LAPORAN TUGAS KECIL

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

Laporan dibuat untuk memenuhi salah satu tugas kecil mata kuliah

IF2211 Strategi Algoritma

Disusun oleh:

I Gede Arya Raditya Parameswara 13520036

PROGRAM STUDI TEKNIK INFORMATIKA

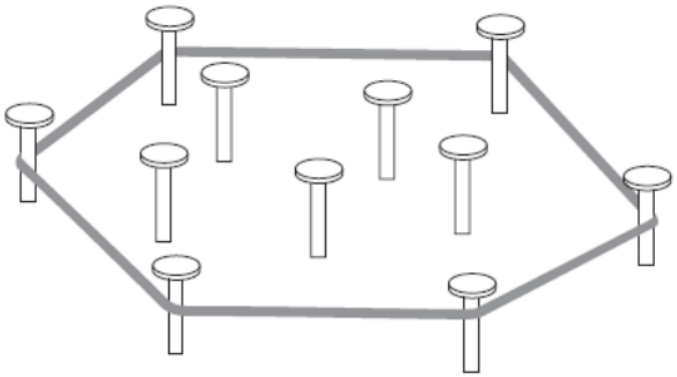
SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

2022

BAB 1
Penerapan Algoritma Divide and Conquer

Pada pembuatan pustaka myConvexHull ini diterapkan algoritma *divide and conquer* untuk mendapatkan titik-titik yang membentuk convex tersebut. Algoritma ini memiliki efektifitas yang sangat baik jika diterapkan pada pustaka myConvexHull ini dibandingkan dengan algoritma *brute force*.



Source: https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2018-2019/Convex-Hull-2019.pdf

Langkah-langkah penyelesaian pustaka myConvexHull menggunakan algoritma *divide and conquer* adalah sebagai berikut,

1. Fungsi ConvexHull pada pustaka menerima parameter berupa list of point berbentuk array 2D yang berisikan titik-titik pada koordinat kartesius berbentuk (x, y).
2. Dilakukan pencarian titik dengan nilai x paling minimum dan titik dengan nilai x paling maksimum.
3. Setelah mendapatkan kedua titik tersebut, program akan melakukan rekursif pada fungsi DivideAndConquerConvexHull dua kali, pertama pada direction 1 (sisi kiri dari garis) dan arah kedua pada sisi sebaliknya.
4. Pada fungsi DivideAndConquerConvexHull ini akan dilakukan beberapa langkah berikut,
    - Mendeklarasi variable idx dengan -1 yang menyatakan index pada array points yang memiliki jarak terjauh dengan garis pada parameter fungsi DivideAndConquerConvexHull.
    - Mengiterasi semua titik pada array of points yang memiliki direction sesuai dengan parameter dan memiliki jarak terjauh dengan garis pada parameter.
    - Jika idx tidak ditemukan atau idx masih bernilai -1 maka garis pada parameter tersebut merupakan bagian dari convexHull sehingga akan dimasukkan array solusi.

- Jika idx ditemukan maka lakukan lagi rekursif fungsi DivideAndConquerConvexHull pada 2 garis (garis pertama yaitu titik pada garis awal pertama dengan titik terjauh dan garis kedua adalah titik terjauh dengan titik pada garis awal kedua) dengan direction yang sama pada

5. Setelah rekursif pada fungsi DivideAndConquerConvexHull selesai maka semua solusi pasangan index convex hull sudah terkumpul pada array dan sisanya tinggal divisualisasikan menggunakan jupyter.

