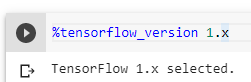
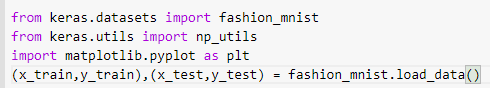
ECT HW9

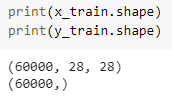
**前置處理**



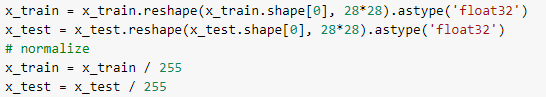
* 先把tensorflow版本指定為1.x，因為目前預設的2版有一些問題



* 把資料即從fashion\_mnist中下載下來，並放進traing、tesing的變數中



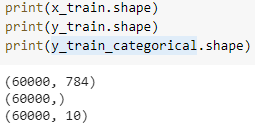
* 用Shape可以觀察出資料類型為28\*28的圖片



* 把input的28 \* 28維度的圖片，用reshape轉成1維的784，並且同除以255來當作正規化 (因為每個點的值為0~255)



* Testing data則是進行one-hot encoding，因為共有10個類別，所以除了類別的位置是1，其他的位置會用0表示，變成10維

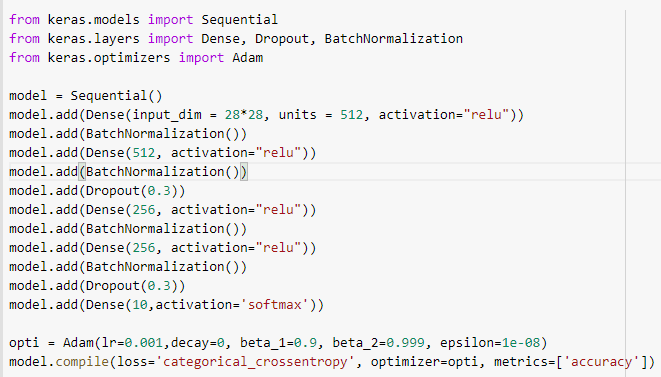


* 可以印出來檢查，input的確變成1維，output也變成10維



* CNN的input則要重新reshape成適當的形式

**(a)**



* 建立NN模型，共有5層Dense



* 建立CNN模型，共有一堆層…太多了

**(b)**

**參數意義For NN :**



* 初始化model



* Add函數用來增加新的層，Dense就是最普通的神經元
* Input\_dim: 代表input的shape，可以對比前處理時的784
* Units: 代表神經元的數量
* Activation: 代表經過Sum(wx + b)之後要用哪個函數來進行非線性激發



* 這是用來使每次神經元output之後，在input進下一層之前進行正規化



* 代表這一層訓練完之後，有0.3的output被拋棄，可以讓結果更加General，有助於提高test的準確度，但train的準確度會暫時下降



* 這一層是我的最後一層，也就是真正的output。
* 使用softmax函數可使output總合為1。他是用每一個output總合當分母，各自output當分子來運算的，因此適合用於分類問題



* 這是設定optimizer，Adam是一種可以自動調整learning rate的方式



* Compile會把上面的設定都編譯起來
* Loss就是loss funciton，metrics代表要印出的資訊



* Fit就是真的開始訓練
* 前兩個參數就是training的input、output
* Batch\_size: 代表每次要訓練幾筆資料
* Epochs: 代表要訓練幾回合
* Validation\_data: 用自己準備的data當作Validation data，若用validation\_split則會從training data中切割



* 可以看整個NN大致有哪些參數

**參數意義 For CNN:**



* 這代表CNN中filter的數量，我初始化最小的數量之後都用這個的倍數



* 初始化模型



* CNN其實跟NN一樣，只是中間的隱藏層是使用Conv2D (Convolutional layer)也就是卷積層
* Filters: 代表filter有多少個，用來找特徵
* Input\_shape: 代表input的shape長怎樣，每張圖片都是28 \* 28的pixel，每種圖片只有單1色調，因此shape = (28,28,1)
* Kernel\_size: 代表抓特徵時，要用多大的filter來抓取，我用3 X 3
* Strides: 代表每次filter滾動的步數，(1, 1)就是指滾動一步
* Activation: 就是activation function



