breast cancer
Masukkan pilihan dataset: 2
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
Masukkan dua pilihan atribut berbeda (co. 1 2): 10 11
```

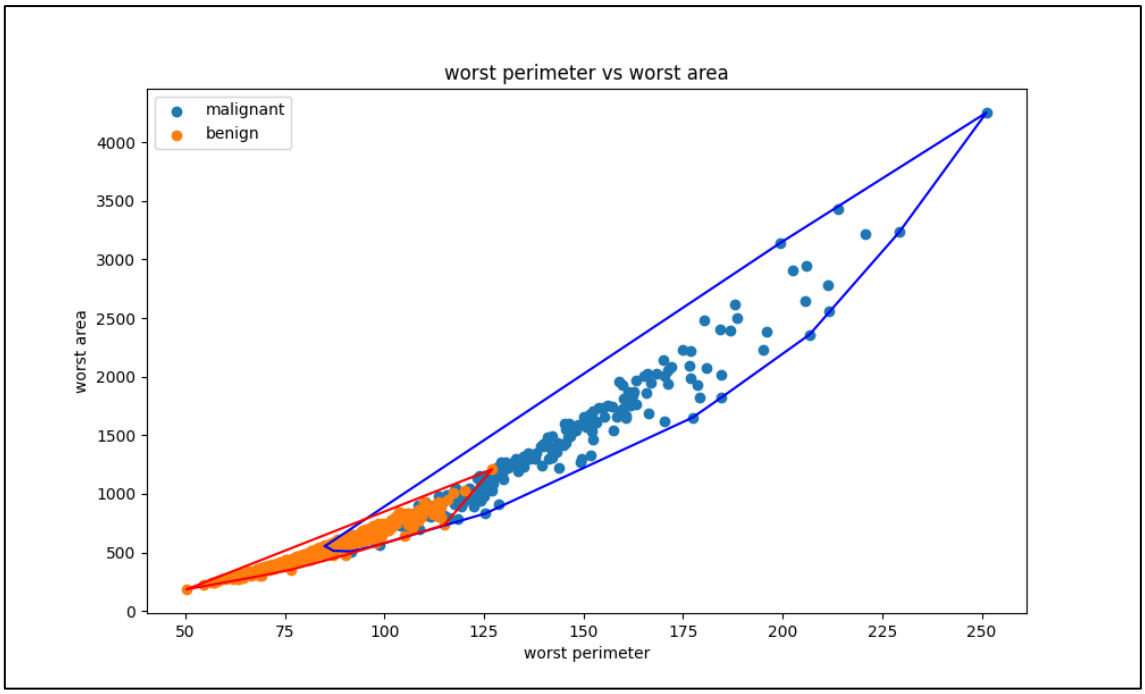


3.3. Dataset Breast Cancer

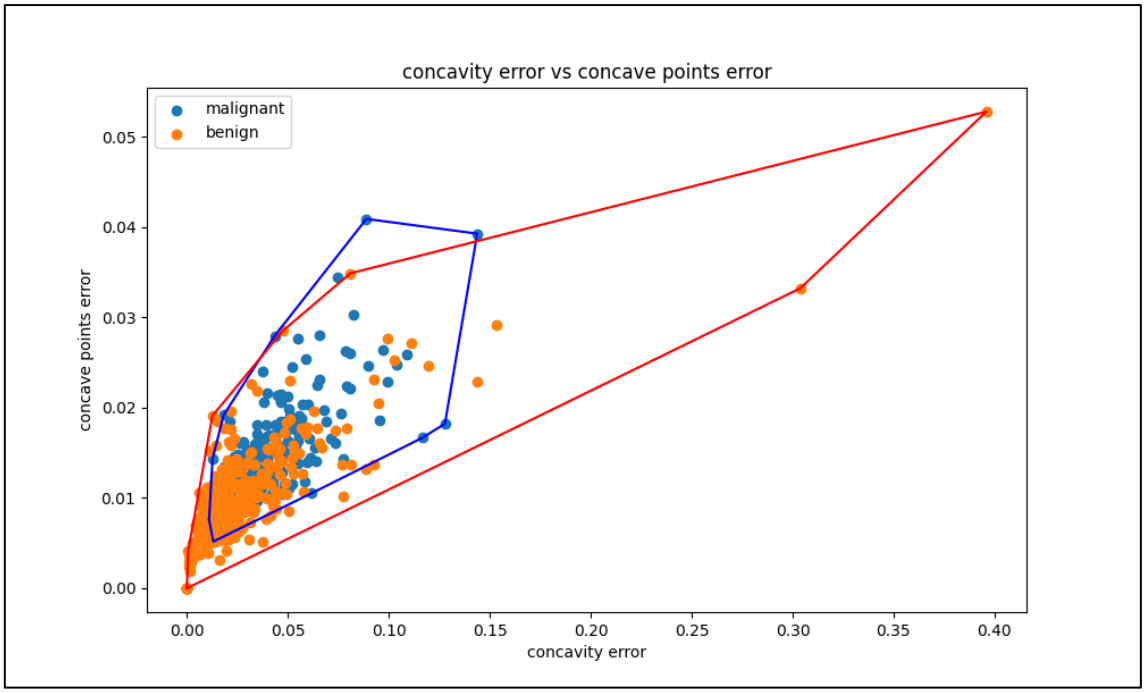
```
Hai kamu... Selamat datang di program visualisasi Convex Hull ini.
Silakan ikuti perintah yang muncul setelah ini, ya!

Daftar dataset:
1. iris
2. wine
3. breast cancer
Masukkan pilihan dataset: 3
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
Masukkan dua pilihan atribut berbeda (co. 1 2): 1 6
```



```
Hai kamu... Selamat datang di program visualisasi Convex Hull ini.
Silakan ikuti perintah yang muncul setelah ini, ya!

Daftar dataset:
1. iris
2. wine
3. breast cancer
Masukkan pilihan dataset: 3
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
Masukkan dua pilihan atribut berbeda (co. 1 2): 23 24
```



```
Hai kamu... Selamat datang di program visualisasi Convex Hull ini.
Silakan ikuti perintah yang muncul setelah ini, ya!

Daftar dataset:
1. iris
2. wine
3. breast cancer
Masukkan pilihan dataset: 3
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
Masukkan dua pilihan atribut berbeda (co. 1 2): 17 18
```



4. Tautan Github

https://github.com/dikyrest/Tucil_2_Stima_2022.git

5. Lampiran

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