Lab1 – Unsupervised machine learning

ให้ทำบน jupyter lab และทำรายงานส่งเป็น pdf

1. นำ Iris data set มาทำการเปรียบขนาดความคล้ายครึ่ง โดยใช้ Hierarchy clustering ของ

   i) ความยาวและความกว้างของกลีบเลี้ยง

   ii) ความยาวและความกว้างของกลีบดอก

   และสรุปผลลัพธ์ที่ได้ออกมา เป็นการจำแนกได้ว่าความใกล้เคียงของดอกอยู่ประมาณไหน ให้ใช้ Rand Index, Entropy(Purity) ในการประเมิณ (2%)

```
import pandas as pd
Iris = pd.read_csv('data/Iris.csv')
Iris
```

[2]:

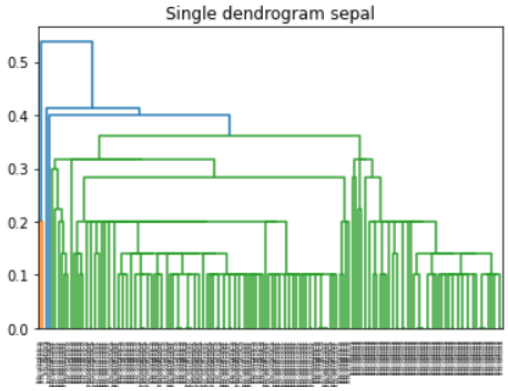
	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	Iris-setosa
1	2	4.9	3.0	1.4	0.2	Iris-setosa
2	3	4.7	3.2	1.3	0.2	Iris-setosa
3	4	4.6	3.1	1.5	0.2	Iris-setosa
4	5	5.0	3.6	1.4	0.2	Iris-setosa
...	...	...	...	...	...	...
145	146	6.7	3.0	5.2	2.3	Iris-virginica
146	147	6.3	2.5	5.0	1.9	Iris-virginica
147	148	6.5	3.0	5.2	2.0	Iris-virginica
148	149	6.2	3.4	5.4	2.3	Iris-virginica
149	150	5.9	3.0	5.1	1.8	Iris-virginica

```
sepal = Iris[["SepalLengthCm","SepalWidthCm"]]
sepal
```

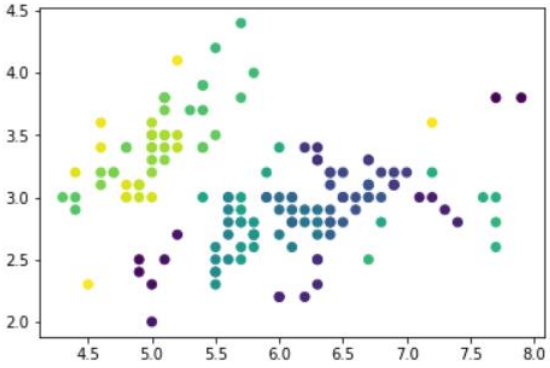
[50]:

	SepalLengthCm	SepalWidthCm
0	5.1	3.5
1	4.9	3.0
2	4.7	3.2
3	4.6	3.1
4	5.0	3.6
...	...	...
145	6.7	3.0
146	6.3	2.5
147	6.5	3.0

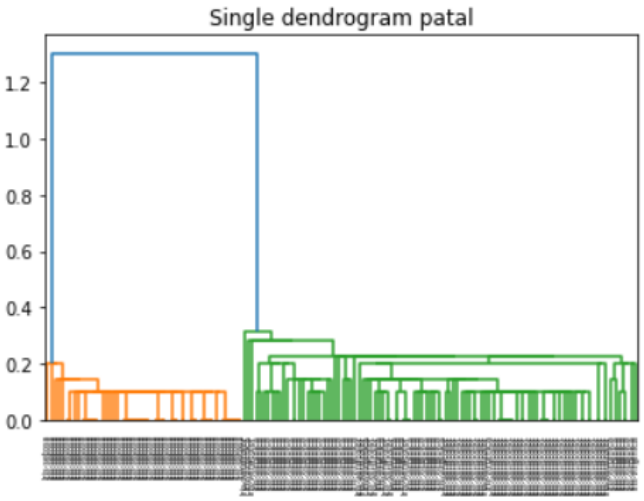
```
from scipy.cluster.hierarchy import dendrogram, linkage, fcluster
import matplotlib.pyplot as plt

linkage_sepal = linkage(sepal, method='single', metric='euclidean')
dendrogram(linkage_sepal, labels=Iris['Species'].to_list())

plt.title('Single dendrogram sepal')
plt.show()
```

