Laporan Tugas Kecil 2
IF 2211 Strategi Algoritma
Library Convex Hull dengan Divide and Conquer

Disusun Oleh:
13520163 Frederik Imanuel Louis

Sekolah Tinggi Elektro dan Informatika
Institut Teknologi Bandung
2022

Daftar Isi

BAB 1
Algoritma Program

1.1 Fungsi convexHull (Fungsi Utama)

Program utama yang menggunakan library akan memanggil fungsi convexHull dengan parameter list titik-titik yang ingin dicari Convex Hullnya. Fungsi convexHull pertama mengurutkan list titik berdasarkan nilai absis dan ordinat yang meningkat. Kemudian, fungsi menetapkan kedua titik tersebut sebagai pivot dan membagi semua titik lain ke dalam dua list, yaitu titik yang berada di sebelah kiri garis pivot dan titik yang berada di sebelah kanan garis pivot. Penentuan posisi titik terhadap pivot akan dibahas pada bagian selanjutnya. Kemudian, fungsi convexHull akan memanggil fungsi convexHullRec yang akan melakukan divide and conquer secara rekursif untuk tiap subproblem (titik sebelah kiri dan kanan). Kemudian, fungsi menggabung solusi dari kedua subproblem dan mengembalikan jawaban yang diperoleh.

1.2 Penentuan Posisi Titik terhadap Dua Pivot (Fungsi getArea)

Pada fungsi convex hull awal dan pada algoritma divide and conquer berikutnya, digunakan fungsi getArea untuk menentukan posisi titik terhadap garis pivot. Fungsi getArea menggunakan shoelace theorem untuk menentukan oriented area dari segitiga yang dibentuk ketiga titik tersebut. Dengan menentukan urutan perhitungan pivot dan titik, dapat diketahui posisi titik terhadap garis pivot. Pada implementasi getArea, digunakan urutan (Pivot1,Pivot2,Titik) dimana titik adalah titik yang diperiksa dan Pivot1 selalu berada di kiri Pivot2. Dengan demikian, jika area yang ditemukan bernilai positif, maka titik berada di sebelah kiri/atas dari pivot, dan di sebelah kanan/bawah jika sebaliknya. Selain itu, nilai absolut dari area juga dapat digunakan untuk menentukan jarak titik dari garis pivot. Makin besar nilai absolut dari area, maka makin besar jarak titik tersebut dari garis pivot. Jarak tersebut akan digunakan pada algoritma divide and conquer.

1.3 Fungsi convexHullRec (Divide and Conquer Rekursif)

Fungsi convexHullRec akan menerima tiga parameter, yaitu list titik pada subproblem yang dikerjakan, pivot kiri, serta pivot kanan. Fungsi terdiri dari tiga bagian, yaitu:

1. Base Case
   Fungsi pertama mengecek apakah subproblem sudah mencapai base case, yaitu saat list titik kosong. Jika base case dicapai, maka fungsi akan mengembalikan jawaban subproblem yaitu pivot kanan. Pivot kiri tidak perlu dikembalikan untuk menghindari pencatatan titik dobel saat proses combine.
2. Divide
   Jika base case tidak dicapai, maka fungsi akan pertama-tama mengiterasi semua titik dengan memanggil getArea(pivot1,pivot2,titik) dan mencatat titik dengan nilai absolut area yang paling besar untuk mencari titik dengan jarak paling jauh dari garis pivot. Kemudian, titik tersebut akan digunakan sebagai pivot berikutnya. Setelah menemukan titik terjauh tersebut, maka fungsi akan mengiterasi semua titik lagi dan memanggil getArea(pivot1,titikMax,titik) dan getArea(titikMax,pivot2,titik), dimana titik dengan area yang positif akan dicatat pada pts1 dan pts2 untuk rekursi selanjutnya.

3. Conquer
   Fungsi akan kemudian memanggil convexHullRec(pts1,pivot1,titikMax) dan convexHullRec(pts2,tiitkMax,pivot2) untuk membagi subproblem menjadi 2 subproblem berikutnya. Pemanggilan fungsi pertama akan mengembalikan titik yang menjadi bagian convex hull diantara pivot1 dan titikMax, dan pemanggilan fungsi kedua akan mengembalikan titik yang menjadi bagian dari convex hull diantara titikMax dan pivot2.
4. Combine
   Fungsi akan mengkonkatenasi kedua hasil yang diperoleh dari penyelesaian subproblem. Karena pada base case fungsi hanya mengembalikan pivot kanan, maka titik-titik pada tiap hasil adalah unik dan saling lepas, sehingga konkatenasi list secara langsung cukup untuk mendapat jawaban subproblem ini. Fungsi akan kemudian mengembalikan hasil konkatenasi tersebut.

