Laporan Tugas Kecil 2 IF2211 Strategi Algortima

Semester II Tahun 2021/2022

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

Christopher Jeffrey
13520055

Algoritma

1. Hilangkan data yang kembar, karena tidak mungkin suatu point pada hull digunakan lebih dari satu kali
2. Urutkan seluruh list berdasarkan absisnya membesar. Jika absisnya sama, urutkan dari ordinat membesar
3. Bayangkan sebuah garis antara point terkecil dengan point terbesar, yang membagi kumpulan point menjadi dua
4. Untuk tiap bagian, jika hanya ada dua buah point pada bagian tersebut, maka kedua point tersebut bagian dari hull
5. Jika lebih dari dua point, bayangkan ada sebuah garis antara point terkecil dan terbesar. Cari point dengan jarak paling jauh dari keduanya, kita sebut point maksimum
6. Bayangkan sebuah garis antara point terkecil dengan point maksimum, yang akan membagi kumpulan point menjadi dua bagian. Begitu juga antara point maksimum dengan point terbesar.
7. Jika point maksimum berada diatas garis point terkecil dan point terbesar, kumpulan point yang akan digunakan adalah bagian atas dari hasil pembagian langkah nomor enam. Jika tidak, yang akan digunakan adalah bagian bawah.
8. Lakukan langkah nomor empat untuk masing-masing kumpulan point hingga tidak ada lagi bagian yang tersisa

Kode Program

```
import numpy as np
from numpy.linalg import norm

pointsReference = []

def myConvexHull(nppoints):
    global pointsReference
    pointsReference = nppoints.tolist()
```

```
    nppoints = np.unique(nppoints, axis = 0) # hilangkan yang kembar, karena
tidak mungkin dua point yang sama ada di dalam hull(dan juga tidak efisien
membiarkan point kembar di algoritma)
    points = nppoints.tolist()
    sortPoints(points)
    pointsAbove, pointsBelow = divideFirstPoints(points)
    hull = DCConvexHull(pointsAbove, []) + DCConvexHull(pointsBelow, [])
    return formatedOutput(hull)

def pointToOriginalIndex(point):
    return pointsReference.index(point)

def sortPoints(points):
    quickSort(points, 0, len(points) - 1)

def partition(arr, low, high):
    i = (low-1) # index of smaller element
    pivot = arr[high] # pivot
    for j in range(low, high):
        # If current element is smaller than or
        # equal to pivot
        if arr[j][0] < pivot[0] or (arr[j][0] == pivot[0] and arr[j][1] <
pivot[1]):
            # increment index of smaller element
            i = i+1
            arr[i], arr[j] = arr[j], arr[i]

    arr[i+1], arr[high] = arr[high], arr[i+1]
    return (i+1)

def quickSort(arr, low, high):
    if len(arr) == 1:
        return arr
    if low < high:
        pi = partition(arr, low, high)
        quickSort(arr, low, pi-1)
        quickSort(arr, pi+1, high)

def DCConvexHull(points,hull):
```

```
    if(len(points) == 2):
        return hull +[[pointToOriginalIndex(points[0]),
pointToOriginalIndex(points[1])]]
    else:
        setOfPoints1,setOfPoints2 = dividePoints(points)
        return hull + DCConvexHull(setOfPoints1, hull) +
DCConvexHull(setOfPoints2, hull)

def findPmaxIndex(points):
    p1= np.array(points[0])
    p2= np.array(points[-1])
    dmax = 0
    dmaxi = 0
    for i in range(0, len(points)):
        p3 = np.array(points[i])
        if(norm(p2-p1) != 0):
            d = norm(np.cross(p2-p1, p1-p3))/norm(p2-p1)
            if(d == dmax):
                if(angle(p1,p2, p3) > angle(p1, p2, points[dmaxi])):
                    dmaxi = i
                    dmax = d

            elif(d > dmax):
                dmaxi = i
                dmax = d
    return dmaxi

def angle(p1, p2, p3):
    v1 = (p2[0] - p1[0], p2[1] - p1[1])
    v2 = (p3[0] - p1[0], p3[1] - p1[1])
    if(np.linalg.norm(v1) == 0 or np.linalg.norm(v2) == 0):
        return 0
    unit_vector_1 = v1 / np.linalg.norm(v1)
    unit_vector_2 = v2 / np.linalg.norm(v2)
    dot_product = np.dot(unit_vector_1, unit_vector_2)
    angle = np.arccos(dot_product)
    return angle

def divideFirstPoints(points):
```