* 這跟NN一樣，用來正規化output再成為下一層的input



* 一樣有Dropout，來拋棄部分output



* 這會縮小圖片，Max代表會抓取選定範圍內最大的值來代表它，其餘皆捨棄，我設定(2,2)就會從2 X 2的方格中，選一個最大的保留，其餘3個捨棄



* 用來攤平卷積後的結果，為了當作普通Dense的input



* Dense部分跟NN相同，不多作介紹



* Output也是跟NN相同



* 優化器和編譯部分也跟NN相同，不加以贅述



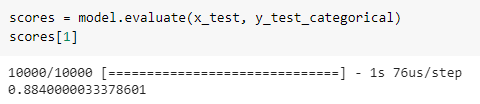
* 然後一樣用fit丟進去訓練



* 用summary可以看各種參數和各層的關係

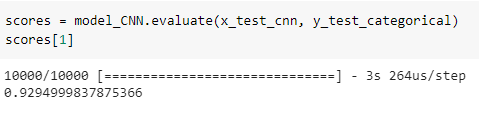
**(c)**

**For NN:**



* 使用evaluate函數，把input、output都換成testing data放進去做評估，可以發現準確度約為88.4%

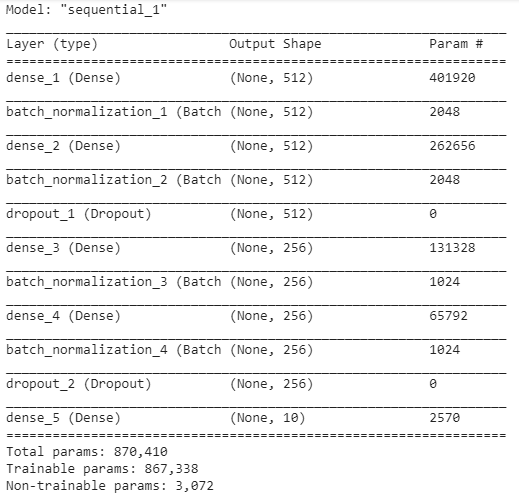
**For CNN:**



* 整體準確度上升至92.9%，比起NN進步許多

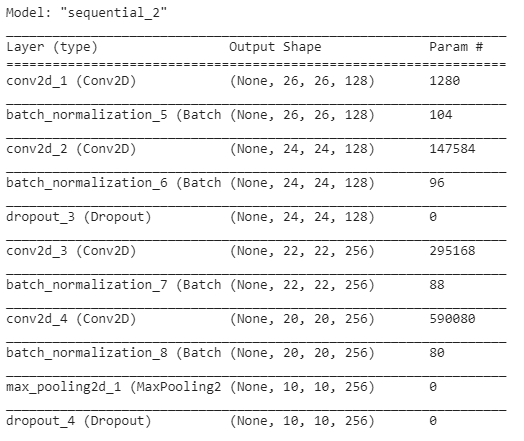
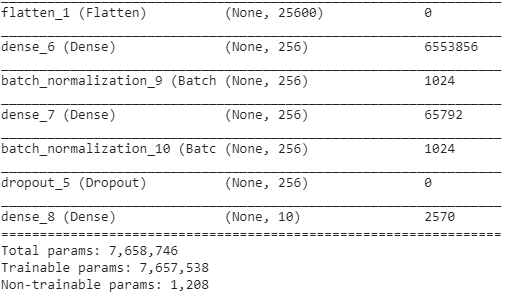
**(d)**

**For NN:**



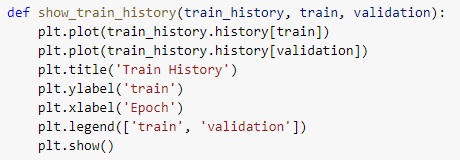
* Model: 代表它的名字
* 然後下面就是各層的資訊，總參數量、訓練參數量、為訓練參數量

