LAPORAN TUGAS KECIL 2

IF2211 Strategi Algoritma

Convex Hull dengan Algoritma Divide and Conquer

oleh

Nadia Mareta Putri Leiden (13520007)

PROGRAM STUDI TEKNIK INFORMATIKA

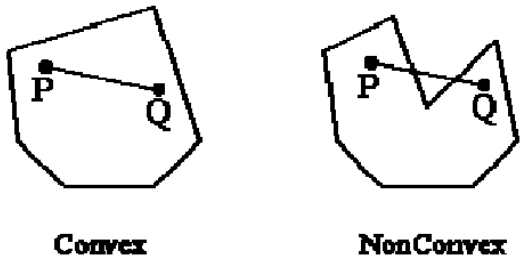
SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

2022

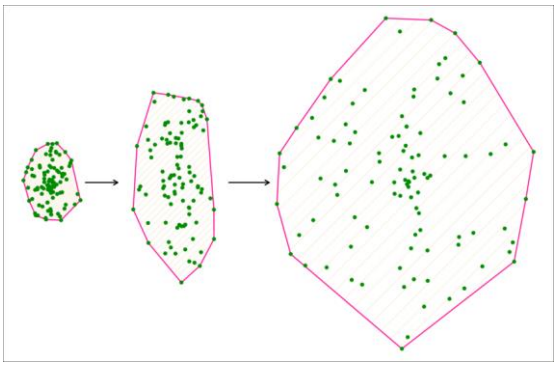
1. Algoritma Divide and Conquer untuk Convex Hull

Sebuah bidang akan disebut sebagai *convex* / konveks jika terdapat dua titik terpisah pada bidang tersebut yang jika disambungkan tidak akan keluar dari bidang. Berikut adalah ilustrasi dari bidang yang disebut konveks (*convex*).



sumber: researchgate.com

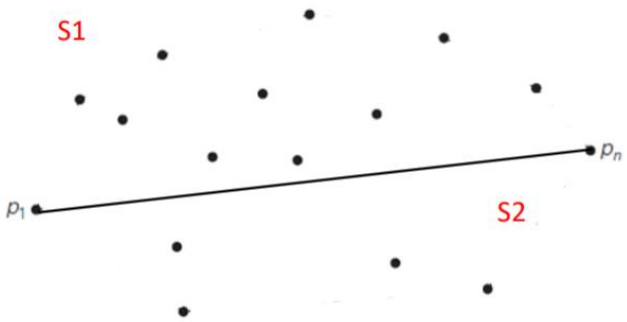
Adapun yang dimaksud dengan *Convex Hull* adalah himpunan suatu titik / *convex* terkecil yang memuat himpunan keseluruhan titik-titik. Kemudian, *Convex Hull* ini digunakan dalam beberapa aplikasi animasi komputer salah satu contohnya adalah dalam *collision detection* yang memanfaatkan *Convex Hull*.



sumber: researchgate.com

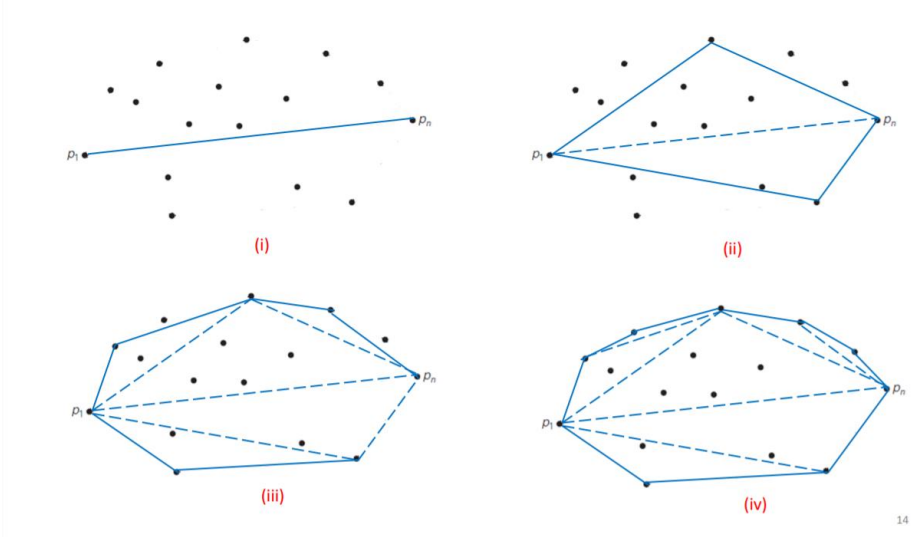
Selain pada animasi komputer, penggunaan *Convex Hull* juga dilakukan dalam bidang statistic yaitu digunakan untuk pendeteksian *outliers* yang terjadi pada beberapa kumpulan data.

