國立成功大學工程科學系

數值方法期末專題

Department of Engineering Science

National Cheng Kung University of Engineering

Practicum of Engineering Science

使用深度學習技術分析學生用餐習慣並實作預測系統

學生：梁哲嘉

指導教授：游濟華 博士

中華民國 111 年 6 月

目 錄

第一章 緒論

第一節 研究動機

一、研究動機

這次的專題報告我所選擇做的主題是深度學習，當初會選擇做這個主題的原因是因為我對於機器學習這塊領域感到很好奇。其實在二年級修習資料結構的時候，教授就有講授了一些 Machine Learning 的知識，而在今年的寒假，我也稍微利用了一些空閒時間，在 YouTube 上觀看了一些李宏毅教授所上傳的開放式課程，從此對於機器學習有了初步的認識，但由於只是看上課影片的關係，還沒有自己實作的經驗，所以在學期初公布課程大綱的時候我感到很驚訝，沒想到居然有機會能夠在這堂課學習到機器學習的相關知識，並實作出自己的作品!

而要利用深度學習去解決什麼樣的問題呢?其實當初我思考了很久，也花了很多時間找尋我覺得分析起來感覺會很有意思的數據，一開始我所選擇的題目是對於手機遊戲的抽卡結果，去預測抽中限定角色的次數，但分析完的結果並不是非常理想，因此只好再另尋題目。後來我在 Kaggle 這個網站上找到了我覺得蠻有趣的數據，是針對美國高中生的相關資料與成績進行的調查，而其中有一項資料我覺得很特別，那就是調查學生有沒有正常在吃午餐這項，因為我在高中時期，時常因為中午很想睡覺，或是由其他活動就選擇不吃午餐，也常常被家人念說三餐要正常才有精神好好動腦，所以就決定使用這份數據，觀察看看能不能從學生的個人資料與成績，去找尋這之間是否有關聯存在。

二、原理性質

深度學習是由機器學習再細分出來的一個領域，它的運作原理主要是用電腦去模擬我們人類大腦的神經元運作，來進行資料的分析。而在架構上面，一般會從輸入層開始，如神經元一般，連接到下一層的神經元，而此時這層的神經元所接收到的數據，是透過 activation function 將上一層的輸入進行加總並轉換而得來

的。如此重複這樣的運作，最終輸出結果的那層神經元即稱為輸出層，而夾在輸入層與輸出層之間的即為隱藏層。

再來是我們要如何利用上面所說的架構去進行訓練，一般我們會使用梯度下降法來設法尋找能夠最佳化學習結果的權重參數，每次訓練都會將參數沿著梯度下降的方向往下走一點，經過反覆的訓練之後，就能找到較符合我們所預期的結果的參數。

第二節 研究目的

根據以上的研究動機及原理，即是想要利用深度神經網路這項技術，去分析 1,000 名美國高中生的相關資料，如:性別、種族、父母學歷、複習習慣，以及各項考試成績，來觀察並預測學生平時的用餐習慣，及其數據之間是否具有一定的關聯性，期許能夠分析出有用的結果。

第三節 研究流程

這次的實驗主要是利用 Python 語言來編寫程式，利用 Keras 來設計與建立深度學習的模型。首先先將學生相關數據的 csv 檔匯入程式，再根據輸入的資料進行資料預處理，將數據切割以 8:2 的比例分割成訓練集與資料集後，放入我所設計的模型進行訓練。

本次實驗將會設定模型的參數值，設置 batch 值為 10、20、50，epoch 值設為 100、200、300 次。然後利用程式去訓練模型並預測結果，觀察 batch 值與 epoch 值對於訓練出來的準確度與 total loss 是否有影響。並將訓練出來的模型，利用 Tkinter 設計圖形化界面，實作出能夠供人使用的預測系統。使用者將能夠將自身的資料填入系統，而系統會當場訓練出模型，並提供預測結果供使用者查看。

第二章 研究方法

一、資料預處理程式碼實作

```
In [1]: import numpy as np
        import pandas as pd
        df = pd.read_csv('StudentsPerformance.csv')
        df = df[['gender','race/ethnicity','parental level of education','test preparation course','math score','reading score','w
        data = np.array(df)
```

圖一：

引入函式庫，將 csv 檔匯入，這邊為了之後設 label 方便，先把要做為 label 的值調換到最後一行。將 data frame 轉換為 ndarray 的形式保存起來。

```
In [2]: def change_value_to_number(array):

            for i in range(len(array)):
                if array[i][0] == 'female':
                    array[i][0] = 0
                elif array[i][0] == 'male':
                    array[i][0] = 1

            for i in range(len(array)):
                if array[i][1] == 'group A':
                    array[i][1] = 0
                elif array[i][1] == 'group B':
                    array[i][1] = 1
                elif array[i][1] == 'group C':
                    array[i][1] = 2
                elif array[i][1] == 'group D':
                    array[i][1] = 3
                elif array[i][1] == 'group E':
                    array[i][1] = 4
```

