

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

## **LAPORAN TUGAS KECIL 2**

Diajukan Untuk Memenuhi Tugas Kecil IF2211 Strategi Algoritma

Semester II tahun 2021/2022

Disusun oleh

**Patrick Amadeus Irawan (13520109)**



**TEKNIK INFORMATIKA**

**INSTITUT TEKNOLOGI BANDUNG**

**BANDUNG**

**2022**

## BAB I

### Penjelasan Algoritma Program

Algoritma *Divide and Conquer* merupakan jenis algoritma program yang bertujuan untuk menyelesaikan permasalahan dengan membagi persoalan menjadi upa-persoalan dengan karakteristik yang sama sehingga dapat diselesaikan dengan algoritma yang telah dibuat. Implementasi algoritma ini lebih natural diungkapkan dalam skema rekursif. Umumnya, algoritma *Divide and Conquer* dibagi menjadi beberapa tahap, yakni:

1. *Divide* : membagi persoalan menjadi beberapa upa-persoalan yang memiliki kemiripan dengan persoalan semula namun berukuran lebih kecil
2. *Conquer (solve)* : menyelesaikan masing-masing upa-persoalan ( secara langsung jika sudah berukuran kecil atau secara rekursif jika masih berukuran besar )
3. *Combine* : menggabungkan solusi masing-masing upa-persoalan sehingga membentuk solusi persoalan semula

Implementasi visualisasi *Convex Hull* dapat diselesaikan dengan jenis algoritma ini. Berikut deskripsi langkah-langkah untuk membentuk suatu *Convex Hull* dari parameter tertentu dari dataset yang dipilih dengan algoritma *Divide and Conquer* :

1. Pada algoritma (fungsi) *myConvexHull*, diterima set titik pembentuk convex polygon , yakni daerah yang dibatasi oleh titik-titik pembentuk convex hull.
2. Pengecekan basis set titik **S**:
  - a. Apabila set titik **S** kosong , maka algoritma rekursi akan berhenti dan mengirimkan set kosong pada set solusi titik pembentuk *convex hull*.
  - b. Apabila set titik **S** berisi 1 buah titik, maka algoritma rekursi akan berhenti dan menambahkan titik tersebut pada set solusi titik pembentuk *convex hull*.
3. Untuk rekursi pertama, akan dicari 2 titik ekstrim yakni **p1** dan **p2** dengan implementasi *quick sort*. Titik ekstrim dicari dengan mengurutkan set titik berdasarkan absis membesar, kemudian ordinat membesar. Garis **g** yang menghubungkan **p1** dan **p2** akan membagi daerah menjadi dua bagian yang direpresentasikan sebagai set titik. Untuk setiap bagian akan lanjut ke tahap 4.
4. Segmen ini adalah rekurens untuk 2 bagian awal yang telah dibagi di tahap 3 dimana kesatuan terbagi menjadi bagian atas dan bagian bawah, algoritma untuk menyelesaikan kedua bagian ini sedikit berbeda dari segi orientasi, yakni sebagai berikut :

- a. Untuk set titik **S** yang berada di bagian atas garis **g** yang dihubungkan oleh **p1** dan **p2**, tahap penyelesaian sebagai berikut:
  - i. Dilakukan pengecekan basis (tahap 2).
  - ii. Akan dicari titik **p\_max** dengan jarak terjauh dari garis **g** , apabila jarak 2 atau lebih titik sama, akan dicari titik yang membentuk sudut terbesar (**p1,p\_max,p2**) . **p\_max** akan berada di atas garis **g**.
  - iii. Garis **L** akan menghubungkan **p1** dengan **p\_max**, sedangkan Garis **R** akan menghubungkan **p\_max** dengan **p2**. Untuk kedua garis, tahap penyelesaian sebagai berikut:
    1. Akan dibentuk set titik **S1** yang merupakan titik-titik yang berada di atas garis **L**. Dengan menganggap **S1** sebagai **S**, **p1** sebagai **p1** ,**p\_max** sebagai **p2**, dan **L** sebagai **g**, rekurens dilanjutkan (tahap 4a).
    2. Akan dibentuk set titik **S1** yang merupakan titik-titik yang berada di atas garis **R**. Dengan menganggap **S1** sebagai **S**, **p1** sebagai **p1** ,**p\_max** sebagai **p2**, dan **R** sebagai **g**, rekurens dilanjutkan (tahap 4a).
- b. Untuk set titik **S** yang berada di bagian bawah garis **g** yang dihubungkan oleh **p1** dan **p2**, tahap penyelesaian sebagai berikut:
  - i. Dilakukan pengecekan basis (tahap 2).
  - ii. Akan dicari titik **p\_max** dengan jarak terjauh dari garis **g** , apabila jarak 2 atau lebih titik sama, akan dicari titik yang membentuk sudut terbesar (**p1,p\_max,p2**) . **p\_max** akan berada di bawah garis **g**.
  - iii. Garis **L** akan menghubungkan **p1** dengan **p\_max**, sedangkan Garis **R** akan menghubungkan **p\_max** dengan **p2**. Untuk kedua garis, tahap penyelesaian sebagai berikut:
    1. Akan dibentuk set titik **S1** yang merupakan titik-titik yang berada di atas garis **L**. Dengan menganggap **S1** sebagai **S**, **p1** sebagai **p1** ,**p\_max** sebagai **p2**, dan **L** sebagai **g**, rekurens dilanjutkan (tahap 4a).
    2. Akan dibentuk set titik **S1** yang merupakan titik-titik yang berada di atas garis **R**. Dengan menganggap **S1** sebagai **S**, **p1** sebagai **p1** , **p\_max** sebagai **p2**, dan **R** sebagai **g**, rekurens dilanjutkan (tahap 4a).

Berikut panduan tambahan untuk menjalankan program yang telah dibuat:

1. Masuk ke folder **src** dan buka file **main.ipynb**
2. Visualisasi dataset uji dapat dihasilkan dari beberapa sumber dataset yang telah disediakan, yakni:
  - i. Iris Plants Dataset
  - ii. Wine Recognition Dataset
  - iii. Breast Cancer Wisconsin (Diagnostic) Dataset

Dataset yang dipilih berjenis klasifikasi untuk memungkinkan pembentukan *convex hull* pada tiap target labelnya. Dataset dapat dipilih dengan memasukkan input sesuai dengan referensi angka yang tersedia di notebook.