Ide dari Algoritma Divide and Conquer untuk *Convex Hull* dilakukan dalam beberapa tahapan sebagai berikut:



sumber: Algoritma Divide and Conquer (2022) Bagian 4, Rinaldi Munir

1. Pertama, membagi dua bagian kumpulan titik-titik menjadi dua himpunan titik-titik yang masing-masing akan diterapkan Algoritma Divide and Conquer



sumber: Algoritma Divide and Conquer (2022) Bagian 4, Rinaldi Munir

2. Kedua, mencari titik terjauh dari masing-masing daerah (S1 dan S2) yang mungkin dari garis dari dua titik terjauh yang telah ditentukan sebelumnya, dalam hal ini adalah titik *p1* dan *pn* pada ilustrasi di atas.
3. Kemudian langkah ini akan terus diulangi dan dibagi menjadi beberapa kasus yang lebih kecil
4. Algoritma berakhir dengan penggabungan partisi-partisi yang telah dilakukan sebelumnya.

2. Source Code Algoritma Divide and Conquer untuk *Convex Hull*

A. Source Code

```
class Point:
    def __init__(self, x, y, idx):
        self.x = x
        self.y = y
        self.idx = idx

#memberikan nomor urut terhadap data yang telah diambil dari dataset Iris
def give_nomor_urut(bucket):
    point_dict = []
    for i in range (len(bucket)):
        p1 = Point(bucket[i][0], bucket[i][1], i)
        point_dict.append(p1)
    return point_dict

#mengambil area konveks sesuai dengan urutan P1-P2 (sehingga penggunaannya tinggal dibalik, jika ingin arah yang berlawanan P2-P1)
#mencari daerah S1 / S2
def area_convex(P1, P2, point_dict):
    area = []
    for i in range (len(point_dict)):
        if(point_dict[i] != P1 and point_dict[i] != P2):
            P3 = point_dict[i]
            det = P1.x*P2.y + P3.x*P1.y + P2.x*P3.y - P3.x*P2.y - P2.x*P1.y - P1.x*P3.y
            if(det < 0):
                area.append(P3)
    return area
```

```
#mencari titik paling jauh setelah mendapatkan kumpulan titik2
def far_point(P1, P2, area):
    array1 =np.array([P1.x, P1.y])
    array2 = np.array([P2.x, P2.y])
    array3 = np.array([area[0].x, area[0].y])
    max = (np.abs(np.cross(array2-array1, array1-array3)) / norm(array2-array1))
    point = area[0]
    for i in range (1, len(area)):
        array3 = np.array([area[i].x, area[i].y])
        length = (np.abs(np.cross(array2-array1, array1-array3)) / norm(array2-array1))
        if(max < length):
            max = length
            point = area[i]
    return point
```

```python
#mencari titik-titik hull yang dikelompokkan menjadi 1 array
def findHull(area, P1, P2, point_dict):
    if(len(area) == 0):
        return [[P1.idx, P2.idx]]
    elif(len(area) == 1):
        return [[P1.idx, area[0].idx], [area[0].idx, P2.idx]]
    else:
        solution = []
        far = far_point(P1, P2, area)
        area_below = area_convex(P1, far, point_dict)
        area_upper = area_convex(far, P2, point_dict)
        below_hull = findHull(area_below, P1, far, point_dict)
        upper_hull = findHull(area_upper, far, P2, point_dict)

        #tambahkan
        for i in below_hull:
            solution.append(i)
        for i in upper_hull:
            solution.append(i)
        return solution
```

```python
def myConvexHull(bucket):
    point_dict = give_nomor_urut(bucket)
    full_solution = []
    #minimum point
    P1 = point_dict[0]
    #maximum point
    P2 = point_dict[0]
    for i in range (1, len(point_dict)):
        current_point = Point(point_dict[i].x, point_dict[i].y, point_dict[i].idx)
        if(P1.x > current_point.x):
            P1 = current_point
        elif(P1.x == current_point.x):
            if(P1.y > current_point.y):
                P1 = current_point
        if(P2.x < current_point.x):
            P2 = current_point
        elif(P2.x == current_point.x):
            if(P2.y < current_point.y):
                P2 = current_point
    #atas / kiri
    left_area = area_convex(P2, P1, point_dict)

    #bawah / kanan
    right_area = area_convex(P1, P2, point_dict)

    #mencari daerah kiri/atas
    upper_solution = findHull(left_area, P2, P1, point_dict)

    #mencari daerah kanan/bawah
    bottom_solution = findHull(right_area, P1, P2, point_dict)
```

```
full_solution = []
for i in range (len(upper_solution) + len(bottom_solution)):
    if(i < len(upper_solution)):
        full_solution.append(upper_solution[i])
    elif(i >= len(upper_solution)):
        full_solution.append(bottom_solution[i-len(upper_solution)])
return full_solution
```

