**居家安全智慧地震盒使用手冊**

注意：

我們於桃園市八德區的桃園防災教育館，透過館內地震平台進行資料蒐集，而館方的儀器震度有弱化，故收集到的資料與後續之資料處理可能與實際發生的地震頗有落差。

**目錄**

[1. 整體架構 2](#_Toc106110573)

[2. 蒐集資料的地震盒(收集地震資料，以訓練機器學習) 4](#_Toc106110574)

[2.1 地震盒 4](#_Toc106110575)

[2.2 資料庫 6](#_Toc106110576)

[2.3 機器學習 8](#_Toc106110577)

[2.4 感測端:(搭配SKTIME模型) 9](#_Toc106110578)

[2.5 傳輸端 10](#_Toc106110579)

[2.6 控制端 11](#_Toc106110580)

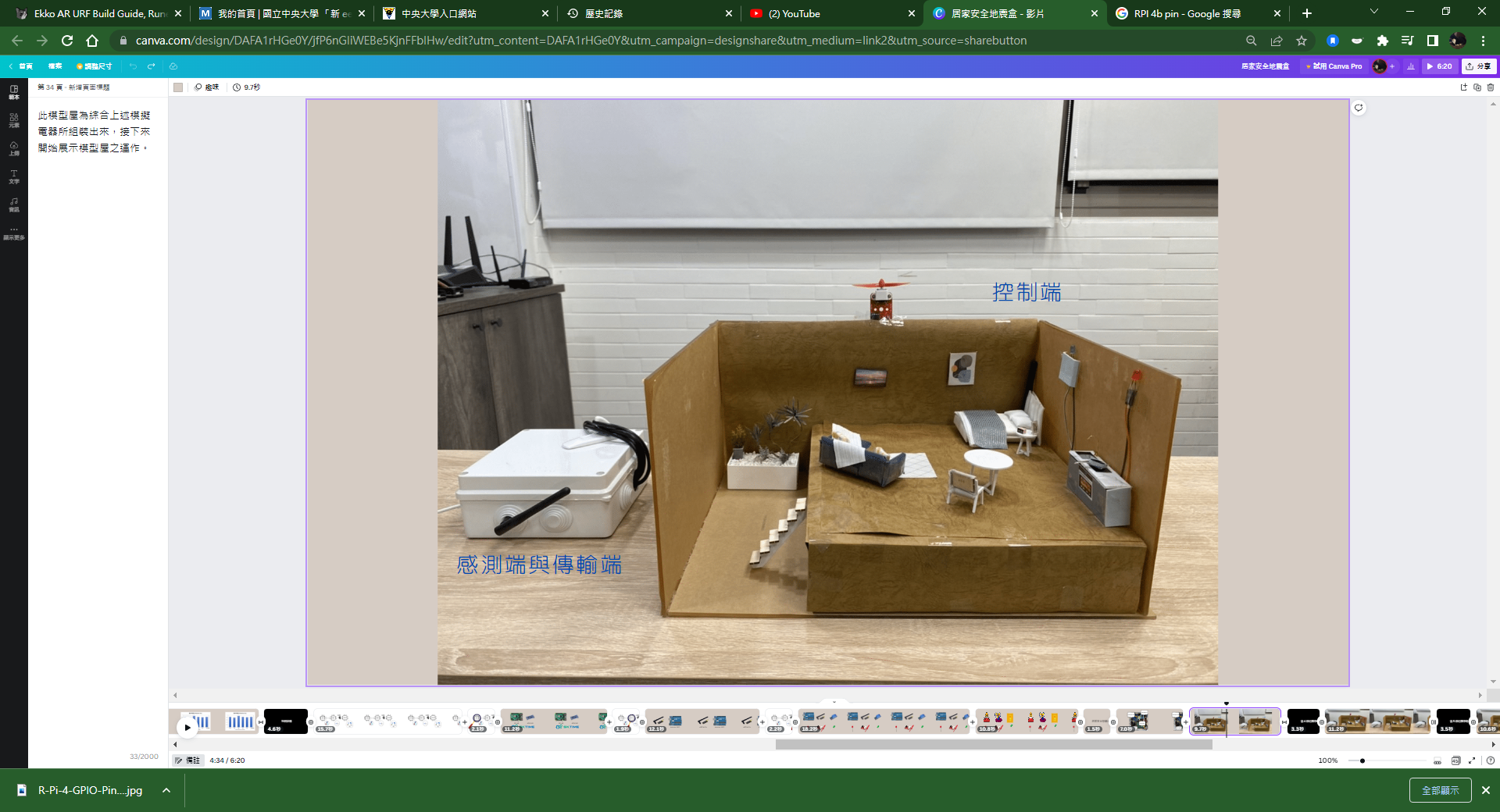
[3. 模型屋 13](#_Toc106110581)

