**LAPORAN TUGAS KECIL 2 IF2211 STRATEGI ALGORITMA**

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

Logo

Description automatically generated

**PUTRI NURHALIZA**

**13520066**

**TEKNIK INFORMATIKA**

**INSTITUT TEKNOLOGI BANDUNG**

**SEMESTER II 2021/202**

# Daftar Isi

[Daftar Isi i](#_Toc96981129)

[A. Algoritma *Divide and Conquer* pada Convex Hull 1](#_Toc96981130)

[B. *Source Code* Program (Bahasa Pemograman Python) 3](#_Toc96981131)

[C. Input dan Output Hasil Eksekusi 7](#_Toc96981132)

[D. *Repository* Program (Github) 10](#_Toc96981133)

[E. Rangkuman Keberjalanan Program 10](#_Toc96981134)

# Algoritma *Divide and Conquer* pada Convex Hull

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. Algoritma *divide and conquer* digunakan pada pustaka myConvexHull untuk menemukan titik-titik terluar yang membentuk convex hull.

Sebelum dilakukan *divide*, titik pada di bidang kartesian dua dimensi diurutkan berdasarkan nilai absis yang menaik, dan ordinat yang menaik jika ada nilai absis yang sama. Kemudian tentukan dan yang merupakan titik ekstrim yang akan membentuk convex hull tersebut. Setelah itu, dilakukan tahap *divide* sebagai berikut.

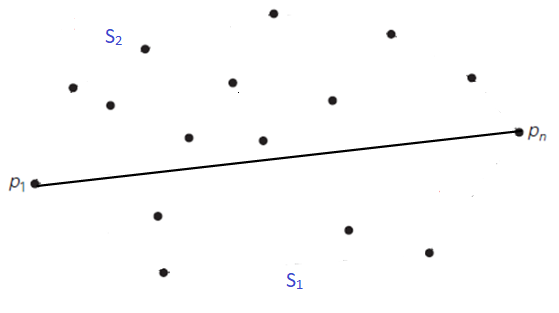
1. dibagi menjadi dan oleh garis yang menghubungkan dan (). Jika ke dianggap memiliki orientasi kedepan, maka berada di sebelah kanan, dan di sebelah kiri. Pemilahan titik-titik ini ditentukan berdasarkan determinan. Misal (), (), dan (), maka determinannya adalah sebagai berikut.

Diagram

Description automatically generated with low confidence

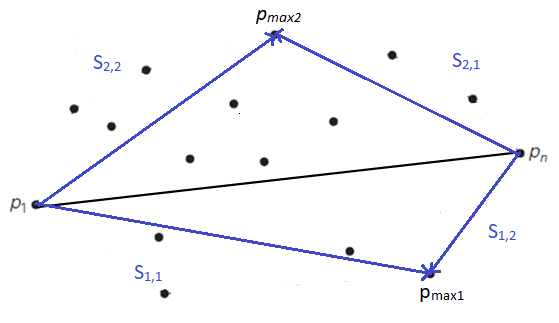
Gambar 1.1. Mencari determinan untuk menentukan arah

Jika hasil determinan positif, artinya titik () berada di sebelah kiri garis , dan sebaliknya. Jika hasil determinan sama dengan 0, artinya titik tersebut berada pada garis sehingga tidak akan dimasukkan ke maupun untuk perhitungan lebih lanjut.