B. Visualisasi Data

```
data = datasets.load_iris()
#create a DataFrame
df = pd.DataFrame(data.data, columns=data.feature_names)
df['Target'] = pd.DataFrame(data.target)
print(df.shape)
df.head()

plt.figure(figsize = (10, 6))
colors = ['b','r','g']
plt.title('Sepal Width vs Sepal Length')
plt.xlabel(data.feature_names[0])
plt.ylabel(data.feature_names[1])
for i in range(len(data.target_names)):
    bucket = df[df['Target'] == i]
    bucket = bucket.iloc[:,[0,1]].values
    hull = myConvexHull(bucket)
    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()
```

Data Sepal Width and Sepal Length

```
data = datasets.load_iris()
#create a DataFrame
df = pd.DataFrame(data.data, columns=data.feature_names)
df['Target'] = pd.DataFrame(data.target)
print(df.shape)
df.head()

plt.figure(figsize = (10, 6))
colors = ['b','r','g']
plt.title('Petal Width vs Petal Length')
plt.xlabel(data.feature_names[2])
plt.ylabel(data.feature_names[3])
for i in range(len(data.target_names)):
    bucket = df[df['Target'] == i]
    bucket = bucket.iloc[:,[2,3]].values
    hull = myConvexHull(bucket)
    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()
```

Data Petal Width and Petal Length

```
data = datasets.load_wine()
#create a DataFrame
df = pd.DataFrame(data.data, columns=data.feature_names)
df['Target'] = pd.DataFrame(data.target)
print(df.shape)
df.head()

plt.figure(figsize = (10, 6))
colors = ['b','r','g']
plt.title('Malic Acid vs Alcohol')
plt.xlabel(data.feature_names[0])
plt.ylabel(data.feature_names[1])
for i in range(len(data.target_names)):
    bucket = df[df['Target'] == i]
    bucket = bucket.iloc[:,[0,1]].values
    hull = myConvexHull(bucket)
    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()
```

Data Malic Acid vs Alcohol

```
data = datasets.load_wine()
#create a DataFrame
df = pd.DataFrame(data.data, columns=data.feature_names)
df['Target'] = pd.DataFrame(data.target)
print(df.shape)
df.head()

plt.figure(figsize = (10, 6))
colors = ['b','r','g']
plt.title('Alcalinity of Ash vs Ash')
plt.xlabel(data.feature_names[2])
plt.ylabel(data.feature_names[3])
for i in range(len(data.target_names)):
    bucket = df[df['Target'] == i]
    bucket = bucket.iloc[:,[2,3]].values
    hull = myConvexHull(bucket)
    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()
```

