Praktikum 5
Hierarchical Clustering

A. Hierarchical Clustering

Hierarchical methods adalah teknik clustering membentuk hirarki atau berdasarkan tingkatan tertentu sehingga menyerupai struktur pohon. Dengan demikian proses pengelompokannya dilakukan secara bertingkat atau bertahap. Biasanya, metode ini digunakan pada data yang jumlahnya tidak terlalu banyak dan jumlah cluster yang akan dibentuk belum diketahui. Di dalam metode hirarki, terdapat dua jenis strategi pengelompokan yaitu agglomerative dan divisive.

Agglomerative (metode penggabungan) adalah strategi pengelompokan hirarki yang dimulai dengan setiap objek dalam satu cluster yang terpisah kemudian membentuk cluster yang semakin membesar. Jadi, banyaknya cluster awal adalah sama dengan banyaknya objek. Sedangkan Divisive (metode pembagian) adalah strategi pengelompokan hirarki yang dimulai dari semua objek dikelompokkan menjadi cluster tunggal kemudian dipisah sampai setiap objek berada dalam cluster yang terpisah. Pada praktikum ini kita hanya akan focus pada metode agglomerative.

B. Metode Agglomerative

Terdapat tiga teknik pengelompokan yang paling dikenal dalam Agglomerative Method, yaitu:

a. Single linkage (jarak terdekat atau tautan tunggal)
   Teknik yang menggabungkan cluster-cluster menurut jarak antara anggota-anggota terdekat di antara dua cluster.
b. Average linkage (jarak rata-rata atau tautan rata-rata)
   Teknik yang menggabungkan cluster-cluster menurut jarak rata-rata pasangan anggota masing-masing pada himpunan antara dua cluster.
c. Complete linkage (jarak terjauh atau tautan lengkap)
   Teknik yang menggabungkan cluster-cluster menurut jarak antara anggota-anggota terjauh di antara dua cluster.

C. Algoritma Metode Agglomerative

a. Hitung matriks jarak

Ada berbagai macam jenis jarak, namun jarak yang sering digunakan adalah Euclidean.

$$d_{i,j} = \sqrt{\sum_{k=1}^{p} (x_{ik} - x_{jk})^2}$$

b. Gabungkan dua cluster terdekat

Jika jarak objek a dengan b memiliki nilai jarak paling kecil dibandingkan jarak antar objek lainnya dalam matriks jarak Euclidean, maka gabungan dua cluster pada tahap pertama adalah d_ab.

c. Perbarui matriks jarak sesuai dengan teknik pengelompokan agglomerative method

Jika d_ab adalah jarak terdekat dari matriks jarak Euclidean, maka rumus untuk metode agglomerative adalah:

$d_{(ab)c} = \min(d_{a,c}; d_{b,c})$	$d_{(ab)c} = \text{avg}(d_{a,c}; d_{b,c})$	$d_{(ab)c} = \max(d_{a,c}; d_{b,c})$
Single Lingkage	Average Linkage	Complete Linkage

d. Ulangi langkah (b) dan (c) sampai hanya tersisa satu cluster

e. Buat dendrogram

Contoh:

Pada praktikum ini akan digunakan scipy dan scikit-learn package dalam Bahasa pemrograman python. Kita juga akan menggunakan Complete Lingkage sebagai Teknik Agglomerative yang diterapkan pada clustering kali ini.

Library yang digunakan dalam clustering

```
import numpy as np
import pandas as pd
from scipy import ndimage
from scipy.cluster import hierarchy
from scipy.spatial import distance_matrix
from matplotlib import pyplot as plt
from sklearn import manifold, datasets
from sklearn.cluster import AgglomerativeClustering
from sklearn.datasets.samples_generator import make_blobs
%matplotlib inline
```

Melakukan Generate Random Data

```
X1, y1 = make_blobs(n_samples=50, centers=[[4,4], [-2, -1], [1, 1], [10,4]], cluster_std=0.9)
```

