

**LAPORAN TUGAS KECIL 2**  
**STRATEGI ALGORITMA IF2211 IMPLEMENTASI CONVEX HULL**  
**MENGGUNAKAN ALGORITMA DIVIDE AND CONQUER**



**Pembuat:**  
**13520136 Vincent Christian Siregar**

## I. Algoritma

1. Data yang akan di olah adalah data yang berisi kumpulan titik sebanyak  $n$  (dengan notasi  $(x,y)$ ) pada bidang dua dimensi.
2. Dua titik dengan nilai absis terendah dan tertinggi diambil dari dataset (misalkan titik absis terendah  $p_1$  dan titik absis tertinggi  $p_2$ ). Titik  $p_1$  dan  $p_2$  termasuk dalam convex hull.
3. Garis yang dibentuk dari  $p_1$  dan  $p_2$  membagi titik-titik yang lain menjadi 2 bagian yaitu  $s_1$  yang berada di sebelah kiri atas garis dan  $s_2$  yang berada di sebelah kanan bawah garis.
4.  $s_1$  dan  $s_2$  diterapkan algoritma decrease and conquer dengan metode rekursif.
5. Titik-titik pada  $s_1$  dan  $s_2$  dicari jarak yang terjauh dari garis yang dibentuk antara  $p_1$  dan  $p_2$ . Titik yang terjauh termasuk dari convex hull (misalkan  $p_3$ ). Setelah itu, titik-titik yang berada di luar garis  $p_1$ - $p_3$ , dan di luar garis  $p_3$ - $p_2$  dilakukan algoritma yang sama hingga tidak terdapat titik di luar garis yang dibentuk.
6. Hasil titik yang merupakan convex hull di merge dalam satu array yang menyimpan data titik-titik yang termasuk convex hull.

## II. Kode Program

A. myConvexHull.py (library Convex Hull)

```
import math

class ConvexHull:
    def __init__(self, bucket):
        self.bucket = bucket

    def getVertices(bucket):
        #ubah input menjadi list
        array = bucket.tolist()

        #mencari index letak titik paling kiri dan paling kanan
        x = []
        for item in array:
            x.append(item[0])
        idx_min, idx_max = getMinMax(x)

        #membagi titik titik menjadi s1 dan s2
        matrix = [[1 for i in range(3)] for i in range(3)]
        matrix[0][0] = array[idx_min][0]
        matrix[0][1] = array[idx_min][1]
        matrix[1][0] = array[idx_max][0]
        matrix[1][1] = array[idx_max][1]

        s1 = []
```

```

s2 = []
for i in range(len(array)):
    matrix[2][0] = array[i][0]
    matrix[2][1] = array[i][1]
    det = getDeterminant(matrix)
    if (det > 0):
        s1.append(i)
    elif (det < 0) :
        s2.append(i)

Hull1 = RecConvexHull(idx_min,idx_max, s1, array, True)
Hull2 = RecConvexHull(idx_min,idx_max, s2, array, False)
invertArray(Hull2)

vertices = [idx_min]
for item in Hull1:
    vertices.append(item)
vertices.append(idx_max)
for item in Hull2:
    vertices.append(item)

return vertices

def getSimplices(vertices):
    simplices = []
    for i in range(len(vertices)):
        s = []
        if i != len(vertices)-1:
            s.append(vertices[i])
            s.append(vertices[i+1])
        else:
            s.append(vertices[i])
            s.append(vertices[0])
        simplices.append(s)

    return simplices

#fungsi untuk membalikan urutan array
def invertArray(array):
    for i in range(math.floor(len(array)/2)):
        temp = array[i]
        array[i] = array[len(array)-1-i]
        array[len(array)-1-i] = temp

#fungsi untuk mencari index dengan nilai min dan max pada array