Data Alcalinity of Ash vs Ash

```
data = datasets.load_wine()
#create a DataFrame
df = pd.DataFrame(data.data, columns=data.feature_names)
df['Target'] = pd.DataFrame(data.target)
print(df.shape)
df.head()

plt.figure(figsize = (10, 6))
colors = ['b','r','g']
plt.title('Total Phenols vs Magnesium')
plt.xlabel(data.feature_names[4])
plt.ylabel(data.feature_names[5])
for i in range(len(data.target_names)):
    bucket = df[df['Target'] == i]
    bucket = bucket.iloc[:,[4,5]].values
    hull = myConvexHull(bucket)
    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()
```

Data Total Phenols vs Magnesium

```
data = datasets.load_breast_cancer()
#create a DataFrame
df = pd.DataFrame(data.data, columns=data.feature_names)
df['Target'] = pd.DataFrame(data.target)
print(df.shape)
df.head()

plt.figure(figsize = (10, 6))
colors = ['b','r','g']
plt.title('Mean Texture vs Mean Radius')
plt.xlabel(data.feature_names[0])
plt.ylabel(data.feature_names[1])
for i in range(len(data.target_names)):
    bucket = df[df['Target'] == i]
    bucket = bucket.iloc[:,[0,1]].values
    hull = myConvexHull(bucket)
    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()
```

Data Breast Cancer Mean Texture vs Mean Radius

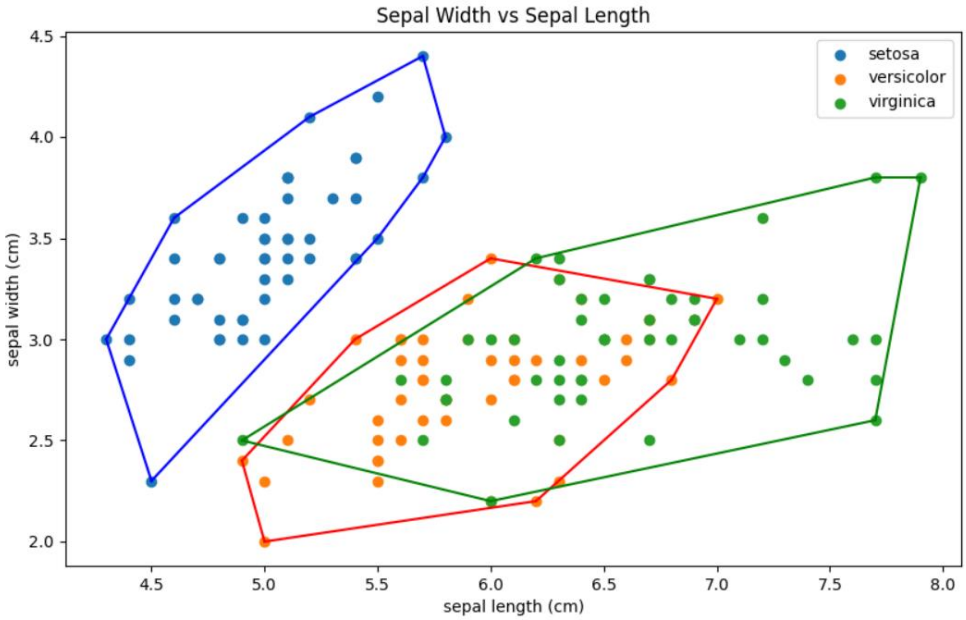
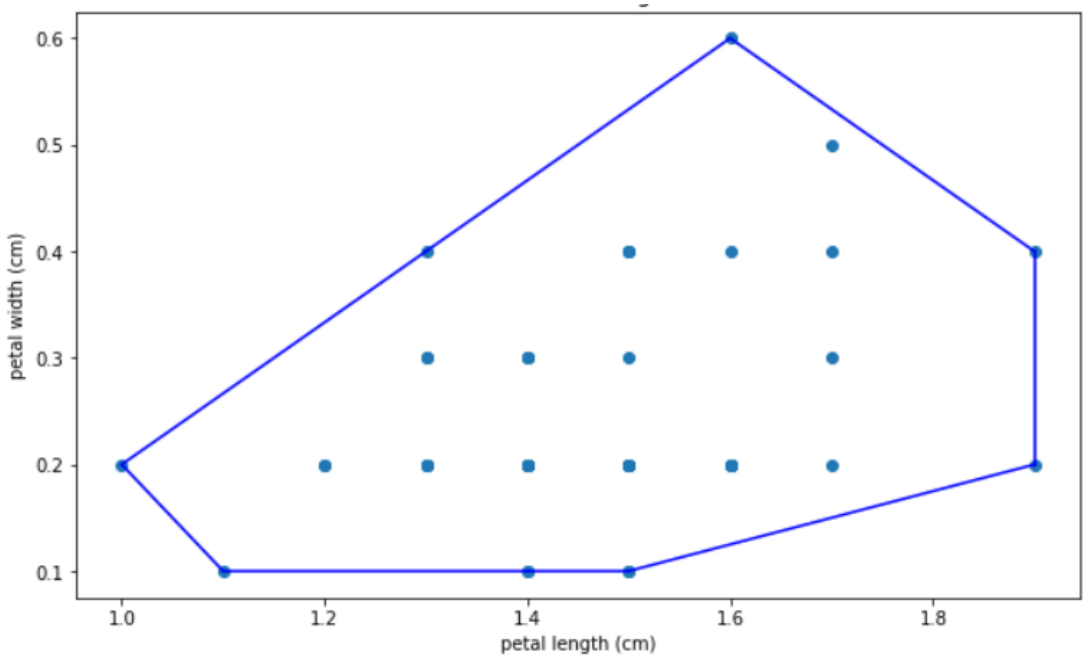
```
data = datasets.load_breast_cancer()
#create a DataFrame
df = pd.DataFrame(data.data, columns=data.feature_names)
df['Target'] = pd.DataFrame(data.target)
print(df.shape)
df.head()

plt.figure(figsize = (10, 6))
colors = ['b','r','g']
plt.title('Mean Area vs Mean Perimeter')
plt.xlabel(data.feature_names[2])
plt.ylabel(data.feature_names[3])
for i in range(len(data.target_names)):
    bucket = df[df['Target'] == i]
    bucket = bucket.iloc[:,[2,3]].values
    hull = myConvexHull(bucket)
    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()
```