[4. 使用版本 13](#_Toc106110582)

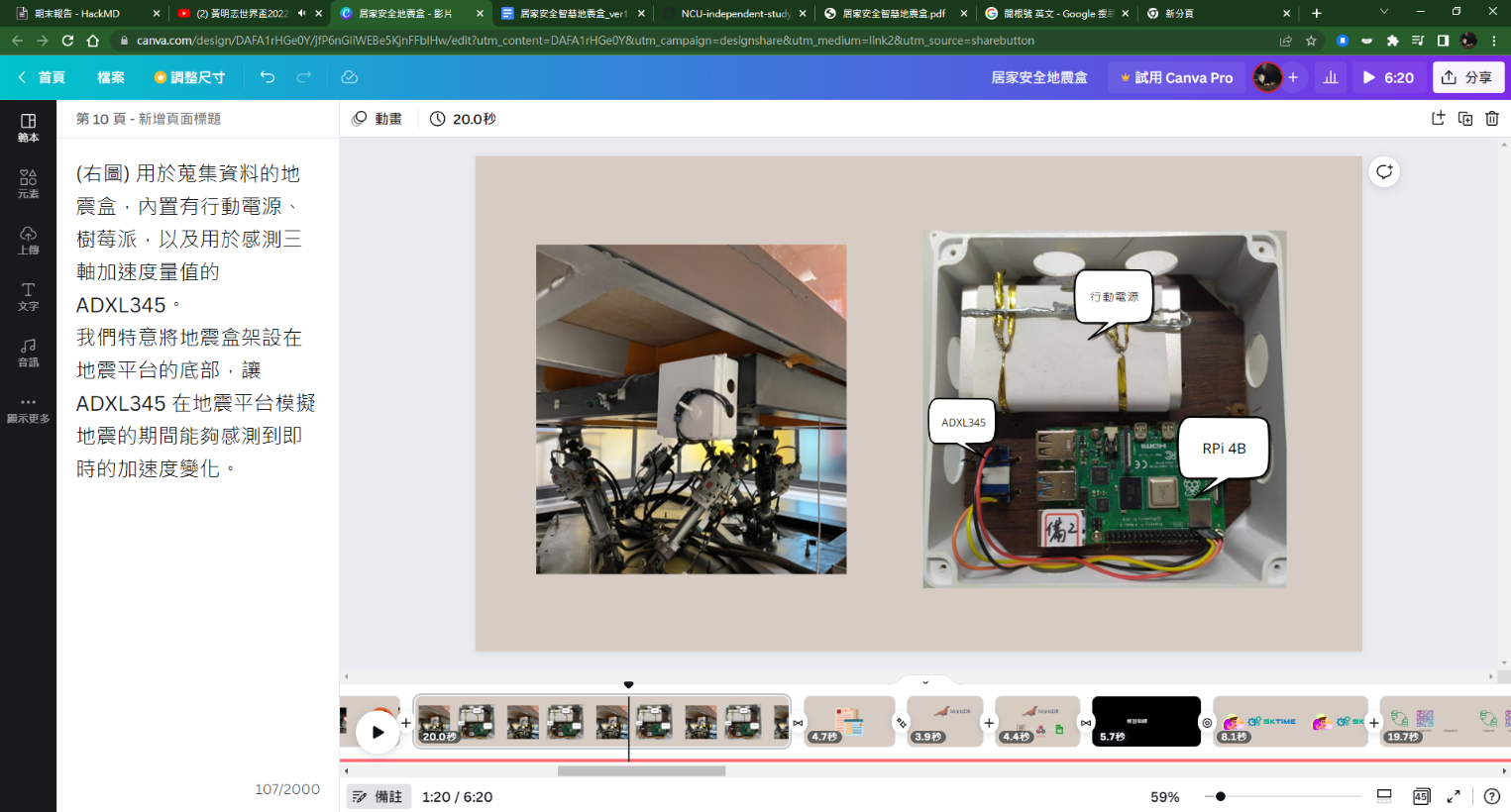
[5. 附錄 14](#_Toc106110583)

