TUGAS KECIL 2

IF2211 Strategi Algoritma

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

Dipersiapkan oleh:

Taufan Fajarama Putrawansyah Ruslanali (13520031)

PROGRAM STUDI TEKNIK INFORMATIKA
SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG

2022

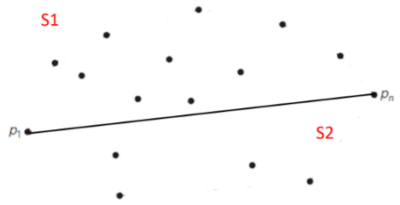
ALGORITMA DIVIDE AND CONQUER

Mencari Titik Ekstrim

Pada input dataset, akan terlebih dahulu dicari dua titik ekstrim, yaitu titik dengan absis dan ordinat terkecil dan titik dengan absis dan ordinat terbesar. Kedua titik ini akan menjadi awal dari penyelesaian convex hull.

Membagi Kumpulan Titik

Kedua titik ekstrim yang sudah dicari, kemudian ditarik garis sehingga membagi kumpulan titik awal menjadi kumpulan titik di kiri/atas garis dan kumpulan titik di kanan/bawah garis.



Lalu, untuk memeriksa apakah suatu titik berada di bagian kiri/atas atau kanan/bawah, digunakan rumus determinan:

$$\begin{vmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{vmatrix} = x_1y_2 + x_3y_1 + x_2y_3 - x_3y_2 - x_2y_1 - x_1y_3$$

Titik (x3,y3) berada di sebelah kiri dari garis ((x1,y1),(x2,y2)) jika hasil determinan positif

Menerapkan Divide and Conquer Secara Rekursif

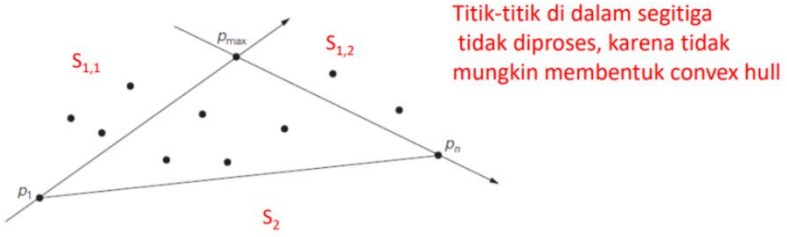
Setelah itu, akan dibentuk convex hull pada kedua bagian yang sudah dibagi, untuk suatu bagian yang sudah dibagi dengan garis yang terbentuk dari titik p1 dan pn, selanjutnya akan dilakukan divide & conquer secara rekursif.

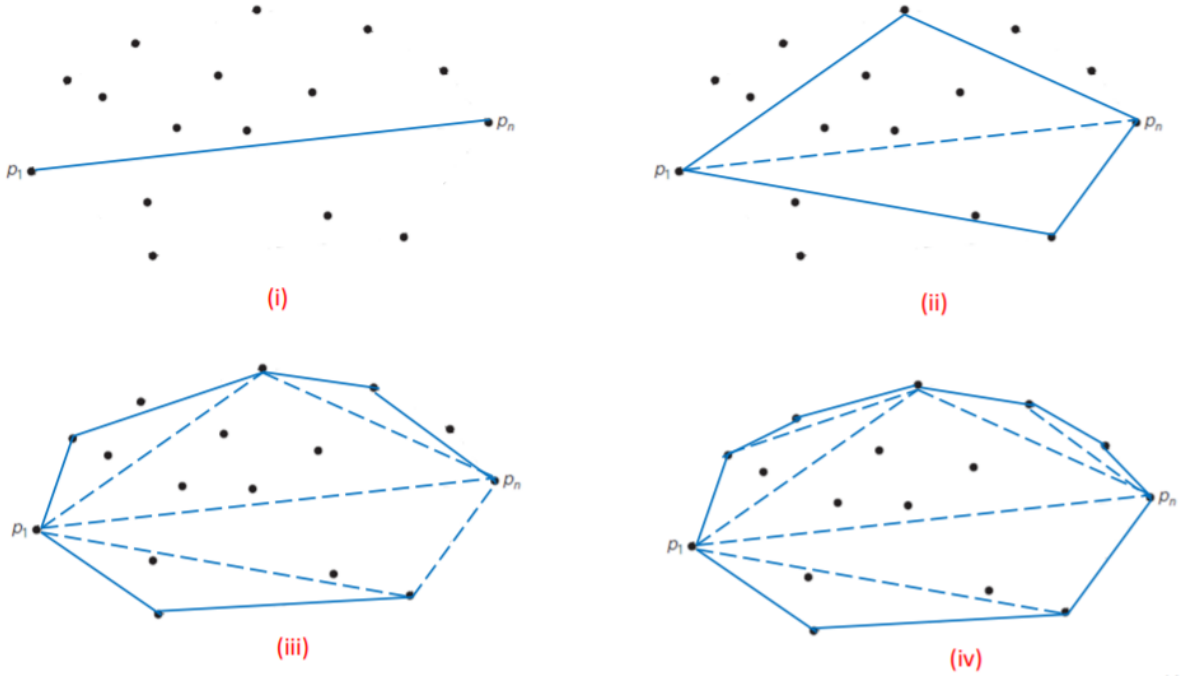
Kasus basis:

- Tidak ada titik lain di bagian tersebut, sehingga titik p1 dan pn adalah pembentuk convex hull bagian tersebut.
- Hanya ada satu titik di bagian tersebut, sehingga ketiga titik akan menjadi pembentuk convex hull di bagian tersebut.

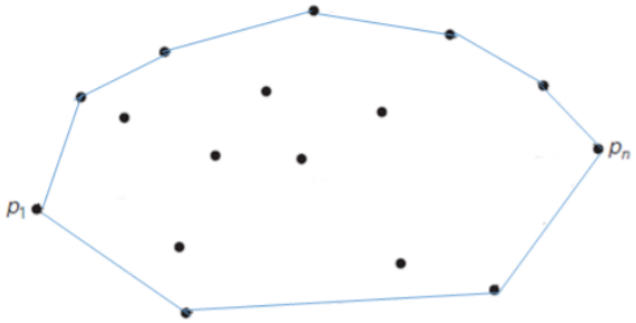
Kasus rekursif:

Cari titik dengan jarak terjauh dari garis p1pn, jika terdapat beberapa titik dengan jarak yang sama, pilih titik dengan sudut terbesar. Selanjutnya, dilakukan pembagian wilayah seperti pada langkah awal sehingga terbentuk segitiga dengan garis p1pn. Kemudian akan dilakukan kembali divide & conquer untuk kumpulan titik yang berada di luar segitiga tersebut, kumpulan titik di dalam segitiga dapat diabaikan. Langkah ini diulangi hingga mencapai kasus basis. Kemudian akan dihasilkan pasangan titik yang membentuk convex hull.





Hasil akhir algoritma:



sumber gambar: Bahan Kuliah IF2211 Strategi Algoritma

SOURCE CODE

Fungsi split

Fungsi split digunakan untuk membagi kumpulan titik menjadi 2 bagian berdasarkan garis yang dibentuk dari dua titik awal. Berikut source codenya:

```
# listP adalah kumpulan seluruh titik, p1 dan p2 adalah indeks titik
# listIdx adalah kumpulan indeks titik
def split(listP, p1, p2, listIdx):
# inisialiasi list index titik di kiri dan kanan garis
  left = []
  right = []
# menghitung determinan setiap titik dan dibagi ke kumpulan titik
  for i in listIdx:
    if (i == p1) or (i == p2):  continue
    else:
      x1 = listP[p1][0]
      y1 = listP[p1][1]
      x2 = listP[p2][0]
      y2 = listP[p2][1]
      x3 = listP[i][0]
      y3 = listP[i][1]
      det = x1*y2 + x3*y1 + x2*y3 - x3*y2 - x2*y1 - x1*y3
      # jika determinan positif maka titik berada di kiri/atas garis
      if (det > 0):
        left.append(i)
      # jika determinan negatif maka titik berada di kanan/bawah garis
      elif (det < 0):
        right.append(i)
      # jika determinan = 0 maka titik berada di garis dan diabaikan
  return left, right  # mengembalikan daftar index titik yang sudah dibagi
```