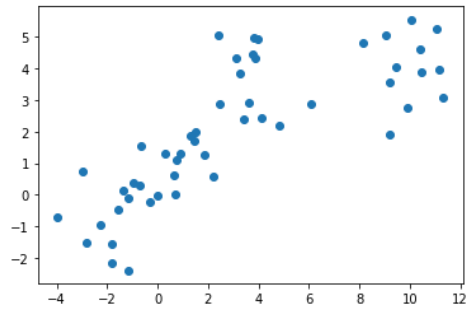
Kita akan menghasilkan satu set data menggunakan kelas make_blobs. Masukkan parameter ini ke make_blobs:

a. n_samples: Jumlah total poin yang dibagi rata di antara cluster. Pilih nomor dari 10-1500.
b. centers: Jumlah pusat yang akan dihasilkan, atau lokasi pusat tetap. Pilih array koordinat x, y untuk menghasilkan pusat. Memiliki 1-10 pusat (mis. Pusat = [[1,1], [2,5]])
c. cluster_std: Deviasi standar dari cluster. Semakin besar angkanya, semakin jauh klusternya. Pilih angka antara 0,5-1,5

Plotting Random Data

```
plt.scatter(X1[:, 0], X1[:, 1], marker='o')
```

```
<matplotlib.collections.PathCollection at 0x229bb65b4a8>
```



Agglomerative Clustering

Kelas Agglomerative Clustering akan membutuhkan dua masukan:

a. n_clusters: Jumlah cluster yang akan dibentuk serta jumlah sentroid yang akan dibuat. Nilainya akan menjadi: 4
b. linkage: Kriteria linkage mana yang akan digunakan. Algoritma akan menggabungkan pasangan cluster yang meminimalkan kriteria ini. Nilainya akan menjadi: 'complete'
   Catatan: Anda disarankan untuk mencoba semuanya dengan 'rata-rata' juga.

```
agglom.fit(X1,y1)
```

```
agglom = AgglomerativeClustering(n_clusters = 4, linkage = 'single')
```

```
# Create a figure of size 6 inches by 4 inches.
plt.figure(figsize=(6,4))

# These two lines of code are used to scale the data points down,
# Or else the data points will be scattered very far apart.

# Create a minimum and maximum range of X1.
x_min, x_max = np.min(X1, axis=0), np.max(X1, axis=0)

# Get the average distance for X1.
X1 = (X1 - x_min) / (x_max - x_min)

# This loop displays all of the datapoints.
for i in range(X1.shape[0]):
    # Replace the data points with their respective cluster value
    # (ex. 0) and is color coded with a colormap (plt.cm.spectral)
    plt.text(X1[i, 0], X1[i, 1], str(y1[i]),
             color=plt.cm.nipy_spectral(agglom.labels_[i] / 10.),
             fontdict={'weight': 'bold', 'size': 9})

# Remove the x ticks, y ticks, x and y axis
plt.xticks([])
plt.yticks([])
#plt.axis('off')



# Display the plot of the original data before clustering
plt.scatter(X1[:, 0], X1[:, 1], marker='.')
# Display the plot
plt.show()
```

Plotting Dendrogram

Ingatlah bahwa matriks jarak berisi jarak dari setiap titik ke titik lain dari sebuah set data. Gunakan fungsi distance_matrix, yang membutuhkan dua input. Gunakan Matriks Fitur, X2 sebagai kedua input dan simpan matriks jarak ke variabel yang disebut dist_matrix. Ingatlah bahwa nilai jarak adalah simetris, dengan diagonal 0. Ini adalah salah satu cara untuk memastikan matriks Anda benar. (cetak dist_matrix untuk memastikannya benar).

```
dist_matrix = distance_matrix(X1,X1)
print(dist_matrix)
```

```
[[0.         0.80425436 0.48340885 ... 0.54772885 0.08696131 0.35288139]
 [0.80425436 0.         1.24063766 ... 0.4645181  0.87502949 1.15140767]
 [0.48340885 1.24063766 0.         ... 0.86536109 0.39688093 0.19242836]
 ...
 [0.54772885 0.4645181  0.86536109 ... 0.         0.58159445 0.83158061]
 [0.08696131 0.87502949 0.39688093 ... 0.58159445 0.         0.27643481]
 [0.35288139 1.15140767 0.19242836 ... 0.83158061 0.27643481 0.        ]]
```