```

```

def getMinMax(array):
    min = array[0]
    max = array[0]
    idx_min = 0
    idx_max = 0
    for i in range(1, len(array)):
        if array[i] < min:
            min = array[i]
            idx_min = i
        elif array[i] > max:
            max = array[i]
            idx_max = i
    return idx_min, idx_max

#fungsi untuk mencari determinan dengan menggunakan ekspansi kofaktor
def getDeterminant(matrix):
    if len(matrix) == 1:
        return matrix[0][0]
    elif len(matrix) > 1:
        sign = 1
        det = 0
        for i in range(len(matrix)):
            a = matrix[i][0]
            m = []
            for j in range(len(matrix)):
                m1 = []
                for k in range(len(matrix[0])):
                    if j != i and k != 0:
                        m1.append(matrix[j][k])
                if (len(m1) != 0):
                    m.append(m1)
            det += sign * (a * getDeterminant(m))
            sign *= -1
        return det

def DividePoint(idx1, idx2, si, bucket, isLeft):
    # inisiasi matrix untuk mencari determinan
    mat = [[1 for i in range(3)] for i in range(3)]
    mat[0][0] = bucket[idx1][0]
    mat[0][1] = bucket[idx1][1]
    mat[1][0] = bucket[idx2][0]
    mat[1][1] = bucket[idx2][1]

    s = []
    if isLeft:

```

```

        for i in si:
            mat[2][0] = bucket[int(i)][0]
            mat[2][1] = bucket[int(i)][1]
            det = getDeterminant(mat)
            if (det > 0):
                s.append(i)
        return s
    else:
        for i in si:
            mat[2][0] = bucket[int(i)][0]
            mat[2][1] = bucket[int(i)][1]
            det = getDeterminant(mat)
            if (det < 0):
                s.append(i)
        return s

def PointToLine(x1,x2,y1,y2):
    distx = x2 - x1
    disty = y2 - y1
    a = disty
    b = -distx
    c = (distx*y1) - (disty*x1)
    return (a,b,c)

def Distance(x1, y1, Line):
    d = abs((Line[0] * x1 + Line[1] * y1 + Line[2])) / (math.sqrt(Line[0]
* Line[0] + Line[1] * Line[1]))
    return d

def RecConvexHull(idx1, idx2, s, bucket, isLeft):
    if len(s) == 0:
        return []
    elif len(s) == 1:
        return s
    else:
        max = 0
        Line =
PointToLine(bucket[idx1][0],bucket[idx2][0],bucket[idx1][1],bucket[idx2][1])
        for i in s:
            idx = int(i)
            distance = Distance(bucket[idx][0], bucket[idx][1], Line)
            if (max < distance):
                max = distance
                id = idx

```

```

        s1 = DividePoint(idx1, id, s, bucket, isLeft)
        arr1 = RecConvexHull(idx1, id, s1, bucket, isLeft)
        s2 = DividePoint(id, idx2, s, bucket, isLeft)
        arr2 = RecConvexHull(id, idx2, s2, bucket, isLeft)

        arr1.append(id)
        for item in arr2:
            arr1.append(item)

        return arr1

self.vertices = getVertices(self.bucket)
self.simplices = getSimplices(self.vertices)

```

## B. Visualizer.py (Visualisasi Convex Hull)

```

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

def check_input_dataset(input):
    try:
        val = int(input)
        if (val >= 1 and val <=3):
            return True
        else:
            return False
    except ValueError:
        return False

def check_input_xy(x,y,length):
    try:
        valx = int(x)
        valy = int(y)
        if (valx >= 0 and valx <=length-1) and (valy >= 0 and valy <=length-1):
            return True
        else:
            return False
    except ValueError:
        return False

def visualisasi(data,x,y):
    df = pd.DataFrame(data.data, columns=data.feature_names)

```

```

df['Target'] = pd.DataFrame(data.target)
plt.figure(figsize = (10, 6))
colors = ['b','r','g']
string = data.feature_names[x] + ' vs ' + data.feature_names[y]
plt.title(string)
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 = ConvexHull(bucket) #bagian ini diganti dengan hasil implementasi
ConvexHull Divide & Conquer
    plt.scatter(bucket[:, 0], bucket[:, 1], label=data.target_names[i])
    for simplex in hull.simplices:
        plt.plot(bucket[simplex, 0], bucket[simplex, 1], colors[i])
plt.legend()
plt.show()

end = False
print("~~ Tucil 2 Convex Hull Visualizer ~~")
print()
print("~~ Dibuat oleh 13520136 ~~")

while(not end):

    valid = False

    while (not valid):
        print()
        print("Pilih dataset yang ingin dicari convex hull:")
        print("1. Iris")
        print("2. Wine")
        print("3. Breast Cancer")
        pilihan = input("Masukkan angka pilihan: ")
        valid = check_input_dataset(pilihan)
        if (not valid):
            print("input salah!")
            print()
    pilihan = int(pilihan)
    if pilihan == 1:
        data = datasets.load_iris()
    elif pilihan == 2:
        data = datasets.load_wine()

