人工智慧之應用
用人工智慧處理工具機之顫振效應

程式介紹:

```
import numpy as np
import matplotlib.pyplot as plt
from os import walk
from sklearn.preprocessing import MinMaxScaler
from sklearn.neural_network import MLPClassifier
from sklearn.model_selection import LeaveOneOut
from sklearn.model_selection import cross_val_predict
from sklearn.metrics import confusion_matrix
from numpy import genfromtxt

train_input=[]
train_input_std=[]
train_output=[]
folder_name=['stable','unstable']
i = 0;
for folder in folder_name:
   path = 'Data/'+str(folder)+'/'
   for root,dirs,files in walk(path):
      for f in files:
         filename = path + f
         print(filename)
         acc = genfromtxt(filename, delimiter=',')
         acc = acc[:,1].tolist()
         train_input.append(acc[60000:80000])
         train_input_std.append(np.std(acc))

         if folder == 'unstable':
            train_output.append(1)
            title = 'Original Signal With Chatter #'
            saved_file_name = 'Fig/Original/unstable_'
         if folder == 'stable':
            train_output.append(0)
            title = 'Original Signal Without Chatter #'
            saved_file_name = 'Fig/Original/stable_'

         plt.figure(figsize=(7,4))
         plt.plot(acc,'b-',lw=1)
         plt.title(title + str(i+1))
         plt.xlabel('Samples')
         plt.ylabel('Acceleration')
         plt.savefig(saved_file_name + str(i+1) + '.png')
         plt.show()
```

```
        i = i + 1

train_input = np.array(train_input_std)
train_output = np.array(train_output)

scaler = MinMaxScaler(feature_range=(0,1))
train_input=scaler.fit_transform(train_input.reshape(-1,1))

loo = LeaveOneOut()
model = MLPClassifier(max_iter=500, batch_size=1, solver='adam')

y_pred = cross_val_predict(model, train_input, train_output, cv=loo)
y_true = train_output

print('Prediction: \t', y_pred)
print('Ground Truth: \t',y_true)


cf_m = confusion_matrix(y_true, y_pred)
print('Confusion Matrix: \n', cf_m)

tn, fp, fn, tp = cf_m.ravel()
accuracy = (tn+tp)/(tn+fp+fn+tp)
print('Accuracy: ', accuracy)
```

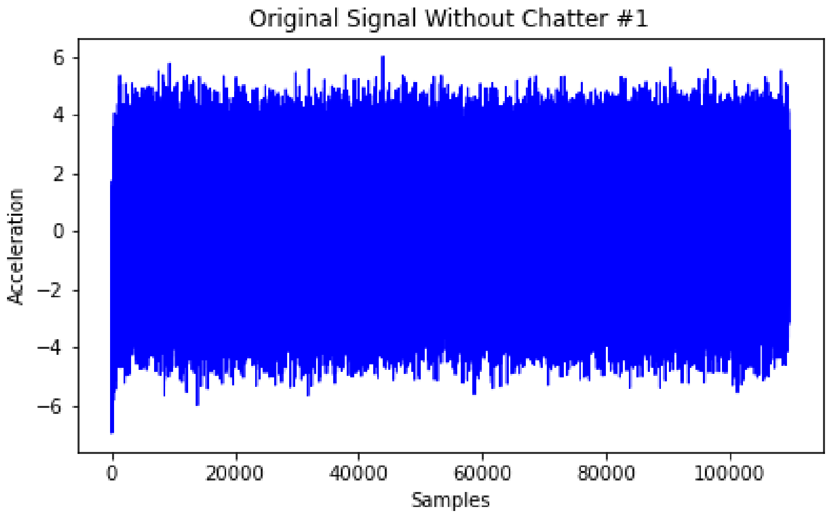
實驗過程:
利用郭秉寰教授的影片中的程式，但因助教給的數據為csv檔，和影片中的mat檔不同，所以做了變化，用了genfromtxt來拿出數據，其餘和郭教授的程式一樣，且助教的數據為x,y,z方向的，也有各用過，而x,y,z方向切換用acc = acc[:,1].tolist()這一行的數字切換就行，x:0，y:1，z:2。

訓練模型:

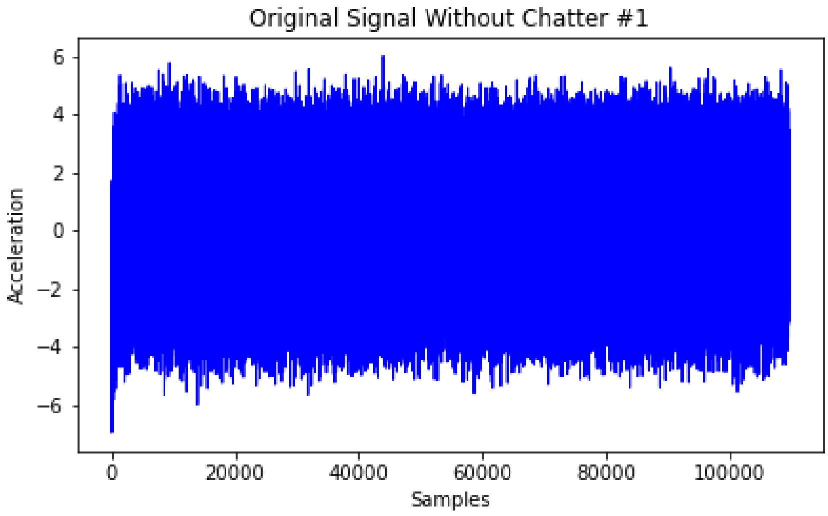
```
train_input = np.array(train_input_std)
train_output = np.array(train_output)

scaler = MinMaxScaler(feature_range=(0,1))
train_input=scaler.fit_transform(train_input.reshape(-1,1))

loo = LeaveOneOut()
model = MLPClassifier(max_iter=500, batch_size=1, solver='adam')

y_pred = cross_val_predict(model, train_input, train_output, cv=loo)
y_true = train_output
```