**For CNN:**

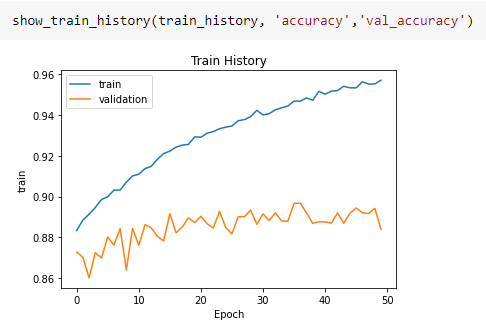
* 參數意義同NN所提，可以注意因為是第二個用Sequential創建的Model，因此名字也有變化了

**(e)**

****

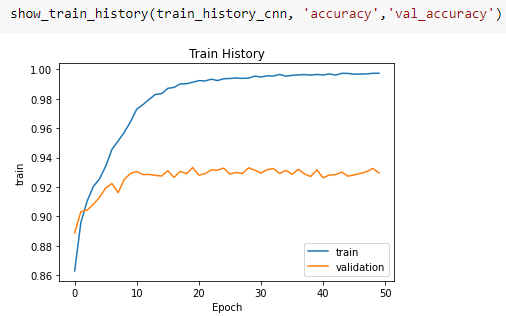
* 使用此函數還繪製圖片

**For NN:**



* 可以看到隨著Epoch上升，training的準確度也越來越高，但validation卻不久就開始浮動了。
* 這其實跟我之前提到的現象一樣，Validation不用等到50個Epoch，就已經訓練得差不多了，開始浮動了

**For CNN:**



* 在training部分比使用NN訓練得更為完整了，Validation部分準確度也更高了，但Validation部分還是很快就開始浮動了

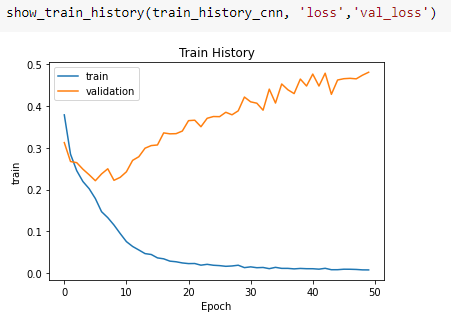
**(f)**

**For NN:**



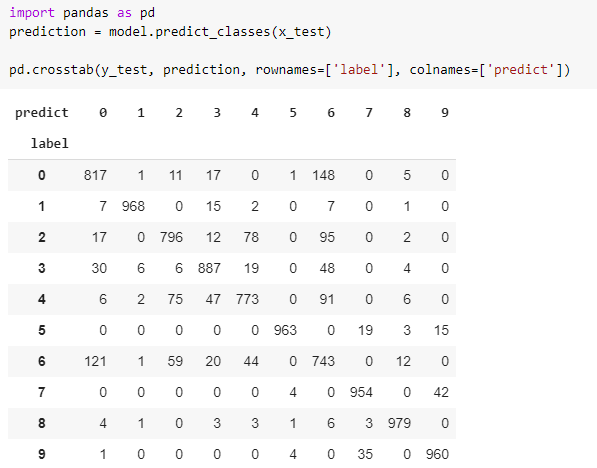
* 如圖所示Loss會隨著訓練而下降，跟Accuracy相反

**For CNN:**



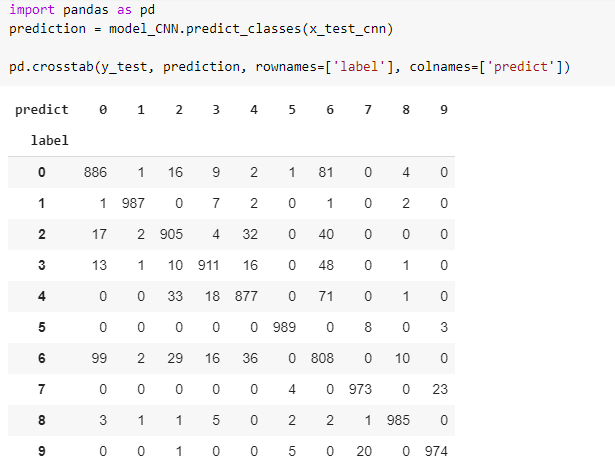
**(g)**

**For NN:**



* 混淆矩陣如上圖所示，大多在斜直線上，代表預測準確度高

**For CNN:**



* 更多數字集中在斜直線上了，預測更準確