```

```

elif pilihan == 3:
    data = datasets.load_breast_cancer()

    print()
    print("Atribut yang dimiliki tabel (<index>. <nama atribut>):")
    for i in range(len(data.feature_names)):
        print(i, end=". ")
        print(data.feature_names[i])

    valid2 = False
    while(not valid2):
        print()
        print("Pilih nilai x dan y yang ingin ditampilkan pada visualisasi convex
hull (masukan indexnya):")
        x = input("Masukkan x: ")
        y = input("Masukkan y: ")
        valid2 = check_input_xy(x,y,len(data.feature_names))
        if (not valid2):
            print("input salah!")

    x = int(x)
    y = int(y)

    visualisasi(data,x,y)

    valid3 = False
    while (not valid3):
        print()
        pilihan = input("Ingin mencari convex hull untuk data yang lain? (y/n)?
")

        if pilihan.lower() == "n":
            valid3 = True
            end = True
        elif pilihan.lower() == "y":
            valid3 = True
        else:
            print("input salah!")

```



### III. Tes Program

#### A. Data Iris

##### a. sepal length vs sepal width

Input:

```
PS D:\ITB\Tucil\Sem 4\Stima\Tucil2Stima\Tucil-2-ConvexHull\src> py Visualizer.py
~~ Tucil 2 Convex Hull Visualizer ~~

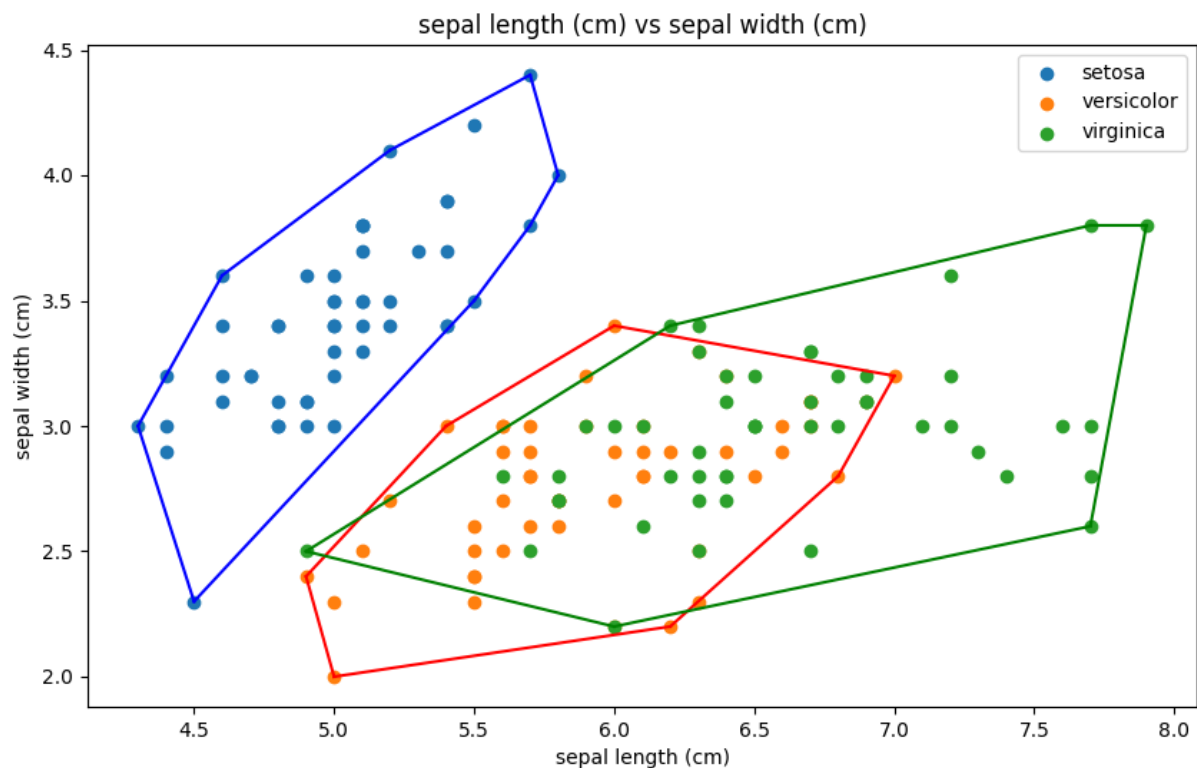
~~ Dibuat oleh 13520136 ~~

Pilih dataset yang ingin dicari convex hull:
1. Iris
2. Wine
3. Breast Cancer
Masukkan angka pilihan: 1

Atribut yang dimiliki tabel (<index>. <nama atribut>):
0. sepal length (cm)
1. sepal width (cm)
2. petal length (cm)
3. petal width (cm)

Pilih nilai x dan y yang ingin ditampilkan pada visualisasi convex hull (masukan indexnya):
Masukkan x: 0
Masukkan y: 1
```

