LAPORAN TUGAS KECIL 2

IF2211 STRATEGI ALGORITMA

IMPLEMENTASI CONVEX HULL DENGAN ALGORITMA DIVIDE AND CONQUER

Disusun oleh:

Wesly Giovano 13520071

TEKNIK INFORMATIKA

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

2022

DAFTAR ISI

1 Algoritma Divide and Conquer untuk Convex Hull

Langkah-langkah algoritma *divide and conquer* dalam implementasi fungsi *convex hull* adalah sebagai berikut.

1. Cari titik ekstrem minimum p_{min} dan maksimum p_{max} dari himpunan titik yang diberikan. Titik minimum/maksimum adalah titik yang memiliki nilai absis terkecil/terbesar. Jika terdapat beberapa titik dengan nilai absis yang sama, pilih titik yang memiliki nilai ordinat terkecil/terbesar.
2. Misalkan p_1 adalah p_{min} dan p_2 adalah p_{max}.
3. Cari titik, misalkan p_3, yang memiliki jarak terjauh dari garis yang melalui titik p_1 dan p_2 sedemikian sehingga p_1, p_2, p_3 secara berurutan membentuk arah *counter-clockwise* (arah ini menjamin titik berada di luar *convex hull* yang terbentuk). Jika terdapat beberapa titik dengan jarak terjauh yang sama, pilih titik yang mengakibatkan $\angle\, p_3\, p_1\, p_2$ memiliki nilai terbesar. Jika tidak ada titik yang memenuhi kondisi tersebut, masukkan p_2 ke dalam himpunan *convex*.
4. Ulangi langkah (3) dengan p_3 sebagai p_2 dan p_3 sebagai p_1. Kedua hasil digabung ke dalam himpunan *convex*.
5. Ulangi langkah (3) dengan p_{max} sebagai p_1 dan p_{min} sebagai p_2.
6. Himpunan yang diperoleh adalah himpunan yang membentuk *convex hull* dari himpunan titik yang diberikan.

2 Source Code Program

Dalam penyusunan laporan ini, dituliskan empat buah file *source code* dengan satu file *myConvexHull* dan tiga file sebagai contoh penggunaan, yaitu *main-iris*, *main-wine*, dan *main-breast-cancer*. Seluruh *source code* program disimpan dalam repository git dengan alamat https://github.com/weslygio/myConvexHull.

2.1 myConvexHull.py

```
import numpy as np


def search_min(points: np.ndarray) -> np.ndarray:
    minpoint = points[0]
    for i in range(1, len(points)):
        if points[i, 0] < minpoint[0]:
            minpoint = points[i]
        elif points[i, 0] == minpoint[0]:
            if points[i, 1] < minpoint[1]:
                minpoint = points[i]
    return minpoint


def search_max(points: np.ndarray) -> np.ndarray:
    maxpoint = points[0]
    for i in range(1, len(points)):
        if points[i, 0] > maxpoint[0]:
```

```
            maxpoint = points[i]
        elif points[i, 0] == maxpoint[0]:
            if points[i, 1] > maxpoint[1]:
                maxpoint = points[i]
    return maxpoint


def dist_point_to_line(p0: np.ndarray, p1: np.ndarray, p2: np.ndarray) -> float:
    """ Calculate distance from point p0 to a line segment from p1 to p2 """
    A = 0.5 * np.linalg.det([[p1[0], p1[1], 1],
                             [p2[0], p2[1], 1],
                             [p0[0], p0[1], 1]])
    b = np.linalg.norm(p2 - p1)
    h = 2 * A / b
    if abs(h) < 1e-5:
        return 0
    else:
        return h


def angle(P: np.ndarray, Q: np.ndarray, R: np.ndarray) -> np.ndarray:
    """ Calculate measure of ∠PQR in radian """
    QP = P - Q
    QR = R - Q
    return np.arccos(np.dot(QP, QR) / (np.linalg.norm(QP) * np.linalg.norm(QR)))


def ConvexHull(points: np.ndarray) -> np.ndarray:
    """
    Create convex hull from given points. Let p1, p2, ..., pn be the return set of
points,
    then the visualization of the convex hull is p1-p2-...-pn-p1.

    :param points: a set of points which is to be made its convex hull
    :return: a set of ordered points which is the convex hull
    """
    minpoint = search_min(points)
    maxpoint = search_max(points)

    hull1 = ConvexHull2(points, minpoint, maxpoint)
    hull2 = ConvexHull2(points, maxpoint, minpoint)
    return np.append(hull1, hull2, axis=0)


def ConvexHull2(points: np.ndarray, p1: np.ndarray, p2: np.ndarray) -> np.ndarray:
    """
    Create convex hull from given points in which p1 and p2 are the initial extreme
points.
    The resulting convex hull is the union of convex hull of outer points and p2.

    :param points: a set of points which is to be made its convex hull on the outer
side
    :param p1: first extreme point
    :param p2: second extreme point
    :return: a set of ordered points which is the convex hull
    """
    subpoints: np.ndarray = np.array([]).reshape(0,2)
    p_extreme = p1
    max_dist = 0

    # Ambil point yang mengarah ke luar
    for testpoint in points:
        dist = dist_point_to_line(testpoint, p1, p2)
```

```
        if dist > 0:
            subpoints = np.append(subpoints, testpoint.reshape(1,2), axis=0)
            if dist > max_dist:
                max_dist = dist
                p_extreme = testpoint
            elif dist == max_dist:
                if angle(testpoint, p1, p2) > angle(p_extreme, p1, p2):
                    p_extreme = testpoint

    if subpoints.size == 0:
        return np.array([p2])

    hull1 = ConvexHull2(subpoints, p1, p_extreme)
    hull2 = ConvexHull2(subpoints, p_extreme, p2)

    return np.append(hull1, hull2, axis=0)
```