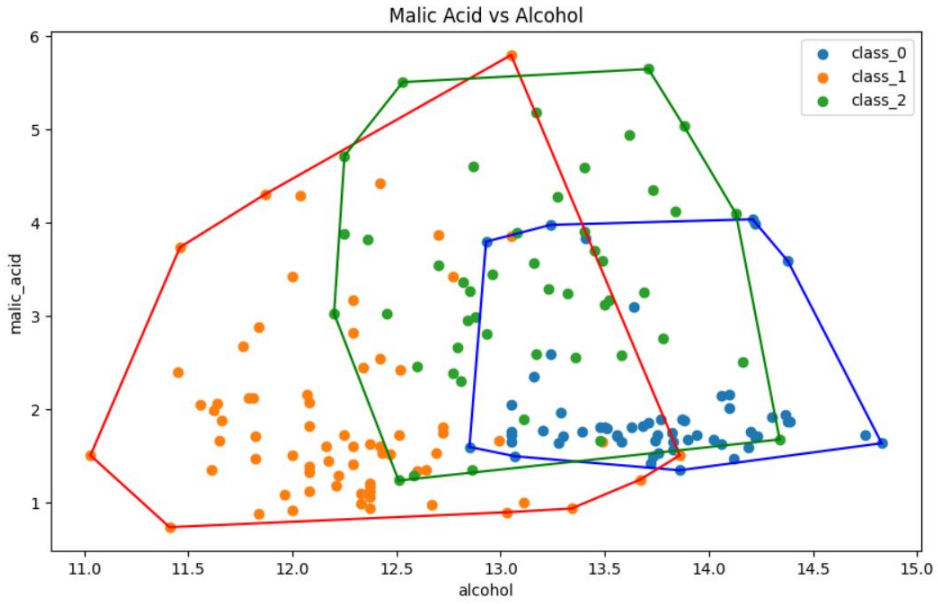
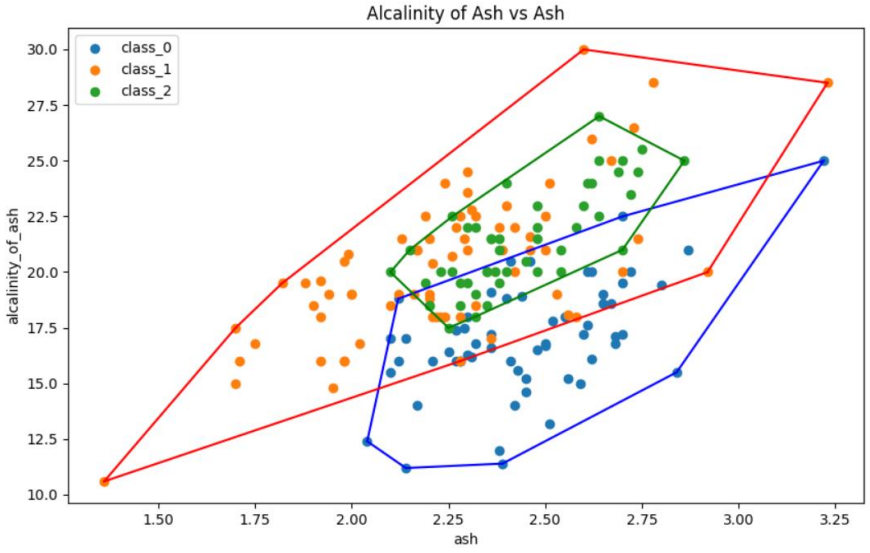
Data Breast Cancer Mean Area vs Mean Parameter