Output:

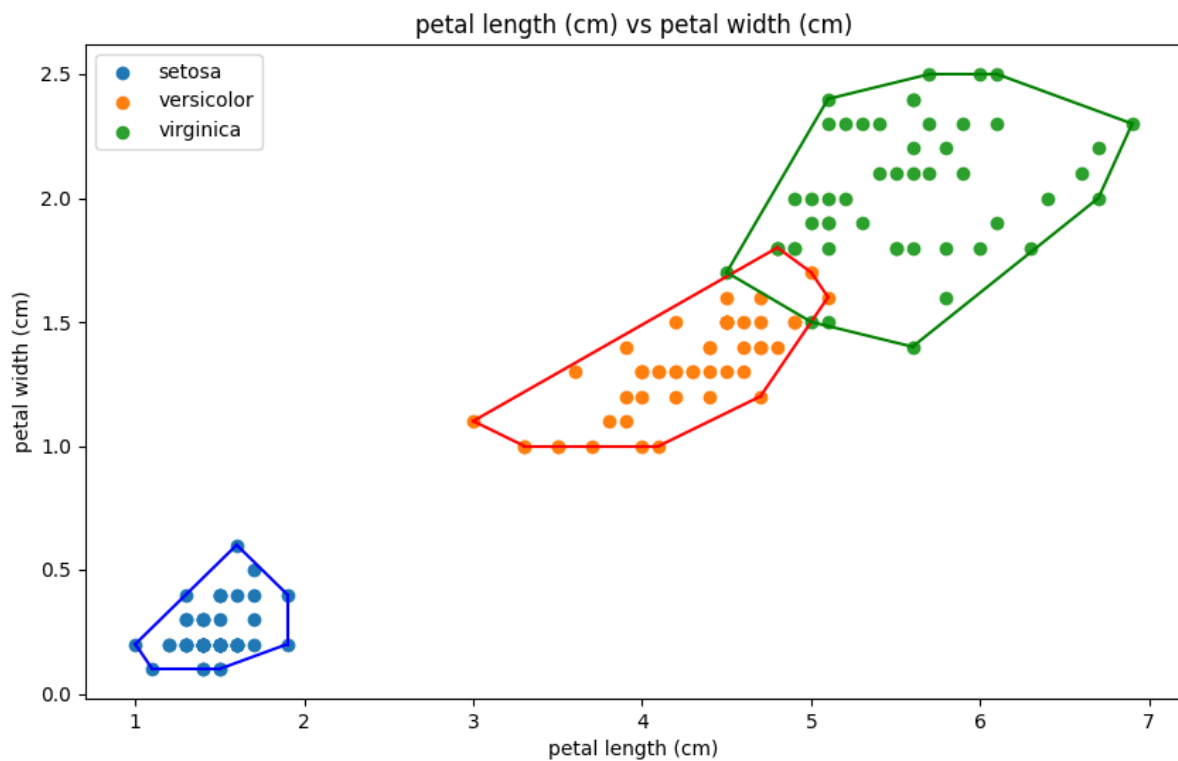


b. petal length vs petal width

Input:

```
Pilih dataset yang ingin dicari convex hull:  
1. Iris  
2. Wine  
3. Breast Cancer  
Masukkan angka pilihan: 1  
  
Atribut yang dimiliki tabel (<index>. <nama atribut>):  
0. sepal length (cm)  
1. sepal width (cm)  
2. petal length (cm)  
3. petal width (cm)  
  
Pilih nilai x dan y yang ingin ditampilkan pada visualisasi convex hull (masukan indexnya):  
Masukkan x: 2  
Masukkan y: 3