3. Pilih 2 dari kolom-kolom yang tersedia, pastikan 2 kolom berbeda (terdapat validasi pembedaan kolom).
4. Program akan menganalisis pilihan kolom dan menghasilkan visualisasi **convex hull** dengan implementasi algoritma yang telah dibuat.

## BAB II

### Implementasi Program

**Bahasa Pemrograman :** Python

**Library :** sklearn, pandas, matplotlib

## Convex Hull Visualizer

Nama : Patrick Amadeus Irawan \ NIM : 13520109 \ Kelas : K-01

---

### Table of Content

1. Library Implementation + Dependency
    - QuickSort implementation
    - Helper Function implementation
    - Convex Hull algorithm implementation
  2. Dataset Pick
  3. Columns Pick
  4. Points QuickSort Implementation
  5. Hull Points Gathering
  6. Visualization
- 

#### ***1. Library Implementation + Dependency***

Base Library and Dependency

```
In [ ]:  
  
# base library  
import pandas as pd  
import matplotlib.pyplot as plt  
  
# datasets property  
from sklearn import datasets
```

#### **QuickSort Implementation for points**

***Sorting all points (X,Y) increasing by X then Y***

```
In [ ]:  
  
def quick_sort(points):  
    if len(points) <= 1:  
        return points  
    else:  
        pivot = points[0]  
        less = [point for point in points[1:] if (point[0] < pivot[0]) or (point[0] == p  
ivot[0] and point[1] < pivot[1])]   
        greater = [point for point in points[1:] if (point[0] > pivot[0]) or (point[0] =  
= pivot[0] and point[1] >= pivot[1])]   
        return quick_sort(less) + [pivot] + quick_sort(greater)
```

### Helper Function Implementation

In [ ]:

```
# Point to Line distance finder
def dist_pt_line(p1,p2,pt):
    """
    Gather linear coefficient of line Ax + By + C = 0 and
    using the distance formula to find the distance between point and line

    params : p1, p2, pt = tuples of (x,y)
    return : float
    """
    A,B,C = (p2[1] - p1[1]) , (p1[0] - p2[0]) , (p1[0]*(p1[1] - p2[1]) + p1[1]*(p2[0] - p1[0]))
    return abs(A*pt[0] + B*pt[1] + C)/((A**2 + B**2)**(0.5))

# linear value finder
def linear_value(p1,p2,pt):
    """
    Gather linear coefficient of line Ax + By + C = 0 and
    input X and Y from pt to find the linear value

    params : p1, p2, pt = tuples of (x,y)
    return : float

    Example :
    5x + 2y + 3
    pt = (2,3)

    5*2 + 2*3 + 3 = 15
    """
    A,B,C = (p2[1] - p1[1]) , (p1[0] - p2[0]) , (p1[0]*(p1[1] - p2[1]) + p1[1]*(p2[0] - p1[0]))
    return A*pt[0] + B*pt[1] + C
```

### Divide and Conquer Convex Hull algorithm Implementation

In [ ]:

```
def myConvexHull(points, p1 = None, p2 = None , types = 0):
    """
    Gather points and find the convex hull of the points (set of points)
    points = initial set of points
    p1,p2 = minimum and maximum point of the set of points (based on quick sort)
    types = 0 : initial state
           1 : upper region
           2 : lower region
    base -> 0 <= len(points) < 2
    recursion -> len(points) >= 2

    params : points = list of tuples of (x,y)
    return : list of tuples of (x,y)
    """

    # ---- Base condition ---- #
    if len(points) == 0:
        return []
    if len(points) == 1:
        return [points[0]]

    # ---- Type 0 : Initial 2 area separation ---- #
    if not types:
        # find the minimum and maximum point
        p_min , p_max = points[0] , points[-1]

        upper,lower = [],[]
        for point in points:
            if linear_value(p_min,p_max,point) < 0:
                upper.append(point)
            elif linear_value(p_min,p_max,point) > 0:
                lower.append(point)
        return [p_min] + myConvexHull(upper,p_min,p_max,1) + [p_max] + myConvexHull(lower,p_min,p_max,-1)

    # ---- Type 1 & 2 : Upper and Lower region ---- #

    # Find the point with the maximum distance to the line
    distance, p_max = 0, None
```

```

for point in points:
    if dist_pt_line(p1,p2,point) > distance:
        distance = dist_pt_line(p1,p2,point)
        p_max = point

# Find the points in the upper and lower region
# left : region between p1 and p_max
# right : region between p_max and p2
left,right = [],[]
for point in points:
    if linear_value(p1,p_max,point) * types < 0:
        left.append(point)
    if linear_value(p_max,p2,point) * types < 0:
        right.append(point)

# Recursion order separation
if types == 1:
    return myConvexHull(left , p1 , p_max, types) + [p_max] + myConvexHull(right , p_max , p2, types)
else:
    return myConvexHull(right , p_max , p2, types) + [p_max] + myConvexHull(left , p1 , p_max, types)

```

### Choose any classification dataset from Toy Datasets properties as listed below

1. Iris Plants Dataset
2. Wine recognition dataset
3. Breast cancer wisconsin (diagnostic) dataset

*Enter the number of wanted dataset within input*

In [ ]:

```

data_sets = [
    (datasets.load_iris(), "Iris Plants Dataset"),
    (datasets.load_wine(), "Wine Recognition Dataset"),
    (datasets.load_breast_cancer(), "Breast Cancer Wisconsin (Diagnostic) Dataset"),
]

data_sets_choice = int(input("Enter the dataset number : "))

while data_sets_choice > len(data_sets) or data_sets_choice < 1:
    data_sets_choice = int(input("Please enter the valid dataset number : "))

data = data_sets[data_sets_choice - 1][0]

```

### Chosen Dataset DataFrame Preview

In [ ]:

```

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

```

### Enumerate Target Label from Dataset

In [ ]:

```

n_target = len(df.Target.value_counts())
d_target = {}
for i in range(n_target):
    d_target[i] = (df.Target.value_counts().index[i])