BAB II
Source Code Program

2.1 Program Convex Hull

```
def convexHull(pts): #recieves list of points
    if(len(pts)<2):
        return pts #needs at least two points
    pts.sort()
    a=pts[0]
    b=pts[len(pts)-1]
    left=[] #left of pivot ab
    right=[] #right of pivot ab
    for p in pts: #traverse all points and divide points
        area=getArea(a,b,p)
        if (area>0.00001):
            left.append(p)
        elif (area<-1*0.00001):
            right.append(p)

    #call recursion
    left=convexHullRec(left,a,b)
    right=convexHullRec(right,b,a)
    return left+right


def getArea(a,b,c): #gets area of triangle ABC
    #if positive, c is to the left of line ab, and right otherwise
    return a[0]*b[1]+b[0]*c[1]+c[0]*a[1]-a[0]*c[1]-b[0]*a[1]-c[0]*b[1]


def convexHullRec(pts,a,b): #recursive function (points, pivot left, pivot right)
    if (len(pts)==0): #meets base case, return right pivot only
        return [b]

    #if not base case, divide and recurse
    #step 1: get furthest point
    maxArea=0
    pivot=[]
    for p in pts:
        area=getArea(a,b,p)
        if(area>maxArea):
            maxArea=area
            pivot=p

    #step 2: get left and right
    left=[]
    right=[]
    for p in pts:
        areaL=getArea(a,pivot,p)
        areaR=getArea(pivot,b,p)
        if (areaL>0.00001):
            left.append(p)
```

```
        elif (areaR>0.00001):
            right.append(p)

    #call recursion
    left=convexHullRec(left,a,pivot)
    right=convexHullRec(right,pivot,b)
    return left+right
```

2.2 Program Utama (Pengguna myConvexHull)

```
# # Tugas Kecil 2 Strategi Algoritma: Convex Hull
# ## 1. Convex Hull Data Iris
# ### Load Data Iris:

# In[1]:


import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import myConvexHull as cv
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)

# ### Plot Convex Hull data Iris (Sepal Length vs Width):

# In[2]:


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
    bucket = bucket.tolist() #convert to list
    hull = cv.convexHull(bucket)
    bucket = np.array(bucket) #convert back to np.array
    plt.scatter(bucket[:, 0], bucket[:, 1], label=data.target_names[i])
    x=[hull[len(hull)-1][0]] #get x coordinates
    y=[hull[len(hull)-1][1]] #get y coordinates
    for p in hull:
        x.append(p[0])
        y.append(p[1])
    plt.plot(x, y, colors[i]) #plot
    plt.legend()
plt.show()
```

```
# ### Plot Convex Hull Data Iris (Petal Length vs Width):

# In[3]:


plt.figure(figsize = (10, 6))
colors = ['b','r','g']
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
    bucket = bucket.tolist() #convert to list
    hull = cv.convexHull(bucket)
    bucket = np.array(bucket) #convert back to np.array
    plt.scatter(bucket[:, 0], bucket[:, 1], label=data.target_names[i])
    x=[hull[len(hull)-1][0]] #get x coordinates
    y=[hull[len(hull)-1][1]] #get y coordinates
    for p in hull:
        x.append(p[0])
        y.append(p[1])
    plt.plot(x, y, colors[i]) #plot
    plt.legend()
plt.show()


# ## 2. Convex Hull Data Wine
# ### Load Data Wine:

# In[4]:


data = datasets.load_wine()
df = pd.DataFrame(data.data, columns=data.feature_names)
df['Target'] = pd.DataFrame(data.target)


# ### Plot Data Wine (Alchohol vs Flavanoids)

# In[5]:


plt.figure(figsize = (10, 6))
colors = ['b','r','g']
plt.title('Alchohol vs Flavanoids')
plt.xlabel(data.feature_names[0])
plt.ylabel(data.feature_names[6])
for i in range(len(data.target_names)):
    bucket = df[df['Target'] == i]
    bucket = bucket.iloc[:,[0,6]].values
    bucket = bucket.tolist() #convert to list
    hull = cv.convexHull(bucket)
    bucket = np.array(bucket) #convert back to np.array
```

```
    plt.scatter(bucket[:, 0], bucket[:, 1], label=data.target_names[i])
    x=[hull[len(hull)-1][0]] #get x coordinates
    y=[hull[len(hull)-1][1]] #get y coordinates
    for p in hull:
        x.append(p[0])
        y.append(p[1])
    plt.plot(x, y, colors[i]) #plot
    plt.legend()
plt.show()


# ### Plot Data Wine (Ash vs Total Phenols)

# In[6]:


plt.figure(figsize = (10, 6))
colors = ['b','r','g']
plt.title('Ash vs Total Phenols')
plt.xlabel(data.feature_names[2])
plt.ylabel(data.feature_names[5])
for i in range(len(data.target_names)):
    bucket = df[df['Target'] == i]
    bucket = bucket.iloc[:,[2,5]].values
    bucket = bucket.tolist() #convert to list
    hull = cv.convexHull(bucket)
    bucket = np.array(bucket) #convert back to np.array
    plt.scatter(bucket[:, 0], bucket[:, 1], label=data.target_names[i])
    x=[hull[len(hull)-1][0]] #get x coordinates
    y=[hull[len(hull)-1][1]] #get y coordinates
    for p in hull:
        x.append(p[0])
        y.append(p[1])
    plt.plot(x, y, colors[i]) #plot
    plt.legend()
plt.show()
```