```

Output:



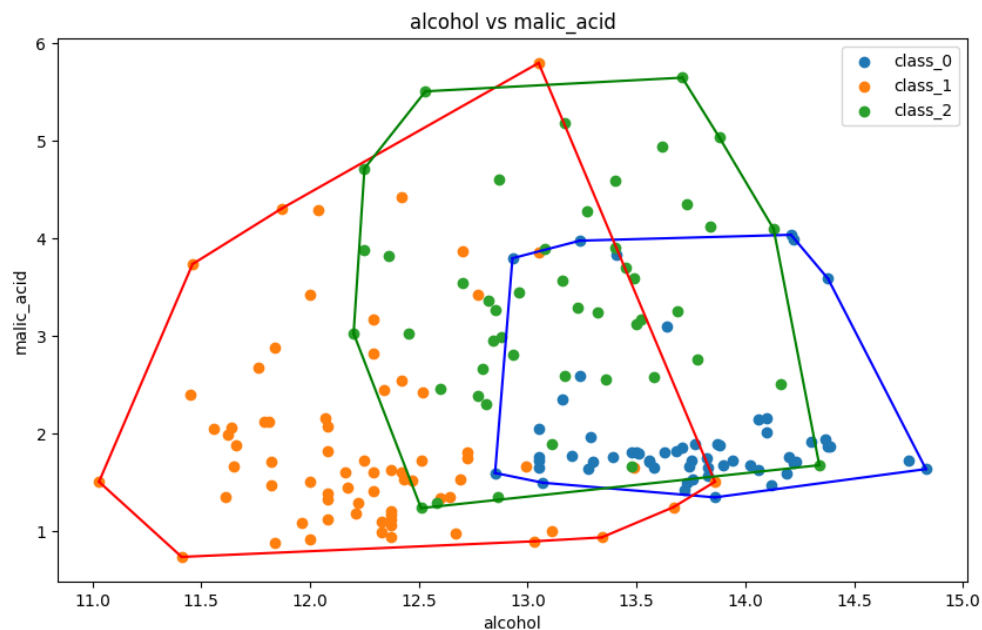
## B. Data Wine

### a. alcoholic vs malic\_acid

Input:

```
Pilih dataset yang ingin dicari convex hull:  
1. Iris  
2. Wine  
3. Breast Cancer  
Masukkan angka pilihan: 2  
  
Atribut yang dimiliki tabel (<index>. <nama atribut>):  
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  
  
Pilih nilai x dan y yang ingin ditampilkan pada visualisasi convex hull (masukan indexnya):  
Masukkan x: 0  
Masukkan y: 1
```

Output:

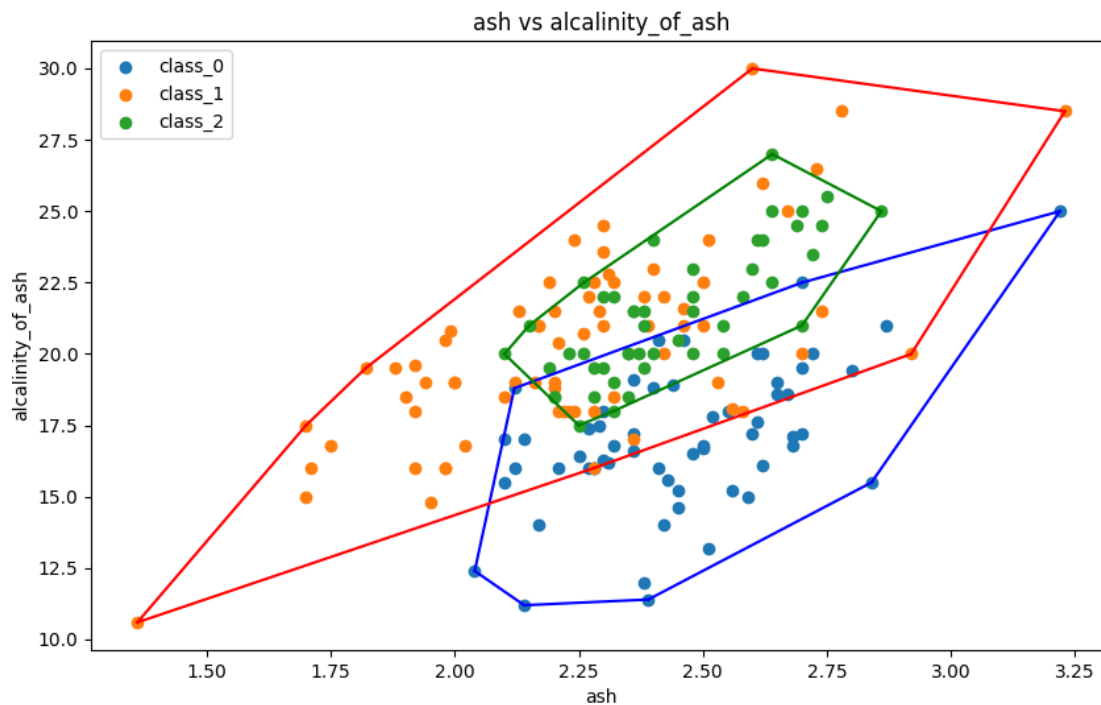


b. ash vs alcalinity\_of\_ash

Input:

```
Pilih dataset yang ingin dicari convex hull:  
1. Iris  
2. Wine  
3. Breast Cancer  
Masukkan angka pilihan: 2  
  
Atribut yang dimiliki tabel (<index>. <nama atribut>):  
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  
  
Pilih nilai x dan y yang ingin ditampilkan pada visualisasi convex hull (masukan indexnya):  
Masukkan x: 2  
Masukkan y: 3
```

Output:

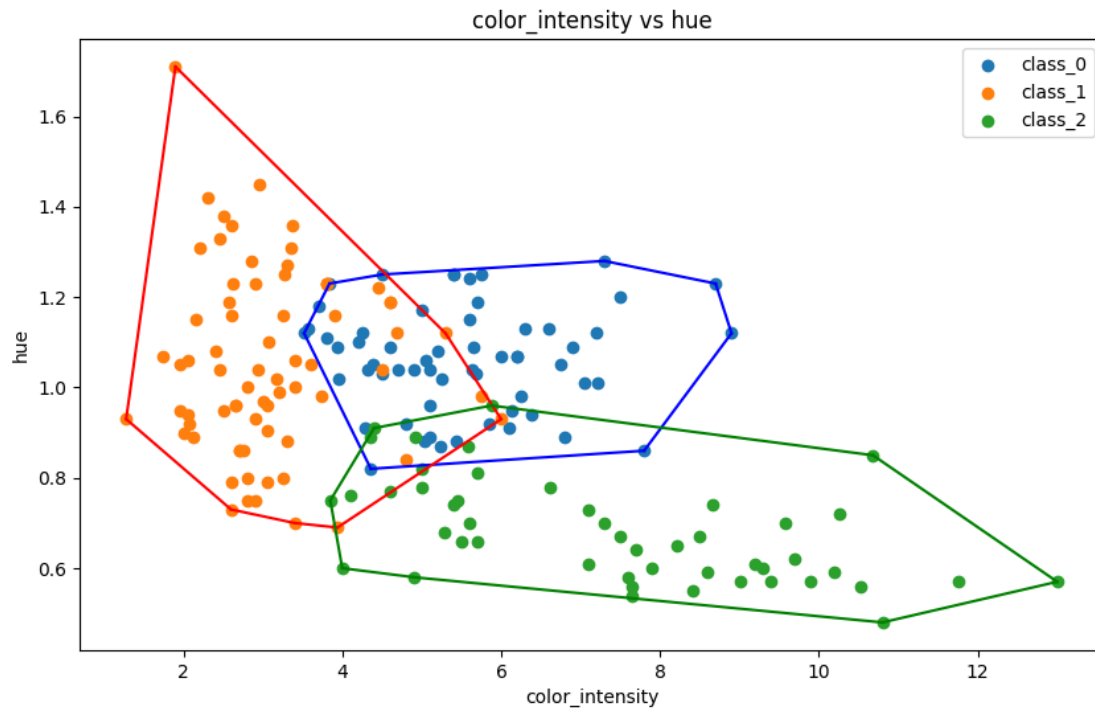


c. color\_intensity vs hue

Input:

```
Pilih dataset yang ingin dicari convex hull:  
1. Iris  
2. Wine  
3. Breast Cancer  
Masukkan angka pilihan: 2  
  
Atribut yang dimiliki tabel (<index>. <nama atribut>):  
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  
  
Pilih nilai x dan y yang ingin ditampilkan pada visualisasi convex hull (masukan indexnya):  
Masukkan x: 9  
Masukkan y: 10
```

Output:



### C. Data Breast Cancer

#### a. mean radius vs mean texture

Input:

Pilih dataset yang ingin dicari convex hull:

1. Iris
2. Wine
3. Breast Cancer

Masukkan angka pilihan: 3

Atribut yang dimiliki tabel (<index>. <nama atribut>):

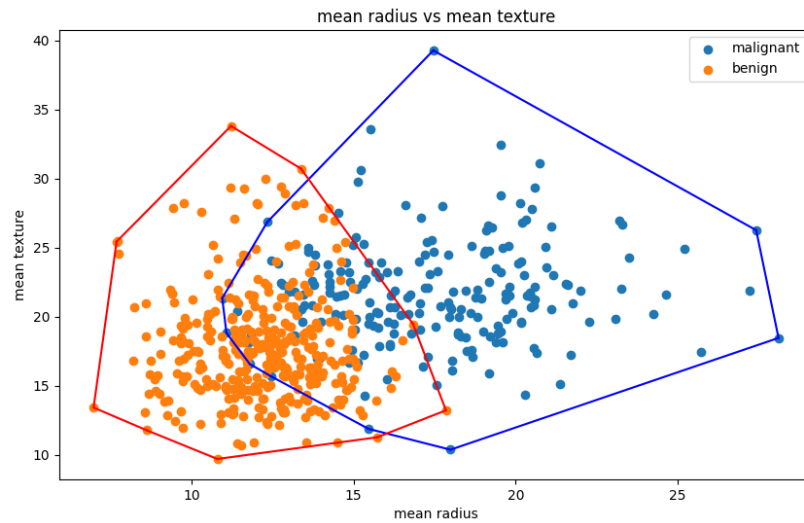
0. mean radius
1. mean texture
2. mean perimeter
3. mean area
4. mean smoothness
5. mean compactness
6. mean concavity
7. mean concave points
8. mean symmetry
9. mean fractal dimension
10. radius error
11. texture error
12. perimeter error
13. area error
14. smoothness error
15. compactness error
16. 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

Pilih nilai x dan y yang ingin ditampilkan pada visualisasi convex hull (masukan indexnya):

