**Basic:**

1. 實作:

我這次是用tensorflow來執行CNN的部分。

我一開始用很簡單的結構，兩層conv2D跟兩層maxpooling交錯，conv的kernel size為3\*3,maxpooling的size為2\*2，最後接flatten及dense，訓練完之後將validation data丟進去測試，出來的結果用f1\_score評分的結果只有0.55。

所以我不斷調整我的model,嘗試不同的conv和maxpool的層數。在過程中發現: conv的kernel size為3\*3 而maxpooling的size為2\*2時，model通常有比較好的表現。最後我的model總共有8層conv，3層maxpooling，結構大致上是這樣:

2\*conv + maxpooling + 2\*conv + maxpooling + 4\*conv + maxpooling，另外在每一層maxpooling前，我都有加一層BatchNormalization，目的為了避免overfitting，也可以使程式跑得更快。在flatten()之後，我有加一層dropout(0.4)，目的是為了要減少參數。而全部的activation function除了最後一層flatten之外，都是用relu。因為是要做binary classifier，所以我最後一層flatten是用sigmoid。 至於compile時的參數，我optimizer選Adam，learning約為0.001，而loss function選'binary\_crossentropy'。

以上的結構可以使我的f1\_score達到0.7左右，比一開始增加許多。

1. 遇到的問題:

(1). 我一開始也有去參考一些有名的CNN model，如VGG, ResNet等，並模仿他們大致上的結構(conv, maxpool, drop數目與順序)，導致我最後的model參數變超多，但是最後的結果也沒有比較好。

(2). 要注意data的陣列形狀，因為傳入model有要求特定的形狀，且這一次的data是grayscale，所以channel要設為1。

**Bonus:**

我的做法是將X\_train丟入basic完成的model，得到的結果稱為X\_train\_bonus。再將X\_train\_bonus與age跟gender接再一起，得到10000\*3的陣列，也就是bonus part的train data。

至於model的選擇，我是選擇使用老師教過的random forest去實作，因為decision tree可能會有深度過深導致overfitting的問題，所以用random forest可以解決這個問題。

利用random forest我的f1\_score最高可以達到0.72的準確率，跟basic部分相比，雖然多了兩筆資料，但提升的程度並不是很多。我推測可能以下原因:因為bonus的training data有部分是透過basic part的預測結果所組成，有可能是因為basic part的預測結果只有約7成的準確度，導致bonus一開始拿來訓練的資料就是不對的，因此最後的結果也會受到影響。

**Advanced:**

1. 實作:

我在這個部分的做法是利用ResNet50，從tensorflow.keras. applications.resnet50 import ResNet50，並直接在model最上層加上Resnet50，之後直接加上flatten及dense，因為是multi-label，所以最後的dense的unit我設為7。而最後一層的activation function我是用sigmoid。

我一開始最後一層的activation適用softmax，但是最後的結果卻不盡理想，預測結果中的七個分類往往只有一個的機率是高的，其他六個的機率都很低，我後來想到我們要做的事”multi-label”而不是”multi-class”，結果可能有很多label， 而每個label 都具有各自獨立的機率值，所以要用sigmoid。

Compile的部分optimizer選Adam， 而loss function的部分，因為是multi-label而不是multi-class，所以我一樣選'binary\_crossentropy'。最後執行出來的結果，利用weighted f1-score評分，可以達到0.72。

1. 實作過程與遇到的問題

(1) 我一開始的做法是用basic part的結構，”分別”訓練七個label，也就是說我有7個model，訓練完成後，再將預測結果合在一起。將結果丟入weighted f1-score，得到0.63。

(2).我其實還有試過用basic part的結構，最後的dense的unit我設為7(意思就是我只用一個model)，出來的結果是0.61，比分開訓練還要來的低。

(3).因為我是用colab去實作這次的作業，但因為colab有限制GPU的使用狀況，所以用12個小時之後，就暫時沒有GPU可以用了==，特別是跑Resnet的部分，沒有GPU的話根本跑不動。而且只要一不小心中斷，全部的東西就要重跑一次。

(3).使用Resnet時，tensorflow有限制傳入的train data要是三維的，我一開始一直傳灰階的影像進去，導致一直報錯。我的解決辦法是將一張圖片重複疊三次，以滿足他的要求。