BAB III
Pengujian Progam

3.1 Data Iris

Load Data Iris:

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import myConvexHull as cv
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()
```

```
(150, 5)
```

Out[1]:

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)	Target
0	5.1	3.5	1.4	0.2	0
1	4.9	3.0	1.4	0.2	0
2	4.7	3.2	1.3	0.2	0
3	4.6	3.1	1.5	0.2	0
4	5.0	3.6	1.4	0.2	0

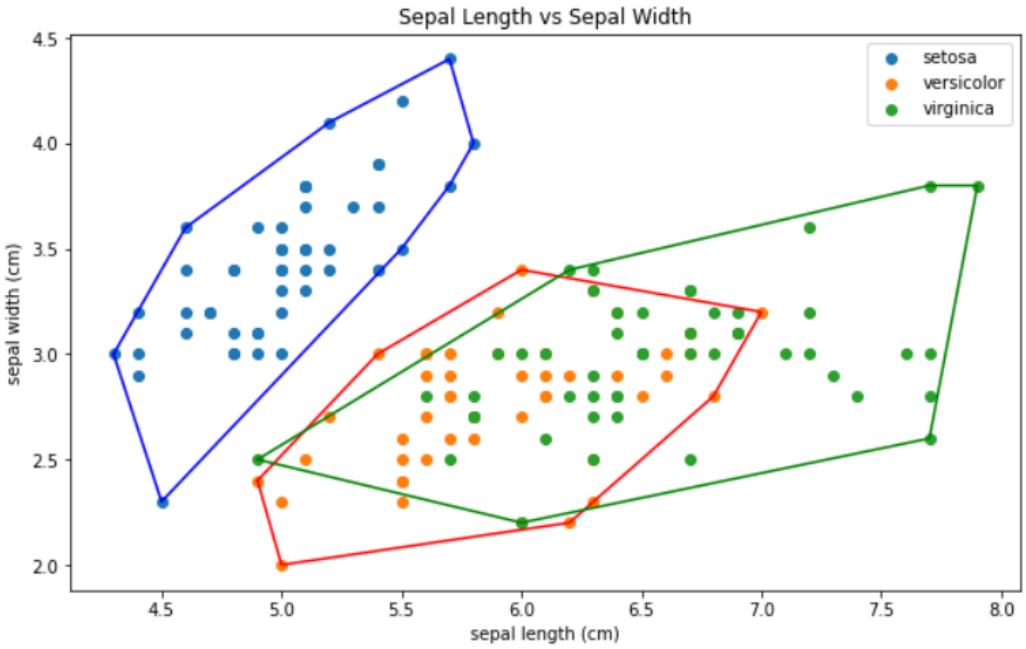
Gambar 1. Load Data Iris

3.1.1 Sepal Length vs Sepal Width

Plot Convex Hull data Iris (Sepal Length vs Width):

In [2]:
```
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
    bucket = bucket.tolist() #convert to list
    hull = cv.convexHull(bucket)
    bucket = np.array(bucket) #convert back to np.array
    plt.scatter(bucket[:, 0], bucket[:, 1], label=data.target_names[i])
    x=[hull[len(hull)-1][0]] #get x coordinates
    y=[hull[len(hull)-1][1]] #get y coordinates
    for p in hull:
        x.append(p[0])
        y.append(p[1])
    plt.plot(x, y, colors[i]) #plot
    plt.legend()
plt.show()
```

Gambar 2. Penggunaan Library Convex Hull pada Sepal Length vs Width