Menggunakan kelas linkage dari hierarki, berikan parameter:

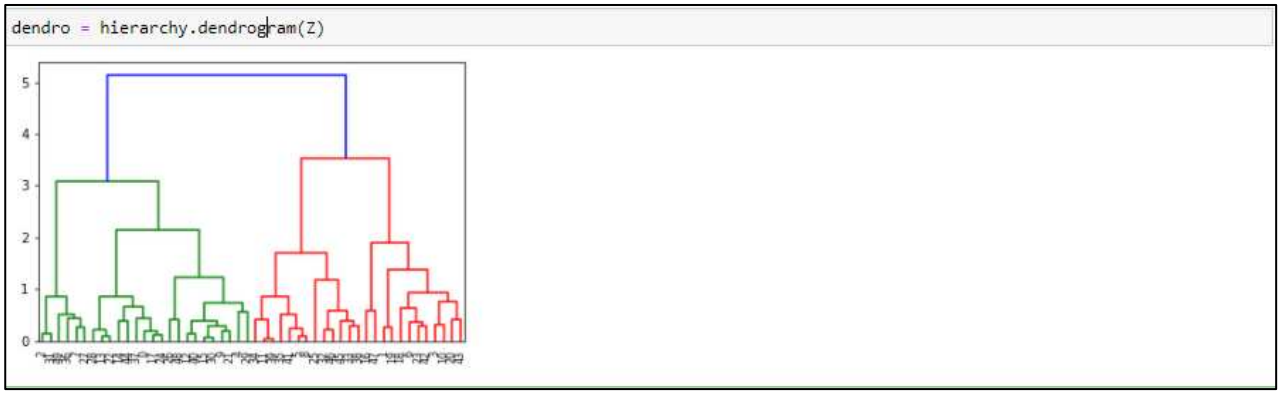
a. Matriks jarak
b. 'complete' untuk complete linkage
c. Simpan hasilnya ke variabel bernama Z

```
Z = hierarchy.linkage(dist_matrix, 'complete')
X = hierarchy.linkage(dist_matrix, 'single')
Y = hierarchy.linkage(dist_matrix, 'average')
```

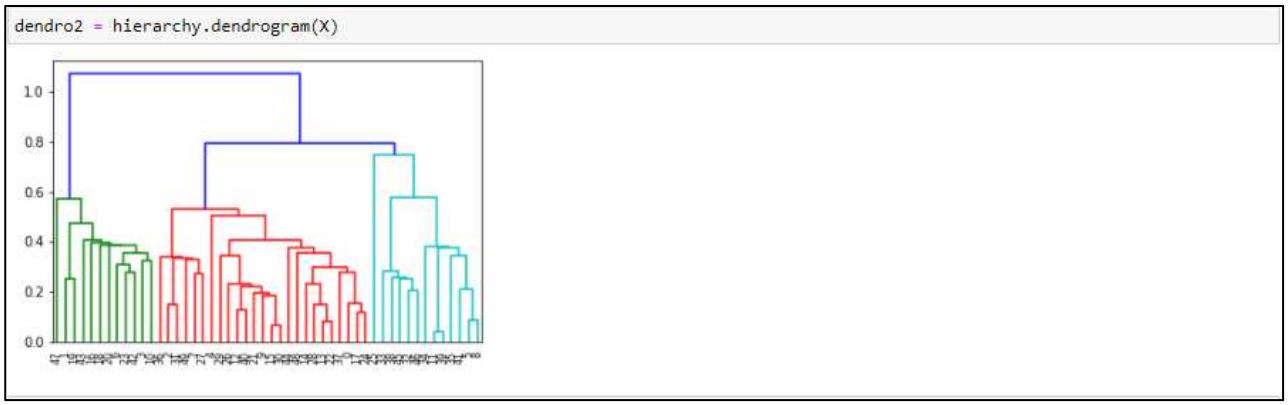
```
D:\anaconda3\lib\site-packages\ipykernel_launcher.py:1: ClusterWarning: scipy.cluster: The symmetric non-negative hollow observ
ation matrix looks suspiciously like an uncondensed distance matrix
  """Entry point for launching an IPython kernel.
D:\anaconda3\lib\site-packages\ipykernel_launcher.py:2: ClusterWarning: scipy.cluster: The symmetric non-negative hollow observ
ation matrix looks suspiciously like an uncondensed distance matrix

D:\anaconda3\lib\site-packages\ipykernel_launcher.py:3: ClusterWarning: scipy.cluster: The symmetric non-negative hollow observ
ation matrix looks suspiciously like an uncondensed distance matrix
  This is separate from the ipykernel package so we can avoid doing imports until
```