Fungsi distanceP

Fungsi distance digunakan untuk menghitung jarak suatu titik terhadap garis yang dibentuk dari dua titik. Berikut source codenya:

```
import numpy as np
# p1, p2, px adalah titik berbentuk numpy array
def distanceP(p1, p2, px):
  d = np.cross(p2-p1,px-p1) / np.linalg.norm(p2-p1)
  return d
```

Fungsi angleP

Fungsi distance digunakan untuk menghitung sudut suatu titik. Berikut source codenya:

```
# px, p1, p2 adalah titik berbentuk numpy array
def angleP(px, p1, p2):
  line1 = px - p1
  line2 = p2 - p1
  cos_angle = np.dot(line1, line2) / (np.linalg.norm(line1) * np.linalg.norm(line2))
  angle = np.arccos(cos_angle)
  return np.degrees(angle)
```

Fungsi farthestP

Fungsi farthestP digunakan untuk mencari index titik terjauh dari garis Berikut source codenya:

```
# listP adalah kumpulan seluruh titik, p1 dan p2 adalah indeks titik
# listIdx adalah kumpulan indeks titik
def farthestP(listP, p1, p2, listIdx):
  # ubah menjadi numpy array untuk mempermudah perhitungan
  npP = np.array(listP)
  np1 = npP[p1]
  np2 = npP[p2]
  # inisialisasi variable penampung
  maxi = 0
  pMax = np.array([0, 0])
  idxMax = 0
  # mencari jarak terjauh
  for i in listIdx:
    d = distanceP(np1, np2, npP[i])
    if (d >= maxi):
      maxi = d
      pMax = npP[i]
      idxMax = i
    # jika jaraknya sama dgn jarak maks sebelumnya, dihitung sudutnya
    elif (d == maxi):
      angleA = angleP(npP[i], np1, np2)
      angleB = angleP(pMax, np1, np2)
      if (angleA > angleB):
        pMax = npP[i]
        idxMax = i
  # mengembalikan index titik dengan jarak terjauh
  return idxMax
```

Fungsi recursiveCHull

Fungsi recursiveCHull digunakan untuk mencari convex hull secara rekursif Berikut source codenya:

```
# listP adalah kumpulan seluruh titik, p1 dan p2 adalah indeks titik
# listIdx adalah kumpulan indeks titik
def recursiveCHull(listP, p1, p2, listIdx):
  # jika tidak ada titik lain, maka convex hullnya adalah pasangan p1 dan p2
  if (len(listIdx) == 0):
    return [[p1, p2]]
  # jika hanya ada satu titik, maka convex hullnya adalah segitiga p1, px, p2
  elif (len(listIdx) == 1):
    return [[p1, listIdx[0]], [listIdx[0], p2]]
  else:
    # mencari titik terjauh dari garis yang dibentuk p1 dan p2
    px = farthestP(listP, p1, p2, listIdx)
    # membagi kumpulan titik berdasarkan garis yang dibentuk
    # antara p1 ke px dan px ke p2
    leftA, rightA = split(listP, p1, px, listIdx)
    leftB, rightB = split(listP, px, p2, listIdx)
    # melakukan divide secara rekursif
    A = recursiveCHull(listP, p1, px, leftA)
    B = recursiveCHull(listP, px, p2, leftB)
    # conquer list pasangan convex hull
    A.extend(B)
    return A
```