```

### 3. Columns Pick

#### Select 2 fields to be projected with Convex Hull

*(Please choose different columns)*

```
In [ ]:
```

```
idx = 0
msg = ""
for i in range(len(df.columns)):
    msg += (str(i+1)+". "+str(df.columns[i]) + "\n")
print("%s dataset have %d columns which enumerated as below : " % (data_sets[data_sets_c
hoice - 1][1], len(df.columns)))
print(msg)
```

```
In [ ]:
```

```
idX,idY = int(input("Pick column 1 to plot : ")),int(input("Pick column 2 to plot : "))
while idX <= 0 or idX > len(df.columns) or idY < 0 or idY > len(df.columns):
    idX,idY = int(input("Pick column 1 to plot : ")),int(input("Pick column 2 to plot : 
"))
while idX == idY:
    idY = int(input("(Please Pick the different column)Pick column 2 to plot : "))
x,y = df.columns[idX-1],df.columns[idY-1]
```

### 4. Points QuickSort Implementation

```
In [ ]:
```

```
# gather and sort points for chosen columns
d_points = {}
for i in d_target:
    d_points[i] = quick_sort([(df[x].values[j] , df[y].values[j]) for j in range(len(df[
x])) if df.Target.values[j] == i])
# ----- ^^^^^ explanation ^^^^^ -----
#
# explanation : for every target value (i), gather all points and sort them by absis then
ordinate
```

### 5. Hull Points Gathering

```
In [ ]:
```

```
hull_points = {}
for i in range(n_target):
    hull_points[i] = myConvexHull(d_points[i])
```



## 6. Visualization

### Final Scatter Plot with Convex Hull

In [ ]:

```
color_pallette = ["blue", "green", "red", "purple", "magenta", "yellow", "black", "white"] # for plotting

f = plt.figure()
f.set_figwidth(7.5)
f.set_figheight(5)

for i in d_points:
    x, y = [pts[0] for pts in d_points[i]], [pts[1] for pts in d_points[i]]
    plt.scatter(x, y, c = color_pallette[i], label = data.target_names[i], s=20)
```

```
for i in hull_points:
    size = len(hull_points[i])
    for j in range(len(hull_points[i])):
        plt.plot([hull_points[i][j % size][0], hull_points[i][(j + 1) % size][0]], [hull_points[i][j % size][1], hull_points[i][(j + 1) % size][1]], c = color_pallette[i], linewidth = 0.75)

plt.title("Convex Hull of %s" % data_sets[data_sets_choice - 1][1])
plt.xlabel(df.columns[int(idX) - 1])
plt.ylabel(df.columns[int(idY) - 1])
plt.legend()

plt.show()
```

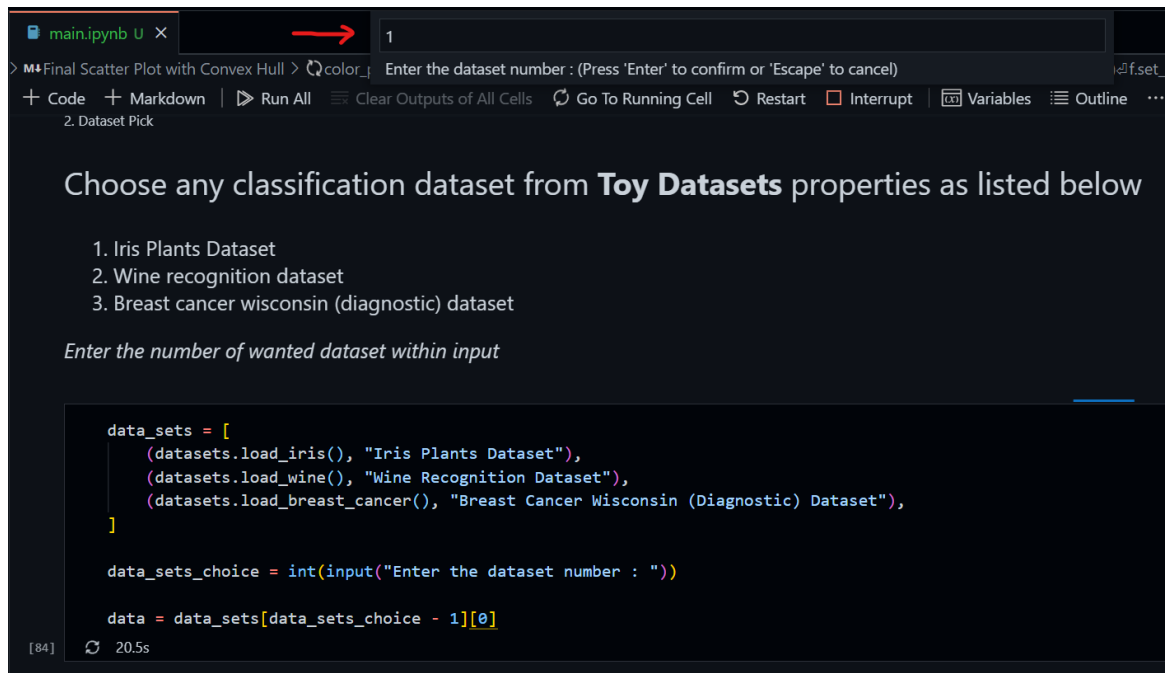
## BAB III

### Hasil Percobaan

#### 1. Iris Plants Dataset

##### INPUT

##### Pemilihan Dataset



```
data_sets = [
    (datasets.load_iris(), "Iris Plants Dataset"),
    (datasets.load_wine(), "Wine Recognition Dataset"),
    (datasets.load_breast_cancer(), "Breast Cancer Wisconsin (Diagnostic) Dataset"),
]

data_sets_choice = int(input("Enter the dataset number : "))

data = data_sets[data_sets_choice - 1][0]
```

