遙感探測--作業六

監督式影像分類--精度評估

108207405 王文智

使用工具

- ArcGIS Pro
- QGIS
- SAGA GIS
- Python
  - Numpy
  - PIL
  - pandas

使用資料

- EO-1 ALI
  - EO1A1170442016203110K0
- 同學提供的地真資料
  (108207406、108207408、108207411)

前情提要

Kappa係數反應的是跟隨機分類成果比較的優劣（-1 <= κ <= 1）

κ＝（該次分類正確程度-隨機分類的正確程度）/（1-隨機分類的正確程度）

隨機分類的正確程度＝分類時挑中第i類的機率*被分為第i類的機率

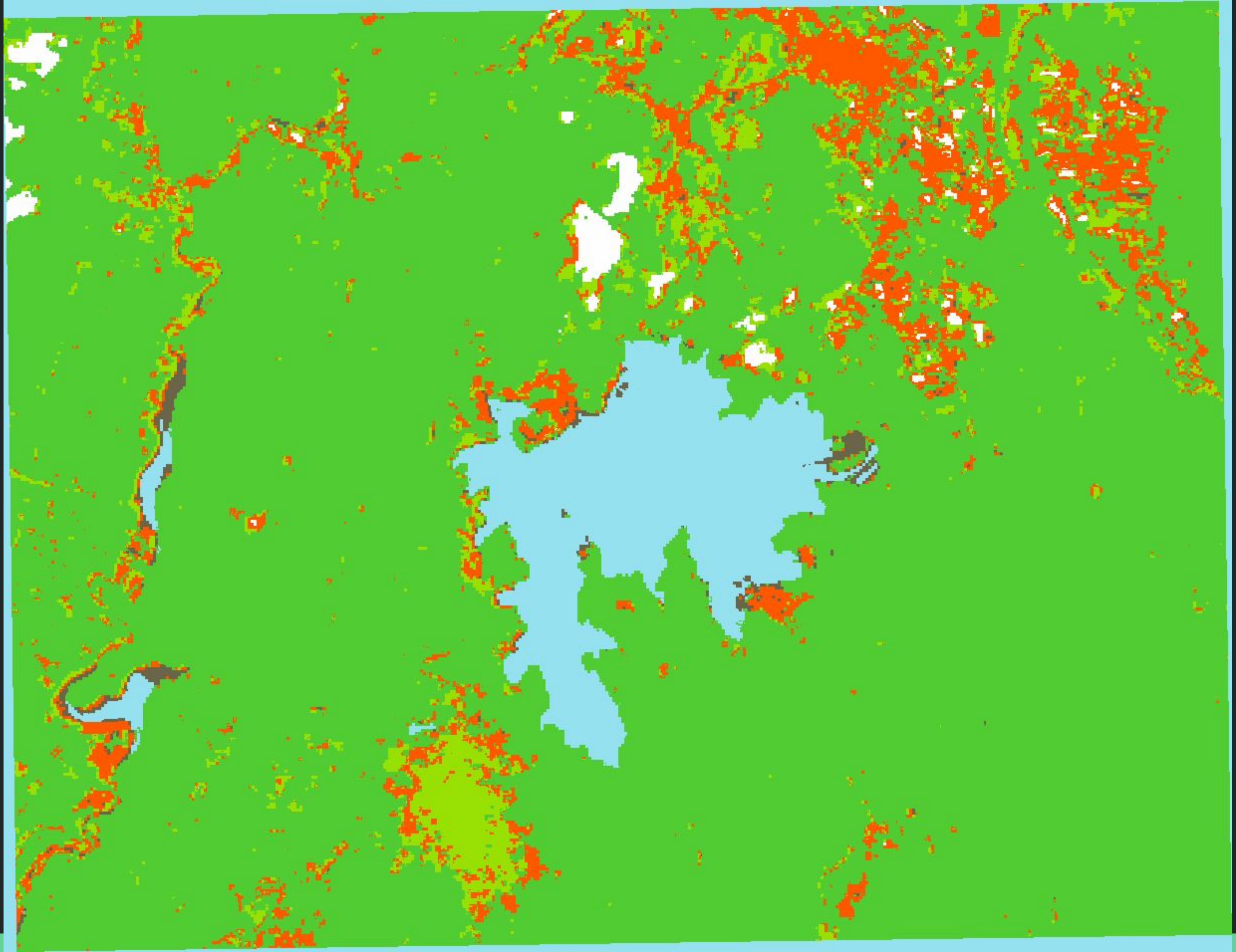
＝（地真資料第i類的個數/總數）*（分類成果第i類的個數/總數）

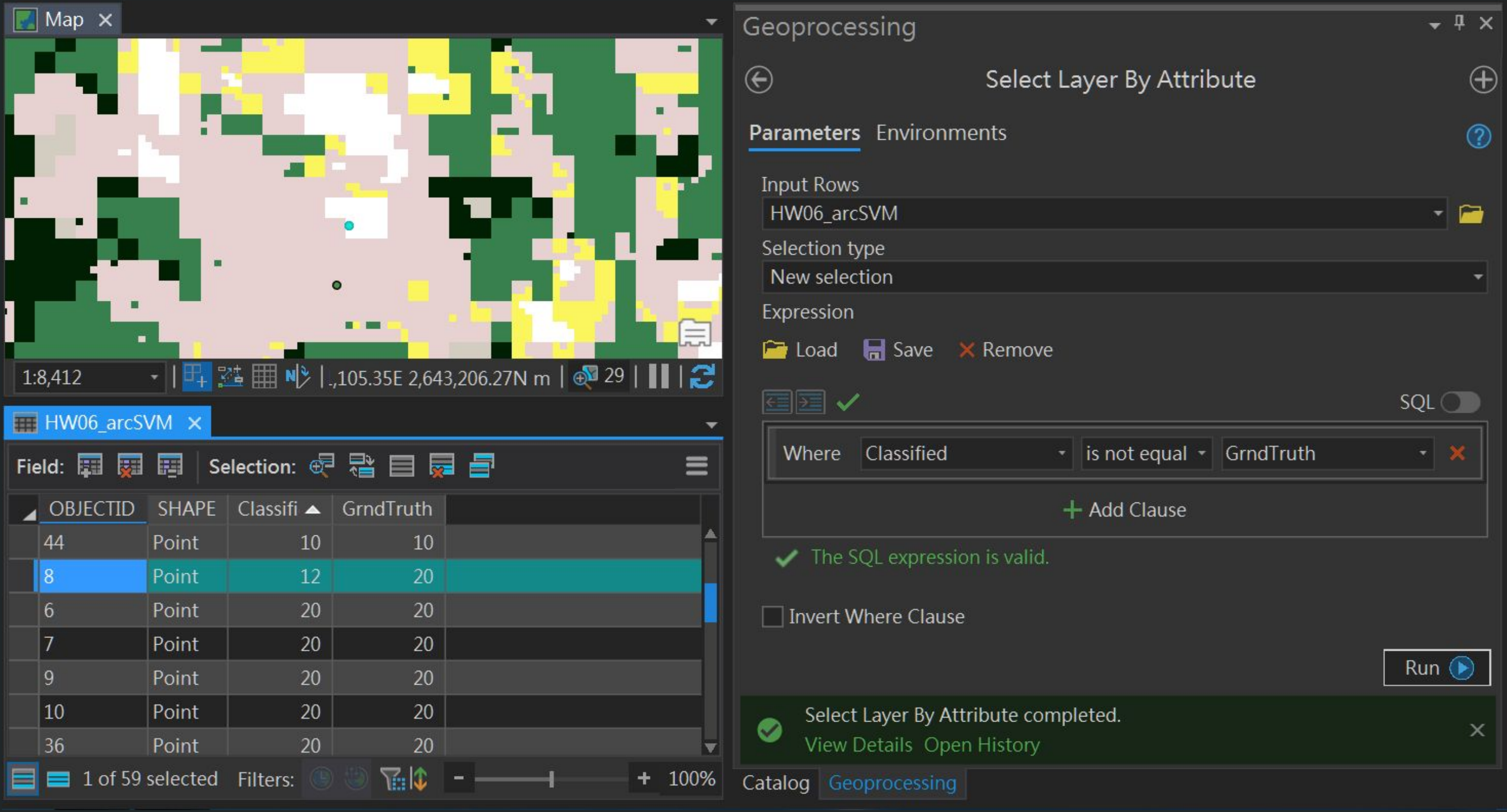
（假設分類間獨立不相干，為獨立事件

κ＝1→完美　　0<κ<1→還行　　κ＝0→跟隨機分的差不多　　κ < 0→比隨機分的還糟糕

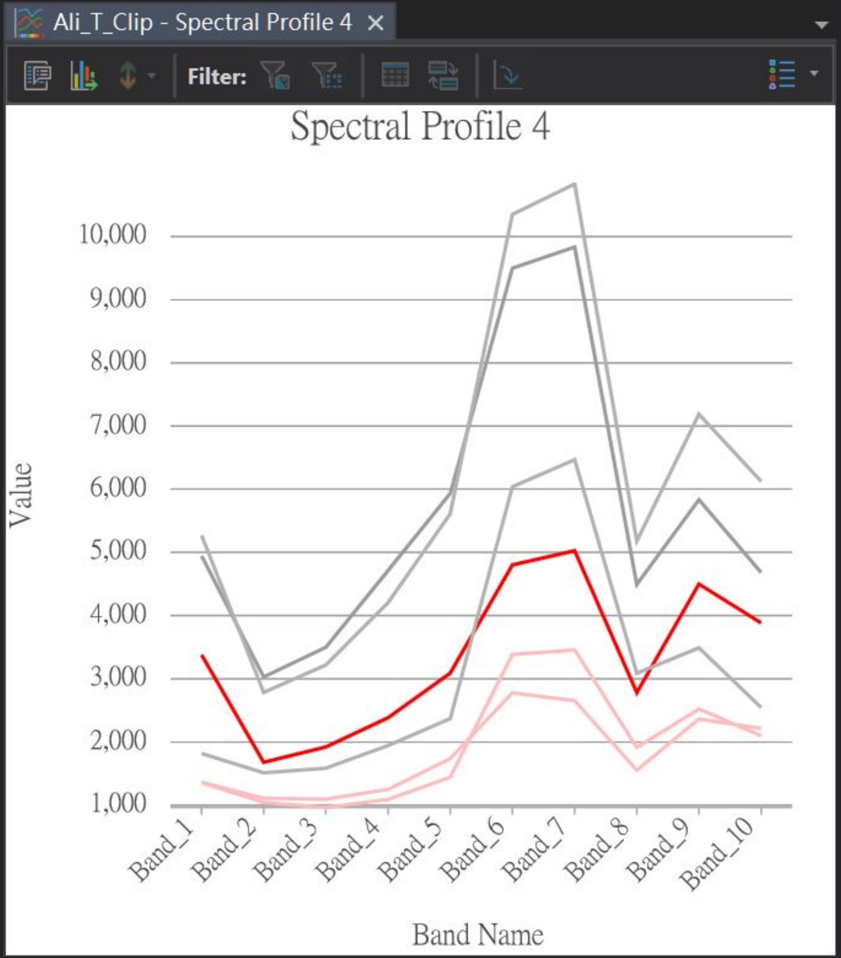
ArcGIS SVM

ClassValue	C_10 (水)	C_12 (雲)	C_20 (建物)	C_42 (植被)	Total	U_Accuracy	Kappa
C_10	15	0	0	0	15	1	0
C_12	0	0	1	0	1	0	0
C_20	0	0	13	0	13	1	0
C_42	0	0	0	30	30	1	0
Total	15	0	14	30	59	0	0
P_Accuracy	1	0	0.92857143	1	0	0.98305085	0
Kappa	0	0	0	0	0	0	0.97286109