1. **整體架構**

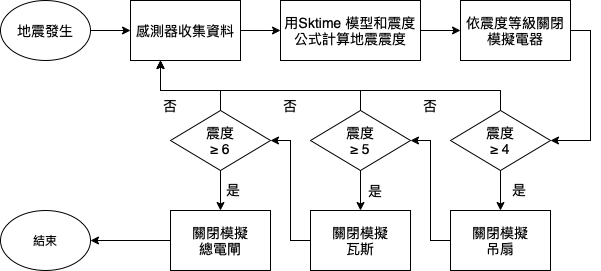
本專題主要分為蒐集資料的地震盒和模型屋(傳輸端，感測端和控制端)。



模型屋圖



蒐集資料的地震盒內部圖



流程圖

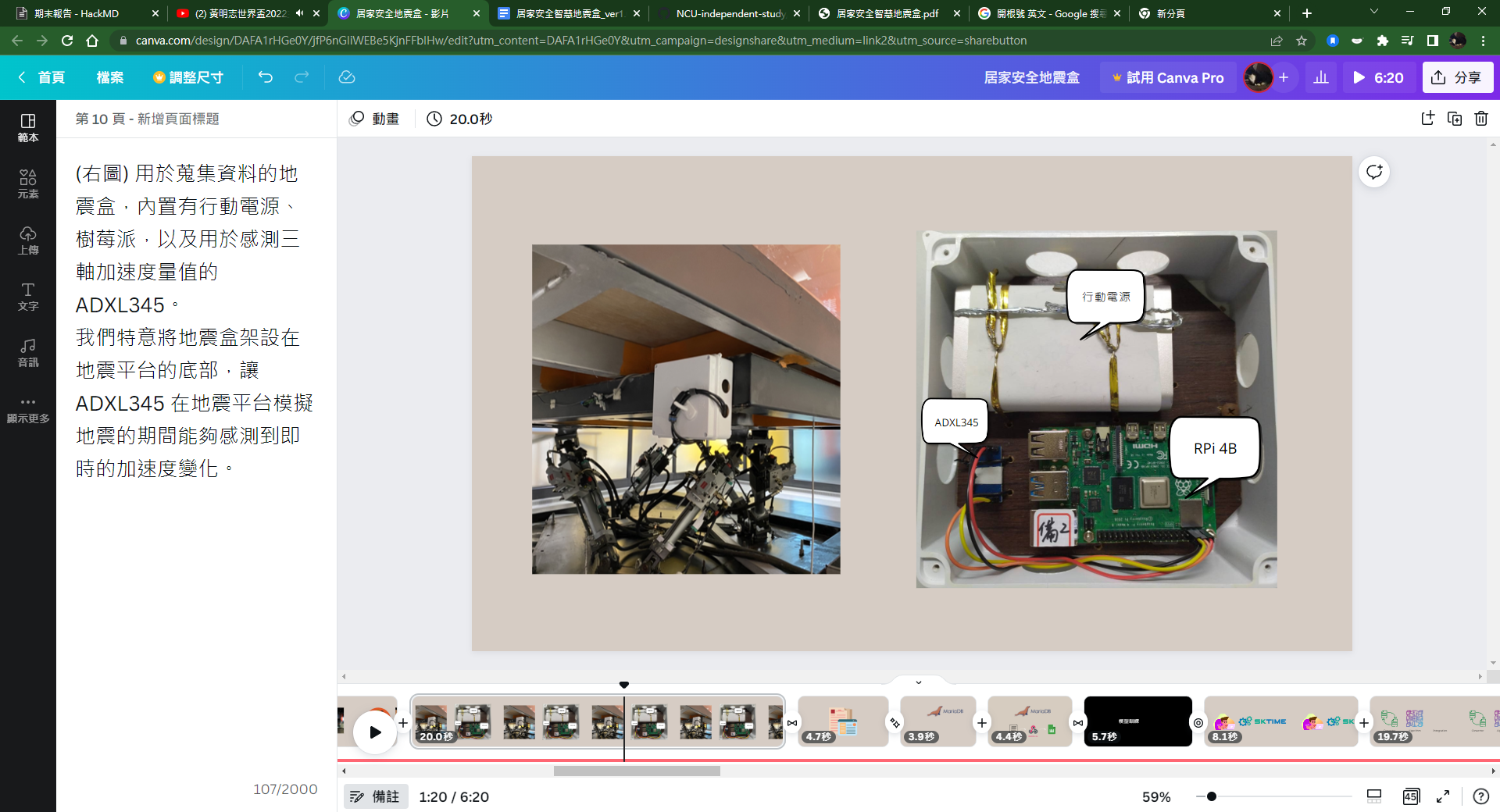
主要先以蒐集資料的地震盒，在桃園防災教育館，透過ADXL345收集x,y,z三軸資料，偵測到的加速度資料，我們透過兩種方式進行存儲，第一種是透過 MariaDB 儲存在樹莓派，另一種則是透過 Webhook 將資料儲存至雲端硬碟。爾後將ADXL345 所蒐集到三維方向上的加速度，計算出各個時間點上加速度的相對量值，並透過Univariate Time Series的形式進行模型訓練。