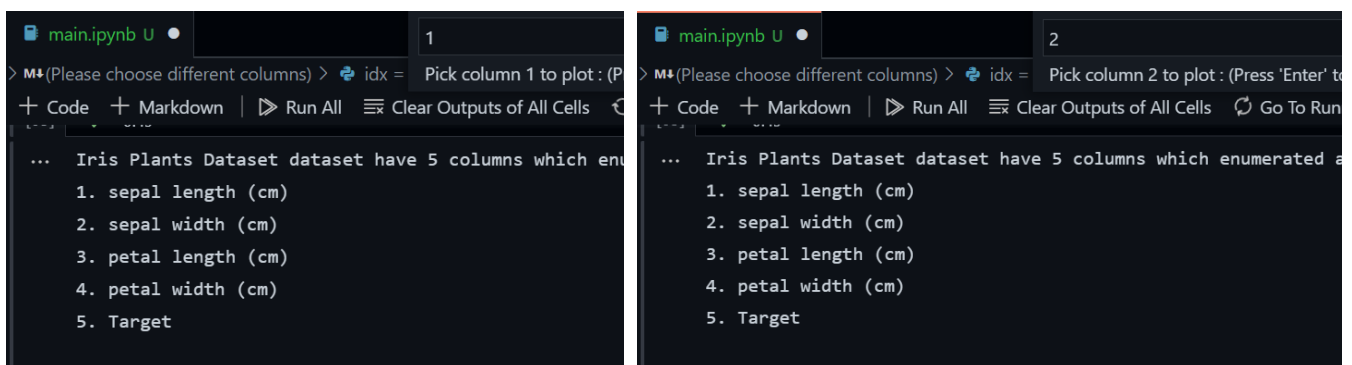
Choose any classification dataset from **Toy Datasets** properties as listed below

1. Iris Plants Dataset
2. Wine recognition dataset
3. Breast cancer wisconsin (diagnostic) dataset

Enter the number of wanted dataset within input

##### Pemilihan Kolom

##### a. sepal length (cm) – sepal width (cm)



```
... Iris Plants Dataset dataset have 5 columns which enumerated as follows:
1. sepal length (cm)
2. sepal width (cm)
3. petal length (cm)
4. petal width (cm)
5. Target
```

```
... Iris Plants Dataset dataset have 5 columns which enumerated as follows:
1. sepal length (cm)
2. sepal width (cm)
3. petal length (cm)
4. petal width (cm)
5. Target
```

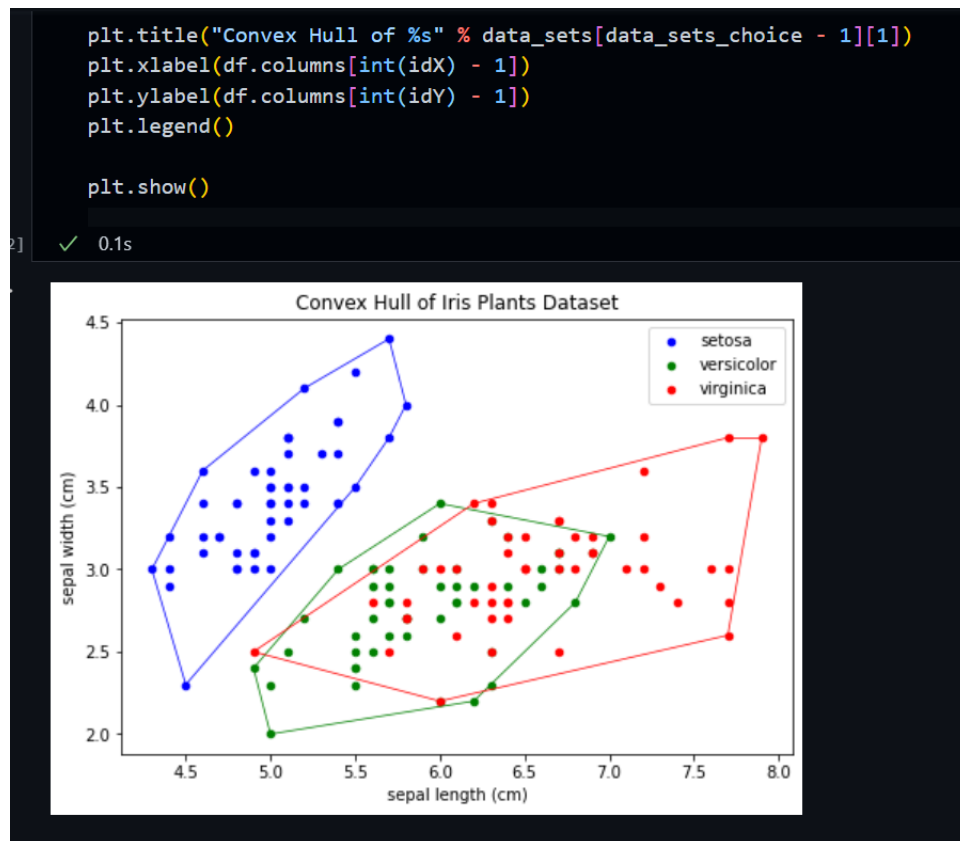
b. petal length (cm) – petal width (cm)

```
main.ipynb U • 3
> M (Please choose different columns) > idx = Pick column 1 to plot: (Press
+ Code + Markdown | Run All Clear Outputs of All Cells
... Iris Plants Dataset dataset have 5 columns which
1. sepal length (cm)
2. sepal width (cm)
3. petal length (cm)
4. petal width (cm)
5. Target
```