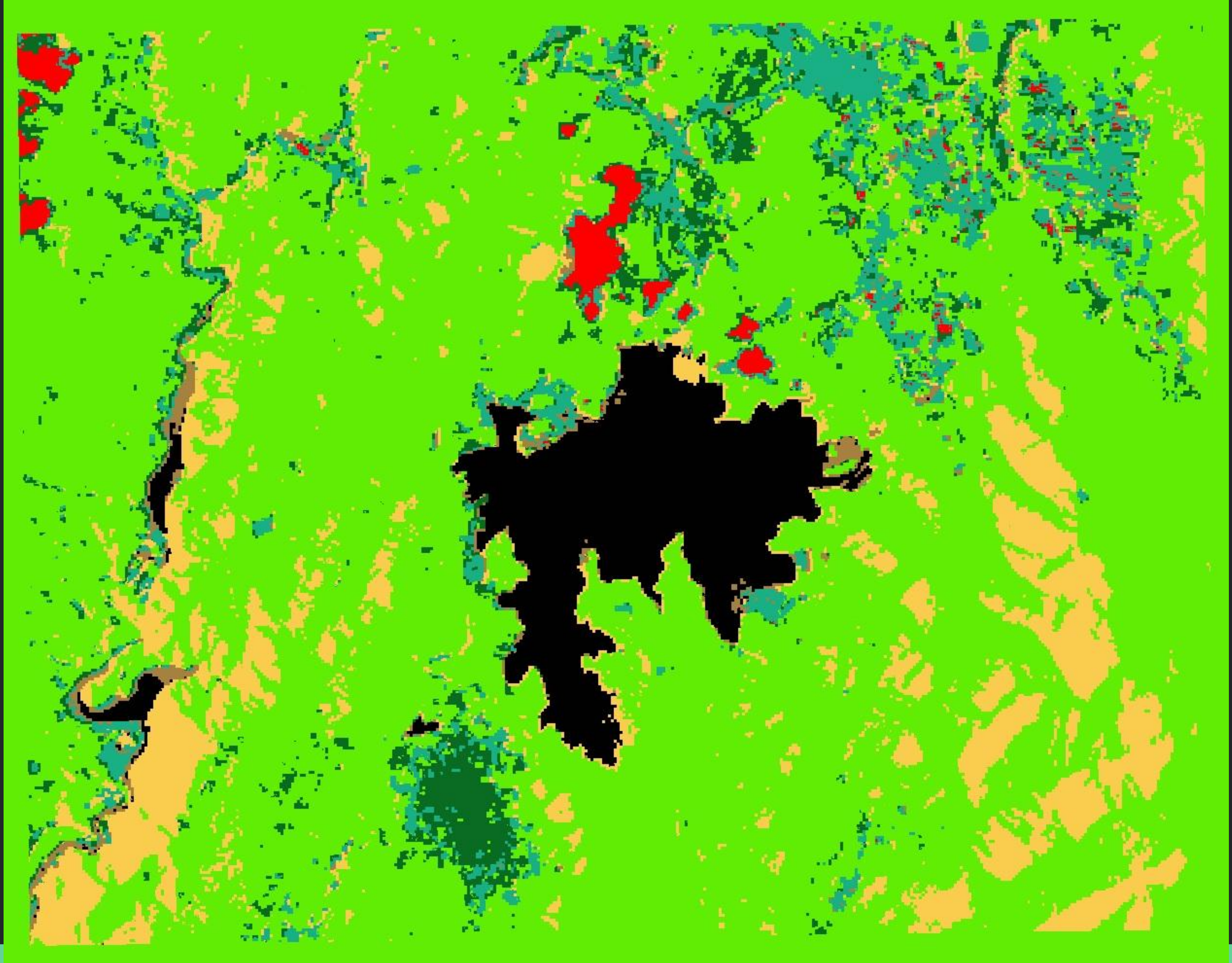

Map
1:8,412
,105.35E 2,643,206.27N m
29
HW06_arcSVM
Field:
Selection:
OBJECTID | SHAPE | Classifi | GrndTruth
44 | Point | 10 | 10
8 | Point | 12 | 20
6 | Point | 20 | 20
7 | Point | 20 | 20
9 | Point | 20 | 20
10 | Point | 20 | 20
36 | Point | 20 | 20
1 of 59 selected
Filters:
100%
Geoprocessing
Select Layer By Attribute
Parameters
Environments
Input Rows
HW06_arcSVM
Selection type
New selection
Expression
Load
Save
Remove
SQL
Where
Classified
is not equal
GrndTruth
Add Clause
The SQL expression is valid.
Invert Where Clause
Run
Select Layer By Attribute completed.
View Details
Open History
Catalog
Geoprocessing