Fungsi myConvexHull

Fungsi myConvexHull adalah fungsi utama yang digunakan untuk mencari convex hull secara keseluruhan. Berikut source codenya:

```
# listP adalah kumpulan seluruh titik
def myConvexHull(listP):
  listIdx = []
  idxMin = 0
  idxMax = 0
  # mencari titik ekstrem untuk p1 dan p2
  for i in range(len(listP)):
    if (listP[i][0] <= listP[idxMin][0]):
      if (listP[i][1] < listP[idxMin][1]):
        idxMin = i
    if (listP[i][0] >= listP[idxMax][0]):
      if (listP[i][1] > listP[idxMax][1]):
        idxMax = i
    listIdx.append(i)
  # bagi kumpulan titik berdasarkan garis p1 dan p2
  p1 = idxMin
  p2 = idxMax
  left, right = split(listP, p1, p2, listIdx)
  # melakukan divide secara rekursif pada bagian kiri/atas dan kanan/bawah garis
  leftHull = recursiveCHull(listP, p1, p2, left)
  rightHull = recursiveCHull(listP, p2, p1, right)
  # conquer list semua pasangan convex hull
  leftHull.extend(rightHull)
  return leftHull
```

Input dataset iris

```
import pandas as pd
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)
print(df.shape)
df.head()
```

```
#visualisasi hasil ConvexHull
import matplotlib.pyplot as plt
from scipy.spatial import ConvexHull
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) # hasil implementasi ConvexHull Divide & Conquer
  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()
```

```
#visualisasi hasil ConvexHull
import matplotlib.pyplot as plt
from scipy.spatial 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
  hull = myConvexHull(bucket) # hasil implementasi ConvexHull Divide & Conquer
  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()
```

Input dataset wine

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

```
#visualisasi hasil ConvexHull
import matplotlib.pyplot as plt
from scipy.spatial import ConvexHull
plt.figure(figsize = (10, 6))
colors = ['b','r','g']
plt.title('Malic Acid vs Alcohol')
plt.xlabel(dataWine.feature_names[0])
plt.ylabel(dataWine.feature_names[1])
for i in range(len(dataWine.target_names)):
  bucket = df[df['Target'] == i]
  bucket = bucket.iloc[:,[0,1]].values
  hull = myConvexHull(bucket) # hasil implementasi ConvexHull Divide & Conquer
  plt.scatter(bucket[:, 0], bucket[:, 1], label=dataWine.target_names[i])
  for simplex in hull:
    plt.plot(bucket[simplex, 0], bucket[simplex, 1], colors[i])
plt.legend()
```

```
#visualisasi hasil ConvexHull
import matplotlib.pyplot as plt
from scipy.spatial import ConvexHull
plt.figure(figsize = (10, 6))
colors = ['b','r','g']
plt.title('Alcalinity of Ash vs Ash')
plt.xlabel(dataWine.feature_names[2])
plt.ylabel(dataWine.feature_names[3])
for i in range(len(dataWine.target_names)):
  bucket = df[df['Target'] == i]
  bucket = bucket.iloc[:,[2,3]].values
  hull = myConvexHull(bucket) # hasil implementasi ConvexHull Divide & Conquer
  plt.scatter(bucket[:, 0], bucket[:, 1], label=dataWine.target_names[i])
  for simplex in hull:
    plt.plot(bucket[simplex, 0], bucket[simplex, 1], colors[i])
plt.legend()
```