Hierachical Clustering biasanya divisualisasikan sebagai dendrogram seperti yang diperlihatkan di cell berikut ini. Setiap gabungan diwakili oleh garis horizontal. Koordinat y dari garis horizontal adalah kesamaan dari dua cluster yang digabungkan, di mana kota-kota dipandang sebagai cluster tunggal. Dengan berpindah dari lapisan bawah ke node atas, dendrogram memungkinkan kita merekonstruksi riwayat penggabungan yang menghasilkan pengelompokan yang digambarkan. Selanjutnya, kita akan menyimpan dendrogram ke variabel yang disebut dendro. Dalam melakukan ini, dendrogram juga akan ditampilkan. Menggunakan kelas dendrogram dari hierarki, berikan parameter X, Y, Z.

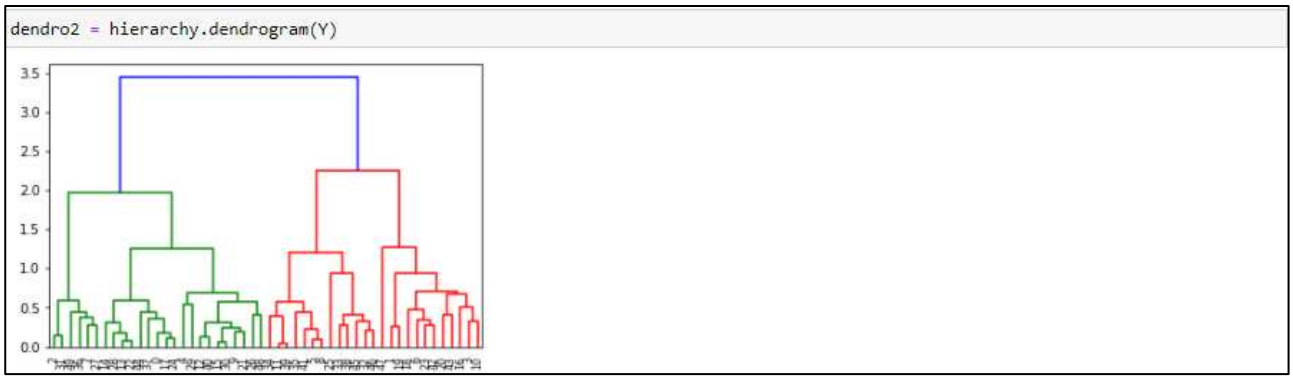
Dendogram untuk Parameter Z



Dendogram untuk Parameter X



Dendogram untuk Parameter Y



Clustering on Vehicle Dataset

Bayangkan sebuah pabrikan mobil telah mengembangkan prototipe untuk kendaraan baru. Sebelum memperkenalkan model baru ke dalam jajarannya, pabrikan ingin menentukan kendaraan yang ada di pasaran yang paling mirip dengan prototipe - yaitu, bagaimana kendaraan dapat dikelompokkan, kelompok mana yang paling mirip dengan model tersebut, dan oleh karena itu model mana mereka akan bersaing.

Tujuan kita di sini, adalah menggunakan metode pengelompokan, untuk menemukan kluster kendaraan yang paling khas. Ini akan meringkas kendaraan yang ada dan membantu manufaktur untuk membuat keputusan tentang model baru secara sederhana.