2.2 main-iris.py

```
import pandas as pd
import matplotlib.pyplot as plt
from sklearn import datasets
from myConvexHull import ConvexHull
from itertools import cycle

data = datasets.load_iris()

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

plt.figure(figsize=(10, 6))
colors = ['b','r','g']
plt.title('Sepal Length vs Sepal Width')
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 = ConvexHull(bucket)
    plt.scatter(bucket[:, 0], bucket[:, 1], label=data.target_names[i])

    hull_iter = cycle(hull)
    p1 = next(hull_iter)
    for _ in range(len(hull)):
        p2 = next(hull_iter)
        plt.plot([p1[0], p2[0]], [p1[1], p2[1]], colors[i])
        p1 = p2

plt.legend()
plt.show()

plt.clf()

plt.figure(figsize=(10, 6))
plt.title('Petal Length vs Petal Width')
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 = ConvexHull(bucket)
    plt.scatter(bucket[:, 0], bucket[:, 1], label=data.target_names[i])

    hull_iter = cycle(hull)
    p1 = next(hull_iter)
    for _ in range(len(hull)):
        p2 = next(hull_iter)
        plt.plot([p1[0], p2[0]], [p1[1], p2[1]], colors[i])
        p1 = p2

plt.legend()
plt.show()
```

2.3 main-wine.py

```
import pandas as pd
import matplotlib.pyplot as plt
from sklearn import datasets
from myConvexHull import ConvexHull
from itertools import cycle

data = datasets.load_wine()

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

plt.figure(figsize=(10, 6))
colors = ['b','r','g']
plt.title('Nonflavanoid phenols vs Total phenols')
plt.xlabel(data.feature_names[7])
plt.ylabel(data.feature_names[5])

for i in range(len(data.target_names)):
    bucket = df[df['Target'] == i]
    bucket = bucket.iloc[:,[7,5]].values
    hull = ConvexHull(bucket)
    plt.scatter(bucket[:, 0], bucket[:, 1], label=data.target_names[i])

    hull_iter = cycle(hull)
    p1 = next(hull_iter)
    for _ in range(len(hull)):
        p2 = next(hull_iter)
        plt.plot([p1[0], p2[0]], [p1[1], p2[1]], colors[i])
        p1 = p2

plt.legend()
plt.show()

plt.clf()

plt.figure(figsize=(10, 6))
plt.title('Alcohol vs Total phenols')
plt.xlabel(data.feature_names[0])
plt.ylabel(data.feature_names[5])

for i in range(len(data.target_names)):
```

```
    bucket = df[df['Target'] == i]
    bucket = bucket.iloc[:,[0,5]].values
    hull = ConvexHull(bucket)
    plt.scatter(bucket[:, 0], bucket[:, 1], label=data.target_names[i])

    hull_iter = cycle(hull)
    p1 = next(hull_iter)
    for _ in range(len(hull)):
        p2 = next(hull_iter)
        plt.plot([p1[0], p2[0]], [p1[1], p2[1]], colors[i])
        p1 = p2

plt.legend()
plt.show()
```