Input dataset breast cancer

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

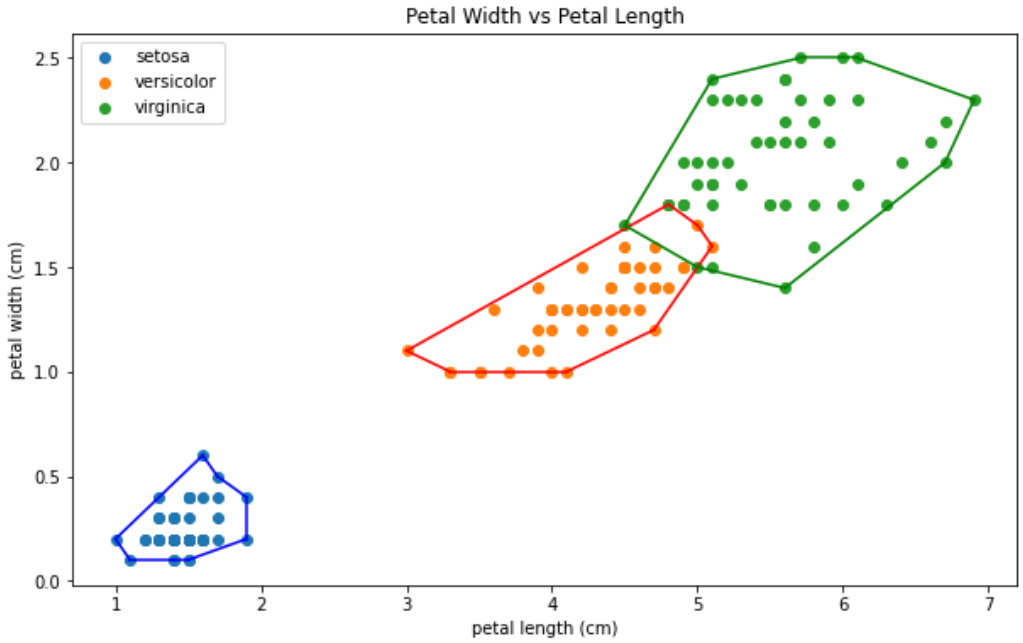
```
#visualisasi hasil ConvexHull
import matplotlib.pyplot as plt
from scipy.spatial import ConvexHull
plt.figure(figsize = (10, 6))
colors = ['b','r','g']
plt.title('Mean Texture vs Mean Radius')
plt.xlabel(databc.feature_names[0])
plt.ylabel(databc.feature_names[1])
for i in range(len(databc.target_names)):
  bucket = df[df['Target'] == i]
  bucket = bucket.iloc[:,[0,1]].values
  hull = myConvexHull(bucket) # hasil implementasi ConvexHull Divide & Conquer
  plt.scatter(bucket[:, 0], bucket[:, 1], label=databc.target_names[i])
  for simplex in hull:
    plt.plot(bucket[simplex, 0], bucket[simplex, 1], colors[i])
plt.legend()
```

```
#visualisasi hasil ConvexHull
import matplotlib.pyplot as plt
from scipy.spatial import ConvexHull
plt.figure(figsize = (10, 6))
colors = ['b','r','g']
plt.title('Mean Area vs Mean Perimeter')
plt.xlabel(databc.feature_names[2])
plt.ylabel(databc.feature_names[3])
for i in range(len(databc.target_names)):
  bucket = df[df['Target'] == i]
  bucket = bucket.iloc[:,[2,3]].values
  hull = myConvexHull(bucket) # hasil implementasi ConvexHull Divide & Conquer
  plt.scatter(bucket[:, 0], bucket[:, 1], label=databc.target_names[i])
  for simplex in hull:
    plt.plot(bucket[simplex, 0], bucket[simplex, 1], colors[i])
plt.legend()
```