Masukkan x: 0

Masukkan y: 1

Output:



b. worst radius vs worst texture

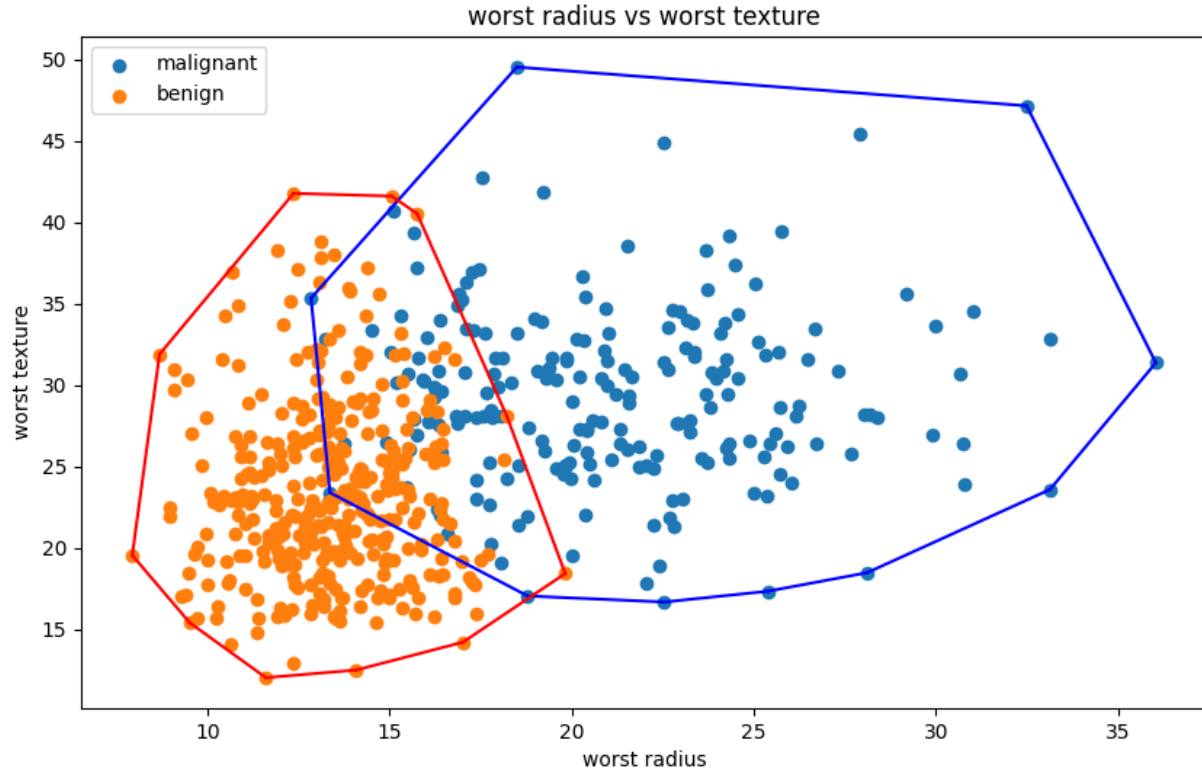
Input:

```
Pilih dataset yang ingin dicari convex hull:
1. Iris
2. Wine
3. Breast Cancer
Masukkan angka pilihan: 3

Atribut yang dimiliki tabel (<index>. <nama atribut>):
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

Pilih nilai x dan y yang ingin ditampilkan pada visualisasi convex hull (masukan indexnya):
Masukkan x: 20
Masukkan y: 21
```

Output:



#### IV. Git Hub Link

<https://github.com/Vincent136/Tucil-2-ConvexHull>

#### V. Tabel

| Poin   | Ya                                  | Tidak                    |
|--|-------------------------------------|--------------------------|
| Pustaka <i>myConvexHull</i> berhasil dibuat dan tidak ada kesalahan  | <input checked="" type="checkbox"/> | <input type="checkbox"/> |
| <i>Convex hull</i> yang dihasilkan sudah benar   | <input checked="" type="checkbox"/> | <input type="checkbox"/> |
| Pustaka <i>myConvexHull</i> dapat digunakan untuk menampilkan <i>convex hull</i> setiap label dengan warna yang berbeda. | <input checked="" type="checkbox"/> | <input type="checkbox"/> |
| <b>Bonus:</b> program dapat menerima input dan menuliskan output untuk dataset lainnya.                                  | <input checked="" type="checkbox"/> | <input type="checkbox"/> |