Map
1:53,189
98,984.92E 2,646,722.04N m
29


Ali_T_Clip - Spectral Profile 4
Filter:
Spectral Profile 4
Value
10,000
9,000
8,000
7,000
6,000
5,000
4,000
3,000
2,000
1,000
Band_1
Band_2
Band_3
Band_4
Band_5
Band_6
Band_7
Band_8
Band_9
Band_10
Band Name

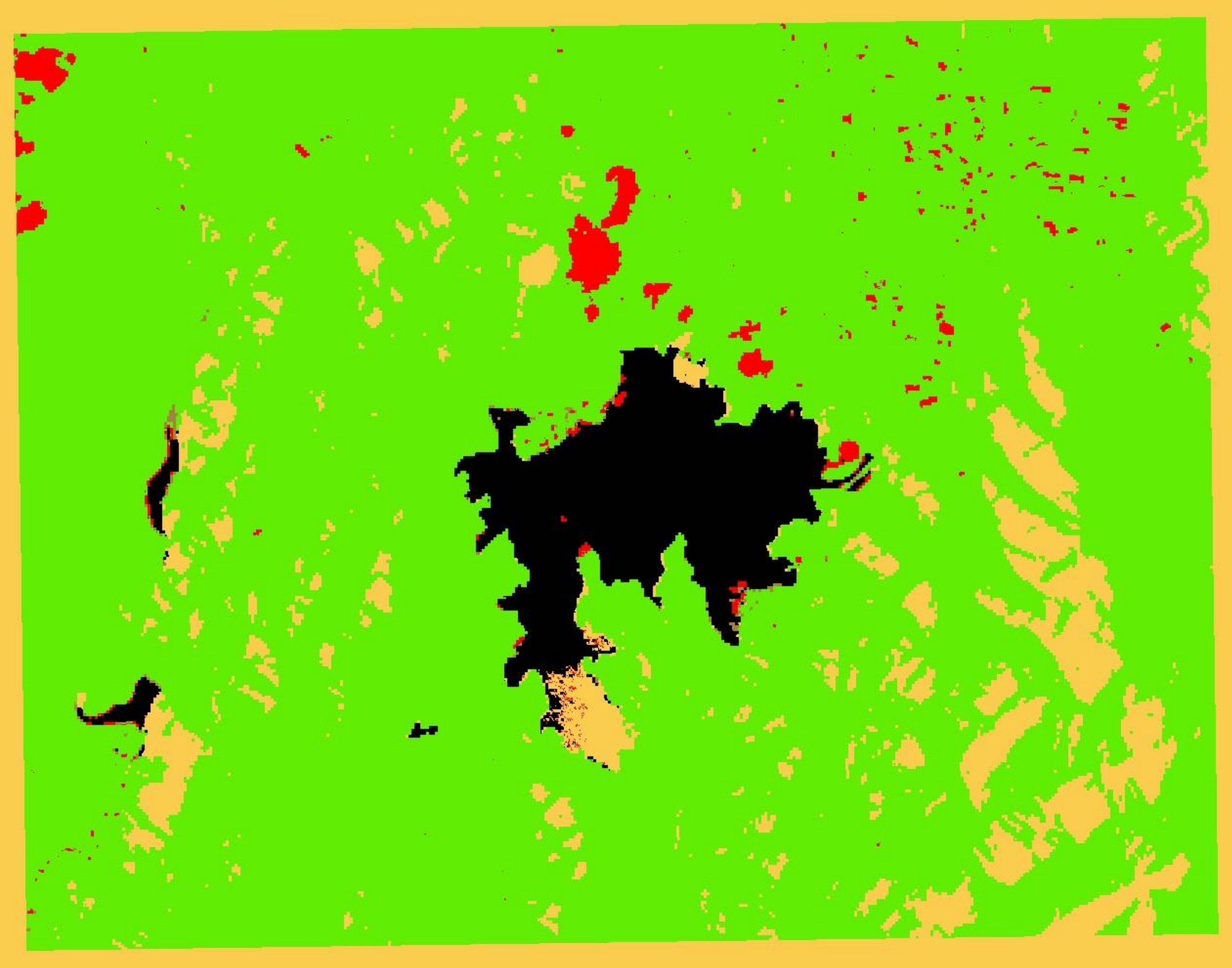
SAGA GIS -- ANN (OpenCV) 15x2

ClassValue	C_0(水)	C_2(建物)	C_4(植被)	Total	U_Accuracy	Kappa
C_0	15	0	0	15	1	0
C_2	0	14	0	14	1	0
C_4	0	0	30	30	1	0
Total	15	14	30	59	0	0
P_Accuracy	1	1	1	0	1	0
Kappa	0	0	0	0	0	1