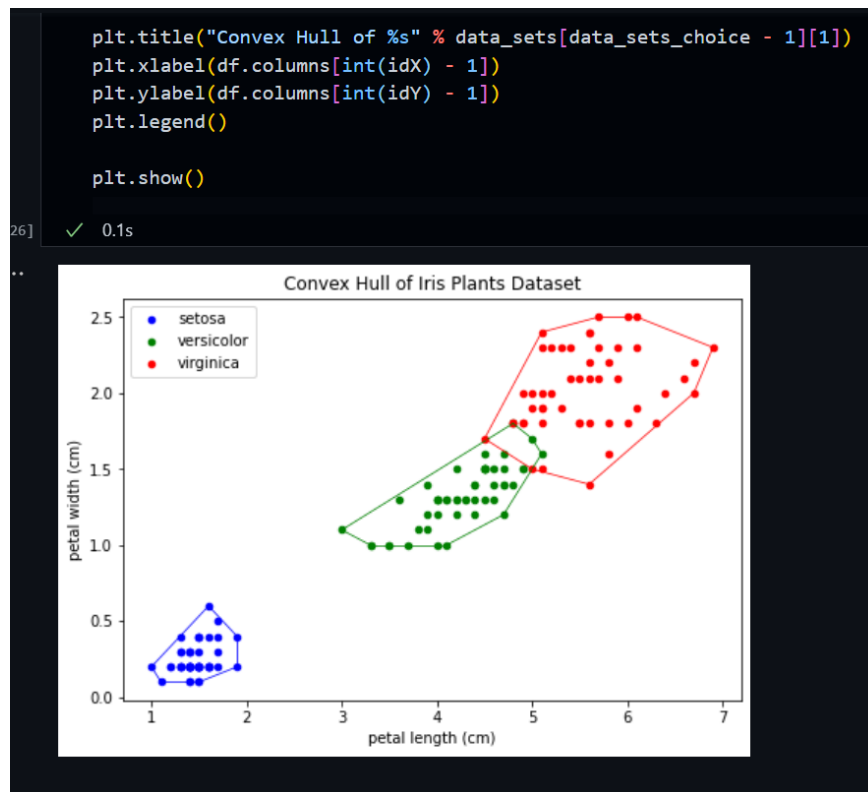
```
main.ipynb U • 4
> M (Please choose different columns) > idx = Pick column 1 to plot: (Press
+ Code + Markdown | Run All Clear Outputs of All Cells
... Iris Plants Dataset dataset have 5 columns which enume
1. sepal length (cm)
2. sepal width (cm)
3. petal length (cm)
4. petal width (cm)
5. Target
```

## OUTPUT

a. sepal length (cm) – sepal width (cm)



b. petal length (cm) – petal width (cm)



2. Wine Recognition Dataset

## INPUT

### Pemilihan Dataset

```
main.ipynb U x 2
src > main.ipynb > M Convex Hull Visualizer Enter the dataset number : (Press 'Enter' to confirm or 'Escape' to cancel)
+ Code + Markdown | ▶ Run All | Clear Outputs of All Cells | Go To Running Cell | Restart | Interrupt | Variables | Outline

2. Dataset Pick

Choose any classification dataset from Toy Datasets properties as listed below

1. Iris Plants Dataset
2. Wine recognition dataset
3. Breast cancer wisconsin (diagnostic) dataset

Enter the number of wanted dataset within input

data_sets = [
    (datasets.load_iris(), "Iris Plants Dataset"),
    (datasets.load_wine(), "Wine Recognition Dataset"),
    (datasets.load_breast_cancer(), "Breast Cancer Wisconsin (Diagnostic) Dataset"),
]

data_sets_choice = int(input("Enter the dataset number : "))

data = data_sets[data_sets_choice - 1][0]
```

[84] 3m 17.8s

## Pemilihan Kolom

a. magnesium – color\_intensity

```
main.ipynb U • 5
M Final Scatter Plot with Convex Hull > color_ Pick column 1 to plot : (Press 'Enter' to confirm)
+ Code + Markdown | Run All Clear Outputs of All Cells Go To Running Cell

... Wine Recognition Dataset dataset have 14 columns which enumerated as follows:
1. alcohol
2. malic_acid
3. ash
4. alcalinity_of_ash
5. magnesium
6. total_phenols
7. flavanoids
8. nonflavanoid_phenols
9. proanthocyanins
10. color_intensity
11. hue
12. od280/od315_of_diluted_wines
13. proline
14. Target
```

```
main.ipynb U X 10
M Final Scatter Plot with Convex Hull > color_ Pick column 2 to plot : (Press 'Enter' to confirm)
+ Code + Markdown | Run All Clear Outputs of All Cells Go To Running Cell

... Wine Recognition Dataset dataset have 14 columns which enumerated as follows:
1. alcohol
2. malic_acid
3. ash
4. alcalinity_of_ash
5. magnesium
6. total_phenols
7. flavanoids
8. nonflavanoid_phenols
9. proanthocyanins
10. color_intensity
11. hue
12. od280/od315_of_diluted_wines
13. proline
14. Target
```