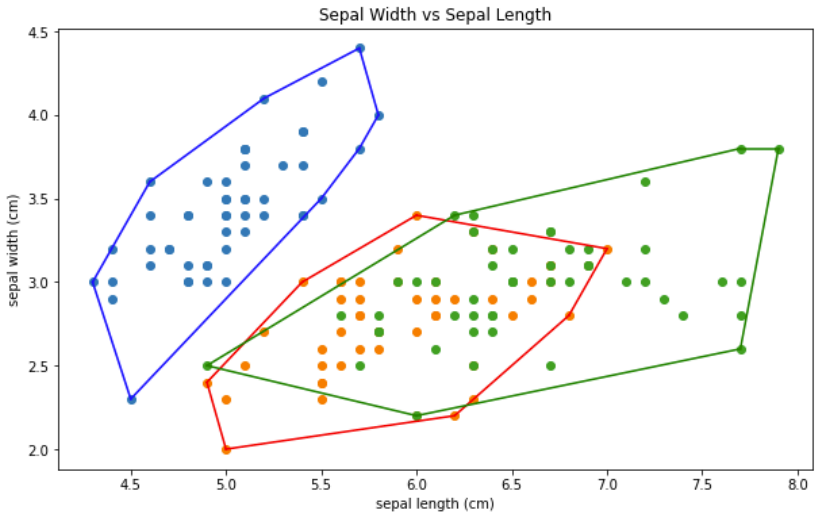
```
    return divideCustom(points[0],points[-1], points)

def dividePoints(points):
    p1 = points[0]
    p2 = points[-1]
    pmaxIndex = findPmaxIndex(points)
    pmax = points[pmaxIndex]

    pointsWithP1Above, pointsWithP1Below = divideCustom(p1, pmax, points)
    pointsWithP2Above, pointsWithP2Below = divideCustom(pmax, p2, points)
    if(crossProduct(p1, p2, pmax) > 0):
        return pointsWithP1Above, pointsWithP2Above
    else:
        return pointsWithP1Below, pointsWithP2Below

def divideCustom(p1, p2, points):
    pointsAbove = [p1,p2]
    pointsBelow = [p1,p2]
    for point in points:
        if(point != p1 and point != p2):
            checker = crossProduct(p1, p2, point)
            if checker < 0:
                pointsBelow.append(point)
            if checker > 0:
                pointsAbove.append(point)
    sortPoints(pointsAbove)
    sortPoints(pointsBelow)
    return pointsAbove, pointsBelow

def crossProduct(p1, p2, p3):
    v1 = (p2[0] - p1[0], p2[1] - p1[1])
    v2 = (p3[0] - p1[0], p3[1] - p1[1])
    x = np.cross(v1,v2)
    return x

class Hull:
    def __init__(self,simplices):
        self.simplices = simplices
```

```
def formatedOutput(hull):
    return Hull(hull)
```