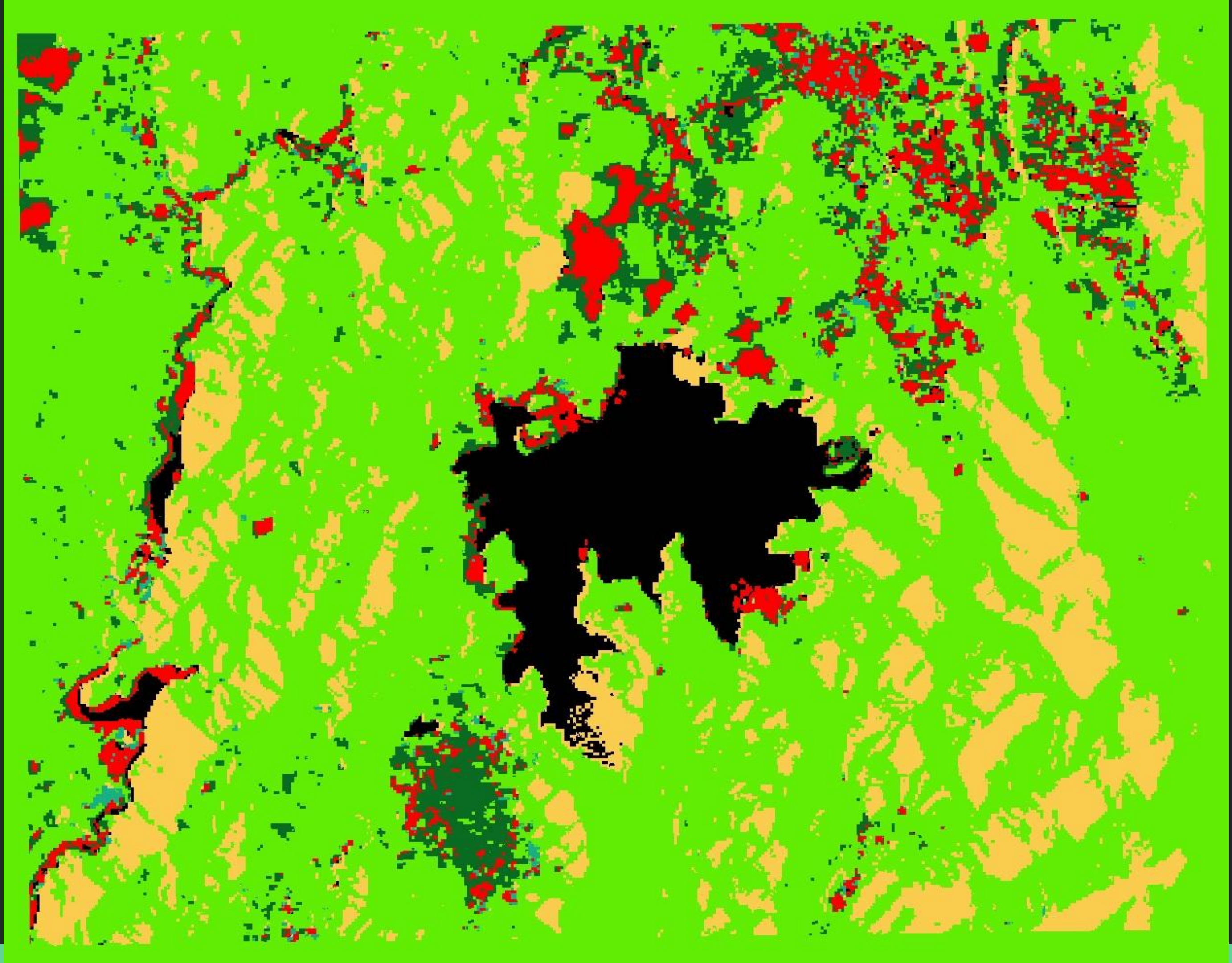
ANN (OpenCV) 6x5

ClassValue	C_0(水)	C_1(雲)	C_2(建物)	C_4(植被)	Total	U_Accuracy	Kappa
C_0	13	0	0	0	13	1	0
C_1	0	0	1	0	1	0	0
C_2	0	0	0	0	0	0	0
C_4	2	0	13	30	45	0.66666667	0
Total	15	0	14	30	59	0	0
P_Accuracy	0.86666667	0	0	1	0	0.72881356	0
Kappa	0	0	0	0	0	0	0.51239669



ANN (OpenCV) 3x3

ClassValue	C_0(水)	C_1(雲)	C_2(建物)	C_4(植被)	Total	U_Accuracy	Kappa
C_0	14	0	0	0	14	1	0
C_1	0	0	12	0	12	0	0
C_2	0	0	2	0	2	1	0
C_4	1	0	0	30	31	0.967741935	0
Total	15	0	14	30	59	0	0
P_Accuracy	0.933333333	0	0.142857143	1	0	0.779661017	0
Kappa	0	0	0	0	0	0	0.668396022



Numpy – Confusion matrix

```
import numpy as np
from PIL import Image
import pandas as pd

GT=np.asarray(Image.open('Rasterize_HW06.tif'))    #GroundTruth
Classified='ANN_15x2.tif'
Clsf=np.asarray(Image.open(Classified))
GTclasses=[0,1,2]     #0:water 1:building 2:forest
ClassTable={0:0,1:2,2:4}     # ground_truth : classified
VdDict={}     #validation pixel's class and [col,row]
Clsf=np.where(Clsf==5,4,Clsf)     #merge class
Class_Truth=[]
Class_Predicted=[]
for Gclass in GTclasses:
        VdDict[Gclass]=np.argwhere(GT==Gclass).tolist()
        for index in VdDict[Gclass]:
                indices=index[0],index[1]
                cla=Clsf[indices]
                Class_Truth.append(ClassTable[Gclass])
                Class_Predicted.append(cla)
                if cla!=ClassTable[Gclass]:
                        print(indices,Gclass,cla)
df=pd.DataFrame(list(zip(Class_Truth,Class_Predicted)),columns =['Truth', 'Predicted'])
CM=pd.crosstab(df['Truth'],df['Predicted'],rownames=['Actual'], colnames=['Predicted'],margins=True)
print(CM.transpose())
```

```
E:\Homework\RemoteSensing\Reference>py ConfusionMatrix.py
(450, 539) 0 2
(505, 886) 0 4
(704, 545) 0 4
(767, 439) 0 2
(801, 641) 0 4
(181, 1154) 1 4
(195, 353) 1 4
(424, 590) 1 4
(445, 581) 1 0
(829, 437) 1 6
Actual       0   2   4  All
Predicted
0           10   1   0   11
2            2   9   0   11
4            3   3  30   36
6            0   1   0    1
All         15  14  30   59
```