Gambar 1.2. Ilustrasi membagi titik-titik menjadi 2 bagian

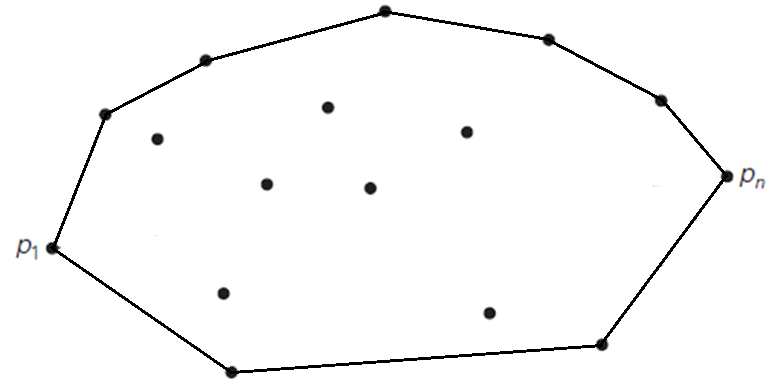
1. Selanjutnya, akan dicari convex hull pada maupun dengan metode rekursif yang kemudian akan membentuk convex hull secara keseluruhan.
2. Pada ,
3. Jika kosong maka tidak ada titik lain yang menjadi pembentuk convex hull pada bagian tersebut.
4. Jika tidak kosong, pilih sebuah titik (misal ) yang berjarak maksimal dari garis . akan terhitung sebagai salah satu titik pembentuk convex hull.
5. Segitiga akan membagi menjadi 3 bagian. Titik-titik yang berada di dalam segitiga maupun di garis atau garis dapat diabaikan.
6. Selanjutnya, tentukan titik titik-titik yang berada di sebelah kanan () dengan melihat determinannya terhadap , serta titik-titik yang berada di sebelah kiri (misal ) dengan melihat determinannya terhadap .
7. Lakukan poin 3 secara rekursif hingga bagian kanan dan kiri kosong.
8. Begitupun pada , lakukan hal yang sama seperti poin 3. Namun, ada sedikit perbedaan pada poin d untuk menjaga konsistensi orientasi. Tentukan titik titik-titik yang berada di sebelah kanan () dengan melihat determinannya terhadap , serta titik-titik yang berada di sebelah kiri (misal ) dengan melihat determinannya terhadap , serta. Hal ini juga dilakukan secara rekursif hingga bagian kanan dan kiri kosong.



Gambar 1.3. Ilustrasi menentukan *pmax*tiap bagian

Pada tahap *combine*, titik , , serta dari hasil rekursi digabungkan sebagai convex hull. Tahap penggabungan ini sebenarnya tidak memerhatikan urutan. Namun, untuk mempermudah plot pada visualisasi, titik-titik tersebut dapat diurutkan searah jarum jam maupun berlawanan arah jarum jam. Sebagai contoh, untuk arah berlawanan jarum jam dapat dilakukan dengan cara berikut,

* + Pada maupun , urutannya adalah (kanan dan kiri berlaku rekursif)
  + Untuk menggabungkan keseluruhan, urutannya adalah



Gambar 1.4. Ilustrasi hasil akhir convex hull

# *Source Code* Program (Bahasa Pemograman Python)

*Source code* program dikelompokkan menjadi tiga.

1. MyConvexHull.py

Pustaka yang digunakan untuk menentukan mendapatkan convex hull.

import numpy as np

def orientation(p1, p2, p3):

    """

        prekondisi: p1, p2, dan p3 merupakan array yang terdefenisi, merepresentasikan titik

        proses: Menentukan posisi titik p3 terhadap garis p1p2 berdasarkan determinan

    """

    det = p1[0]\*p2[1] + p2[0]\*p3[1] + p3[0]\*p1[1] - p3[0]\*p2[1] - p2[0]\*p1[1] - p1[0]\*p3[1]

    if (np.allclose(det, 0)): det = 0.0

    if (det > 0): return 1          # left

    elif (det < 0): return -1       # right

    return 0                        # in line

def pointMax(sub, p1, p2):

    """

        prekondisi: array sub tidak kosong

                    p1 dan p2 merupakan array yang terdefenisi, merepresentasikan titik-titik

        proses: menentukan titik pada array sub yang memiliki jarak terjauh terhadap garis p1p2

                jika ada beberapa titik dengan jarak terjauh, akan terpilih yang pertama

    """

    max = 0.0

    pMax = np.array([])

    for p in sub:

        d = np.linalg.norm(np.cross(p2-p1, p1-p))/np.linalg.norm(p2-p1)

        if (d > max):

            max = d

            pMax = p

    return pMax

def addHull(sub, pBegin, pEnd):