BAB 2
Source Code Program

myConvexHull.py > ConvexHull

```python
def DivideAndConquerConvexHull(res, bucket, IdxLine1, IdxLine2, direction) :
    # PointLine1 dan PointLine2 merupakan titik koordinat yang jika disatukan akan membentuk garis awal
    PointLine1 = bucket[IdxLine1]
    PointLine2 = bucket[IdxLine2]

    # idx merupakan index titik/point yang memiliki jarak terjauh dan garis awal dan berada pada direction
    # yang sesuai dengan parameter
    idx = -1

    # maxDist merupakan jarak terjauh dari semua kemungkinan titik/point dengan garis awal dan berada pada direction
    # direction yang sesuai dengan parameter
    maxDist = 0

    # Melakukan iterasi untuk mencari nilai idx dan maxDist
    for i in range(len(bucket)) :
        # Persamaan dibawah didapatkan dari rumus jarak titik dengan garis pada bidang koordinat
        curDist = ((bucket[i][1] - PointLine1[1]) * (PointLine2[0] - PointLine1[0])
                   - (PointLine2[1] - PointLine1[1]) * (bucket[i][0] - PointLine1[0]))
        if curDist * direction >= 0 and abs(curDist) > maxDist :
            idx = i
            maxDist = abs(curDist)

    # Jika idx tidak berubah maka tidak mendapatkan titik yang sesuai dengan direction dari garis
    # awal, artinya garis awal ini merupakan bagian dari Convex Hull
    if idx == -1 :
        newLineRes = [IdxLine1, IdxLine2]
        if newLineRes not in res :
            res.append(newLineRes)
        return res

    # Jika nilai idx tidak -1 lakukan kembali iterasi pada kedua garis baru hasil dari pasangan
    # titik awal pertama dengan titik terjauh dan titik terjauh dengan titik awal kedua
    res = DivideAndConquerConvexHull(res, bucket, IdxLine1, idx, direction)
    res = DivideAndConquerConvexHull(res, bucket, idx, IdxLine2, direction)

    # Mengembalikan array pasangan index Convex Hull
    return res
```

```python
def ConvexHull(bucket) :
    # Deklarasi minX, maxX yang merupakan titik dengan nilai x paling minimum dan paling maksimum
    minX = 0
    maxX = 0

    # res merupakan array pasangan index yang merupakan solusi dari Convex Hull
    res = []

    # Mencari nilai minX dan maxX
    for i in range(len(bucket)) :
        if bucket[i][0] < bucket[minX][0] :
            minX = i
        if bucket[i][0] > bucket[maxX][0] :
            maxX = i

    # Melakukan rekursi algoritma divide and conquer pada kedua sisi dari garis minX dan maxX
    res = DivideAndConquerConvexHull(res, bucket, minX, maxX, 1)
    res = DivideAndConquerConvexHull(res, bucket, minX, maxX, -1)

    # Mengembalikan array beranggotakan pasangan titik yang merupakan garis-garis Convex Hull
    return res
```

BAB 3
Dokumentasi Input dan Output

1. Dataset load_iris

- Petal Length vs Petal Width

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn import datasets
data = datasets.load_iris()
#create a DataFrame
df = pd.DataFrame(data.data, columns=data.feature_names)
df['Target'] = pd.DataFrame(data.target)
```

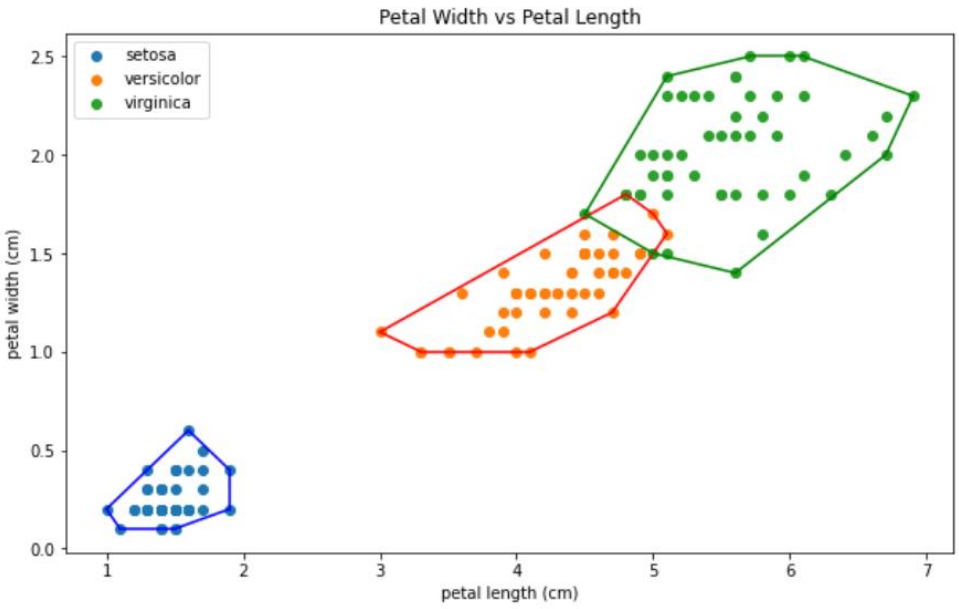
✓ 0.8s