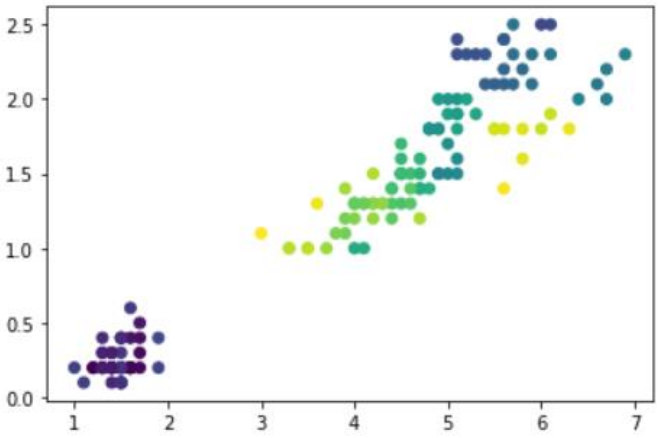



```
sepal_cls = fcluster(linkage_sepal, t=0, criterion='distance')
plt.scatter(sepal["SepalLengthCm"],sepal["SepalWidthCm"],c=sepal_cls)
plt.show()
```



ของ petal ทำคล้ายๆ กัน





การประเมิณผล cluster ของ model

Rand index

```
from sklearn.metrics import cluster
sepal_rand_ind = cluster.rand_score(Iris['Species'],sepal_cls)
sepal_rand_ind
```

ผลลัพธ์

Rand sepal = 0.6722147651006711

Rand petal = 0.6798210290827741

Entropy

```
contingency_matrix = cluster.contingency_matrix(Iris['Species'],sepal_cls)
contingency_matrix
```

```
[65]: array([[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
              0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
              0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
              0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 2, 1, 1, 1, 1, 2,
              1, 2, 3, 1, 2, 1, 1, 1, 1, 2, 1, 2, 2, 1, 1, 3, 1, 1, 2, 1,
              1, 1, 1, 1, 0, 1],
             [0, 0, 1, 0, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1,
              0, 0, 1, 0, 0, 0, 1, 1, 2, 1, 0, 0, 1, 0, 1, 2, 1, 1, 0, 1, 1, 1,
              0, 0, 0, 0, 1, 0, 1, 0, 2, 1, 2, 1, 1, 1, 1, 2, 2, 0, 1, 1, 0, 1,
              1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
              0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
              0, 0, 0, 0, 0, 0],
             [1, 1, 0, 1, 0, 0, 0, 0, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 2, 2, 0,
              1, 1, 1, 1, 1, 3, 0, 0, 1, 1, 1, 2, 0, 1, 0, 0, 1, 1, 1, 0, 0, 0,
              1, 1, 1, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 2, 1, 0, 0, 1, 0,
              0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
              0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
              0, 0, 0, 0, 1, 0]], dtype=int64)
```

```
import numpy as np
mij = contingency_matrix
mi = contingency_matrix.sum(axis=0)
pij = mij/mi

log2pij = np.log2(pij,out=np.zeros_like(pij), where=(pij!=0))
print(pij.round(2))
print(log2pij.round(2))

ei = pij*log2pij
ei = -1*ei.sum(axis=0)
print('e_i \n', ei.round(2))

m = contingency_matrix.sum()

entropy = ((mi/m)*ei).sum()
print('entropy =', entropy )
```

Entropy sepal = 0.1567318333621796

Entropy petal = 0.01836591668108979

Purity

```
purity_sepal = np.sum(np.amax(contingency_matrix, axis=0)) / np.sum(contingency_matrix)
purity_sepal
```