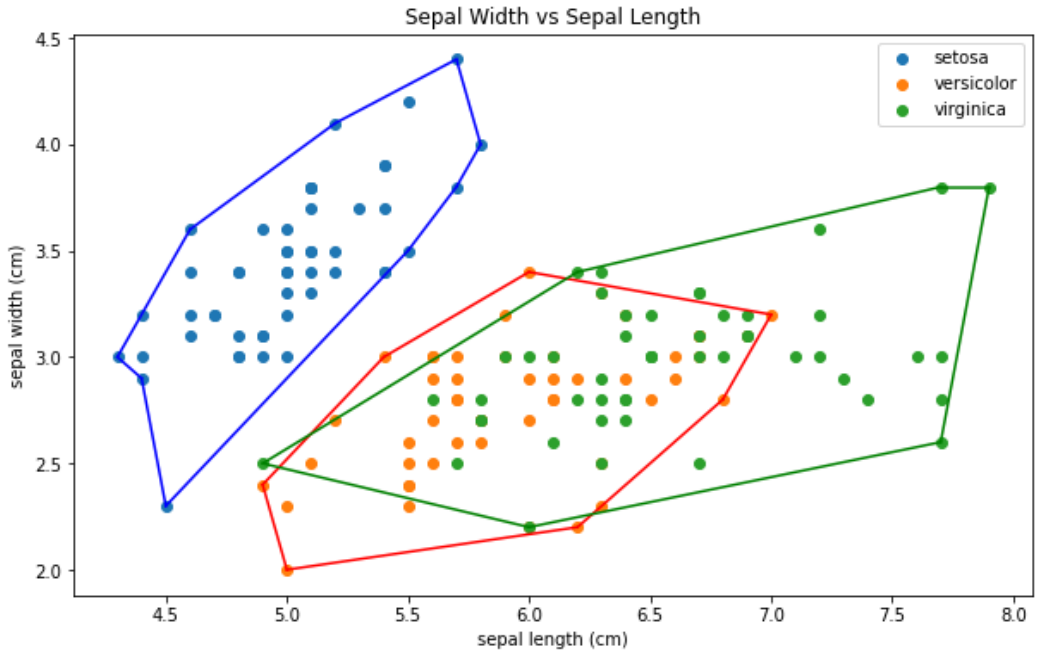
CONTOH INPUT/OUTPUT

Dataset iris

1) Petal Width vs Petal Length

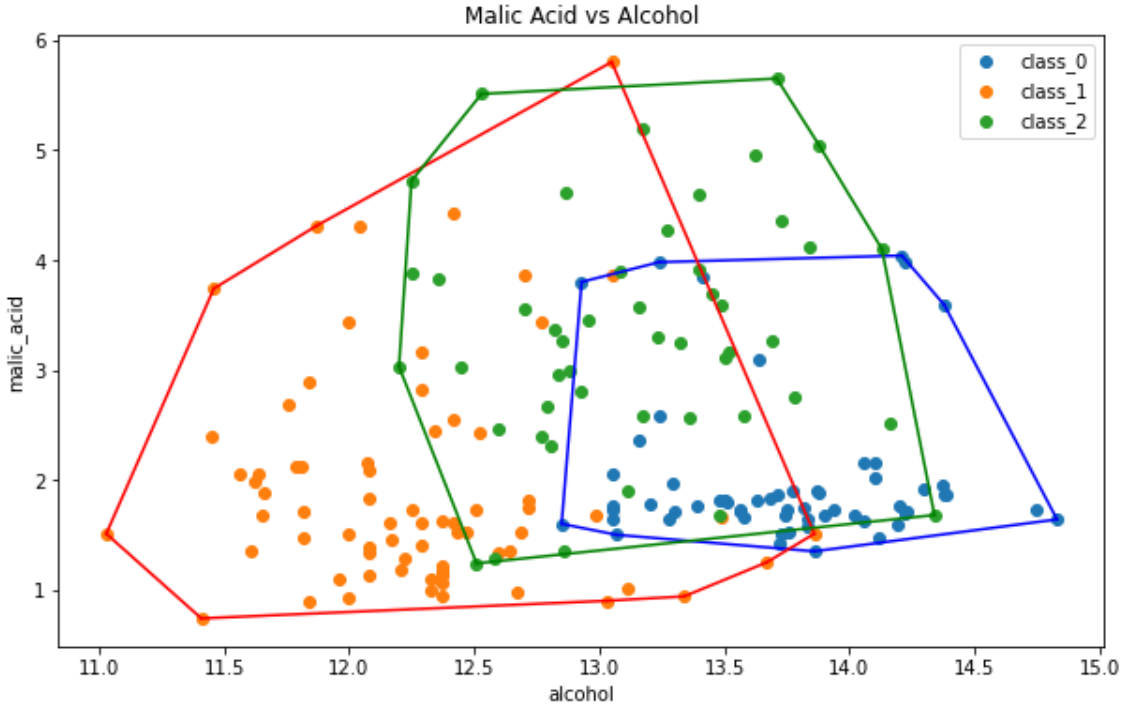


2) Sepal Width vs Sepal Length

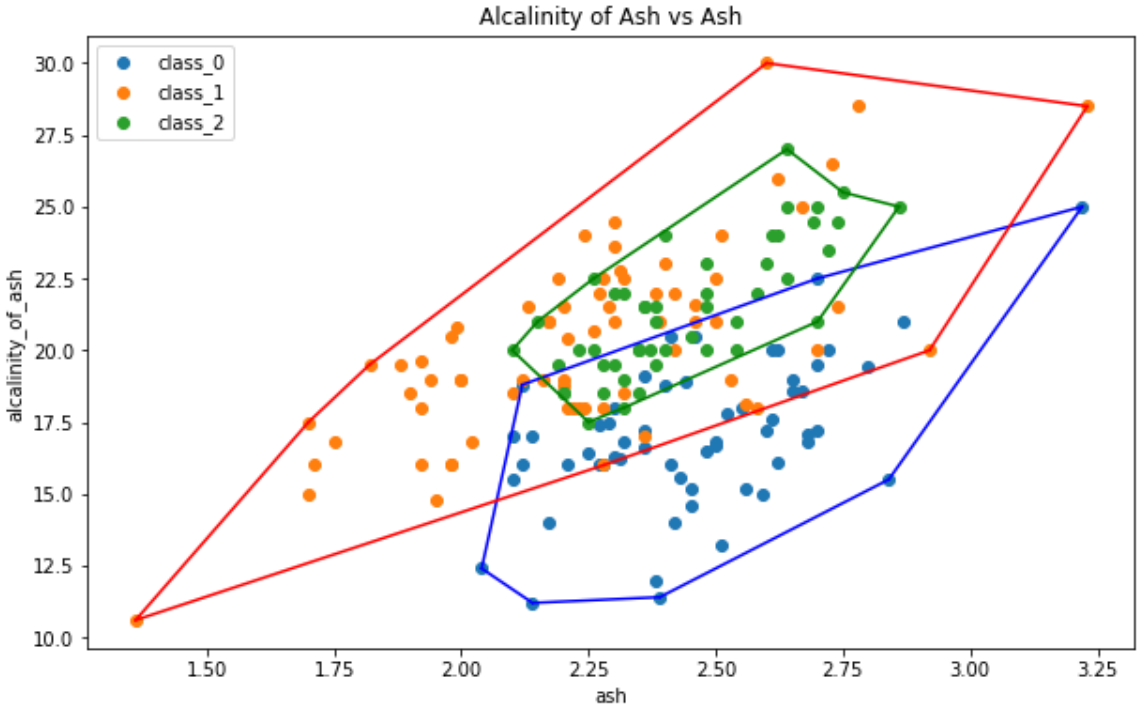


Dataset wine