```
#visualisasi hasil ConvexHull
import matplotlib.pyplot as plt
from myConvexHull import ConvexHull
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
    cumBucket = []
    for buck in bucket :
        cumBucket.append([buck[0], buck[1]])
    hull = ConvexHull(cumBucket)
    plt.scatter(bucket[:, 0], bucket[:, 1], label=data.target_names[i])
    for simplex in hull:
        plt.plot([cumBucket[simplex[0]][0], cumBucket[simplex[1]][0]],
                 [cumBucket[simplex[0]][1], cumBucket[simplex[1]][1]], colors[i])
plt.legend()
```

✓ 0.4s

C



- Sepal Length vs Sepal Width

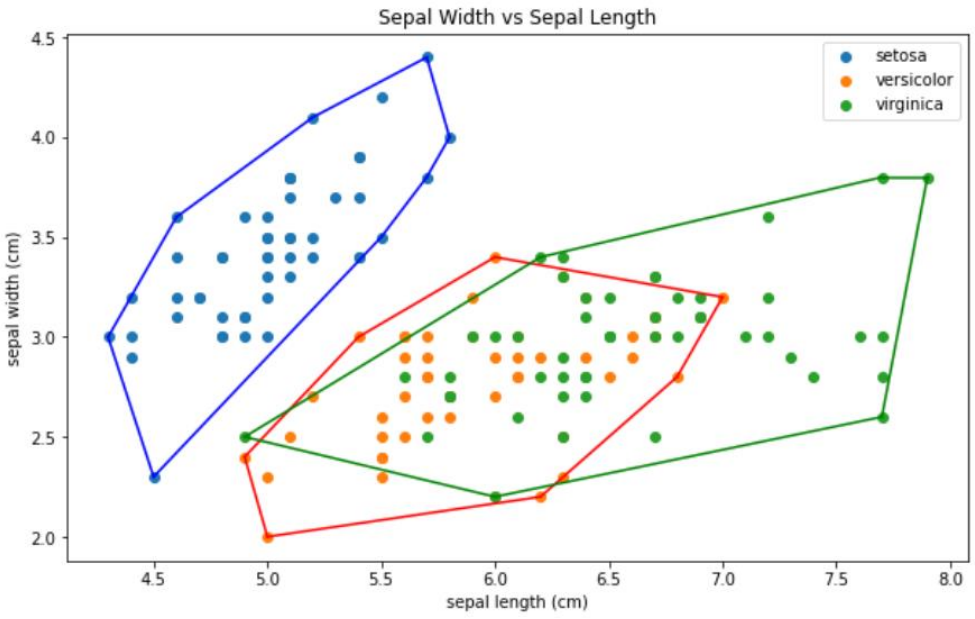
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn import datasets
data = datasets.load_iris()
#create a DataFrame
df = pd.DataFrame(data.data, columns=data.feature_names)
df['Target'] = pd.DataFrame(data.target)
```

✓ 0.8s

```
#visualisasi hasil ConvexHull
import matplotlib.pyplot as plt
from myConvexHull import ConvexHull
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
    cumBucket = []
    for buck in bucket :
        cumBucket.append([buck[0], buck[1]])
    hull = ConvexHull(cumBucket)
    plt.scatter(bucket[:, 0], bucket[:, 1], label=data.target_names[i])
    for simplex in hull:
        plt.plot([cumBucket[simplex[0]][0], cumBucket[simplex[1]][0]],
                 [cumBucket[simplex[0]][1], cumBucket[simplex[1]][1]], colors[i])
plt.legend()
```