Purity sepal = 0.9266666666666666

Purity sepal = 0.9933333333333333

2. จงใช้ SOM ในการหา BMU ของความใกล้เคียงกันของข้อมูลในไฟล์ Healthcare-dataset-stroke-data.csv ระหว่าง Age กับ Average of glucose level และ Residence_type เป็นตัวจำแนก และใช้ K-mean ในการจำแนกกราฟโดยใช้ neuron ใน SOM ดูความคล้ายคลึง ใช้ Silhouette Coefficient ในการประเมิณ (2%)

```
import pandas as pd
df = pd.read_csv('data/Healthcare-dataset-stroke-data.csv')
df
```

[3]:

	id	gender	age	hypertension	heart_disease	ever_married	work_type	Residence_type	avg_glucose_level
0	9046	Male	67.0	0	1	Yes	Private	Urban	228.69
1	51676	Female	61.0	0	0	Yes	Self-employed	Rural	202.21
2	31112	Male	80.0	0	1	Yes	Private	Rural	105.92
3	60182	Female	49.0	0	0	Yes	Private	Urban	171.23
4	1665	Female	79.0	1	0	Yes	Self-employed	Rural	174.12
...	...	...	...	...	...	...	...	...	...
194	23410	Female	72.0	0	0	Yes	Private	Rural	97.92
195	64373	Male	59.0	0	0	Yes	Private	Urban	200.62
196	58267	Male	70.0	1	0	Yes	Private	Rural	242.52
197	35684	Male	69.0	0	0	Yes	Private	Rural	93.81
198	18937	Male	79.0	0	0	Yes	Private	Rural	114.77

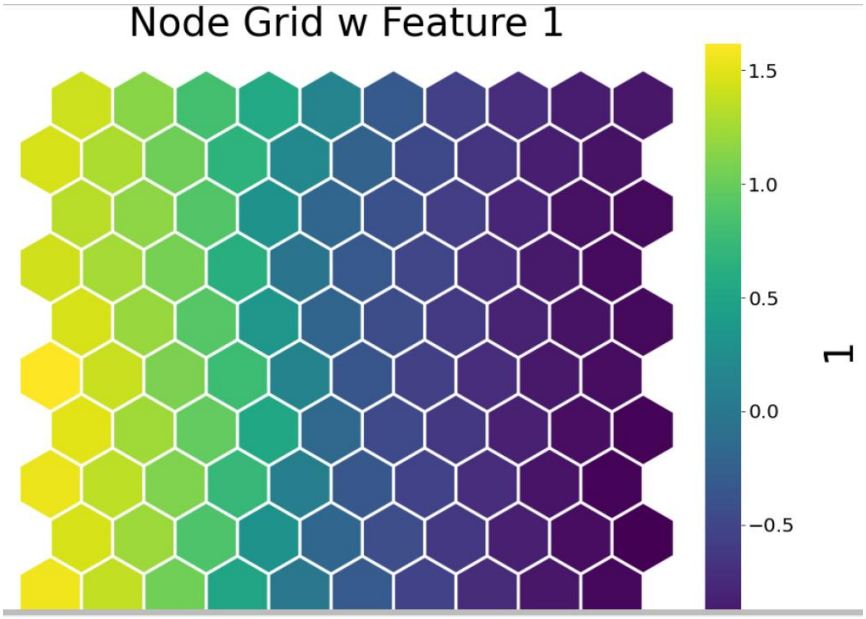
```
from sklearn.preprocessing import StandardScaler

X = df[["age","avg_glucose_level"]]
Y = df[["Residence_type"]]

scaler = StandardScaler()
scaler= scaler.fit(X.values)

X_scale = scaler.transform(X.values)
print(X_scale.shape)
print(X.shape)
```

