第六組\_DM\_hw03

組員:R07725049吳姿君、R08725008周若涓、R08725010陳亦珊、R08725030徐薇尹

**一、研究動機與目的**

目前 MNIST dataset 用 Keras 來建 CNN 的數字分類模型，accuracy 通常都大於 0.9，因此，我們想測試透過 Cluster 的方式，瞭解 Supervised learning 與 Unsupervised learning 的 accuracy 差距，並探討背後的原因。

**二、資料描述**

MNIST 是 Keras 提供的手寫數字 dataset，MNIST 中的圖片是 **28 像素 x 28 像素**，裡面的數字分別有從 0 到 9，共 10 種數字，每一張圖片就可以用 28 x 28 = 784 個數字來紀錄，因此 print(x\_train.shape) 的輸出告訴我們有 **60000 張訓練圖片**，每張圖片都有 784 個數字，而 print(y\_train.shape) 的輸出告訴我們的是這 60000 張訓練圖片的真實的 label 答案；同樣的，print(x\_test.shape) 的輸出告訴我們有 **10000 張測試圖片**，每張圖片也有 784 個數字組成，對應 print(y\_test.shape) 的輸出告訴我們的是這 10000 張訓練圖片的真實的 label 答案。

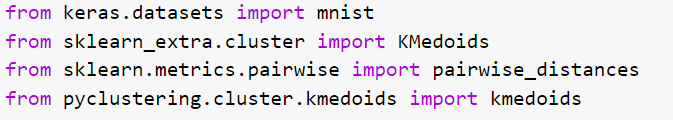
**三、實驗方法**

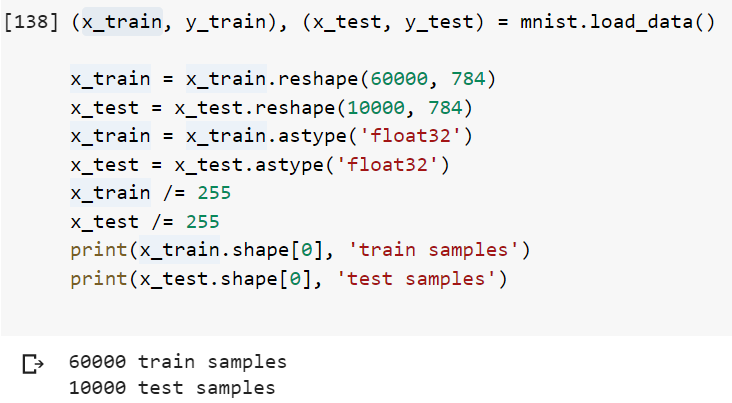
**1.0 K-Medoids**

**1.1 引入模組及資料準備**

在K-Medoids中，主要模型在 sklearn\_extra.cluster 裡面的 KMedoids，K-Medoids 的套件只支援 3.6 以上的Python版本。

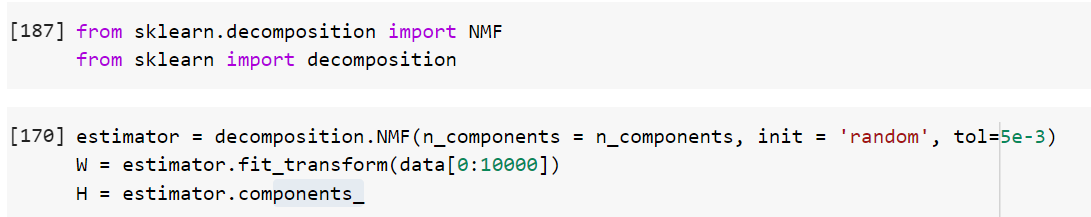
載入 MNIST dataset，並將每張圖片的 array 中的值標準化，使其介於 0 到 1 之間。