✓ 0.4s



2. Dataset load_wine (Alcohol vs Malic Acid)

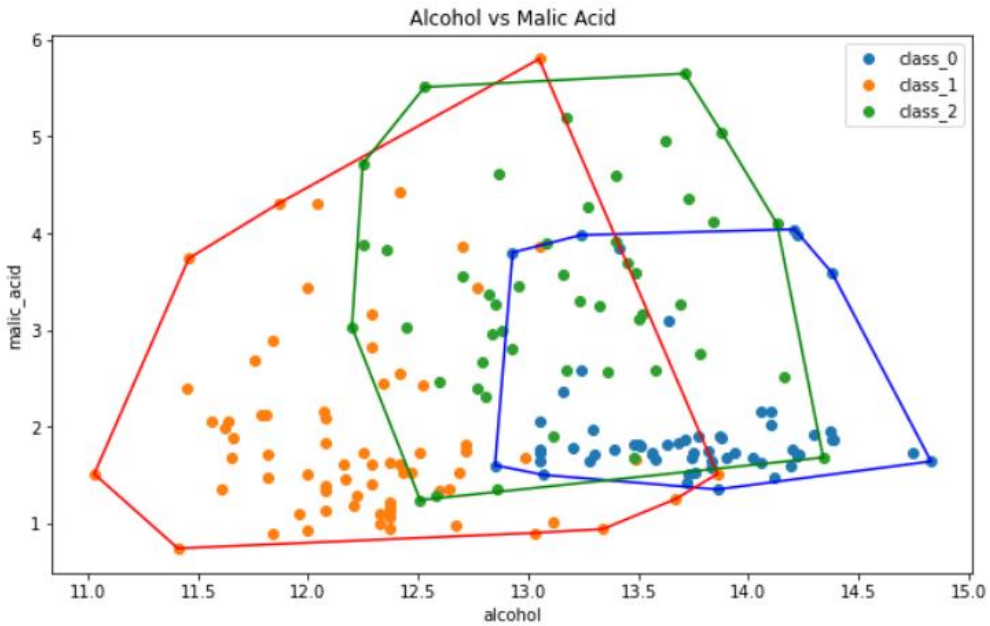
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn import datasets
data = datasets.load_wine()
#create a DataFrame
df = pd.DataFrame(data.data, columns=data.feature_names)
df['Target'] = pd.DataFrame(data.target)
```

✓ 0.5s

```
#visualisasi hasil ConvexHull
import matplotlib.pyplot as plt
from myConvexHull import ConvexHull
plt.figure(figsize = (10, 6))
colors = ['b','r','g']
plt.title('Alcohol vs Malic Acid')
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
    cumBucket = []
    for buck in bucket :
        cumBucket.append([buck[0], buck[1]])
    hull = ConvexHull(cumBucket)
    plt.scatter(bucket[:, 0], bucket[:, 1], label=data.target_names[i])
    for simplex in hull:
        plt.plot([cumBucket[simplex[0]][0], cumBucket[simplex[1]][0]],
                 [cumBucket[simplex[0]][1], cumBucket[simplex[1]][1]], colors[i])
plt.legend()
```

✓ 0.3s



3. Dataset load_digits (Pixel_0_3 vs Pixel_7_3)

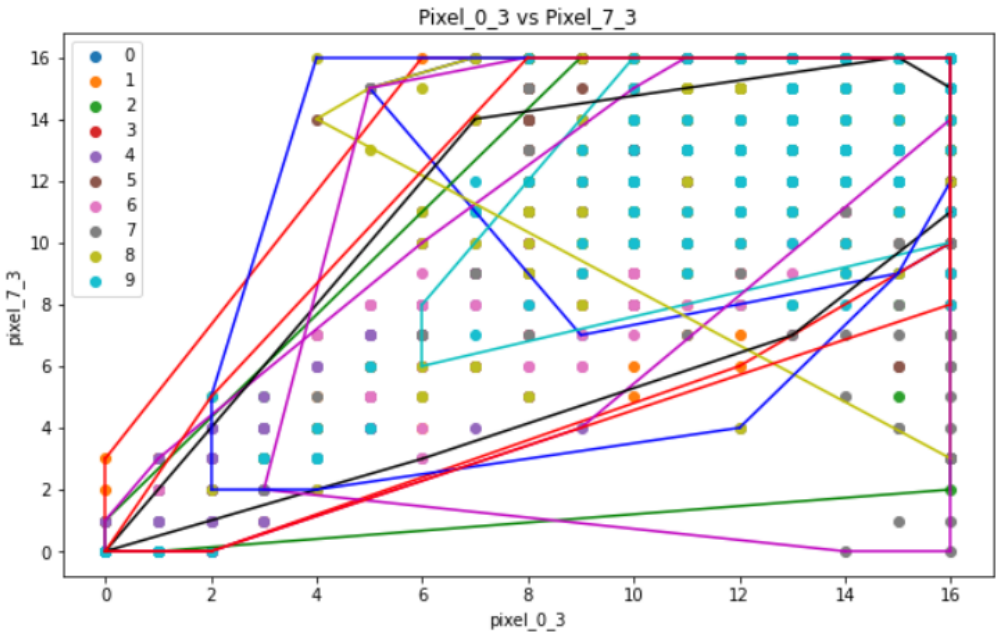
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn import datasets
data = datasets.load_digits()
#create a DataFrame
df = pd.DataFrame(data.data, columns=data.feature_names)
df['Target'] = pd.DataFrame(data.target)
```