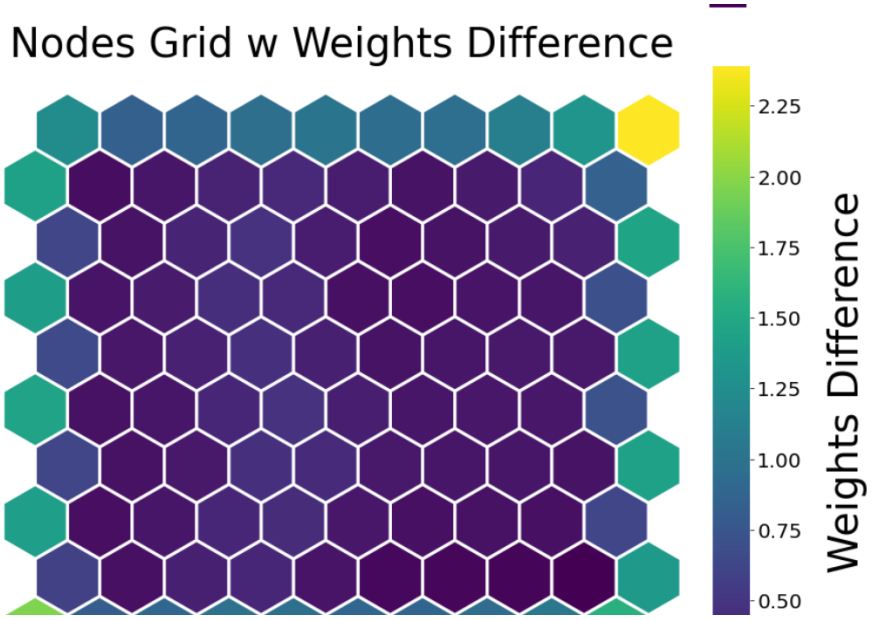
```
(199, 2)
(199, 2)
```

```
import simpsom as sps

# train
net = sps.SOMNet(10, 10, X_scale, PBC=True)
net.train()
```

```
  Periodic Boundary Conditions active.
  The weights will be initialized with PCA.
  The map will be trained with the batch algorithm.
  Training SOM... done!
# model
cls = net.cluster(X_scale, clus_type='KMeans') #'MeanShift') #
```

# plot graph
net.nodes_graph(colnum=1)
net.diff_graph()

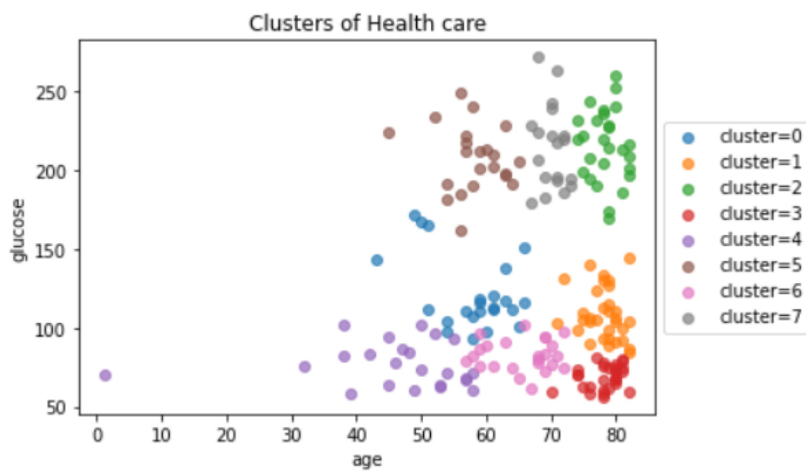




```
# predict
import numpy as np
cls_id = np.zeros([len(X_scale)])
cls_id
```

```
[12]: array([0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
             0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
             0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
             0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
             0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
             0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
             0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
             0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
             0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
             0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
             0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
             0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.])
```

```
import matplotlib.pyplot as plt
for i in range(len(cls)):
    cls_id[cls[i]] = i
    plt.scatter(X.values[cls[i], 0],
                X.values[cls[i], 1], label='cluster='+str(i), alpha=.7)


plt.title("Clusters of Health care")
plt.xlabel("age")
plt.ylabel("glucose")
plt.legend(loc='center left', bbox_to_anchor=(1, 0.5));
plt.show()
```



```
[15]: cls_id
```

```
[15]: array([7., 5., 1., 0., 2., 2., 3., 6., 6., 3., 3., 0., 0., 2., 2., 0., 5.,
       2., 6., 5., 7., 5., 2., 2., 1., 1., 0., 5., 7., 5., 5., 4., 2., 2.,
       4., 1., 2., 6., 0., 4., 3., 4., 1., 6., 5., 2., 1., 0., 1., 4., 1.,
       3., 1., 1., 5., 5., 3., 2., 7., 5., 5., 3., 6., 4., 1., 1., 7., 1.,
       6., 3., 3., 7., 0., 5., 4., 3., 2., 1., 0., 0., 6., 6., 1., 6., 4.,
       0., 0., 3., 4., 1., 2., 3., 4., 3., 4., 2., 6., 5., 6., 6., 6., 2.,
       2., 3., 0., 6., 4., 5., 1., 4., 1., 3., 1., 5., 6., 7., 6., 3., 4.,
       1., 4., 4., 2., 5., 7., 1., 1., 3., 2., 3., 7., 1., 7., 4., 7., 7.,
       1., 2., 2., 2., 1., 0., 3., 6., 7., 0., 6., 3., 4., 7., 2., 7., 3.,
       6., 4., 2., 4., 5., 7., 3., 3., 6., 4., 2., 6., 1., 5., 3., 1., 1.,
       0., 3., 0., 1., 3., 7., 3., 3., 1., 2., 4., 0., 4., 3., 0., 0., 0.,
       2., 1., 5., 5., 0., 0., 7., 6., 5., 7., 6., 1.])
```