****

****

**1.2 降維前處理**

使用 NMF 做降維的處理，NMF 分解得到的 W 矩陣相當於從原始矩陣中提取的特徵，這邊只取訓練資料 60000 筆中的 10000筆，是因為大約超過 10000 筆，降維以及之後 K-Medoids 的 model 就要跑很久，因此，這邊就取訓練資料中的 10000 筆來做處理。

****

**1.3 利用 K-Medoids 建立 cluster model**

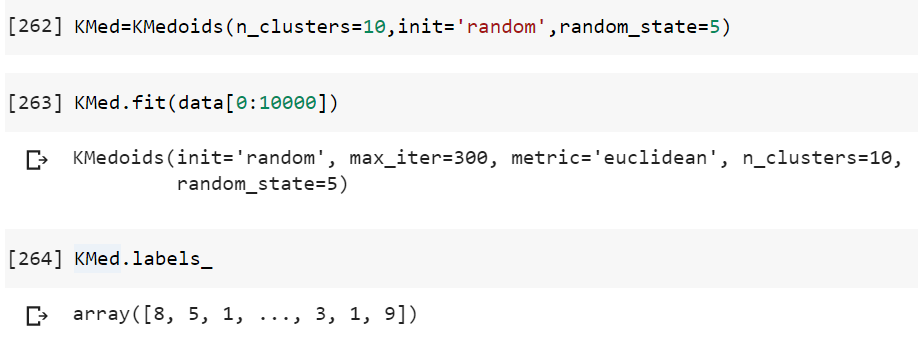
Step 1: 先決定想要把樣本分成幾類，比方說 n\_cluster=10，即將資料分為 10 類

Step 2: 預設為隨機給定 K 個中心點 ，這些中心點都為實體點，隨機選定樣本點

Step 3: 計算每個樣本與每個中心點之間的距離(預設算距離的方式為 Euclidean)，劃分與最近近的中心點為一群

Step 4: 每個群體中分別計算樣本點之間的距離，選取讓所有距離和最小的樣本點為新中心點，並以新中心點取代舊中心點

Step 5: 重複步驟3與4(預設的 max\_iter = 300)，直到中心點不再變動為止

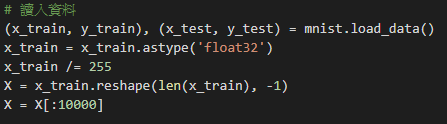
****

**2.0 Hierarchical Clustering**

**2.1 使用套件**

資料集從keras.dataset中import，存下X\_train。訓練套件使用sklearn的cluster.AgglomerativeClustering做階層分群。

**2.2 資料前處理**

****

MNIST已經是整理好的dataset，所以就只做了標準化X /= 225。因為10000筆資料沒有花太多時間訓練，所以沒有進行降維就直接拿去做training。

**2.3 model參數調整**

****

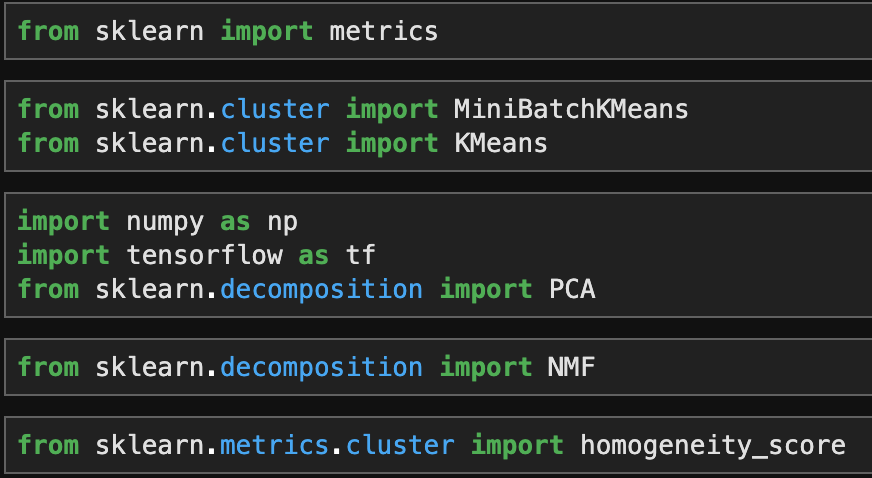
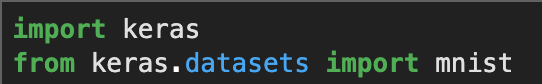
n\_clusters: 預設為2，要分成幾群。因為dataset是0~9的手寫辨識，所以理論上10類會有最好的效果。