✓ 0.1s

```
#visualisasi hasil ConvexHull
import matplotlib.pyplot as plt
from myConvexHull import ConvexHull
plt.figure(figsize = (10, 6))
colors = ['b','r','g','c','m','y','k','m','b','r','g','c']
plt.title('Pixel_0_3 vs Pixel_7_3')
plt.xlabel(data.feature_names[3])
plt.ylabel(data.feature_names[59])

for i in range(len(data.target_names)):
    bucket = df[df['Target'] == i]
    bucket = bucket.iloc[:,[3,59]].values
    cumBucket = []
    for buck in bucket :
        cumBucket.append([buck[0], buck[1]])
    hull = ConvexHull(cumBucket)
    plt.scatter(bucket[:, 0], bucket[:, 1], label=data.target_names[i])
    for simplex in hull:
        plt.plot([cumBucket[simplex[0]][0], cumBucket[simplex[1]][0]],
                 [cumBucket[simplex[0]][1], cumBucket[simplex[1]][1]], colors[i])
plt.legend()
```

✓ 0.6s



4. Dataset load_breast_cancer (Mean Radius vs Mean Texture)

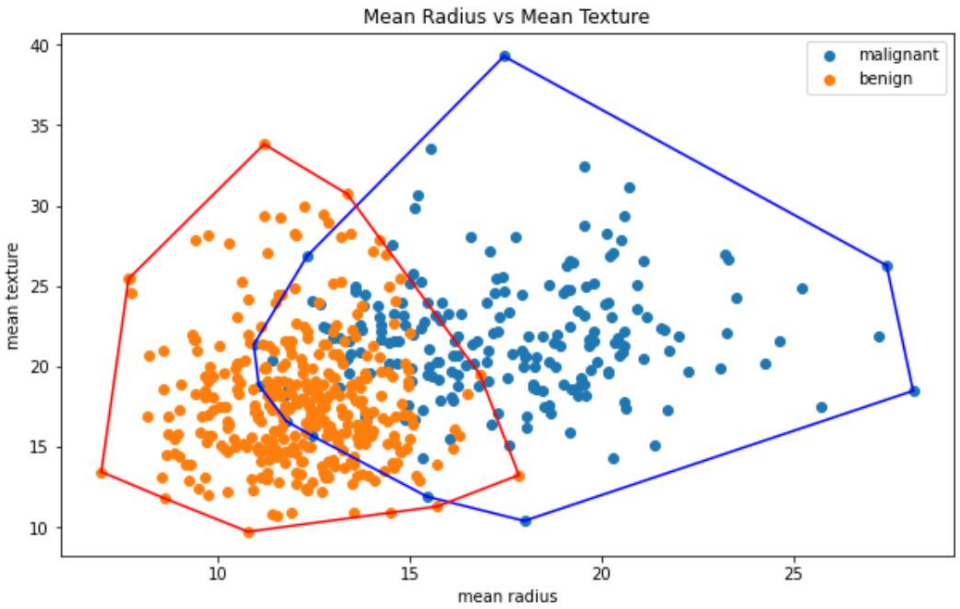
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn import datasets
data = datasets.load_breast_cancer()
#create a DataFrame
df = pd.DataFrame(data.data, columns=data.feature_names)
df['Target'] = pd.DataFrame(data.target)
```

✓ 0.6s

```
#visualisasi hasil ConvexHull
import matplotlib.pyplot as plt
from myConvexHull import ConvexHull
plt.figure(figsize = (10, 6))
colors = ['b','r']
plt.title('Mean Radius vs Mean Texture')
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
    cumBucket = []
    for buck in bucket :
        cumBucket.append([buck[0], buck[1]])
    hull = ConvexHull(cumBucket)
    plt.scatter(bucket[:, 0], bucket[:, 1], label=data.target_names[i])
    for simplex in hull:
        plt.plot([cumBucket[simplex[0]][0], cumBucket[simplex[1]][0]],
                 [cumBucket[simplex[0]][1], cumBucket[simplex[1]][1]], colors[i])
plt.legend()
```

✓ 0.2s



BAB IV
Checklist & Link Github

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

Link Github: https://github.com/gedearyarp/convex-hull