Evaluation

Silhouette Coefficient

```
from sklearn.metrics import silhouette_score

sil_coeff = silhouette_score(X, cls_id, metric='euclidean')

sil_coeff = 0.18583855137939662
```

3. จงใช้ K-Mean จำแนกกลุ่ม 3 กลุ่ม ของ Salary_Data.csv และใช้ Sum square error ในการประเมิณ (2%)

```
import pandas as pd
df = pd.read_csv('data/Salary_Data.csv')
df
```

[1]:

	YearsExperience	Salary
0	1.1	39343.0
1	1.3	46205.0
2	1.5	37731.0
3	2.0	43525.0
4	2.2	39891.0
5	2.9	56642.0
6	3.0	60150.0
7	3.2	54445.0
8	3.2	64445.0
9	3.7	57189.0
10	3.9	63218.0

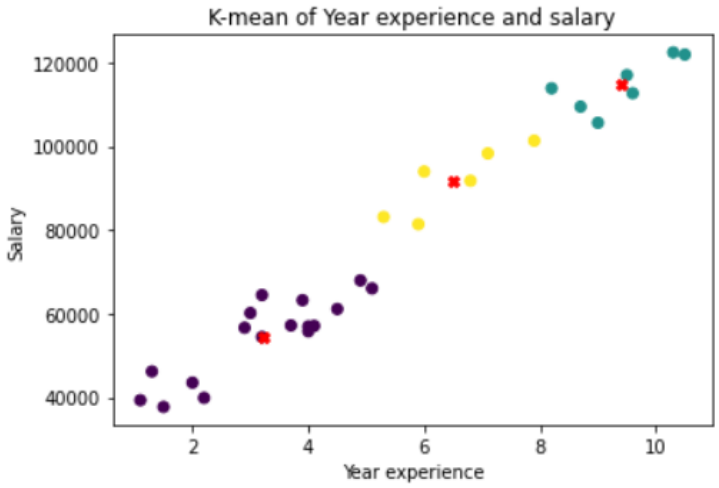
```
from sklearn.cluster import KMeans
model = KMeans(n_clusters=3,random_state = 0)
model.fit(df)

cls_id = model.labels_
cls_id
```

```
[5]: array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 2, 2, 2, 2,
           2, 1, 1, 1, 1, 1, 1, 1])
```

```
centroid = model.cluster_centers_
centroid
```

```
[6]: array([[3.21176471e+00, 5.45702353e+04],
            [9.40000000e+00, 1.14670286e+05],
            [6.50000000e+00, 9.16173333e+04]])
```

```
import matplotlib.pyplot as plt
plt.scatter(df["YearsExperience"],df["Salary"], c=cls_id)
plt.scatter(centroid[:,0], centroid[:,1], marker="X", c="r")
plt.title("K-mean of Year experience and salary")
plt.xlabel("Year experience")
plt.ylabel("Salary")
plt.show()
```



Evaluation
Sum square error

```
sse = model.inertia_
```

```
sse = 2056842710.938375
```