affinitystr: linkage的計算，也是使用預設的euclidean。

linkage: 使用預設的ward，會最小化merge cluster的variance。

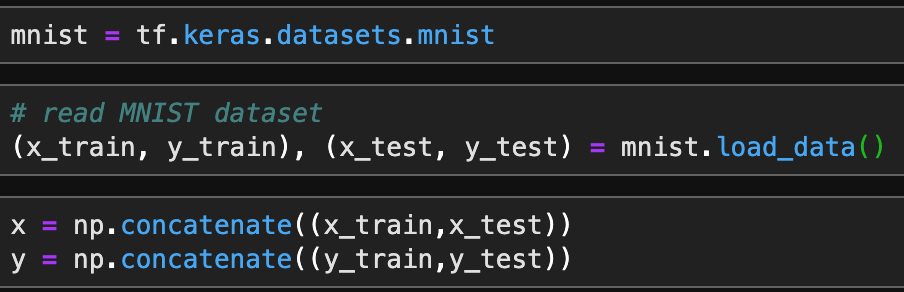
**3.0 K-Means**

**3.1 使用套件**

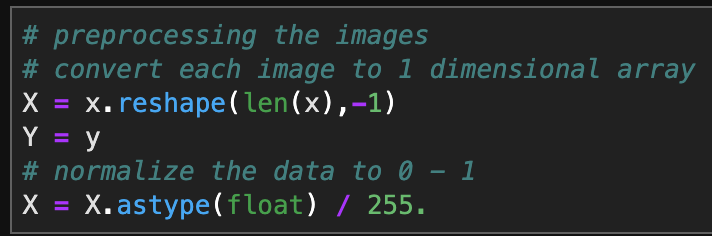


**3.2 訓練模型過程**

讀入MNIST資料集。

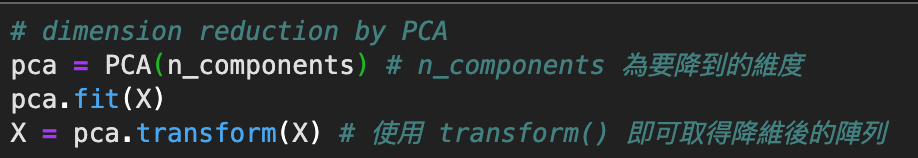
****

由於資料是圖片，是二維資料，所以首先做二維轉一維的處理(reshape)。再來圖片是灰階，灰階圖片pixel value介於0~255區間，為了求處理方便，做除以255的標準化處理。