b. malic\_acid – hue

```
main.ipynb U • 2
c > main.ipynb > M Convex Hull Visualizer Pick column 1 to plot : (Press 'Enter' to confirm)
+ Code + Markdown | Run All Clear Outputs of All Cells Go To Running Cell

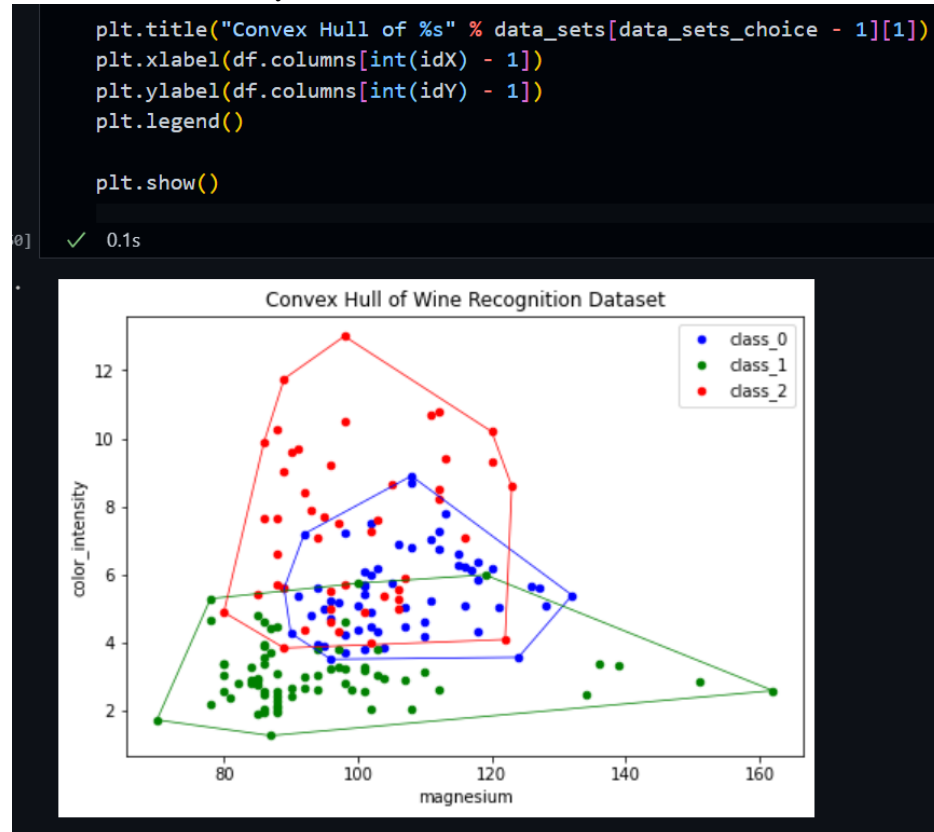
... Wine Recognition Dataset dataset have 14 columns which enumerated as follows:
1. alcohol
2. malic_acid
3. ash
4. alcalinity_of_ash
5. magnesium
6. total_phenols
7. flavanoids
8. nonflavanoid_phenols
9. proanthocyanins
10. color_intensity
11. hue
12. od280/od315_of_diluted_wines
13. proline
14. Target
```

```
main.ipynb U • 11
src > main.ipynb > M Convex Hull Visualizer Pick column 2 to plot : (Press 'Enter' to confirm)
+ Code + Markdown | Run All Clear Outputs of All Cells Go To Running Cell

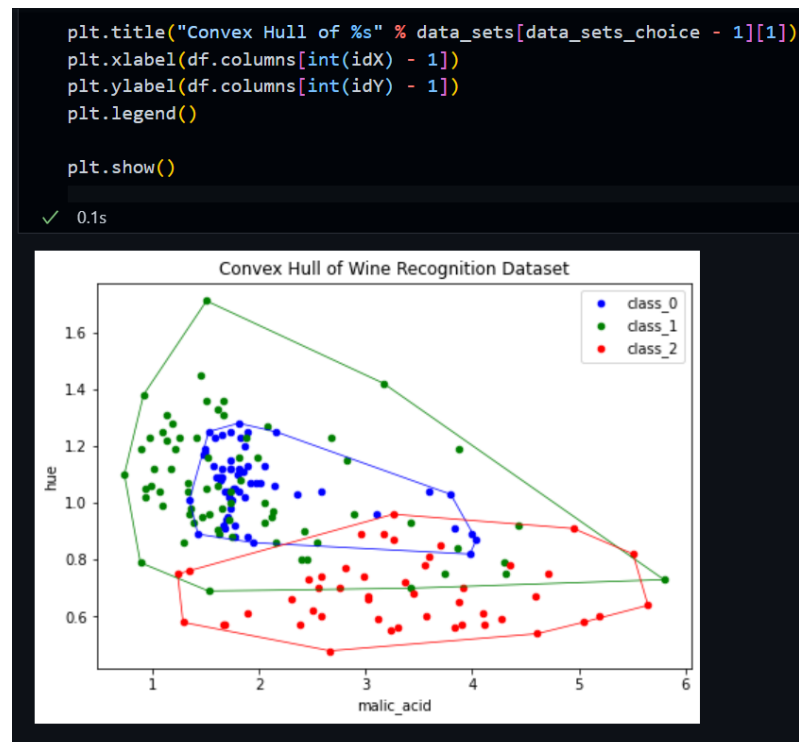
... Wine Recognition Dataset dataset have 14 columns which enumerated as follows:
1. alcohol
2. malic_acid
3. ash
4. alcalinity_of_ash
5. magnesium
6. total_phenols
7. flavanoids
8. nonflavanoid_phenols
9. proanthocyanins
10. color_intensity
11. hue
12. od280/od315_of_diluted_wines
13. proline
14. Target
```