Screenshot

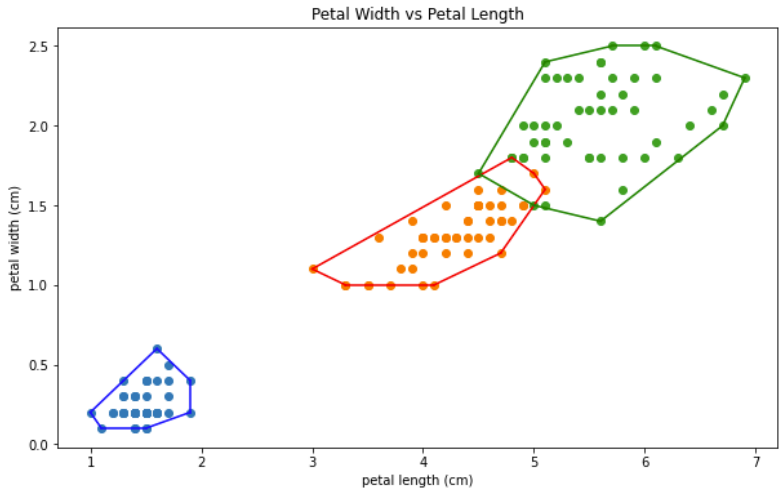
a. visualisasi data iris, sepal width vs sepal length

```
Iris Data
(150, 5)
   sepal length (cm)  sepal width (cm)  petal length (cm)  petal width (cm)  \
0                5.1               3.5                1.4               0.2
1                4.9               3.0                1.4               0.2
2                4.7               3.2                1.3               0.2
3                4.6               3.1                1.5               0.2
4                5.0               3.6                1.4               0.2

   Target
0       0
1       0
2       0
3       0
4       0
```



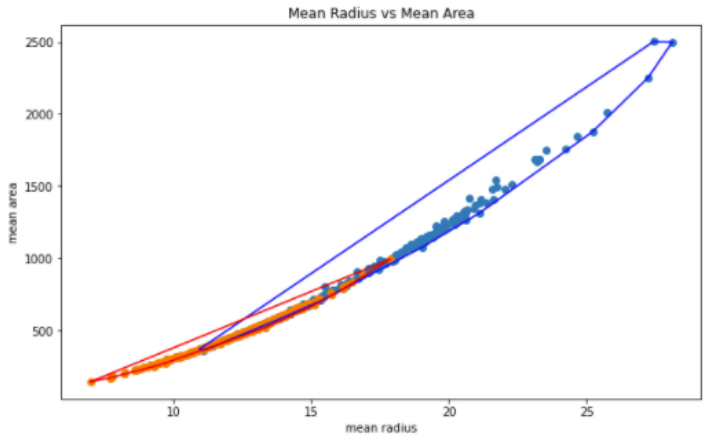
b. visualisasi data iris, petal width vs petal length

```
Iris Data
(150, 5)
   sepal length (cm)  sepal width (cm)  petal length (cm)  petal width (cm)  \
0                5.1               3.5                1.4               0.2
1                4.9               3.0                1.4               0.2
2                4.7               3.2                1.3               0.2
3                4.6               3.1                1.5               0.2
4                5.0               3.6                1.4               0.2

   Target
0       0
1       0
2       0
3       0
4       0
```



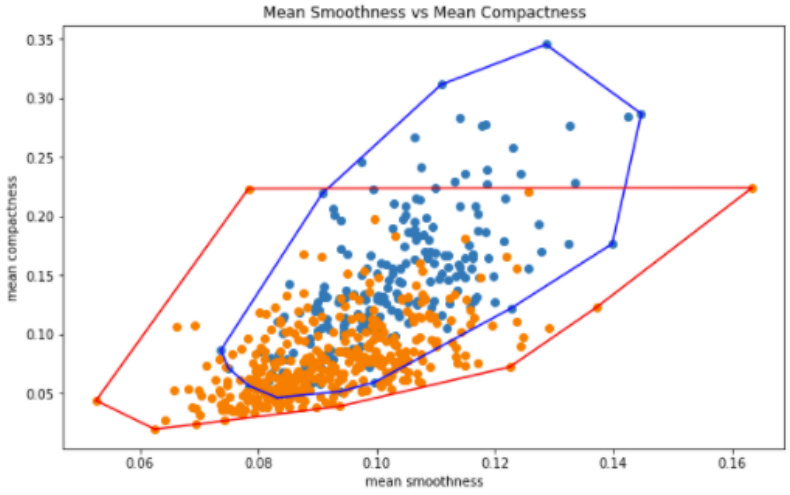
c. visualisasi data breast cancer, mean radius vs mean area

```
Breast Cancer Data
(569, 31)
   mean radius  mean texture  mean perimeter  mean area  mean smoothness  \
0        17.99         10.38          122.80     1001.0          0.11840
1        20.57         17.77          132.90     1326.0          0.08474
2        19.69         21.25          130.00     1203.0          0.10960
3        11.42         20.38           77.58      386.1          0.14250
4        20.29         14.34          135.10     1297.0          0.10030

   mean compactness  mean concavity  mean concave points  mean symmetry  \
0           0.27760          0.3001              0.14710         0.2419
1           0.07864          0.0869              0.07017         0.1812
2           0.15990          0.1974              0.12790         0.2069
3           0.28390          0.2414              0.10520         0.2597
4           0.13280          0.1980              0.10430         0.1809

   mean fractal dimension  ...  worst texture  worst perimeter  worst area  \
0                 0.07871  ...          17.33           184.60      2019.0
1                 0.05667  ...          23.41           158.80      1956.0
2                 0.05999  ...          25.53           152.50      1709.0
3                 0.09744  ...          26.50            98.87       567.7
4                 0.05883  ...          16.67           152.20      1575.0

   worst smoothness  worst compactness  worst concavity  worst concave points  \
0            0.1622             0.6656           0.7119                0.2654
...
2          0.3613                0.08758       0
3          0.6638                0.17300       0
4          0.2364                0.07678       0

[5 rows x 31 columns]
```