    """

        prekondisi: array sub terdefenisi, boleh kosong

                    pBegin dan pEnd merupakan array yang terdefenisi, merepresentasikan titik

                    dan merupakan convex hull yang sudah diketahui di sisi tersebut

        proses: menentukan convex hull pada suatu bagian, dilakukan secara rekursif

    """

    if len(sub) == 0:

        return np.empty((0,2), float)

    else:

        pMax = pointMax(sub, pBegin, pEnd)

        result = right = left = np.empty((0,2), float)

        for point in sub:

            if orientation(pBegin, pMax, point) == 1: left = np.append(left, [point], axis=0)

            elif orientation(pEnd, pMax, point) == -1: right = np.append(right, [point], axis=0)

        result = np.append(result, addHull(right, pMax, pEnd), axis=0)

        result = np.append(result, [pMax], axis=0)

        result = np.append(result, addHull(left, pBegin, pMax), axis=0)

        return result

def myConvexHull(points):

    """

        prekondisi: array points terdefenisi, minimal terdapat 2 titik

        proses: melakukan divide and conquer untuk menentukan convex hull keseluruhan

    """

    points = points[np.lexsort((points[:, 1], points[:, 0]))]

    p1 = points[0]

    pn = points[-1]

    result = right = left = np.empty((0,2), float)

    for point in points:

        if orientation(p1, pn, point) == 1: left = np.append(left, [point], axis=0)

        elif orientation(p1, pn, point) == -1: right = np.append(right, [point], axis=0)

    result = np.append(result, [p1], axis=0)

    result = np.append(result, addHull(right, pn, p1), axis=0)

    result = np.append(result, [pn], axis=0)

    result = np.append(result, addHull(left, p1, pn), axis=0)

    return result

1. Visualization.py

Menampilkan visualisasi convex hull yang dihasilkan dari suatu dataset dan atribut yang dipilih *user*. Visualisasi akan ditampilkan dengan *pop* *up* matplotlib. User perlu menutup *preview* tersebut untuk melanjutkan program.

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

from MyConvexHull import myConvexHull

def visualization(data, x, y, dsName):

    """

        prekondisi: data terdefinisi untuk suatu dataset dsName, x dan y indeks atribut data yang valid

        proses: membentuk dataframe

                menentukan convex hull dari data dengan myConvexhull()

                menampilkan visualisasi

                menyimpan visualisasi jika diinginkan user

    """

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

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

    plt.figure(figsize = (10, 6))

    colors = ['b','orange','g','r', 'purple', 'brown', 'pink', 'gray', 'y', 'cyan']

    title = f"{data.feature\_names[x].upper()} vs {data.feature\_names[y].upper()} ({dsName})"

    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

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

        hull = myConvexHull(bucket)

        # print(hull)

        plt.plot(np.append(hull[:, 0], hull[0, 0]), np.append(hull[:, 1], hull[0, 1]), colors[i])

    plt.legend()

    save = input("\nSave image (y/n)? ")

    if (save == 'y'):

        plt.savefig('../test/' + title + '.png')

    plt.show()

1. main.py

Program utama yang berisi alur program serta *interaksi* dengan user. Termasuk sebuah fungsi untuk *load* dataset yang diinginkan. Program akan terus berjalan jika *user* masih ingin mencoba tes yang lain.

from Visualization import \*

from sklearn import datasets

def loadData(opt):

    """

        prekondisi: opt memiliki nilai yang valid untuk refer ke dataset (1-4)

        proses: meload data yang sesuai

                menginput pilihan attribute dari user

                memanggil fungsi visualisasi jika valid

    """

    dsNames = ["IRIS", "WINE", "BREAST\_CANCER", "DIGITS"]

    if (opt == 1): data = datasets.load\_iris()

    elif (opt == 2): data = datasets.load\_wine()

    elif (opt == 3): data = datasets.load\_breast\_cancer()

    elif (opt == 4): data = datasets.load\_digits()

    print(f"\nAvailable attributes: ")

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

        print(f"{i+1}. {data.feature\_names[i]}")

    print(f"\nChoose attribute (x and y): ")

    x = int(input("x: ")) - 1

    y = int(input("y: ")) - 1

    maxCol = len(data.feature\_names) - 1

    if (x < 0 or x > maxCol or y < 0 or y > maxCol):

        print("The choosen attribute is invalid")

    else:

        visualization(data, x, y, dsNames[opt-1])