2.4 main-breast-cancer.py

```
import pandas as pd
import matplotlib.pyplot as plt
from sklearn import datasets
from myConvexHull import ConvexHull
from itertools import cycle

data = datasets.load_breast_cancer()

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

plt.figure(figsize=(10, 6))
colors = ['orangered','royalblue']
plt.title('Smoothness vs Compactness')
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 = ConvexHull(bucket)
    plt.scatter(bucket[:, 0], bucket[:, 1], label=data.target_names[i], c=colors[i])

    hull_iter = cycle(hull)
    p1 = next(hull_iter)
    for _ in range(len(hull)):
        p2 = next(hull_iter)
        plt.plot([p1[0], p2[0]], [p1[1], p2[1]], colors[i])
        p1 = p2

plt.legend()
plt.show()

plt.clf()

plt.figure(figsize=(10, 6))
plt.title('Concavity vs Concave points')
plt.xlabel(data.feature_names[6])
plt.ylabel(data.feature_names[7])

for i in range(len(data.target_names)):
    bucket = df[df['Target'] == i]
```

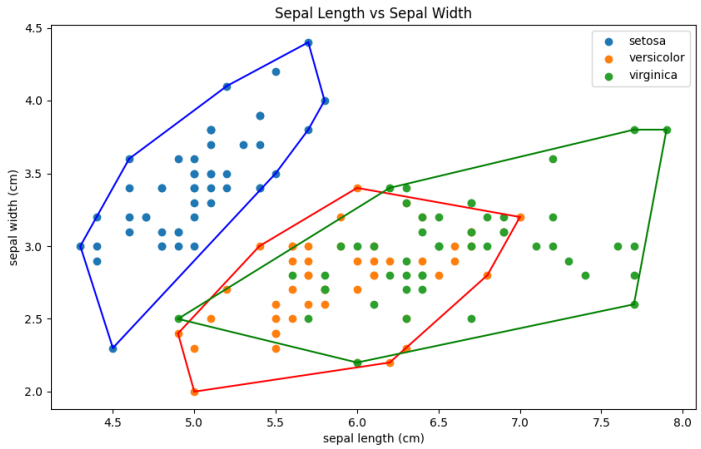
```
    bucket = bucket.iloc[:,[6,7]].values
    hull = ConvexHull(bucket)
    plt.scatter(bucket[:, 0], bucket[:, 1], label=data.target_names[i], c=colors[i])

    hull_iter = cycle(hull)
    p1 = next(hull_iter)
    for _ in range(len(hull)):
        p2 = next(hull_iter)
        plt.plot([p1[0], p2[0]], [p1[1], p2[1]], colors[i])
        p1 = p2

plt.legend()
plt.show()
```

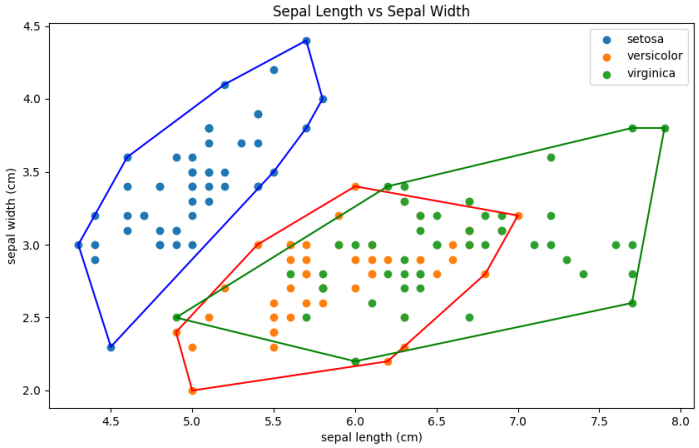
3 Eksperimen

Eksperimen dilakukan terhadap tiga buah *dataset*, yaitu *iris*, *wine*, dan *breast cancer*. Berikut adalah hasil visualisasi.