將訓練完成的機器學習套用到感測端，當感測端偵測到地震發生，偵測到的加速度資料，由機器學習判斷出當前地震震級，再由傳輸端無線傳輸(nRF24lL01+)至控制端，控制端再以nRF24lL01+ 接收傳輸端的訊息，當接收到特定地震級數時，便可利用繼電器來關閉對應的模擬電器。總而言之，大致運作的情形可參照上方之流程圖。

1. **蒐集資料的地震盒(收集地震資料，以訓練機器學習)**
   1. **地震盒**

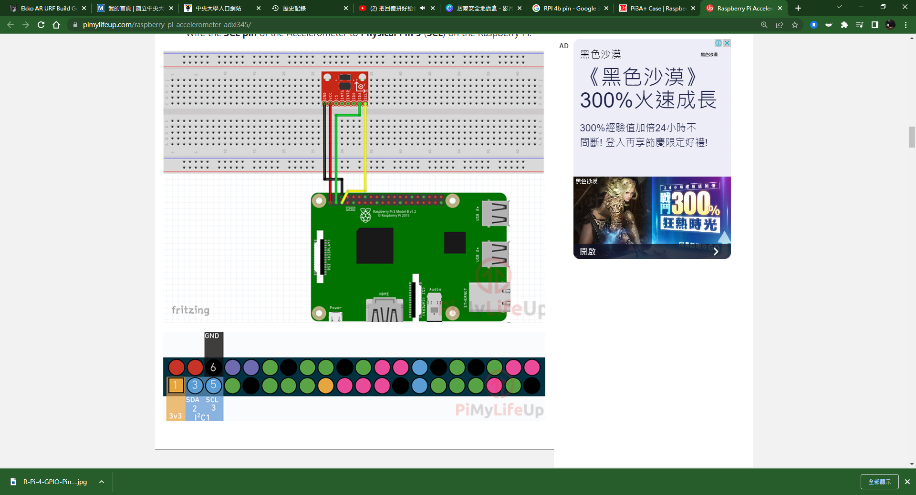
* 詳盡程式碼：

[NCU-independent-study/Final.py at main · JSHT/NCU-independent-study (github.com)](https://github.com/JSHT/NCU-independent-study/blob/main/3.model-using/Final.py)