# ---------------------------- PROGRAM UTAMA --------------------------------

print("===================LINEAR SEPARABILITY TEST===================")

isRun = True

while (isRun):

    print("\n1. Iris")

    print("2. Wine")

    print("3. Breast\_cancer")

    print("4. Digits")

    opt = int(input("Choose dataset: "))

    if (opt > 0 and opt <= 4):

        loadData(opt)

    else:

        print("Dataset not available")

    run = input("\nTry another test (y/n)? ")

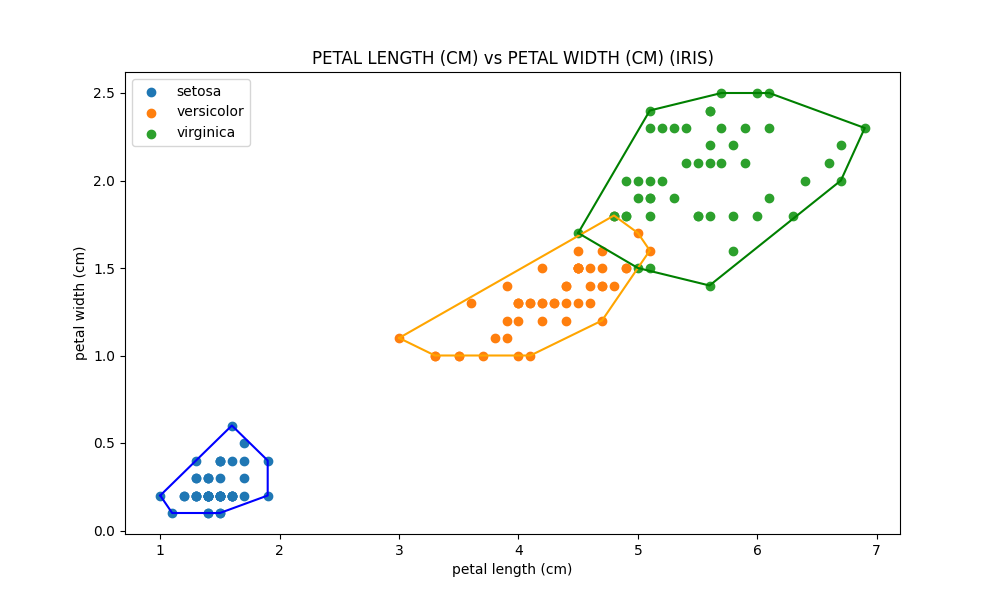
    isRun = True if run == 'y' else False

# Input dan Output Hasil Eksekusi

Program akan meminta *user* untuk menginput pilihan dataset yang ingin diuji beserta atributnya. Hasil visualisasi akan disimpan di dalam folder */test* jika diinginkan.

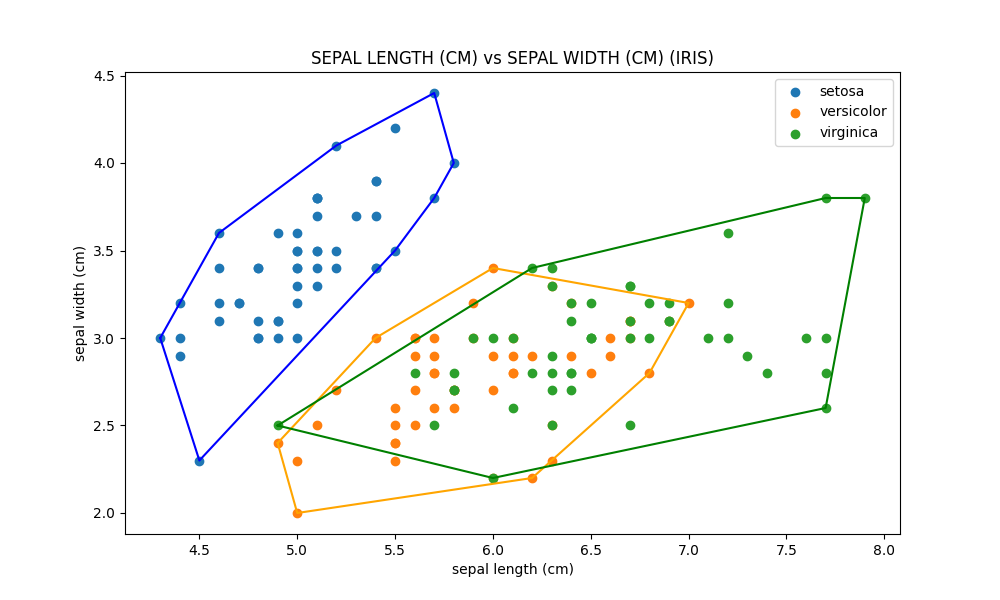
1. Petal length vs Petal Width (IRIS)