成果:
X方向:

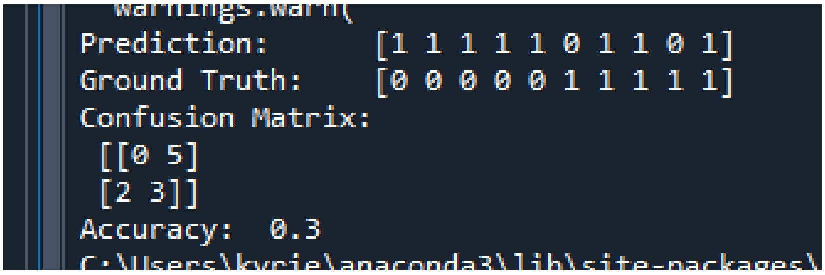
unstable 的其中一張圖



stable 的其中一張圖



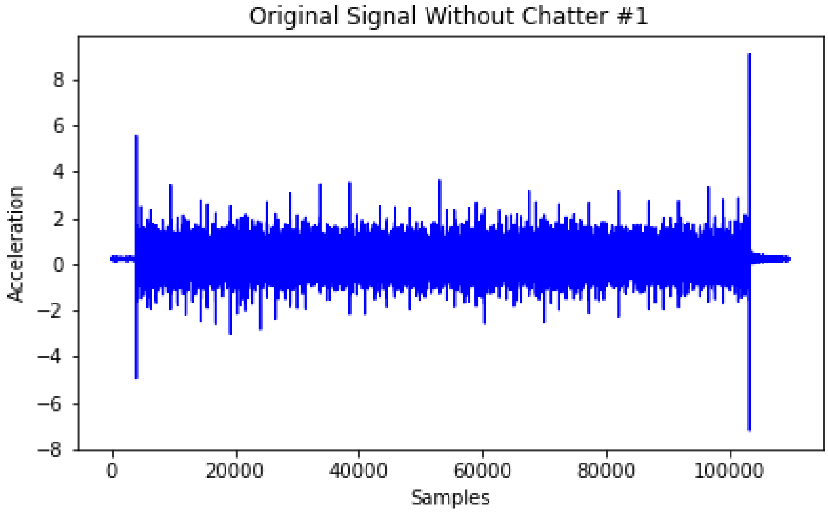
前面看得出兩張圖幾乎沒差，所以測試出的成果準確率也很低，以下是 60000~80000 的成果，這算準了，有些區間準確度甚至是 0



```
  warnings.warn(
Prediction:        [1 1 1 1 1 0 1 1 0 1]
Ground Truth:      [0 0 0 0 0 1 1 1 1 1]
Confusion Matrix:
 [[0 5]
 [2 3]]
Accuracy:  0.3
C:\Users\kyrie\anaconda3\lib\site-packages\
```

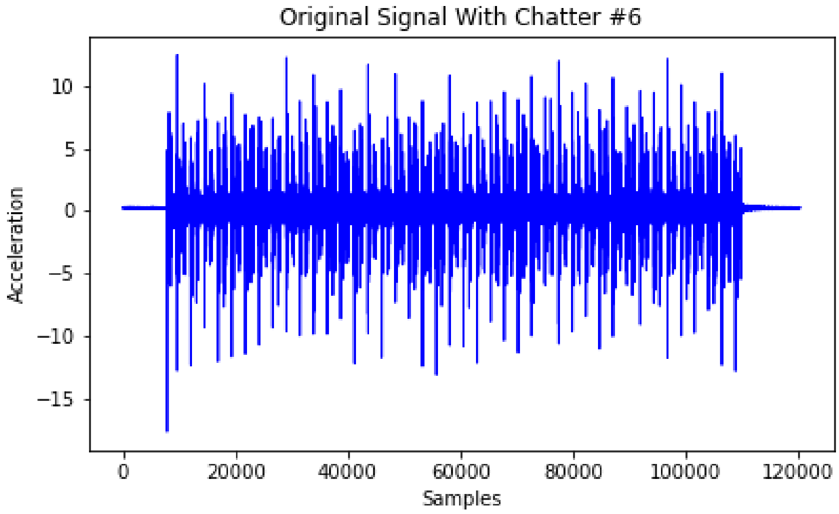
Y 方向:

stable 的其中一張圖



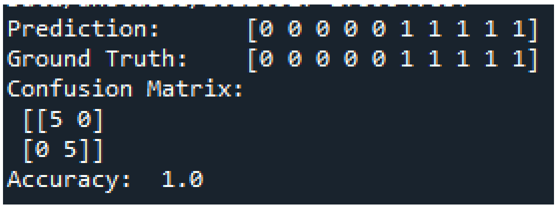
Original Signal Without Chatter #1

unstable 的其中一張圖



Original Signal With Chatter #6

前面看得出兩張圖的差距，而測出的數據也都準確度為 1，相當成功



```
Prediction:      [0 0 0 0 0 1 1 1 1 1]
Ground Truth:    [0 0 0 0 0 1 1 1 1 1]
Confusion Matrix:
 [[5 0]
 [0 5]]
Accuracy:  1.0
```

Z 方向因和 Y 一樣準，所以就沒放上來

由實驗成果得知，X 方向訊號做為有無顫振的判斷相當失敗，而 YZ 方向則準確度皆為 1，用來判斷顫振依據是相當成功的，可以信任。