

Code:

```
109611099_顏彥臣_MLP_HW6.py - C:\prog\ML\hw6\109611099_顏彥臣_MLP_HW6.py (3.8.3)
File Edit Format Run Options Window Help

import numpy as np
import mglearn
import pandas as pd
import sklearn
import matplotlib.pyplot as plt
from sklearn.linear_model import Ridge
from sklearn.model_selection import train_test_split
from sklearn.neural_network import MLPClassifier
from sklearn.svm import SVC
#####
def readHw6(inFileName):
    # init
    recArr = []
    clsArr = []

    # open input text data file, format is given
    inFile = open(inFileName, 'r')
    s = inFile.readline() # skip

    row = 0
    while True:
        s = inFile.readline()
        datal = s.strip() # remove leading and ending blanks
        if (len(datal) <= 0):
            break

        # since we use append, value must be created in the loop
        value = []

        str4 = datal.split(',') # array of 4 str

        # convert to real
        for ix in range(3):
            value.append( eval(str4[ix]) )
        # end for

        target = eval(str4[3])

        recArr.append(value) ; # add 1 record at end of array
        clsArr.append(target) ; # add 1 record at end of array

        row = row+1 # total read counter
    # end while
    # close input file
    inFile.close()
    # convert list to Numpy array
    npXY = np.array(recArr)
    npC = np.array(clsArr)
    # pass out as Numpy array

Ln: 25 Col: 17
```

```

# convert list to Numpy array
npXY = np.array(recArr)
npC = np.array(clsArr)
# pass out as Numpy array
return npXY, npC
# end function

def TrainTestSplit_Fold(X, y, fold, test_size):
    # safety check
    if (fold < 0):
        fold = 0

    # 輸入
    numValue = X.size
    rows = len(X)
    cols = int(numValue/rows)

    # safety check
    rmns = numValue % rows
    if (rmns != 0):
        print("ERROR - missing data in X")

    t0 = fold * test_size
    t1 = t0 + test_size
    # safety check
    if (t1 > rows):
        print("ERROR - out of bound")
        t1 = rows

    fea_test = X[t0:t1,:]
    tar_test = y[t0:t1]

    dr = [t0+x for x in range(test_size)]

    fea_train = np.delete(X, dr, 0)
    tar_train = np.delete(y, dr, 0)
    return fea_train, fea_test, tar_train, tar_test
# end function

#####
def Train_model_5fold(model,X_5fold,y_5fold,X_check,y_check):
    num_folds=5
    numRec=len(X_5fold)
    testsize=int(numRec/num_folds)
    #####
    total_train = 0
    total_test = 0
    for fold in range(num_folds):
        X_train, X_test, y_train, y_test = TrainTestSplit_Fold(X_5fold, y_5fold, fold, test_size)
        lr=model.fit(X_train,y_train)
        train_s = lr.score(X_train, y_train)
        test_s = lr.score(X_test, y_test)

        total_train += train_s
        total_test += test_s
    #average
    average_train = total_train/num_folds