圖二：

（註:這邊後來發現應該使用 one-hot-coding 效果會更好，後面將會檢討。）

將數據中不是以數字形式的資料轉換成以數值的方式表示，為每種資料賦予一個數值。

```
In [3]: def slice_training_value(array):
            x_train = array[0:800,:7]
            y_train = []

            y_train_list = []
            for i in range(800):
                train_label = array[i][7]
                y_train_list.append(train_label)

            y_train = np.array(y_train_list)

            return x_train, y_train
```

```
In [4]: def slice_testing_value(array):
            x_test = array[801:1000,:7]
            y_test = []

            y_test_list = []
            for i in range(801,1000,1):
                test_label= array[i][7]
                y_test_list.append(test_label)

            y_test = np.array(y_test_list)

            return x_test, y_test
```

圖三：

將前 800 筆資料切割出來，x_train 設為用來訓練的訓練集，y_train 則為訓練集的 label。

並以同樣的方式將剩下的 200 筆資料切割出來，並設為測試集及其 label。

```
In [5]: from keras.utils import np_utils

        def categorize_label(y_train, y_test):
          y_train = np_utils.to_categorical(y_train, 2)
          y_test = np_utils.to_categorical(y_test, 2)

          return y_train, y_test
```

圖四：

將 label 的資料改成以 one hot encoding 的形式表示。

```
In [6]: from keras.models import Sequential
        from keras.layers import Dense, Activation
        from keras import optimizers
        from keras import losses

        def analyzing_data(x_train, x_test, y_train, y_test):
          model = Sequential()

          model.add(Dense(units = 7, activation='relu'))
          model.add(Dense(units = 1000, activation='relu'))
          model.add(Dense(units = 1000, activation='relu'))
          model.add(Dense(units = 1000, activation='relu'))
          model.add(Dense(units = 2, activation='softmax'))

          model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
          model.fit(x_train, y_train, batch_size = 20, epochs = 200)

          result = model.evaluate(x_test, y_test)
          print('Total loss',result[0])
          print('Acc:',result[1])

          model.save('analyze.ann')
```

圖五:

資料預處理完畢後，將數據放進設計好的模型裡面，這次的實驗設計了 3 個隱藏層，而輸出層則是由 2 個神經元組成，預期能夠訓練出的效果為:一個代表有正常吃午餐的機率，另一個則為少吃或是沒吃午餐的機率。

而經過多次試驗，發覺 optimizer 使用 Adam optimizer 效果最佳，loss function 則採用分類交叉熵，batch 及 epoch 大小最終設為 20 及 200。最後將訓練完的模型保存起來。

二、圖形化介面程式碼實作:

（由於篇幅有限，故此部分程式碼將由下一部分的成果展示出來）

第三章 資料分析與結果

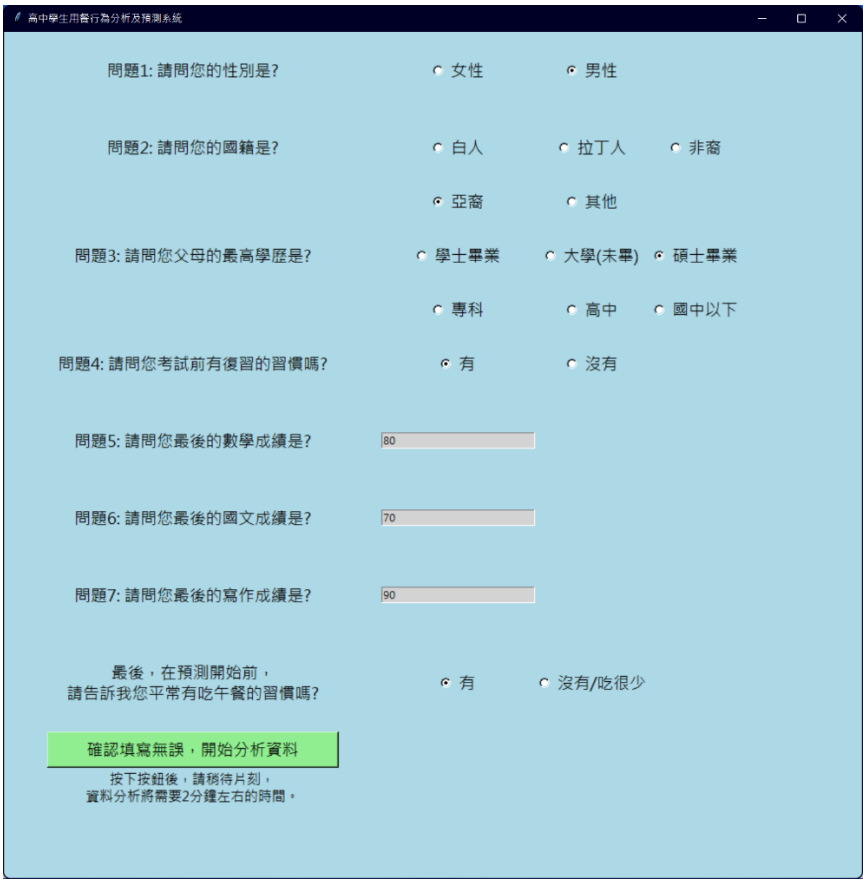
第一節 結果展示

圖六：

首先是歡迎介面，這邊會請使用者先請使用者輸入自己的名稱，按下確定鍵後會進入下一個頁面。

圖七:

輸入完畢後會進入說明介面，告訴使用者這個系統是如何運作的，並要求使用者接下來必須填入自己的相關資料，按下「我知道了!」鍵後進入下一個頁面。



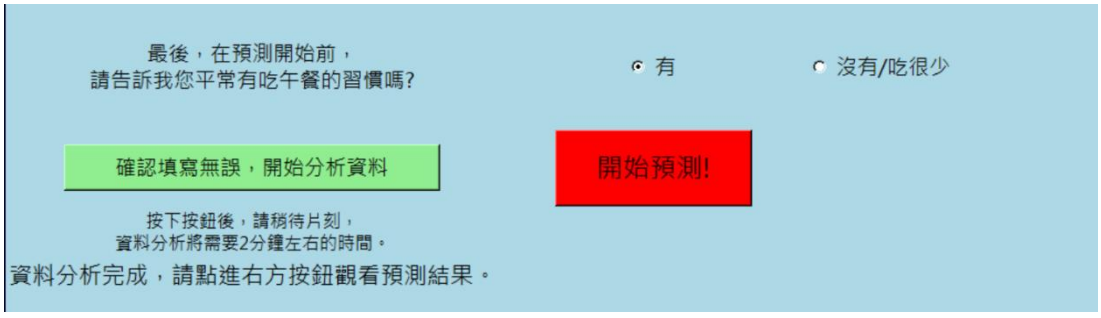
圖八:

這邊就請使用者依照問題，將答案依照提供的答案形式填入進去，這裡所使用的是 Tkinter 提供的 radiobutton 及 entry。這邊在撰寫 code 時有使用到一個驗證的小技巧，在填入成績時會先驗證使用者輸入的是數字，否則無法填入欄位裡面。

資料填寫完畢後，就按下「確認填寫無誤，開始分析資料」的按鈕，這時程式便會開始訓練模型，而訓練的過程經過實測大約需要 2 分鐘的時間，所以必須請使用者稍待片刻。

圖九：

從這邊可以看到，預測的準確率大約在 70%左右。



圖十：

此為訓練的過程，訓練完成後，視窗右邊會跳出「開始預測」的按鈕。

圖十一：

按下預測鍵後，系統會以 message box 的形式跳出提示視窗，告訴使用者預測結果以及他當初所填入的結果。

第二節 結果比較

在這次的實驗裡面，我試著調整模型的 batch 大小及 epoch 值，來觀察究竟會不會對訓練結果產生影響。

Batch 值 / Epoch 值	10	20	50
100	68.8%	66.8%	62.8%
200	60.3%	70.3%	67.8%
300	66.3%	70.1%	65.3%

表一:(上表數值為預測準確度)

從以上的結果可以發現，不論是 batch 值或是 epoch 值，都不能調得太高或是太低，否則訓練的成果反而會造成反效果。像是當 epoch 值調得太高的話，就有可能會造成 over-fitting，或是 batch 大小太大的話，會使得訓練資料的隨機性減低，使得預測結果的準確度降低。

因此最後我所設定的 batch 大小為 20，epoch 值為 200，以此作為預測系統所使用的參數。

第四章 研究心得

這次的期末專題對我來說是一個很新奇的嘗試，也是有點具挑戰性的挑戰。因為自己從來沒有利用深度學習實作過作品，對於視窗程式設計也算是只有初步認識而已，所以在撰寫 code 的時候其實遇到了不少的挫折，像是在將數據丟入模型裡時，一直產生格式不符合的 bug 導致無法進行訓練，或是在使用 one hot encoding 時一直無法成功，只好改以 label encoding 進行處理等等。

原本以為處理完訓練的部分之後就輕鬆了，沒想到在設計介面時花了更多的時間，因為想要在這次的作品中放入更多課堂上所沒提到的功能，像是:產生彈出視窗、限定輸入數字、插入圖片…等等，所以花了一些時間去摸索那些功能的使用方法，沒想到 Tkinter 所含的功能真的蠻多的呢!在寫程式的時候原本只是在猜會不會有這個功能，結果去查了之後發現還真的有。總體來說，雖然做這個作品花了我蠻多時間，常常寫到看日出，但是在此同時，我也對機器學習以及介面設計這塊有了更深的認識，算是收穫滿滿吧!謝謝教授能夠提供這個機會讓我們學習。

第五章 相關連結

1. Github 連結： **https://github.com/johnnyhitpo/school**

2. Google Drive 連結:

https://drive.google.com/drive/folders/1AWCsgvpgKrHr_VnhAL0OJbRu3AQWoq57?usp=sharing