Download Data

Untuk mendownload data, gunakan !wget.

Read Data

Fitur-fiturnya termasuk harga dalam ribuan (price), ukuran mesin (engine_s), horsepower (horsepow), jarak sumbu roda (wheelbas), lebar (width), panjang (height), berat trotoar (curb_wgt), kapasitas bahan bakar (fuel_cap) dan efisiensi bahan bakar (mpg).

```
filename = 'E:\cars_clus.csv'

#Read csv
pdf = pd.read_csv(filename)
print ("Shape of dataset: ", pdf.shape)

pdf.head(5)
```

```
Shape of dataset:  (159, 16)
```

	manufact	model	sales	resale	type	price	engine_s	horsepow	wheelbas	width	length	curb_wgt	fuel_cap	mpg	lnsales	partition
0	Acura	Integra	16.919	16.36	0	21.5	1.8	140	101.2	67.3	172.4	2.639	13.2	28	2.828	0
1	Acura	TL	39.384	19.875	0	28.4	3.2	225	108.1	70.3	192.9	3.517	17.2	25	3.673	0
2	Acura	CL	14.114	18.225	0	*null*	3.2	225	106.9	70.6	192	3.47	17.2	26	2.647	0
3	Acura	RL	8.588	29.725	0	42	3.5	210	114.6	71.4	196.6	3.85	18	22	2.15	0
4	Audi	A4	20.397	22.255	0	23.99	1.8	150	102.6	68.2	178	2.998	16.4	27	3.015	0

Data Cleaning

mari kita cukup menghapus dataset dengan membuang baris yang memiliki nilai null.

```
print ("Shape of dataset before cleaning: ", pdf.size)
pdf[[ 'sales', 'resale', 'type', 'price', 'engine_s',
       'horsepow', 'wheelbas', 'width', 'length', 'curb_wgt', 'fuel_cap',
       'mpg', 'lnsales']] = pdf[['sales', 'resale', 'type', 'price', 'engine_s',
       'horsepow', 'wheelbas', 'width', 'length', 'curb_wgt', 'fuel_cap',
       'mpg', 'lnsales']].apply(pd.to_numeric, errors='coerce')
pdf = pdf.dropna()
pdf = pdf.reset_index(drop=True)
print ("Shape of dataset after cleaning: ", pdf.size)
pdf.head(5)
```

```
Shape of dataset before cleaning:  2544
Shape of dataset after cleaning:  1872
```

	manufact	model	sales	resale	type	price	engine_s	horsepow	wheelbas	width	length	curb_wgt	fuel_cap	mpg	lnsales	partition
0	Acura	Integra	16.919	16.360	0.0	21.50	1.8	140.0	101.2	67.3	172.4	2.639	13.2	28.0	2.828	0
1	Acura	TL	39.384	19.875	0.0	28.40	3.2	225.0	108.1	70.3	192.9	3.517	17.2	25.0	3.673	0
2	Acura	RL	8.588	29.725	0.0	42.00	3.5	210.0	114.6	71.4	196.6	3.850	18.0	22.0	2.150	0
3	Audi	A4	20.397	22.255	0.0	23.99	1.8	150.0	102.6	68.2	178.0	2.998	16.4	27.0	3.015	0
4	Audi	A6	18.780	23.555	0.0	33.95	2.8	200.0	108.7	76.1	192.0	3.561	18.5	22.0	2.933	0

Feature Selection

```
featureset = pdf[['engine_s',  'horsepow', 'wheelbas', 'width', 'length', 'curb_wgt', 'fuel_cap', 'mpg']]
```