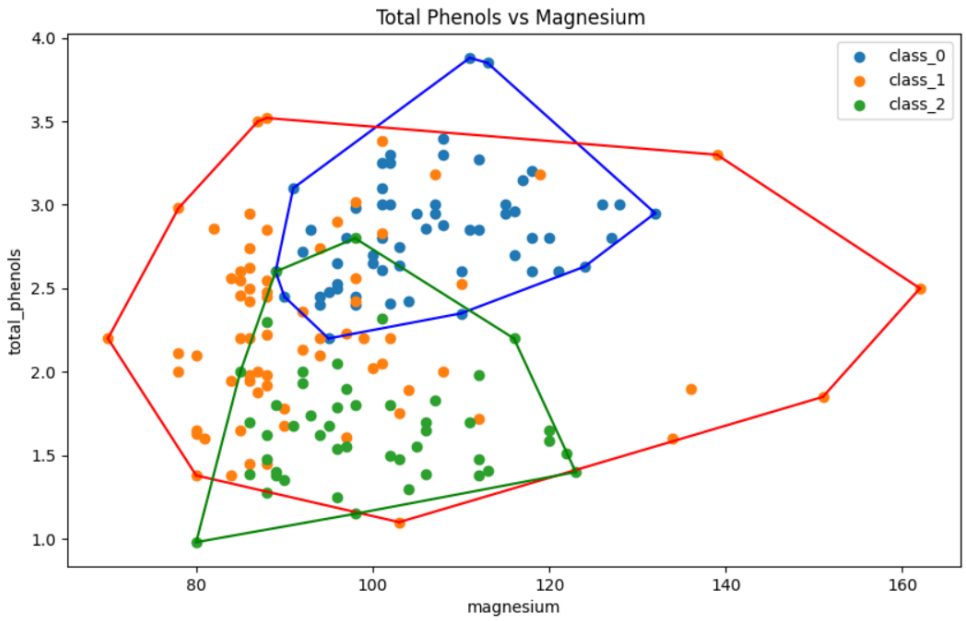
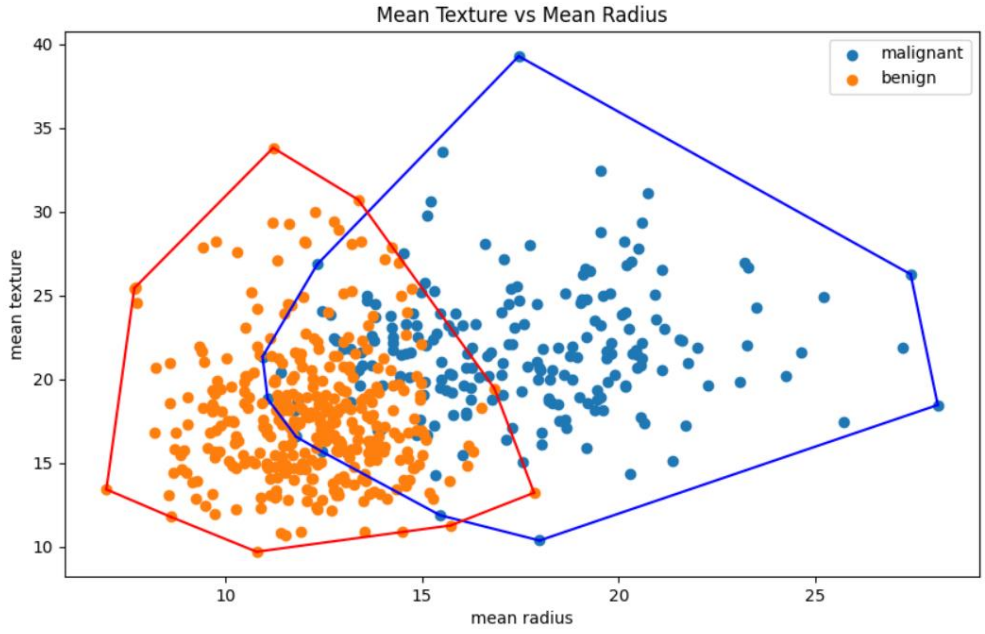
```
data = datasets.load_breast_cancer()
#create a DataFrame
df = pd.DataFrame(data.data, columns=data.feature_names)
df['Target'] = pd.DataFrame(data.target)
print(df.shape)
df.head()

plt.figure(figsize = (10, 6))
colors = ['b','r','g']
plt.title('Mean Compactness vs Mean Smoothness')
plt.xlabel(data.feature_names[4])
plt.ylabel(data.feature_names[5])
for i in range(len(data.target_names)):
    bucket = df[df['Target'] == i]
    bucket = bucket.iloc[:,[4,5]].values
    hull = myConvexHull(bucket)
    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()
```