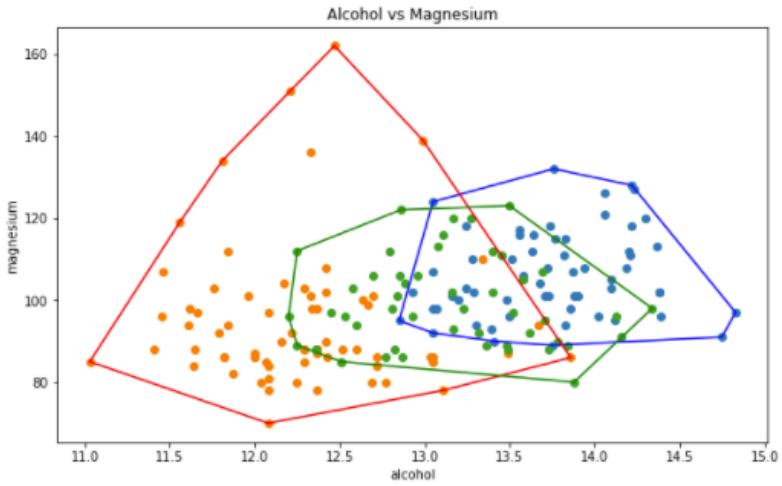
d. visualisasi data breast cancer, mean smoothness vs mean compactness

```
Breast Cancer Data
(569, 31)
   mean radius  mean texture  mean perimeter  mean area  mean smoothness  \
0        17.99         10.38          122.80     1001.0          0.11840
1        20.57         17.77          132.90     1326.0          0.08474
2        19.69         21.25          130.00     1203.0          0.10960
3        11.42         20.38           77.58      386.1          0.14250
4        20.29         14.34          135.10     1297.0          0.10030

   mean compactness  mean concavity  mean concave points  mean symmetry  \
0           0.27760          0.3001              0.14710         0.2419
1           0.07864          0.0869              0.07017         0.1812
2           0.15990          0.1974              0.12790         0.2069
3           0.28390          0.2414              0.10520         0.2597
4           0.13280          0.1980              0.10430         0.1809

   mean fractal dimension  ...  worst texture  worst perimeter  worst area  \
0                 0.07871  ...          17.33           184.60      2019.0
1                 0.05667  ...          23.41           158.80      1956.0
2                 0.05999  ...          25.53           152.50      1709.0
3                 0.09744  ...          26.50            98.87       567.7
4                 0.05883  ...          16.67           152.20      1575.0

   worst smoothness  worst compactness  worst concavity  worst concave points  \
0            0.1622             0.6656           0.7119                0.2654
...
2          0.3613                0.08758       0
3          0.6638                0.17300       0
4          0.2364                0.07678       0

[5 rows x 31 columns]
```