Text

Description automatically generated 

1. Sepal length vs Sepal Width (IRIS)

Text

Description automatically generated 

1. Alcohol vs Malic\_acid (WINE)

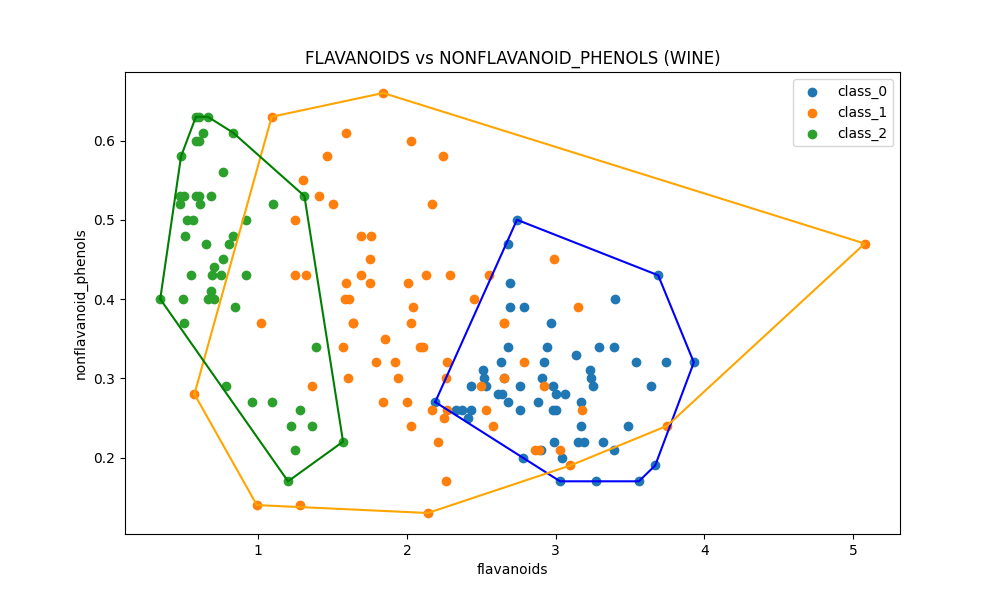
Text

Description automatically generated Chart, radar chart

Description automatically generated

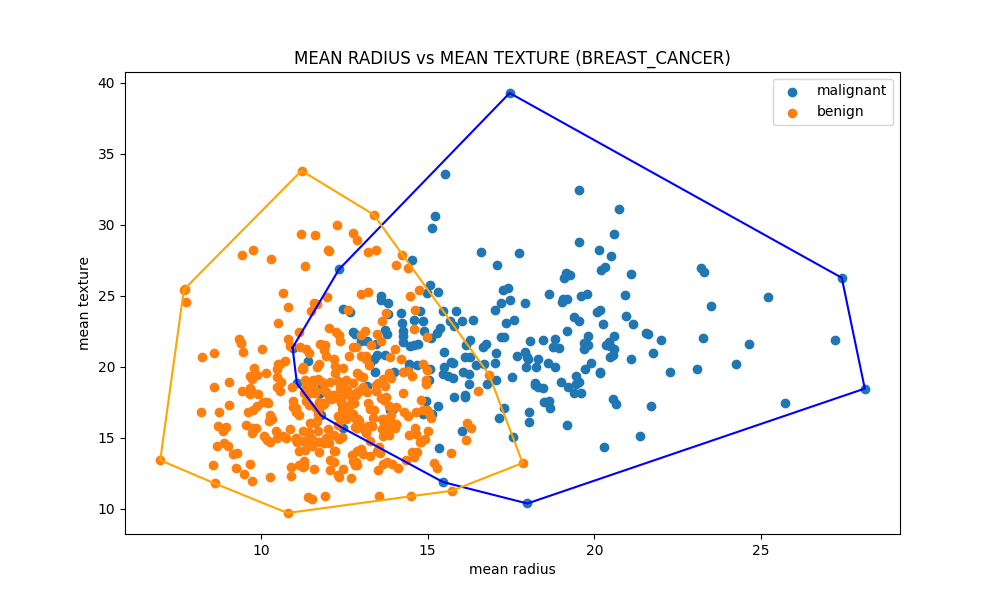
1. Flavanoids vs Nonflavanoid\_phenols (WINE)