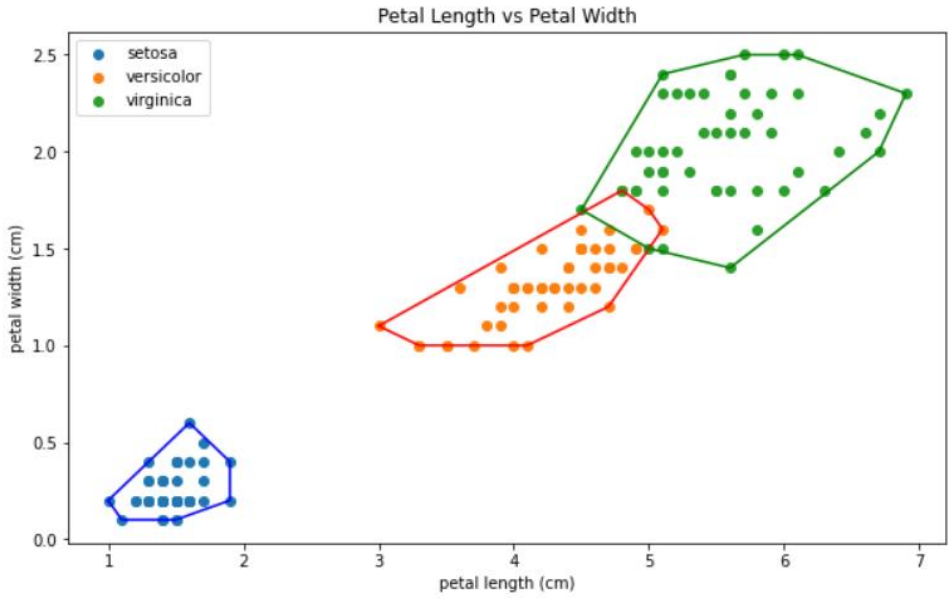
Gambar 3. Plot Convex Hull Sepal Length vs Width

3.1.2 Petal Length vs Petal Width

Plot Convex Hull Data Iris (Petal Length vs Width):

In [3]:
```
plt.figure(figsize = (10, 6))
colors = ['b','r','g']
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
    bucket = bucket.tolist() #convert to list
    hull = cv.convexHull(bucket)
    bucket = np.array(bucket) #convert back to np.array
    plt.scatter(bucket[:, 0], bucket[:, 1], label=data.target_names[i])
    x=[hull[len(hull)-1][0]] #get x coordinates
    y=[hull[len(hull)-1][1]] #get y coordinates
    for p in hull:
        x.append(p[0])
        y.append(p[1])
    plt.plot(x, y, colors[i]) #plot
    plt.legend()
plt.show()
```

Gambar 4. Penggunaan Library Convex Hull pada Petal Length vs Width



Gambar 5. Plot Convex Hull Petal Length vs Width

3.2 Data Wine

Load Data Wine:

In [4]:
```
data = datasets.load_wine()
df = pd.DataFrame(data.data, columns=data.feature_names)
df['Target'] = pd.DataFrame(data.target)
print(df.shape)
df.head()
```