* 硬體：ADXL345、Raspberry Pi 4B、行動電源



蒐集資料的地震盒內部圖



ADXL345和Raspberry Pi 4B 接線圖

* ADXL345設定：

import busio …(1)

import adafruit\_adxl34x…(2)

i2c = busio.I2C(board.SCL, board.SDA)…(3)

(1) 導入Adafruit的 busio 模塊。這個模塊包含各種不同的庫來處理各種串行協議。使用該模塊庫來處理I2C串行協議。

(2) 現在導入 adafruit\_adxl34x 庫。它包含了從ADXL345加速度計中讀取信息所需的所有代碼，並讓操作變得非常簡單。  
(3) 利用 busio 庫為當前板子的SCL和SDA引腳準備一個I2C連接。將其存儲到 i2c 變量中。

* 程式運作:

while True:

accV = []

for i in range(19):

# using adxl345 to get acceleration information

[x, y, z] = accelerometer.acceleration

tmpX = (x-xpre)\*100………………………………………. (5)

tmpY = (y-ypre)\*100………………………………………. (6)

tmpZ = (z-zpre)\*100……………………………………….. (7)

acc = sqrt((tmpX)\*\*2+(tmpY)\*\*2+(tmpZ)\*\*2)………….... (8)

accV.append(acc)

[xpre, ypre, zpre] = [x, y, z]

print([x, y, z])

time.sleep(0.1)

print('prediction')

seismic = model.predict(pd.DataFrame([[pd.Series(accV)]]))[0]

每兩秒會判別一次地震。

(5)-(7) 的100是為單位換算

(8) 加速度量值公式(單位cm/(s^2))公式參考附錄[2]

* 1. **資料庫**
* 詳盡程式碼：

[NCU-independent-study/1.data-collecting at main · JSHT/NCU-independent-study (github.com)](https://github.com/JSHT/NCU-independent-study/tree/main/1.data-collecting)

* MariaDB(本地)設定:

安裝MariaDB到樹莓派上 - 附錄[5]

安裝 MySQL Connector 模組

開啟 Windows 中的命令提示自元，使用pip 安裝 Python 的 MySQL Connector 模組：

pip install mysql-connector-python

* 程式運作:

try:

# 連接 MySQL/MariaDB 資料庫

connection = mysql.connector.connect(

host='localhost', # 主機名稱

database='data7', # 資料庫名稱1

user='pi', # 帳號

password='') # 密碼

# data 1 ~ 10

if connection.is\_connected():

# 顯示資料庫版本

db\_Info = connection.get\_server\_info()

print("資料庫版本：", db\_Info)

sql = "INSERT INTO data (x, y, z) VALUES (%s, %s, %s);"

cursor = connection.cursor()

print("start")

while(1):

# 新增資料

new\_data = accelerometer.acceleration

cursor.execute(sql, new\_data)

# 確認資料有存入資料庫

connection.commit()

# print(new\_data)

* Webhook(雲端)設定:

在本地建立一台 localhost 伺服器來接收訊息

$ Forwarding http://3b2ac1ae.ngrok.io -> 127.0.0.1:4567

(若接收位置使用 Let's Encrypt 的免費憑證，有可能發生傳輸失敗的狀況)

* 程式運作:

while True:

xv = []

yv = []

zv = []

for i in range(20):