Normalization

Sekarang kita dapat menormalkan set fitur. MinMaxScaler mengubah fitur dengan menskalakan setiap fitur ke kisaran tertentu. Ini secara default (0, 1). Artinya, penaksir ini menskalakan dan menerjemahkan setiap fitur secara individual sehingga berada di antara nol dan satu.

```
from sklearn.preprocessing import MinMaxScaler
x = featureset.values #returns a numpy array
min_max_scaler = MinMaxScaler()
feature_mtx = min_max_scaler.fit_transform(x)
feature_mtx [0:5]
```

```
array([[0.11428571, 0.21518987, 0.18655098, 0.28143713, 0.30625832,
        0.2310559 , 0.13364055, 0.43333333],
       [0.31428571, 0.43037975, 0.3362256 , 0.46107784, 0.5792277 ,
        0.50372671, 0.31797235, 0.33333333],
       [0.35714286, 0.39240506, 0.47722343, 0.52694611, 0.62849534,
        0.60714286, 0.35483871, 0.23333333],
       [0.11428571, 0.24050633, 0.21691974, 0.33532934, 0.38082557,
        0.34254658, 0.28110599, 0.4       ],
       [0.25714286, 0.36708861, 0.34924078, 0.80838323, 0.56724368,
        0.5173913 , 0.37788018, 0.23333333]])
```

Clustering With Scipy

Dalam agglomerative clustering, pada setiap iterasi, algoritme harus memperbarui matriks jarak untuk mencerminkan jarak cluster yang baru terbentuk dengan cluster yang tersisa di hutan. Metode berikut ini didukung di Scipy untuk menghitung jarak antara cluster yang baru terbentuk dan masing-masing: - single - complete - average - weighted – centroid

```
import scipy
leng = feature_mtx.shape[0]
D = scipy.zeros([leng,leng])
for i in range(leng):
    for j in range(leng):
        D[i,j] = scipy.spatial.distance.euclidean(feature_mtx[i], feature_mtx[j])
```

Complete Linkage

```
import pylab
import scipy.cluster.hierarchy
Z = hierarchy.linkage(D, 'complete')
```

Pada dasarnya, pengelompokan hierarki tidak memerlukan jumlah kluster yang telah ditentukan sebelumnya. Namun, dalam beberapa aplikasi kami menginginkan partisi dari cluster terputus-putus seperti di clustering datar. Jadi Anda bisa menggunakan cutline:

```
from scipy.cluster.hierarchy import fcluster
max_d = 3
clusters = fcluster(Z, max_d, criterion='distance')
clusters
```

```
array([ 1,  5,  5,  6,  5,  4,  6,  5,  5,  5,  5,  5,  4,  4,  5,  1,  6,
        5,  5,  5,  4,  2, 11,  6,  6,  5,  6,  5,  1,  6,  6, 10,  9,  8,
        9,  3,  5,  1,  7,  6,  5,  3,  5,  3,  8,  7,  9,  2,  6,  6,  5,
        4,  2,  1,  6,  5,  2,  7,  5,  5,  5,  4,  4,  3,  2,  6,  6,  5,
        7,  4,  7,  6,  6,  5,  3,  5,  5,  6,  5,  4,  4,  1,  6,  5,  5,
        5,  6,  4,  5,  4,  1,  6,  5,  6,  6,  5,  5,  5,  7,  7,  7,  2,
        2,  1,  2,  6,  5,  1,  1,  1,  7,  8,  1,  1,  6,  1,  1],
      dtype=int32)
```

Selain itu, Anda dapat menentukan jumlah cluster secara langsung:

```
from scipy.cluster.hierarchy import fcluster
k = 5
clusters = fcluster(Z, k, criterion='maxclust')
clusters
```

```
array([1, 3, 3, 3, 3, 2, 3, 3, 3, 3, 3, 3, 2, 2, 3, 1, 3, 3, 3, 3, 2, 1,
       5, 3, 3, 3, 3, 3, 1, 3, 3, 4, 4, 4, 4, 2, 3, 1, 3, 3, 3, 2, 3, 2,
       4, 3, 4, 1, 3, 3, 3, 2, 1, 1, 3, 3, 1, 3, 3, 3, 3, 2, 2, 2, 1, 3,
       3, 3, 3, 2, 3, 3, 3, 3, 2, 3, 3, 3, 3, 2, 2, 1, 3, 3, 3, 3, 3, 2,
       3, 2, 1, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 1, 1, 1, 1, 3, 3, 1, 1, 1,
       3, 4, 1, 1, 3, 1, 1], dtype=int32)
```