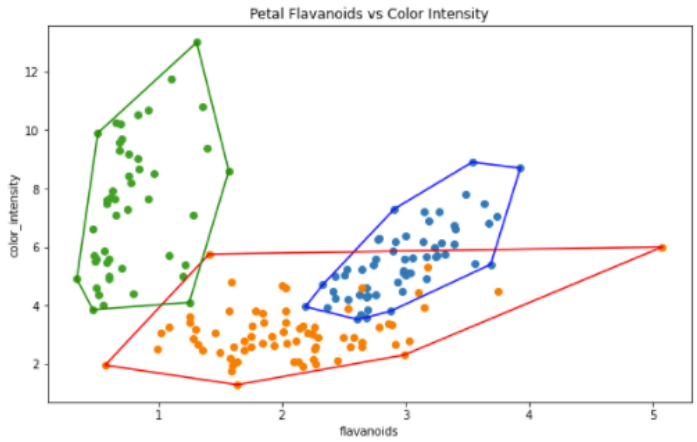
e. visualisasi data wine, alcohol vs magnesium

```
Wine Data
(178, 14)
   alcohol  malic_acid   ash  alcalinity_of_ash  magnesium  total_phenols  \
0    14.23        1.71  2.43               15.6      127.0           2.80
1    13.20        1.78  2.14               11.2      100.0           2.65
2    13.16        2.36  2.67               18.6      101.0           2.80
3    14.37        1.95  2.50               16.8      113.0           3.85
4    13.24        2.59  2.87               21.0      118.0           2.80

   flavanoids  nonflavanoid_phenols  proanthocyanins  color_intensity   hue  \
0        3.06                  0.28             2.29             5.64  1.04
1        2.76                  0.26             1.28             4.38  1.05
2        3.24                  0.30             2.81             5.68  1.03
3        3.49                  0.24             2.18             7.80  0.86
4        2.69                  0.39             1.82             4.32  1.04

   od280/od315_of_diluted_wines  proline  Target
0                          3.92   1065.0       0
1                          3.40   1050.0       0
2                          3.17   1185.0       0
3                          3.45   1480.0       0
4                          2.93    735.0       0
```



f. visualisasi data wine, petal flavanoids vs color intensity

```
Wine Data
(178, 14)
   alcohol  malic_acid   ash  alcalinity_of_ash  magnesium  total_phenols  \
0    14.23        1.71  2.43               15.6      127.0           2.80
1    13.20        1.78  2.14               11.2      100.0           2.65
2    13.16        2.36  2.67               18.6      101.0           2.80
3    14.37        1.95  2.50               16.8      113.0           3.85
4    13.24        2.59  2.87               21.0      118.0           2.80

   flavanoids  nonflavanoid_phenols  proanthocyanins  color_intensity   hue  \
0        3.06                  0.28             2.29             5.64  1.04
1        2.76                  0.26             1.28             4.38  1.05
2        3.24                  0.30             2.81             5.68  1.03
3        3.49                  0.24             2.18             7.80  0.86
4        2.69                  0.39             1.82             4.32  1.04

   od280/od315_of_diluted_wines  proline  Target
0                          3.92   1065.0       0
1                          3.40   1050.0       0
2                          3.17   1185.0       0
3                          3.45   1480.0       0
4                          2.93    735.0       0
```



Alamat Drive

https://github.com/christojeffrey/convex-hull

Poin	Ya	Tidak
1. Pustaka myConvexHull berhasil dibuat dan tidak ada kesalahan	V	
2. Convex hull yang dihasilkan sudah	V	

benar		
3. Pustaka myConvexHull dapat digunakan untuk menampilkan convex hull setiap label dengan warna yang berbeda.	V	
4. Bonus: program dapat menerima input dan menuliskan output untuk dataset lainnya.	V	

Referensi

https://www.kite.com/python/answers/how-to-get-the-angle-between-two-vectors-in-python
https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2021-2022/Algoritma-Divide-and-Conquer-(2022)-Bagian4.pdf
https://www.geeksforgeeks.org/convert-python-list-to-numpy-arrays/
https://www.geeksforgeeks.org/python-program-for-quicksort/