```
(178, 14)
```

Out[4]:

	alcohol	malic_acid	ash	alcalinity_of_ash	magnesium	total_phenols	flavanoids	nonflavanoid_phenols	proanthocyanins	color_intensity	hue	od280/od315
0	14.23	1.71	2.43	15.6	127.0	2.80	3.06	0.28	2.29	5.64	1.04	
1	13.20	1.78	2.14	11.2	100.0	2.65	2.76	0.26	1.28	4.38	1.05	
2	13.16	2.36	2.67	18.6	101.0	2.80	3.24	0.30	2.81	5.68	1.03	
3	14.37	1.95	2.50	16.8	113.0	3.85	3.49	0.24	2.18	7.80	0.86	
4	13.24	2.59	2.87	21.0	118.0	2.80	2.69	0.39	1.82	4.32	1.04	

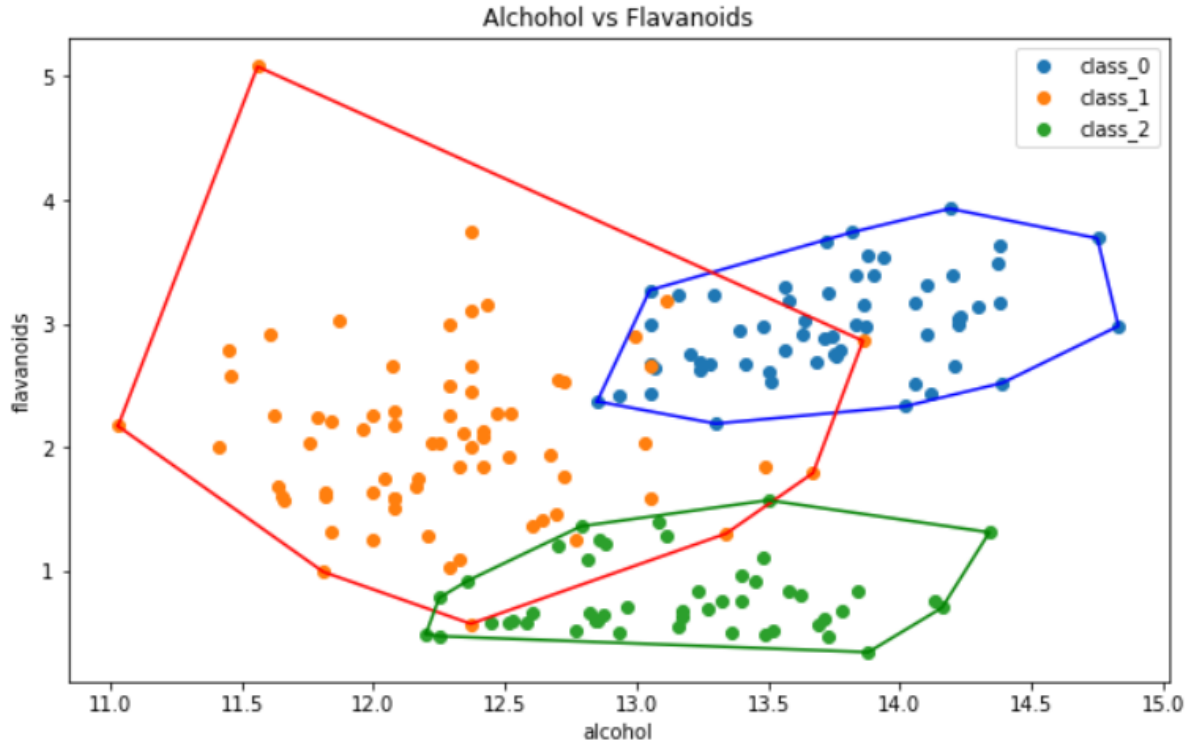
Gambar 6. Load Data Wine

3.2.1 Alchohol vs Flavanoids

Plot Data Wine (Alchohol vs Flavanoids)

In [5]:
```
plt.figure(figsize = (10, 6))
colors = ['b','r','g']
plt.title('Alchohol vs Flavanoids')
plt.xlabel(data.feature_names[0])
plt.ylabel(data.feature_names[6])
for i in range(len(data.target_names)):
    bucket = df[df['Target'] == i]
    bucket = bucket.iloc[:,[0,6]].values
    bucket = bucket.tolist() #convert to list
    hull = cv.convexHull(bucket)
    bucket = np.array(bucket) #convert back to np.array
    plt.scatter(bucket[:, 0], bucket[:, 1], label=data.target_names[i])
    x=[hull[len(hull)-1][0]] #get x coordinates
    y=[hull[len(hull)-1][1]] #get y coordinates
    for p in hull:
        x.append(p[0])
        y.append(p[1])