Plotting Dendogram

```
fig = pylab.figure(figsize=(18,50))
def llf(id):
    return '[%s %s %s]' % (pdf['manufact'][id], pdf['model'][id], int(float(pdf['type'][id])) )

dendro = hierarchy.dendrogram(Z,  leaf_label_func=llf, leaf_rotation=0, leaf_font_size =12, orientation = 'right')
```

Clustering With Scikit-Learn

```
dist_matrix = distance_matrix(feature_mtx,feature_mtx)
print(dist_matrix)
```

```
[[0.         0.57777143 0.75455727 ... 0.28530295 0.24917241 0.18879995]
 [0.57777143 0.         0.22798938 ... 0.36087756 0.66346677 0.62201282]
 [0.75455727 0.22798938 0.         ... 0.51727787 0.81786095 0.77930119]
 ...
 [0.28530295 0.36087756 0.51727787 ... 0.         0.41797928 0.35720492]
 [0.24917241 0.66346677 0.81786095 ... 0.41797928 0.         0.15212198]
 [0.18879995 0.62201282 0.77930119 ... 0.35720492 0.15212198 0.        ]]
```

Sekarang, kita bisa menggunakan fungsi 'AgglomerativeClustering' dari scikit-learn library untuk mengelompokkan dataset. AgglomerativeClustering melakukan pengelompokan hierarkis menggunakan pendekatan bottom-up. Kriteria keterkaitan menentukan metrik yang digunakan untuk strategi penggabungan:

A. Ward meminimalkan jumlah perbedaan kuadrat dalam semua cluster. Ini adalah pendekatan meminimalkan varians dan dalam pengertian ini mirip dengan fungsi tujuan k-means tetapi ditangani dengan pendekatan hierarki aglomeratif.

B. Maximum atau Complete Linkage meminimalkan jarak maksimum antara pengamatan pasangan cluster.

C. Average Linkage meminimalkan rata-rata jarak antara semua pengamatan pasangan cluster.

```
agglom = AgglomerativeClustering(n_clusters = 6, linkage = 'complete')
agglom.fit(feature_mtx)
agglom.labels_
```

```
array([1, 2, 2, 1, 2, 3, 1, 2, 2, 2, 2, 2, 3, 3, 2, 1, 1, 2, 2, 2, 5, 1,
       4, 1, 1, 2, 1, 2, 1, 1, 1, 5, 0, 0, 0, 3, 2, 1, 2, 1, 2, 3, 2, 3,
       0, 3, 0, 1, 1, 1, 2, 3, 1, 1, 1, 2, 1, 1, 2, 2, 2, 3, 3, 3, 1, 1,
       1, 2, 1, 2, 2, 1, 1, 2, 3, 2, 3, 1, 2, 3, 5, 1, 1, 2, 3, 2, 1, 3,
       2, 3, 1, 1, 2, 1, 1, 2, 2, 2, 1, 1, 1, 1, 1, 1, 1, 2, 1, 1, 1,
       2, 0, 1, 1, 1, 1, 1], dtype=int64)
```

Dan, kita bisa menambahkan field baru ke dataframe kita untuk menunjukkan cluster dari setiap baris:

```
pdf['cluster_'] = agglom.labels_
pdf.head()
```

	manufact	model	sales	resale	type	price	engine_s	horsepow	wheelbas	width	length	curb_wgt	fuel_cap	mpg	lnsales	partition	cluster_
0	Acura	Integra	16.919	16.360	0.0	21.50	1.8	140.0	101.2	67.3	172.4	2.639	13.2	28.0	2.828	0	1
1	Acura	TL	39.384	19.875	0.0	28.40	3.2	225.0	108.1	70.3	192.9	3.517	17.2	25.0	3.673	0	2
2	Acura	RL	8.588	29.725	0.0	42.00	3.5	210.0	114.6	71.4	196.6	3.850	18.0	22.0	2.150	0	2
3	Audi	A4	20.397	22.255	0.0	23.99	1.8	150.0	102.6	68.2	178.0	2.998	16.4	27.0	3.015	0	1
4	Audi	A6	18.780	23.555	0.0	33.95	2.8	200.0	108.7	76.1	192.0	3.561	18.5	22.0	2.933	0	2