## OUTPUT

a. magnesium – color\_intensity



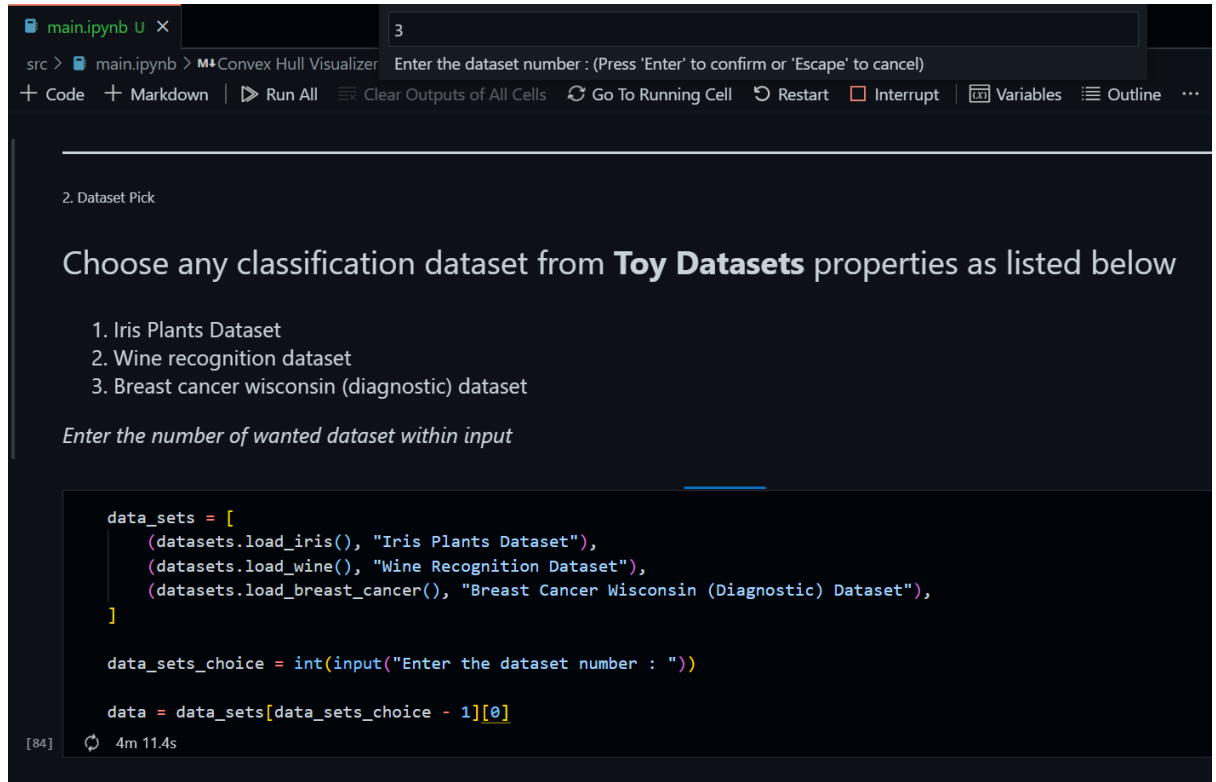
b. malic\_acid – hue



### 3. Breast Cancer Wisconsin (diagnostic) Dataset

## INPUT

### Pemilihan Dataset



```
src > main.ipynb > MConvex Hull Visualizer Enter the dataset number : (Press 'Enter' to confirm or 'Escape' to cancel)
+ Code + Markdown | Run All | Clear Outputs of All Cells | Go To Running Cell | Restart | Interrupt | Variables | Outline ...

2. Dataset Pick

Choose any classification dataset from Toy Datasets properties as listed below

1. Iris Plants Dataset
2. Wine recognition dataset
3. Breast cancer wisconsin (diagnostic) dataset

Enter the number of wanted dataset within input

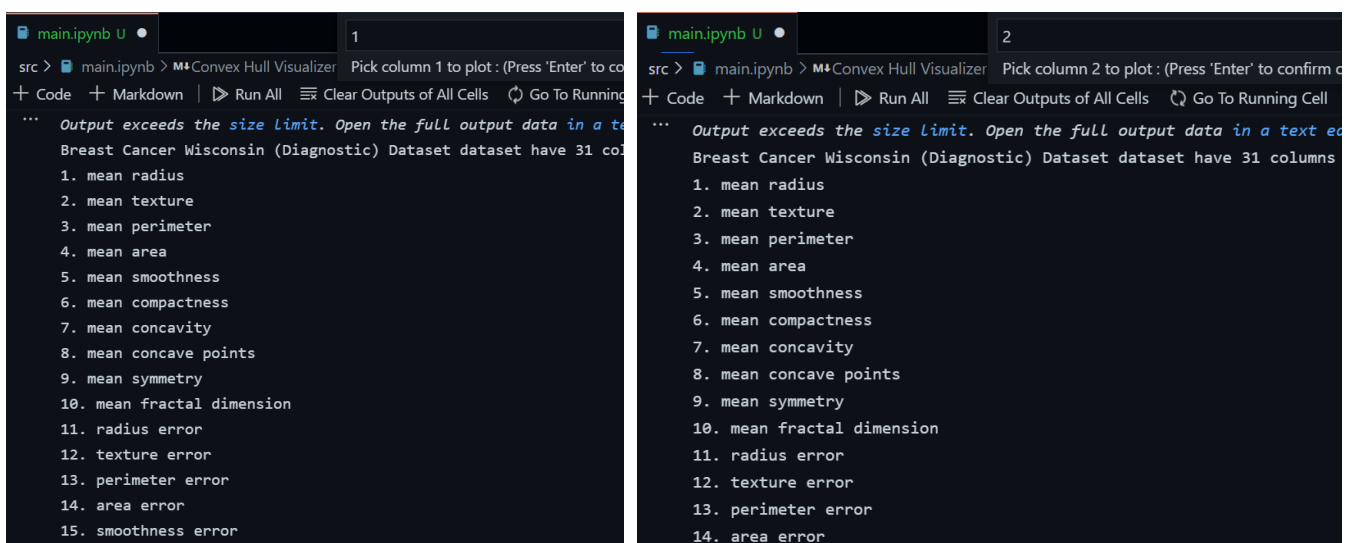
data_sets = [
    (datasets.load_iris(), "Iris Plants Dataset"),
    (datasets.load_wine(), "Wine Recognition Dataset"),
    (datasets.load_breast_cancer(), "Breast Cancer Wisconsin (Diagnostic) Dataset"),
]

data_sets_choice = int(input("Enter the dataset number : "))

data = data_sets[data_sets_choice - 1][0]
```

### Pemilihan Kolom

#### a. mean radius – mean texture



```
src > main.ipynb > MConvex Hull Visualizer Pick column 1 to plot : (Press 'Enter' to confirm or 'Escape' to cancel)
+ Code + Markdown | Run All | Clear Outputs of All Cells | Go To Running Cell

... Output exceeds the size limit. Open the full output data in a text editor ...
Breast Cancer Wisconsin (Diagnostic) Dataset dataset have 31 columns
1. mean radius
2. mean texture
3. mean perimeter
4. mean area
5. mean smoothness
6. mean compactness
7. mean concavity
8. mean concave points
9. mean symmetry
10. mean fractal dimension
11. radius error
12. texture error
13. perimeter error
14. area error
15. smoothness error

src > main.ipynb > MConvex Hull Visualizer Pick column 2 to plot : (Press 'Enter' to confirm or 'Escape' to cancel)
+ Code + Markdown | Run All | Clear Outputs of All Cells | Go To Running Cell

... Output exceeds the size limit. Open the full output data in a text editor ...
Breast Cancer Wisconsin (Diagnostic) Dataset dataset have 31 columns
1. mean radius
2. mean texture
3. mean perimeter
4. mean area
5. mean smoothness
6. mean compactness
7. mean concavity
8. mean concave points
9. mean symmetry
10. mean fractal dimension
11. radius error
12. texture error
13. perimeter error
14. area error
```