[x, y, z] = accelerometer.acceleration

xv.append(x) #append(在列表末尾添加新的對象)

yv.append(y)

zv.append(z)

time.sleep(0.1)

#每兩秒收集一次資料

url = ""

#整合webhook URL

* 1. **機器學習**
* 詳盡程式碼：

[NCU-independent-study/2.time-series-classification at main · JSHT/NCU-independent-study (github.com)](https://github.com/JSHT/NCU-independent-study/tree/main/2.time-series-classification)

* 完整原理與教學:

**國立台灣大學電機工程學系 李宏毅副教授的教學影片**- 附錄[3]

**國立台灣大學**資訊工程學系 林軒田教授的教學影片- 附錄[4]

* 資料前處理:

透過對 ADXL345 所蒐集到三維方向上的加速度，計算出各個時間點上加速度的相對量值Acc(relative)(附錄2)

* 特徵提取:

Random Interval Feature Extractor：將一段時間序列隨機切成多個時間片段並從中提取出特徵，例如最大值 (Max)，最小值 (Min)，數值變化幅度。

tsFresh Feature Extractor：基於 transformer 架構出的模型，透過深度學習的方式提取時間序 列中的各項基本特徵，例如峰值數量 (Number Peaks)，平均值 (Mean)甚至是更為複雜的特徵 ，例如時間反轉對稱統計量 (time reversal symmetry statistic)。

* 時間序列分類:

時間序列森林分類器 (Time Series Forest Classifier)：基於時間間隔特徵樣式的分類模型 ，藉由分析各時間片段中包含的資訊，找出較為顯著的特徵並進行分類。

透過對時間序列數據 進行特徵提取，並將提取出的特徵傳入時間序列分類器進行訓練

* 1. **感測端:(搭配SKTIME模型)**
* 詳盡程式碼：

[NCU-independent-study/modelwithML at main · JSHT/NCU-independent-study (github.com)](https://github.com/JSHT/NCU-independent-study/blob/main/3.model-using/modelwithML)

* 硬體：ADXL345，Raspberry Pi 4B
* 用途：ADXL345 取得地震時的加速度值，再透過機器學習模型判斷出對應震度。
* 完整code中:
  + Sktime是一個專門用於處理時間序列問題的 機器學習套件，與 scikit-learn 兼容，其特點為：

a. 擁有針對時間序列問題最先進的演算法

b. 擁有各式各樣的時間序列特徵轉換器

c. 可以整合各類不同的演算法，進行機器學習模型的訓練

這些特點可以有效解決各種時間序列問題。

* + Pandas為一種Python-based data的分析工具包

import pandas as pd 即導入完成

* + Joblib是一組在Python中提供輕量級管道的工具
  + 導入time series random forest model+model = joblib.load('modelwithML')