3.1 Dataset Iris

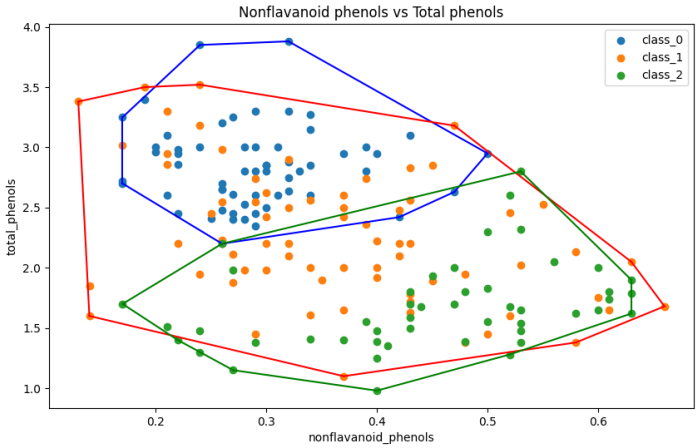
Pada dataset iris, dua buah plot dibentuk, dengan plot pertama adalah “Sepal Length vs Sepal Width” dan plot kedua adalah “Petal Length vs Petal Width”.

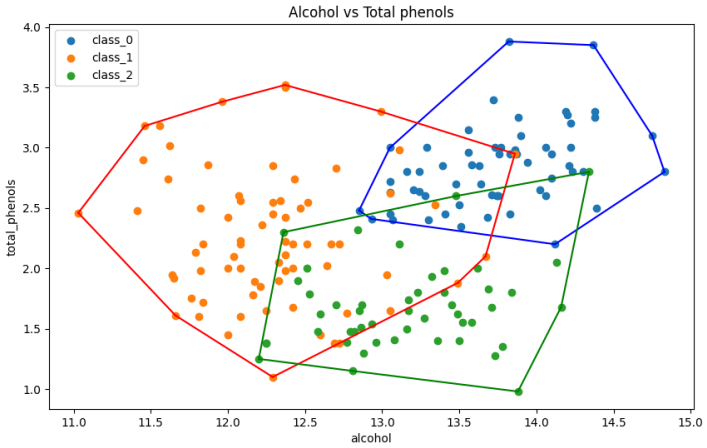




3.2 Dataset Wine

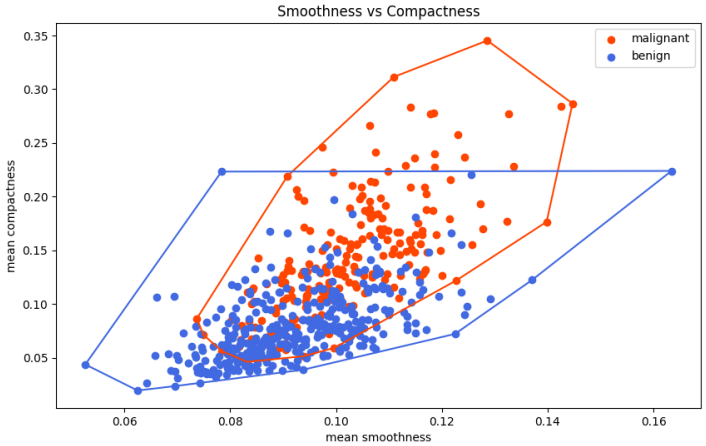
Pada dataset wine, dua buah plot dibentuk, dengan plot pertama adalah “Nonflavonoid phenols vs Total phenols” dan plot kedua adalah “Alcohol vs Total phenols”.

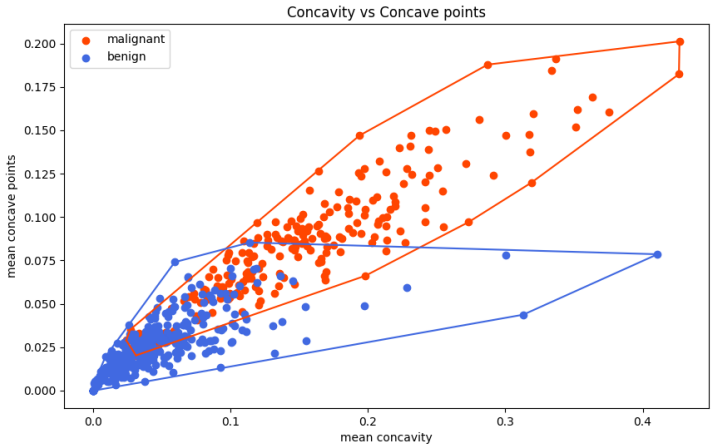




3.3 Dataset Breast Cancer

Pada dataset breast cancer, dua buah plot dibentuk, dengan plot pertama adalah “Smoothness vs Compactness” dan plot kedua adalah “Concavity vs Concave points”.





4 Tabel Penyelesaian Tugas Kecil

Poin	Ya	Tidak
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.	√	