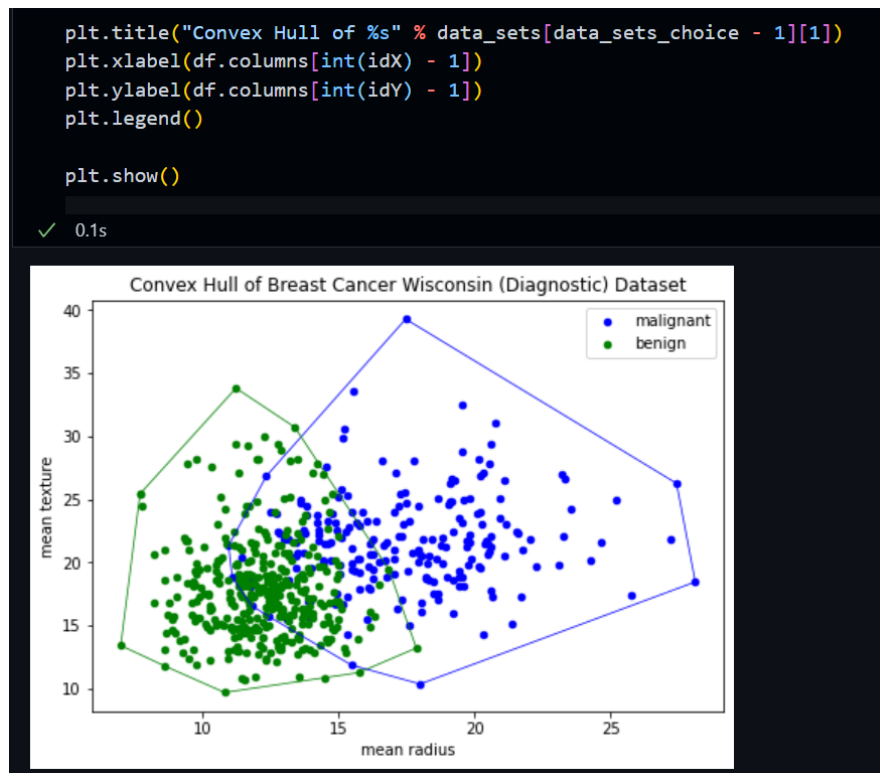
b. mean perimeter – mean symmetry

```
main.ipynb U 3
src > main.ipynb > M+Convex Hull Visualizer Pick column 1 to plot : (Press 'Enter' to confirm)
+ Code + Markdown | Run All Clear Outputs of All Cells Go To Running Cell
Breast Cancer Wisconsin (Diagnostic) Dataset dataset have 31 columns
1. mean radius
2. mean texture
3. mean perimeter
4. mean area
5. mean smoothness
6. mean compactness
7. mean concavity
8. mean concave points
9. mean symmetry
10. mean fractal dimension
11. radius error
12. texture error
13. perimeter error
14. area error
15. smoothness error
16. compactness error
17. concavity error
18. concave points error
```

```
main.ipynb U 9
src > main.ipynb > M+Convex Hull Visualizer Pick column 2 to plot : (Press 'Enter' to confirm)
+ Code + Markdown | Run All Clear Outputs of All Cells Go To Running Cell
Breast Cancer Wisconsin (Diagnostic) Dataset dataset have 31 columns
1. mean radius
2. mean texture
3. mean perimeter
4. mean area
5. mean smoothness
6. mean compactness
7. mean concavity
8. mean concave points
9. mean symmetry
10. mean fractal dimension
11. radius error
12. texture error
```

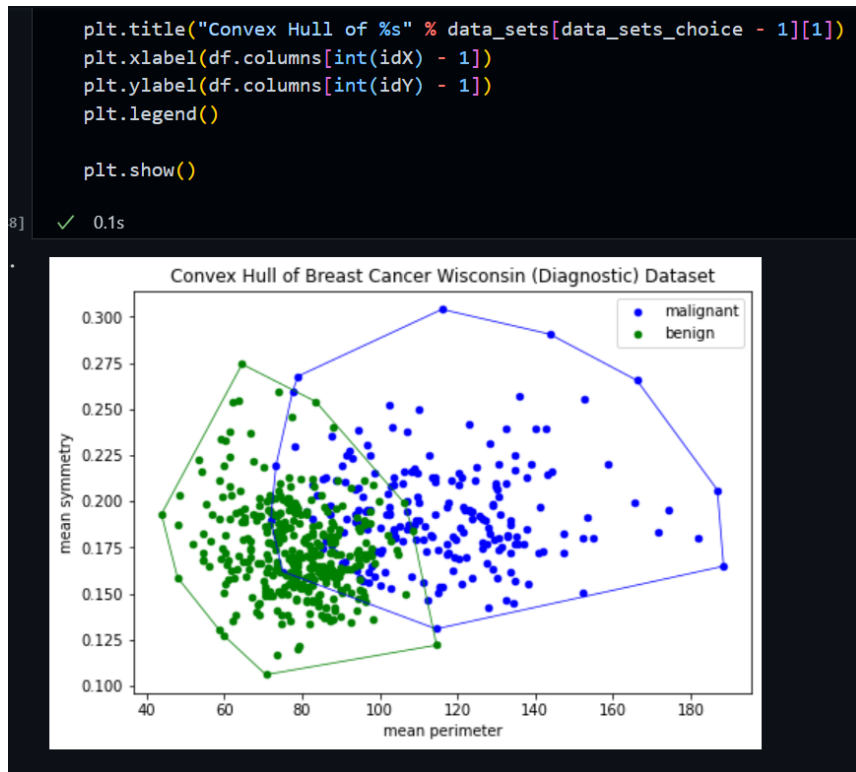
## OUTPUT

a. mean radius – mean texture





- b. mean perimeter – mean symmetry



## LAMPIRAN

1. Alamat Drive code program (berserta testcase dan docs)

<https://drive.google.com/drive/folders/1QEOZR3IQ9c9eDEq15kg1Ut-QJlf3Kpe?usp=sharing>

2. Cek List

| Poin  | Ya | Tidak |
|---|----|-------|
| 1. Pustaka <i>myConvexHull</i> berhasil dibuat dan tidak ada kesalahan  | v  |       |
| 2. <i>Convex hull</i> yang dihasilkan sudah benar   | v  |       |
| 3. Pustaka <i>myConvexHull</i> dapat digunakan untuk menampilkan <i>convex hull</i> setiap label dengan warna yang berbeda. | v  |       |
| 4. <b>Bonus:</b> program dapat menerima input dan menuliskan output untuk dataset lainnya.                                  | v  |       |