```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""

Example 3.5 k-fold cross-validationStochastic for Gradient Descent (SDG)
using the MINST data set

Developed for Machine Learning for Mechanical Engineers at the University of
South Carolina

@author: austin_downey
"""

import IPython as IP
IP.get_ipython().magic('reset -sf')

import numpy as np
import scipy as sp
import pandas as pd
from scipy import fftpack, signal # have to add
import matplotlib as mpl
import matplotlib.pyplot as plt
import sklearn as sk
from sklearn import linear_model
from sklearn import pipeline
from sklearn import datasets

cc = plt.rcParams['axes.prop_cycle'].by_key()['color']
plt.close('all')


#%% Load your data

# Fetch the MNIST dataset from openml
mnist = sk.datasets.fetch_openml('mnist_784',as_frame=False)
X = mnist['data']    # load the data
Y = np.asarray(mnist['target'],dtype=int)  # load the target

# Split the data set up into a training and testing data set
X_train = X[0:60000,:]
X_test = X[60000:,:]
Y_train = Y[0:60000]
Y_test = Y[60000:]

#%% Train a Stochastic Gradient Descent classifier

# Extract a subset for our "5-detector".
Y_train_5 = (Y_train == 5)
Y_test_5 = (Y_test == 5)

sgd_clf = sk.linear_model.SGDClassifier()

#%% Build a 5-detector several times so see the variation in metrics returned by SGD

# Solve the model multiple times to see the variations
for i in range(5):

    # Train the model and eturn the predictions made by SGD
    sgd_clf.fit(X_train, Y_train_5)
    Y_train_pred = sgd_clf.predict(X_train)

    # Use SK learn model to retunr metrics
    accuracy = sk.metrics.accuracy_score(Y_train_5, Y_train_pred)
    precision = sk.metrics.precision_score(Y_train_5, Y_train_pred)
    recall = sk.metrics.recall_score(Y_train_5, Y_train_pred)
    print('accuracy is '+str(np.round(accuracy,4))+'; precision is '+str(np.round(
    precision,4))+
```

```
            '; recall is '+str(np.round(recall,4)))


#%% Build a 5-detector using k-fold cross-validation

# From our example we can see quite a variation in results for a model, making it
# hard to select the proper model. However, k-fold cross-validation can help with this.


# make a prediction using the k-fold method to split up the data set. Again,
# solve the model multiple times to see the variations
for i in range(5):

    # Train the model and eturn the predictions made by SGD
    Y_train_pred = sk.model_selection.cross_val_predict(sgd_clf, X_train, Y_train_5, cv=3
    )

    # Use SK learn model to retunr metrics
    accuracy = sk.metrics.accuracy_score(Y_train_5, Y_train_pred)
    precision = sk.metrics.precision_score(Y_train_5, Y_train_pred)
    recall = sk.metrics.recall_score(Y_train_5, Y_train_pred)
    print('accuracy is '+str(np.round(accuracy,4))+'; precision is '+str(np.round(
    precision,4))+
          '; recall is '+str(np.round(recall,4)))


```