    average_test = total_test/num_folds

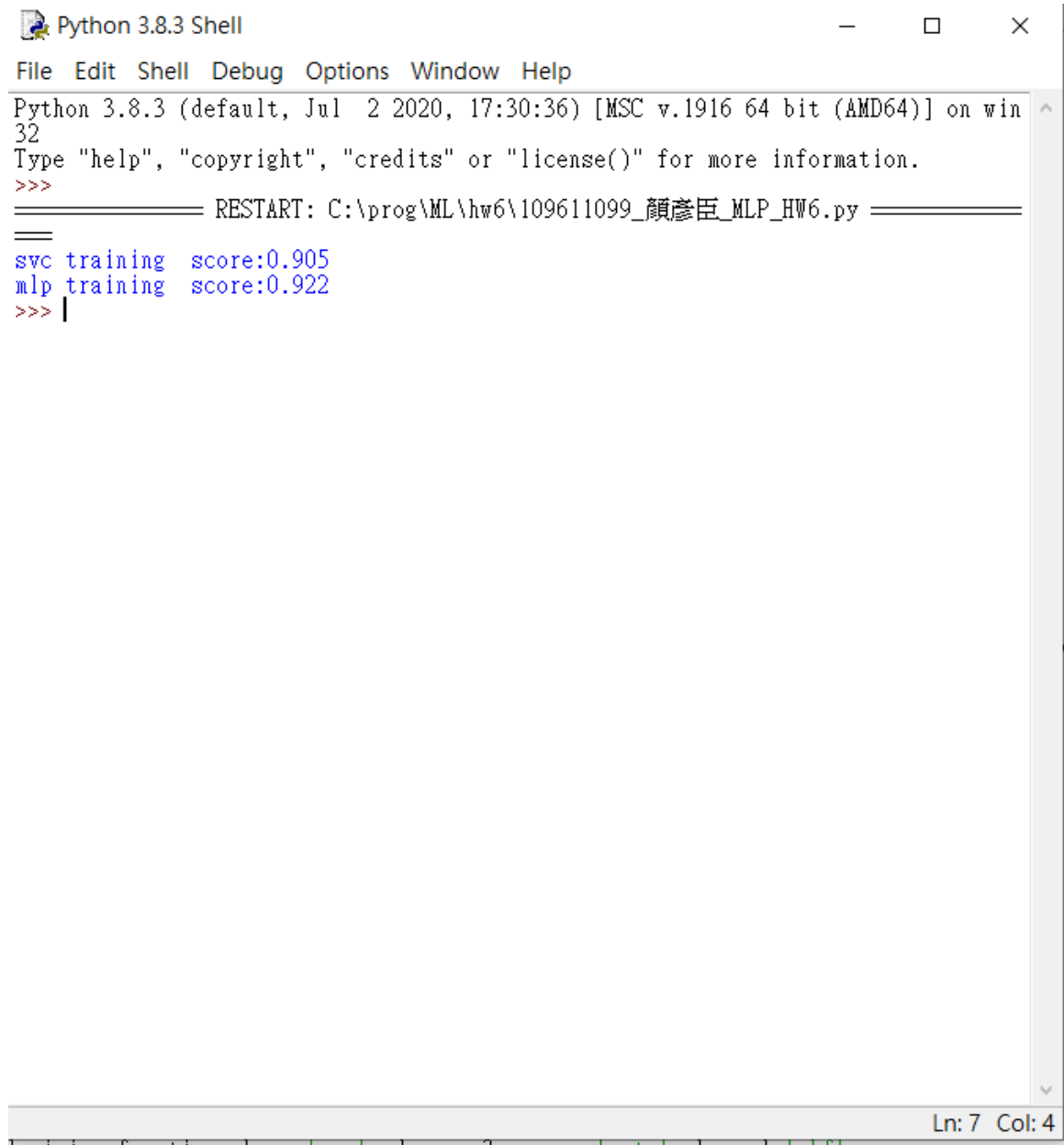
    fea_train = np.delete(X, dr, 0)
    tar_train = np.delete(y, dr, 0)
    return fea_train, fea_test, tar_train, tar_test
# end function

#####
def Train_model_5fold(model,X_5fold,y_5fold,X_check,y_check):
    num_folds=5
    numRec=len(X_5fold)
    testsize=int(numRec/num_folds)
    #####
    total_train = 0
    total_test = 0
    for fold in range(num_folds):
        X_train, X_test, y_train, y_test = TrainTestSplit_Fold(X_5fold, y_5fold, fold, test_size)
        lr=model.fit(X_train,y_train)
        train_s = lr.score(X_train, y_train)
        test_s = lr.score(X_test, y_test)

        total_train += train_s
        total_test += test_s
    #average
    average_train = total_train/num_folds
    average_test = total_test/num_folds
    print("Train/Test average score: {:.3f}/{:.3f}".format(average_train, average_test))
    #5 fold
    lr=model.fit(X_5fold, y_5fold)
    fold5_s=model.score(X_5fold, y_5fold)
    check_s=model.score(X_check, y_check)
    return fold5_s,check_s
#####
X,y=readHv6("C:\prog\ML\hw6\hv6_haberman.csv")
X_train,y_train=X,y
model = SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
decision_function_shape='ovr', degree=3, gamma='auto', kernel='rbf',
max_iter=1, probability=False, random_state=0, shrinking=True,
tol=0.001, verbose=False)
model.fit(X_train,y_train)
print("svc training score:{:.3f}".format(model.score(X_train,y_train)))
#mlp = MLPClassifier(solver='lbfgs', activation='relu', alpha=0.001,hidden_layer_sizes=(50,500), random_state=1,max_iter=10000,verbose=10,learning_rate_init=.1)
mlp = MLPClassifier(solver='lbfgs', activation='relu', alpha=0.001,hidden_layer_sizes=(50,700), random_state=1,max_iter=10000,verbose=10,learning_rate_init=.1)
#mlp = MLPClassifier(solver='sgd', activation='relu', alpha=1e-4, hidden_layer_sizes=(5, 5), random_state=1,max_iter=100, verbose=True, learning_rate_init=.1)
mlp.fit(X_train,y_train)
print("mlp training score:{:.3f}".format(mlp.score(X_train,y_train)))

```

Result:



```
Python 3.8.3 Shell
File Edit Shell Debug Options Window Help
Python 3.8.3 (default, Jul 2 2020, 17:30:36) [MSC v.1916 64 bit (AMD64)] on win
32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\prog\ML\hw6\109611099_顏彥臣_MLP_HW6.py =====
>>>
svc training score:0.905
mlp training score:0.922
>>> |
```

Ln: 7 Col: 4

Parament:

MLPClassifier(solver='lbfgs',activation='relu',alpha=0.001,hidden_layer_sizes=(50,700),random_state=1,max_iter=10000,verbose=10,learning_rate_init=.1)

SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
decision_function_shape='ovr', degree=3, gamma='auto', kernel='rbf',
max_iter=-1, probability=False, random_state=0, shrinking=True,
tol=0.001, verbose=False)