Text

Description automatically generated 

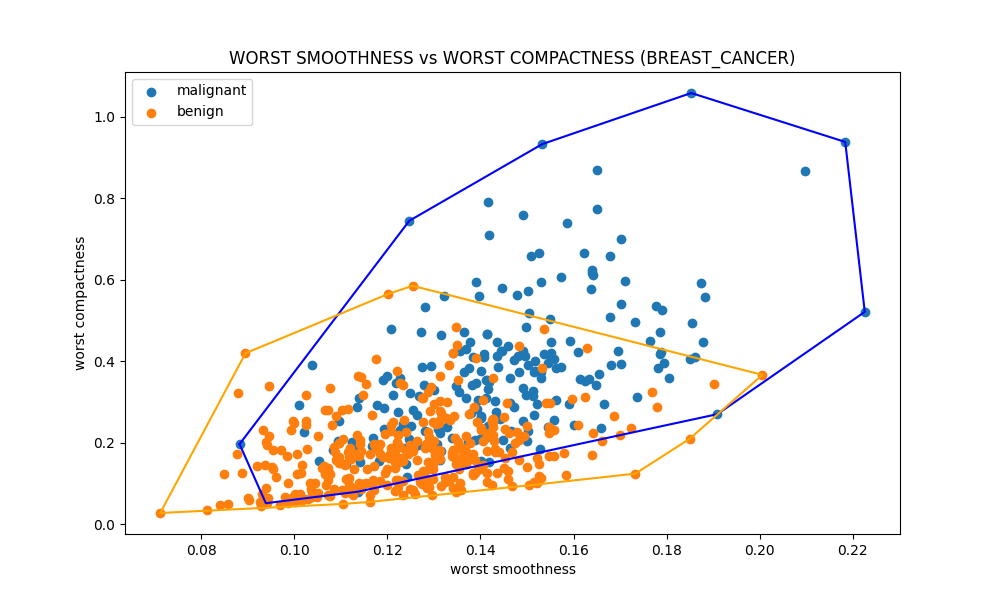
1. Mean Radius vs Mean Texture (BREAST\_CANCER)