Clustering Dengan Scikit-Learn

```
import matplotlib.cm as cm
n_clusters = max(agglom.labels_)+1
colors = cm.rainbow(np.linspace(0, 1, n_clusters))
cluster_labels = list(range(0, n_clusters))

# Create a figure of size 6 inches by 4 inches.
plt.figure(figsize=(16,14))

for color, label in zip(colors, cluster_labels):
    subset = pdf[pdf.cluster_ == label]
    for i in subset.index:
            plt.text(subset.horsepow[i], subset.mpg[i],str(subset['model'][i]), rotation=25)
    plt.scatter(subset.horsepow, subset.mpg, s= subset.price*10, c=color, label='cluster'+str(label),alpha=0.5)
#    plt.scatter(subset.horsepow, subset.mpg)
plt.legend()
plt.title('Clusters')
plt.xlabel('horsepow')
plt.ylabel('mpg')
```

Seperti yang Anda lihat pada grafik yang dihasilkan, distribusi setiap cluster menggunakan scatter plot, tetapi tidak begitu jelas di mana letak pusat dari setiap cluster. Selain itu, ada 2 jenis kendaraan di dataset kita, "truk" (nilai 1 di kolom jenis) dan "mobil" (nilai 1 di kolom jenis). Jadi, kami menggunakannya untuk membedakan kelas, dan meringkas cluster. Pertama kami menghitung jumlah kasus di setiap kelompok:

```
pdf.groupby(['cluster_','type'])['cluster_'].count()
```

```
cluster_  type
0         1.0      6
1         0.0     47
          1.0      5
2         0.0     27
          1.0     11
3         0.0     10
          1.0      7
4         0.0      1
5         0.0      3
Name: cluster_, dtype: int64
```

Sekarang kita dapat melihat karakteristik masing-masing cluster:

```
agg_cars = pdf.groupby(['cluster_','type'])['horsepow','engine_s','mpg','price'].mean()
agg_cars
```

		horsepow	engine_s	mpg	price
cluster_	type				
0	1.0	211.666667	4.483333	16.166667	29.024667
1	0.0	146.531915	2.246809	27.021277	20.306128
	1.0	145.000000	2.580000	22.200000	17.009200
2	0.0	203.111111	3.303704	24.214815	27.750593
	1.0	182.090909	3.345455	20.181818	26.265364
3	0.0	256.500000	4.410000	21.500000	42.870400
	1.0	160.571429	3.071429	21.428571	21.527714
4	0.0	55.000000	1.000000	45.000000	9.235000
5	0.0	365.666667	6.233333	19.333333	66.010000

Plotting Dendogram

```
plt.figure(figsize=(16,10))
for color, label in zip(colors, cluster_labels):
    subset = agg_cars.loc[(label,),]
    for i in subset.index:
        plt.text(subset.loc[i][0]+5, subset.loc[i][2], 'type='+str(int(i)) + ', price='+str(int(subset.loc[i][3]))+'k')
    plt.scatter(subset.horsepow, subset.mpg, s=subset.price*20, c=color, label='cluster'+str(label))
plt.legend()
plt.title('Clusters')
plt.xlabel('horsepow')
plt.ylabel('mpg')
```

TUGAS:

1. Lakukan agglomerative clustering untuk dataset random yang tersedia di atas dengan single linkage dan average linkage! Jelaskan perbedaannya!
2. Lakukan agglomerative clustering menggunakan scipy dan scikit-learn dengan single linkage dan average linkage untuk dataset cars_clustering! Jelaskan perbedaannya!
3. Lakukan agglomerative clustering menggunakan scipy dan scikit-learn dengan single linkage, average linkage, dan complete linkage untuk dataset iris! Jelaskan perbedaannya!