1) Malic Acid vs Alcohol

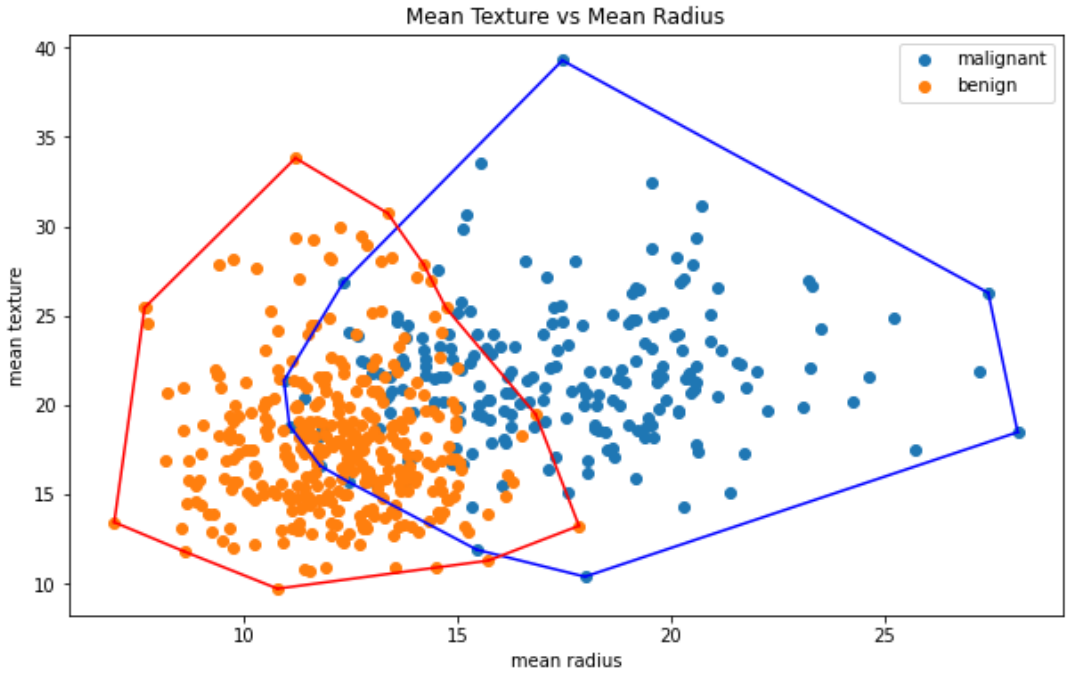


2) Alcalinity of Ash vs Ash

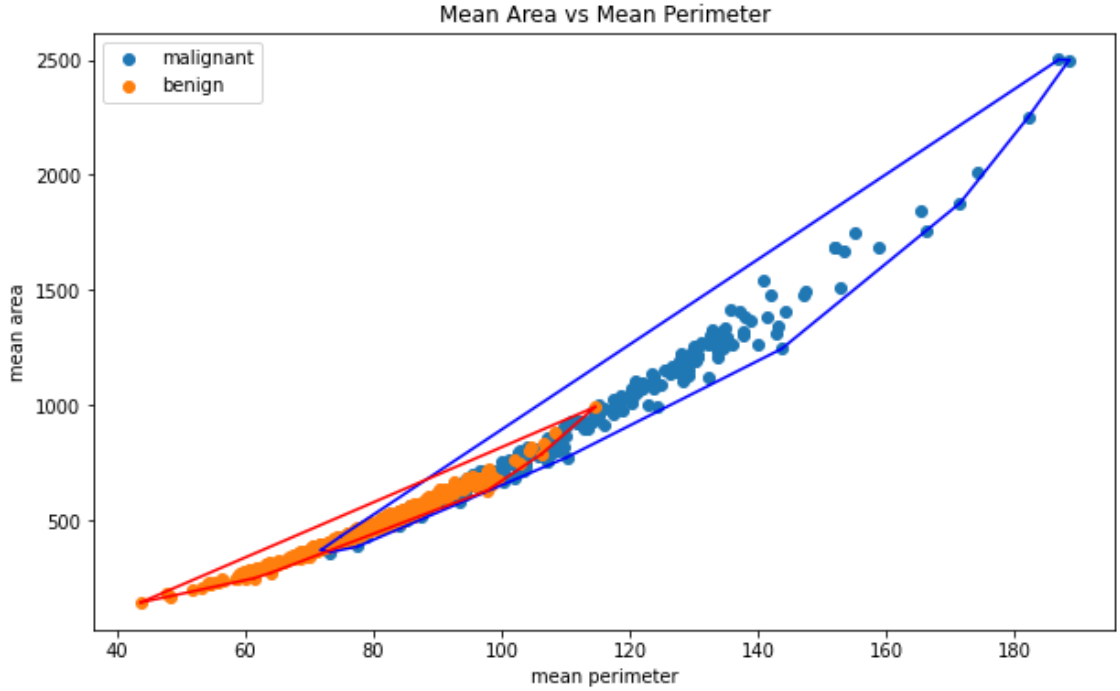


Dataset breast cancer

1) Mean Texture vs Mean Radius



2) Mean Area vs Mean Perimeter



LAMPIRAN

Link Github/Google Colab

https://github.com/roastland/Tucil2_13520031
https://colab.research.google.com/drive/1XgadgG6QXANYkBunLkwVn45uN8jSimF4?usp=sharing

Check List Asisten

No.	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.	✓	