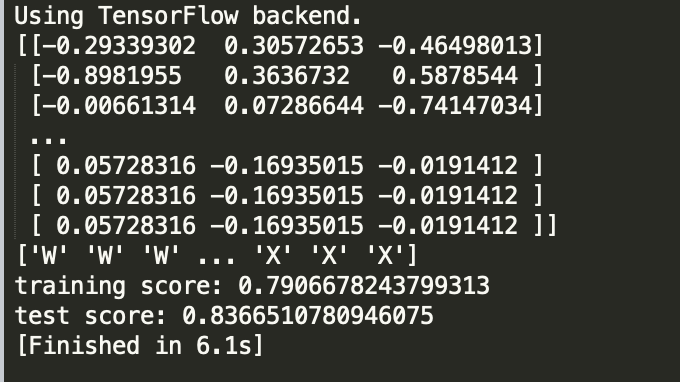
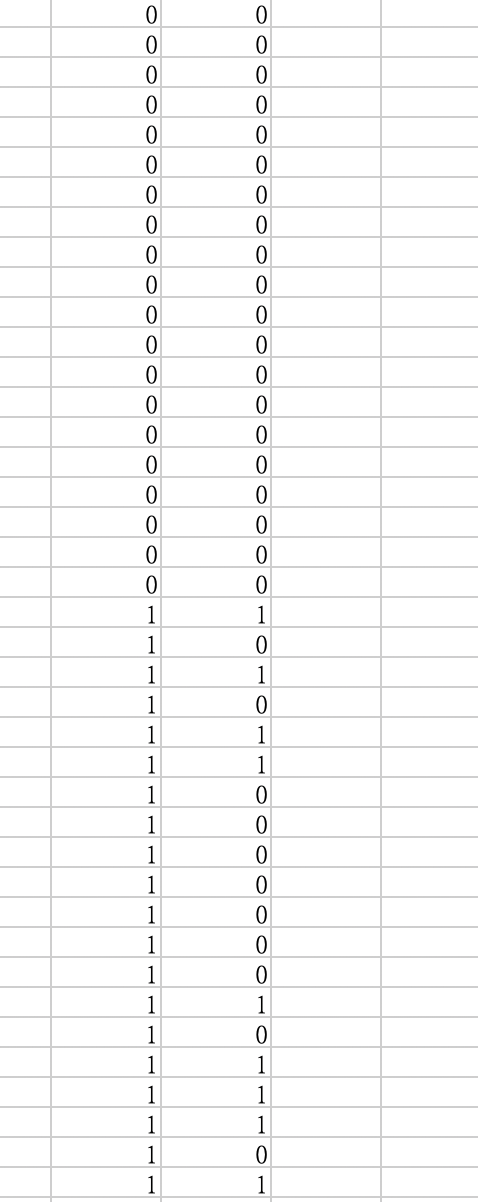
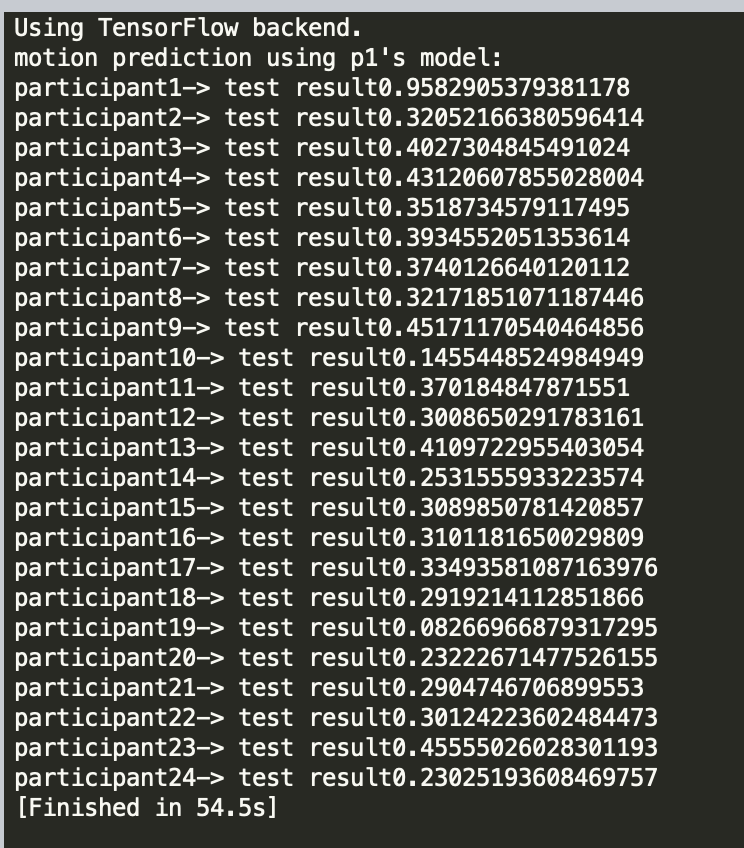
B10601002 廖品捷 github:

Part C:

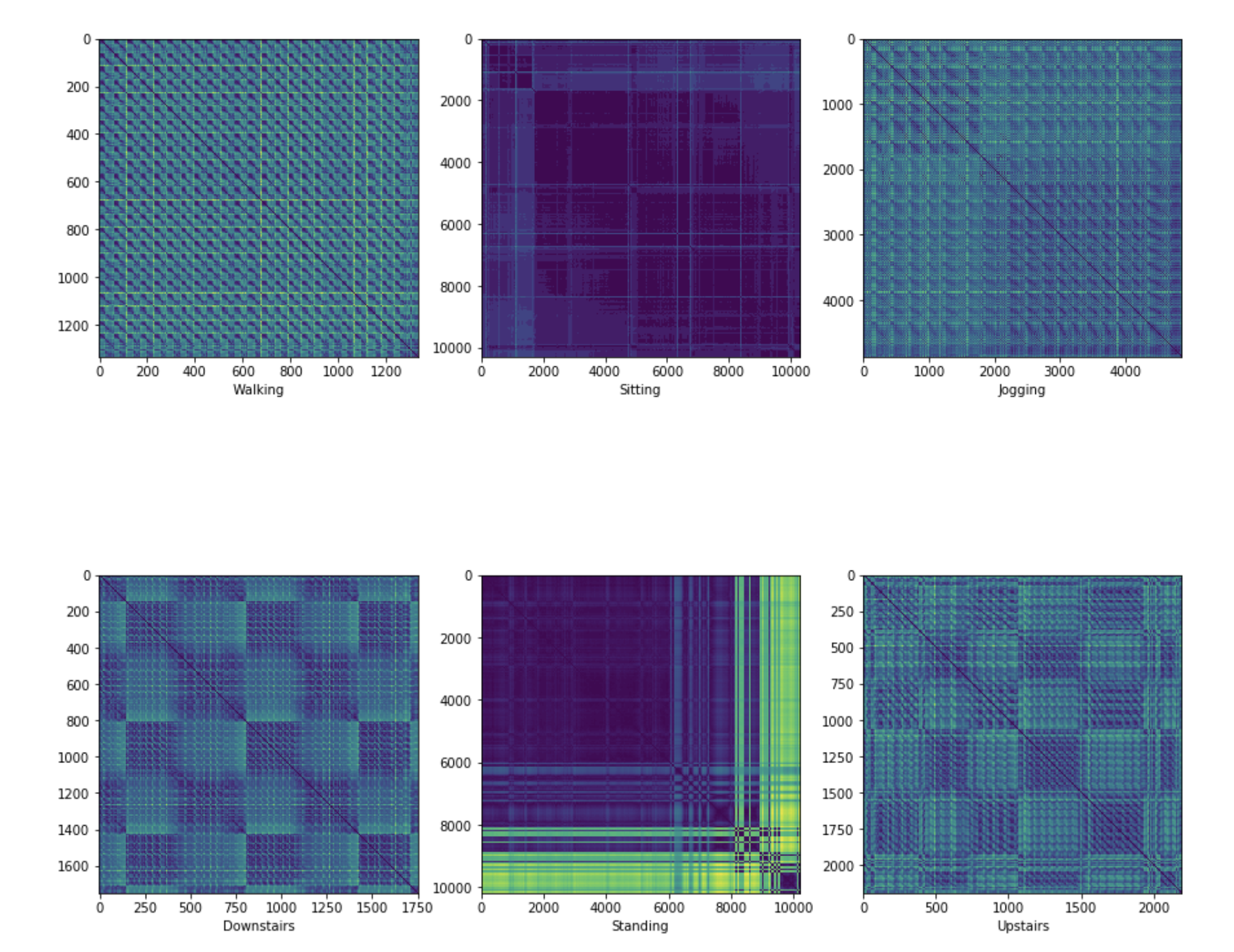
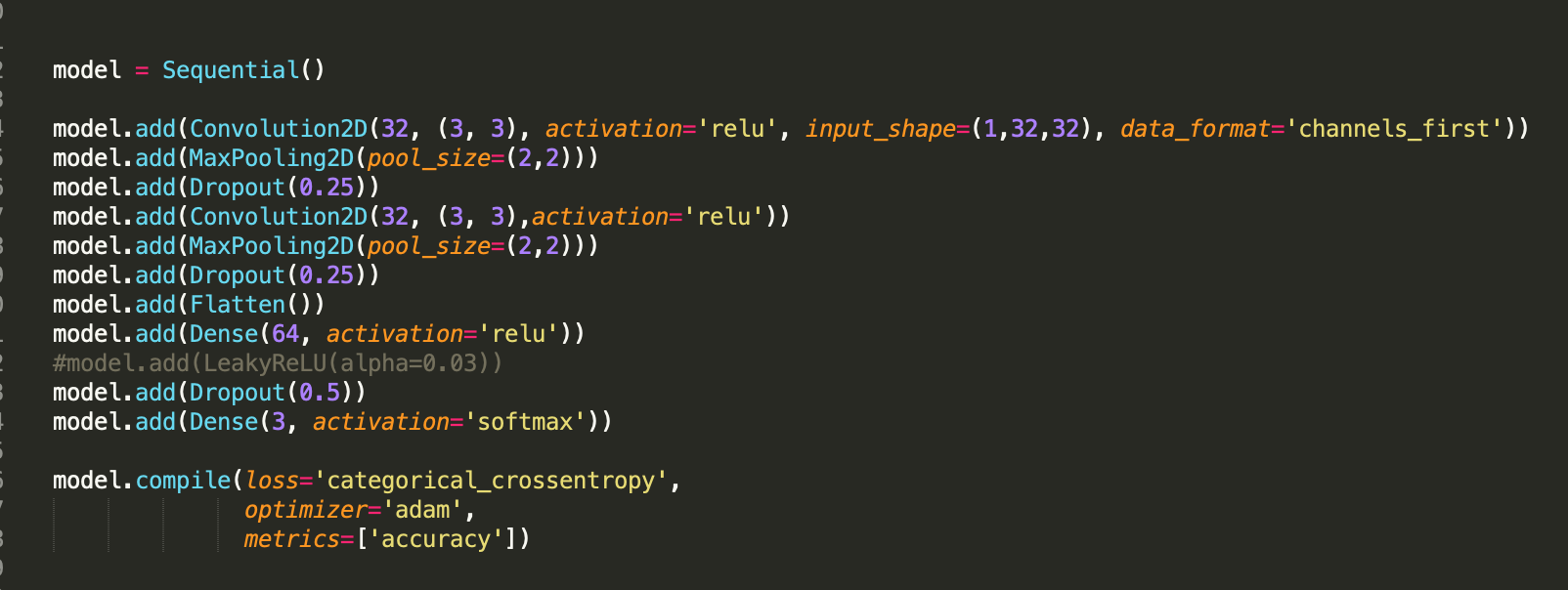
(a)上次用sliding window搭配knn進行分析（window\_size = 3），預測出來最好的情況大概在73%左右，如果訓練資料雜訊太多會造成test accuracy更低，重點是要跑很久，mac感覺要爆炸了，這次直接使用classifier(extreme gradient boosting)直接對資料進行分，結果成功率進步不少（code放在github (a)資料夾裡）。我是將所有資料(train)存起來餵給classifier建一個model，然後再把新的motion sensor data丟給model predict，用.score查看正確率後print出一個pred\_outptut檔來對照預測結果與實際motion（已經過labelencoder）

(b)(c)一開始我用sub1的所有motion的資料建了一個model(用的是xgboost的classifier, 將其切成test 0.1跟 train 0.9，因為資料乾淨結果相當高，都在90%上下)，將model存在xgb\_model.sav裡接著再去對其他使用者進行cross-people study但結果不盡理想，在分析sub1的動態時準確率相當高，但其他人的準確率大概只在30-40％左右，經過觀察後，發現應該是和測試者的身高體種和走路習慣有關。



為了提高cross\_people的準確性，我參考了kaggle kermel的做法。實作方式是透過cnn的方式將讀入資料轉成圖片餵給model，透過[3\*3]sliding window卷積取得特徵值和pooling等步驟後進行預測，結果正確率相當高。雖然這個做法跟我的作法很像，只是train的data是全部測試者的資料，但也讓我發現原來cnn不只能處理圖形，也可以拿來處理連續時間資料！

將data轉為圖片(sub1)

參考程式中的recurrent\_plot

CNN model

reference:\

<https://www.kaggle.com/tigurius/recuplots-and-cnns-for-time-series-classification>