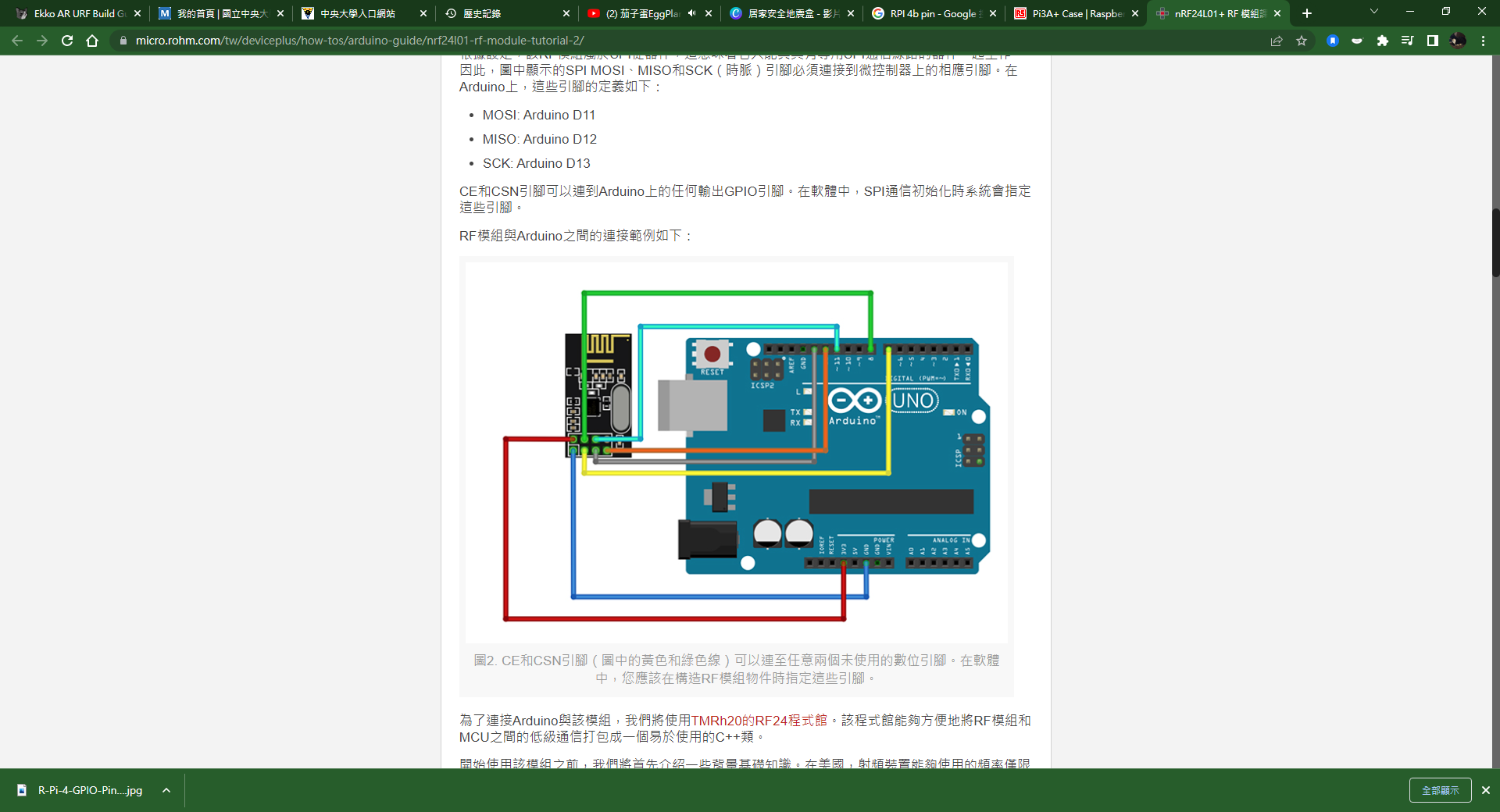
對該模型有更多的興趣可參考附錄[7]

* 1. **傳輸端**
* 硬體：nRF24lL01+，Arduino uno，USB2.0傳輸線(透過USB2.0傳輸線，連接感測端與Arduino uno)
* 詳盡程式碼：

[NCU-independent-study/4.arduino-wireless-data-transfer at main · JSHT/NCU-independent-study (github.com)](https://github.com/JSHT/NCU-independent-study/tree/main/4.arduino-wireless-data-transfer)

* 用途: 將震度資訊透過無線傳輸的方式傳到控制端





nRF24lL01+ 和Arduino uno接線如圖

無線傳輸程式主要分為transferData和receiveData:

**transferData**

#nRF24L01+ 接線:

RF24 radio(7, 8);

byte address[6] = "1Node";

#用nRF24lL01+傳輸資料給其他arduino:

void loop() {

radio.stopListening();

if(Serial.available()){

char text = Serial.read();

radio.write(&text, sizeof(text));

**receiveData:**

#nRF24L01+ 接線:

RF24 radio(7, 8);

byte address[6] = "1Node";