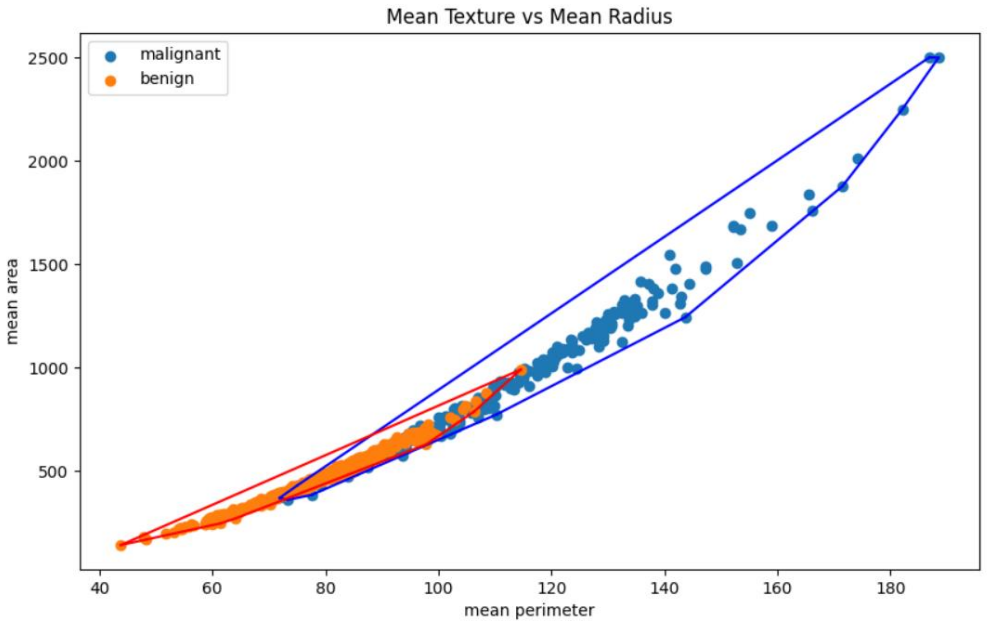
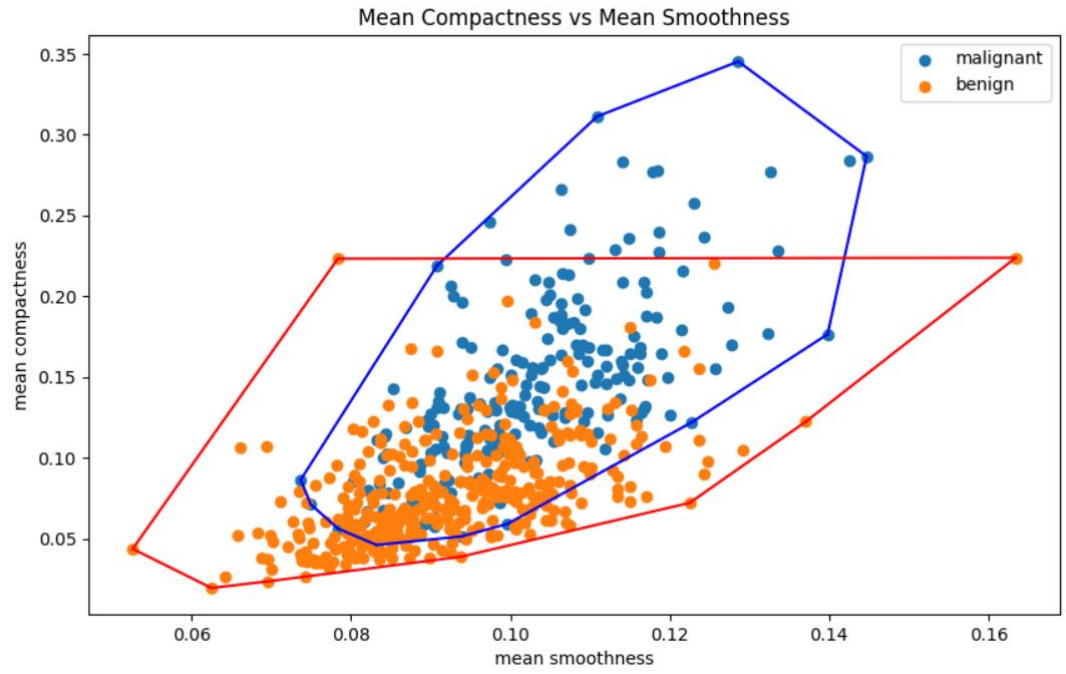
Data Breast Cancer Mean Compactness vs Mean Smoothness

3. Screenshot Pengujian Kode

No	Screenshot
1	
2	Petal error di rekursif, hanya muncul satu saja 

3	 Malic Acid vs Alcohol class_0 class_1 class_2 malic_acid alcohol 1 2 3 4 5 6 11.0 11.5 12.0 12.5 13.0 13.5 14.0 14.5 15.0
4	 Alcalinity of Ash vs Ash class_0 class_1 class_2 alcalinity_of_ash ash 10.0 12.5 15.0 17.5 20.0 22.5 25.0 27.5 30.0 1.50 1.75 2.00 2.25 2.50 2.75 3.00 3.25

5	 Total Phenols vs Magnesium class_0 class_1 class_2 total_phenols magnesium 4.0 3.5 3.0 2.5 2.0 1.5 1.0 80 100 120 140 160
6	 Mean Texture vs Mean Radius malignant benign mean texture mean radius 40 35 30 25 20 15 10 10 15 20 25

7	
8	

4. LAMPIRAN

Link github: https://github.com/KorbanFidas2A/Tucil2_13520007.git

https://github.com/KorbanFidas2A/Tucil2_13520007

Poin	Yes	No
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. Bonus: program dapat menerima input dan menuliskan output untuk dataset lainnya.	√	