Numpy 改

```
import numpy as np
from PIL import Image
import pandas as pd
def ConfusionMatrix(GrndTruth,Classified):  #Rasterize_GrndTruth.tif,Classified.tif
    GT=np.asarray(Image.open(GrndTruth))
    Clsf=np.asarray(Image.open(Classified))
    GTclasses={0:'water',1:'building',2:'forest'}
    ClassTable={0:'water',1:'cloud',2:'building',3:'other',4:'forest',5:'forest',6:'other'} #All classes
    VdDict={} #validation pixel's class and [col,row]
    Class_Truth=[]
    Class_Predicted=[]
    for Gvalue in GTclasses:
        VdDict[Gvalue]=np.argwhere(GT==Gvalue).tolist() #index.type(np.array->list)
        Gclass=GTclasses[Gvalue] #Classname(Ground Truth)
        for index in VdDict[Gvalue]:
            indices=index[0],index[1] #row,colum
            Cvalue=Clsf[indices]    #Corresponed pixel value in 'Classified'
            cla=ClassTable[Cvalue] #Classname(Classified)
            Class_Truth.append(Gclass)
            Class_Predicted.append(cla)
            if cla!=Gclass:
                print(indices,Gclass,cla)
    df=pd.DataFrame(list(zip(Class_Truth,Class_Predicted)),columns =['Truth', 'Predicted'])
   CM=pd.crosstab(df['Predicted'],df['Truth'],margins=True)
    print(CM)
    p0=0;p1=0
    n=CM['All']['All']
    for cla in GTclasses.values():
        p0+=CM[cla][cla]/n
        p1+=CM[cla]['All']/n*CM['All'][cla]/n
    kappa=(p0-p1)/(1-p1)
    print("Cohen's Kappa:%.5f"%kappa)
    return CM,kappa
ConfusionMatrix('Rasterize_HW06.tif','ANN_15x2.tif')
```

```
(783, 627) water forest
(76, 835) building cloud
(163, 1140) building cloud
(172, 959) building cloud
(177, 339) building cloud
(179, 1106) building cloud
(207, 667) building cloud
(406, 576) building cloud
(422, 572) building cloud
(427, 567) building cloud
(611, 777) building cloud
(618, 781) building cloud
(811, 423) building cloud
Truth        building  forest  water  All
Predicted
building            2       0      0    2
cloud              12       0      0   12
forest              0      30      1   31
water               0       0     14   14
All                14      30     15   59
```

Numpy – For loop

```
CMdict={}
Kappadict={}
from pathlib import Path
dir='Classified'
tifs=Path(dir).glob('*.tif')
for file in tifs:
    filename=str(file)[len(dir)+1:-4]
    CMdict[filename],Kappadict[filename]=ConfusionMatrix('Rasterize_HW06.tif',file)
KappaDF=pd.DataFrame({'Node x Layer':Kappadict.keys(),'Kappa':Kappadict.values()})
print(KappaDF.sort_values(by=['Kappa']))
```

	Node x Layer	Kappa
7	ANN_6x5	0.512397
5	ANN_3x1	0.668396
6	ANN_3x3	0.668396
0	ANN_10x2	0.972861
3	ANN_15x3	0.972861
8	ANN_7x3	0.972861
1	ANN_10x3	0.972874
2	ANN_15x2	1.000000
4	ANN_20x2	1.000000
9	ANN_7x4	1.000000

小結

1. 預處裡
   a. 雲、雲影 ➔ CloudMask or 取一段時間的中位數
   https://gis.stackexchange.com/questions/393828/median-vs-mean-in-gee

   但因sensor通過時間相同，日照條件相近，地形造成的陰影尚未除去

   b. 地形陰影 ➔ Topographic Correction (DEM+太陽位置)
   Review of Shadow Detection and De-shadowing Methods in RemoteSensing
2. ANN
   a. Layer不是越多越好，這次的資料大約 2層就夠
   b. 各Layer的Node也不是越多越好，大約 7~20個較為合適

報告結束

謝謝大家