    plt.plot(x, y, colors[i]) #plot
    plt.legend()
plt.show()
```

Gambar 7. Penggunaan Library Convex Hull pada Alchohol vs Flavonoids



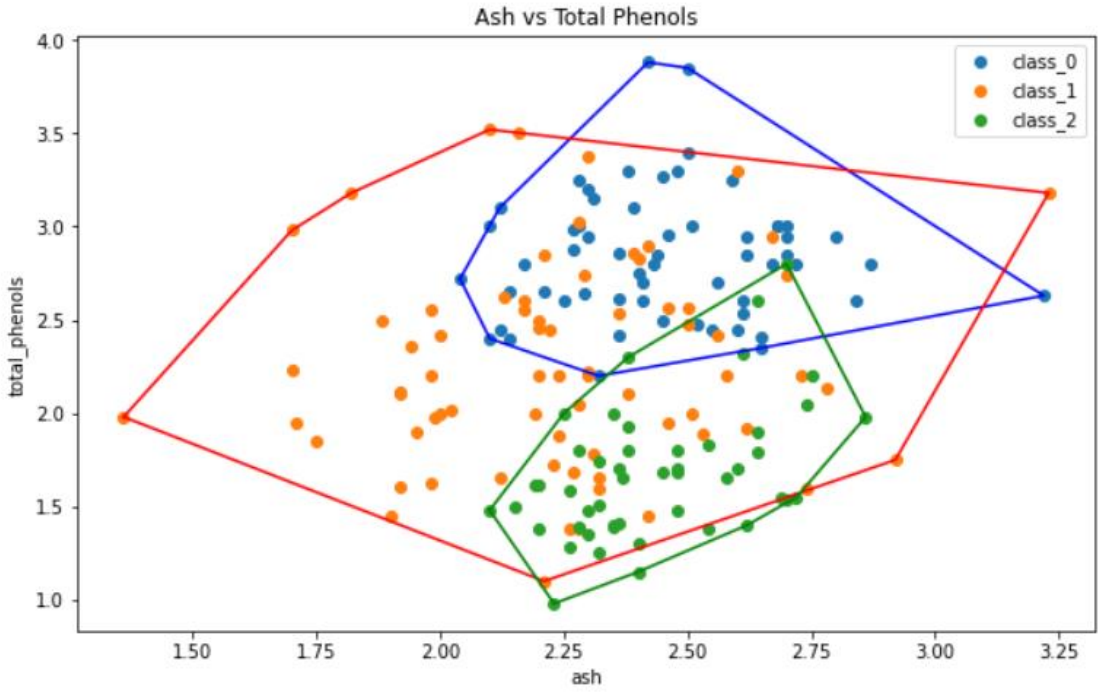
Gambar 8. Plot Convex Hull Alchohol vs Flavonoids

3.2.2 Ash vs Total Phenols

Plot Data Wine (Ash vs Total Phenols)

In [6]:
```
plt.figure(figsize = (10, 6))
colors = ['b','r','g']
plt.title('Ash vs Total Phenols')
plt.xlabel(data.feature_names[2])
plt.ylabel(data.feature_names[5])
for i in range(len(data.target_names)):
    bucket = df[df['Target'] == i]
    bucket = bucket.iloc[:,[2,5]].values
    bucket = bucket.tolist() #convert to list
    hull = cv.convexHull(bucket)
    bucket = np.array(bucket) #convert back to np.array
    plt.scatter(bucket[:, 0], bucket[:, 1], label=data.target_names[i])
    x=[hull[len(hull)-1][0]] #get x coordinates
    y=[hull[len(hull)-1][1]] #get y coordinates
    for p in hull:
        x.append(p[0])
        y.append(p[1])
    plt.plot(x, y, colors[i]) #plot
    plt.legend()
plt.show()
```

Gambar 9. Penggunaan Library Convex Hull pada Ash vs Total Phenols



Gambar 10. Plot Convex Hull Ash vs Total Phenols

Lampiran

Tabel 1. Cek List Status Program

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.	✓	

Repositori program: https://github.com/dxt99/STIMA-TC2