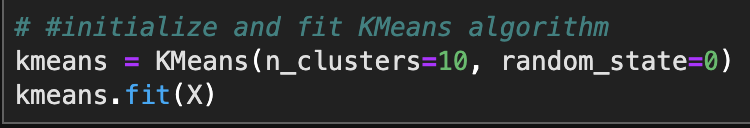
****

選擇以PCA作為降維的方法，分飽嘗試保留100%, 80%, 60%, 40%的特徵，以每20%遞減。

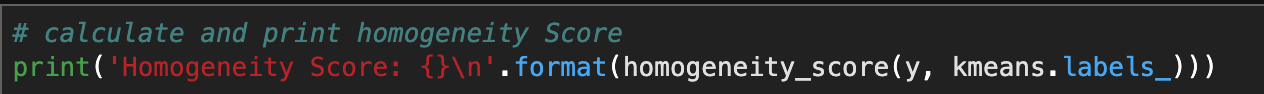
****

****

套用Kmeans模型，設定分成10群。



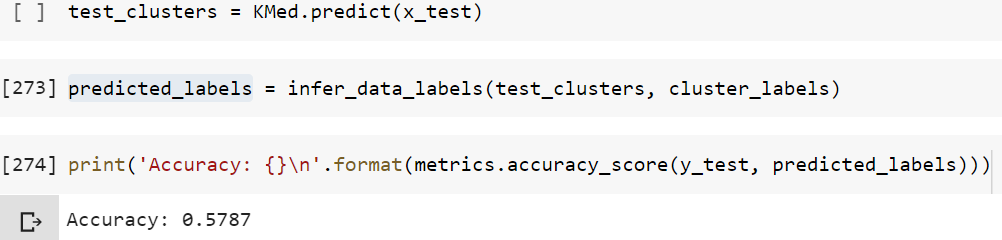
**印出整體分群的**Homogeneity Score



**四、實驗結果與分析**

**1. K-Medoids**

將測試資料丟進 cluster 數量為 10 的 K-Medoids model 後，在 accuracy 為 0.57 左右，另外還測試了不同 cluster 數量的結果，當 cluster 數量為 11 accuracy 為 0.5014、cluster 數量為 9 accuracy 為 0.5011、cluster 數量為 8 accuracy 為 0.4601、cluster 數量為 7 accuracy 為 0.4516，結論為**當 cluster 數量為 10 的 accuracy 是最高的**，當 cluster 數量下降，accuracy 也會遞減。

****

**2. Hierachycal cluster**

分為不同數量群時，所得到的evaluation結果如下：

|  |  |  |
| --- | --- | --- |
| **Cluster** | **Homogeneity score** | **Completeness score** |
| 8 | 0.6143 | 0.7220 |
| 9 | 0.6378 | 0.6930 |
| 10 | 0.6545 | 0.6707 |
| 11 | 0.6546 | 0.6516 |
| 12 | 0.6827 | 0.6512 |

Homogeneity score是分群結果中，每群只包含同一class的程度，類似群內差。而Completeness score是同一class的資料被分到同一cluster的程度。這兩個都是0~1之間的數字。

觀察發現8~12群的數字，homogeneity為遞增，completeness是遞減。符合群越多越集中(h越高)、群越少則越容易將同類分為一群(c越高)，但也犧牲h。

這個方法用在cluster的evaluation，可以觀察cluster的分類為何，以及可能需要怎麼調整群的數量。但缺點就是它仍需要ground truth，sklearn文件中表示實務上幾乎不會用此方式，然而本資料集具備y\_train，仍適合這個method。

**3. K-Means**

實驗結果如下，有分別嘗試使用全部特徵跟降維的處理：

ex: 50%表示取一半的特徵。

|  |  |
| --- | --- |
| Dimension | Homogeneity Score |
| 100%(28\*28=784) | 0.47434297267369585 |
| 80% | 0.4742866487053316 |
| 60% | 0.47624143267384983 |
| 40% | 0.4748912605455121 |

從實驗結果中可以發現，在降維處理後的分群結果還是沒有達到很好的Homogeneity Score。我認為這是因為在處理圖片的時候還是不能忽略影像特徵(例如影像的輪廓、骨架等），對於分群來說，每個維度應該要是能夠獨立解釋的特徵，會是比較理想的。例如以文本分群來說，每個維度可以是重要的用字。但是如果直接將圖片集丟進去分群，在手寫辨識上面存在著一些自然的結構變化(字體的歪斜等等)，所以就算同樣是在寫一個”7”，每個手寫出來的7也不會每一格pixel value都一樣，每一個特徵僅僅是該pixel的值。所以我認為下次如果想要再用分群的方式區分不同圖片的話，應該選擇一些比較有鑑別力的特徵，例如該圖案的topological properties、spatial moments等，這些特徵比較不會受到旋轉、扭曲的影響，比較適合拿來當分群的特徵，而不是直接暴力式地把整張圖片放進去分群。