Text

Description automatically generated 

1. Worst Smoothness vs Worst Compactness (BREAST\_CANCER)

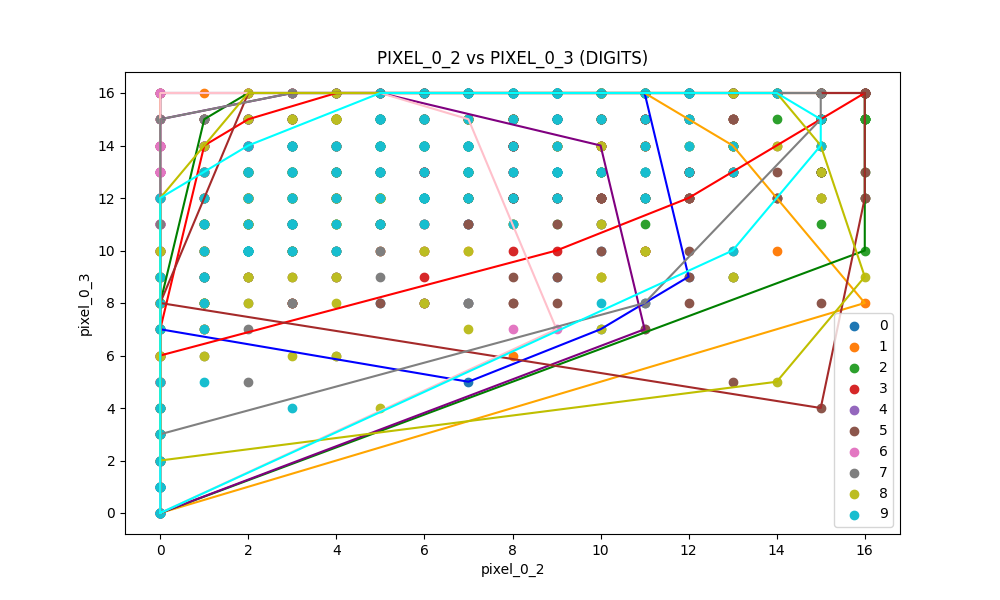
Text

Description automatically generated

1. Pixel\_0\_2 vs Pixel\_0\_3 (DIGITS)

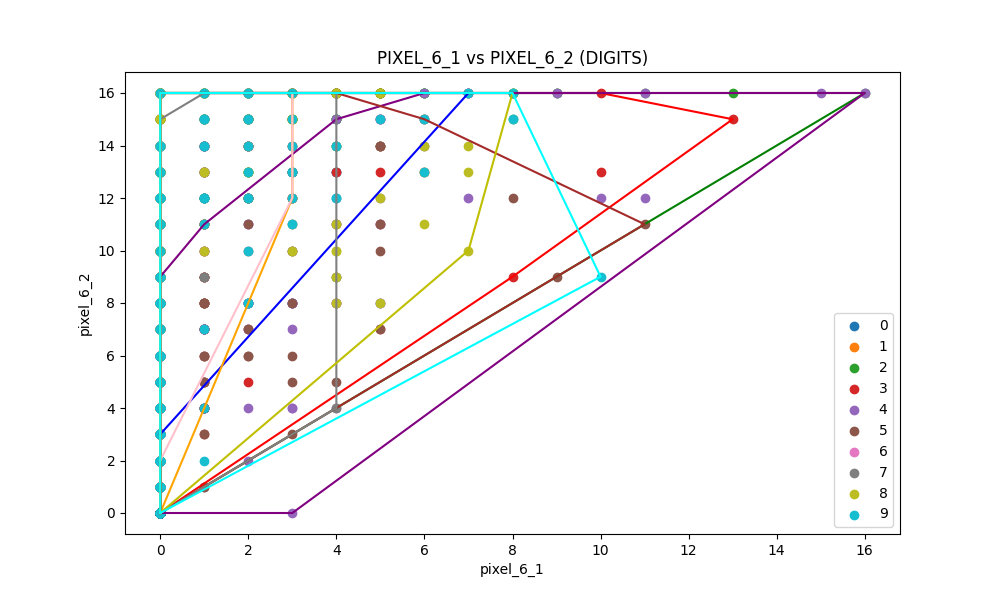
Text

Description automatically generated with medium confidence A picture containing text, keyboard

Description automatically generated

1. Pixel\_6\_1 vs Pixel\_6\_2 (DIGITS)

Text

Description automatically generated with medium confidence

# *Repository* Program (Github)

Untuk menjalankan program, *source code* bisa didapatkan di

https://github.com/Putriliza/Tucil2Stima-ConvexHull

# Rangkuman Keberjalanan Program

|  |  |  |
| --- | --- | --- |
| **Poin** | **Ya** | **Tidak** |
| 1. Pustaka myConvexHull berhasil dibuat dan tidak ada kesalahan | ✓ | - |
| 1. Convex hull yang dihasilkan sudah benar | ✓ | - |
| 1. Pustaka myConvexHull dapat digunakan untuk menampilkan convex hull setiap label dengan warna yang berbeda. | ✓ | - |
| 1. Bonus: program dapat menerima input dan menuliskan output untuk dataset lainnya. | ✓ | - |