#用nRF24lL01+從其他arduino接收資料

while(radio.available()){radio.read(&text, sizeof(text));}

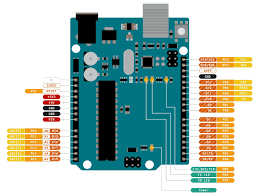
Serial.println(text[0]);

int earthquake = text[0] - '0';

* 1. **控制端**
* 硬體：nRF24lL01+，Arduino uno，繼電器，LED紅燈和綠燈，模組化風扇(樹莓派不能直接控制 5V )
* 詳盡程式碼：

[NCU-independent-study/4.arduino-wireless-data-transfer/transferData at main · JSHT/NCU-independent-study (github.com)](https://github.com/JSHT/NCU-independent-study/tree/main/4.arduino-wireless-data-transfer/transferData)

* 用途:以nRF24lL01+接收傳輸端的訊息，當接收到特定地震級數時，便可利用繼電器來關閉模擬電器。



a. nRF24lL01+ 和Arduino uno 接線同傳輸端。

b. 風扇: INA接數位腳位9，INB接數位腳位10。

c. 紅燈:接數位腳位5。

d. 綠燈:接數位腳位2。

* 程式運作:

#風扇: INA接數位腳位9，INB接數位腳位10

int INA = 9;

int INB = 10;

#紅燈:接數位腳位5:

int redLEDPin = 5;

#綠燈:接數位腳位2:

int greenLEDPin = 2;

#由震度控制關閉相對應模擬電器

if(earthquake >= 4){

digitalWrite(INA, LOW);

Serial.println("close the Fan");

}

if(earthquake >= 5){

digitalWrite(redLEDPin, HIGH);

Serial.println("close Red LED");

}

if(earthquake >= 6){

digitalWrite(greenLEDPin, HIGH);

Serial.println("close Green LED");

}

}

1. **模型屋**

* 模型屋:

迷你屋DIY小屋:

https://reurl.cc/q5aZVn

* 壓克力板:

https://reurl.cc/zZOrDa

30\*20x2

30\*30x2

30\*40x2

1. **使用版本**

python: 3.7.9

樹莓派: Raspberry Pi OS (64-bit)

套件:

RF24 1.4.2

sktime: 0.9.0

scikit-learn: 0.24.2

MariaDB 1.0.9

adafruit-circuitpython-adxl34x 1.12.2

numpy: 1.19.3

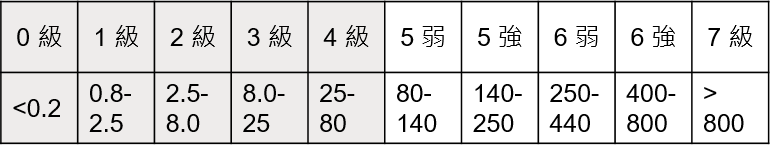
pandas 1.1.5

joblib: 1.1.0

1. **附錄**

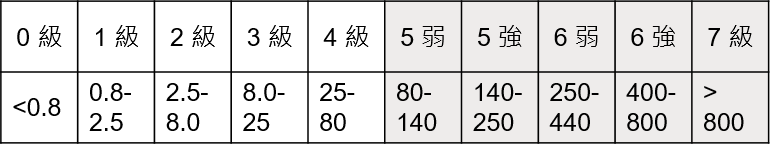
[1]公式：

最大地表加速度



公式：

最大地表速度



(2) 加速度量值公式:

[2]

公式參考: https://makerpro.cc/2018/03/low\_power\_radio\_earthquake\_sensor/

[3] https://www.youtube.com/watch?v=Ye018rCVvOo&list=PLJV\_el3uVTsMhtt7\_Y6sgTHGHp1Vb2P2J&ab\_channel=Hung-yiLee

[4]

https://zh-tw.coursera.org/learn/ntumlone-mathematicalfoundations

[5]

https://raspberrytips.com/install-mariadb-raspberry-pi/

[6]

https://stripe.com/docs/webhooks

[7]

https://reurl.cc/longGq

https://reurl.cc/Lme4Y4