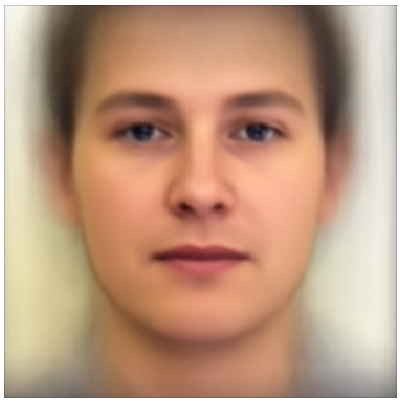
A. PCA of colored faces

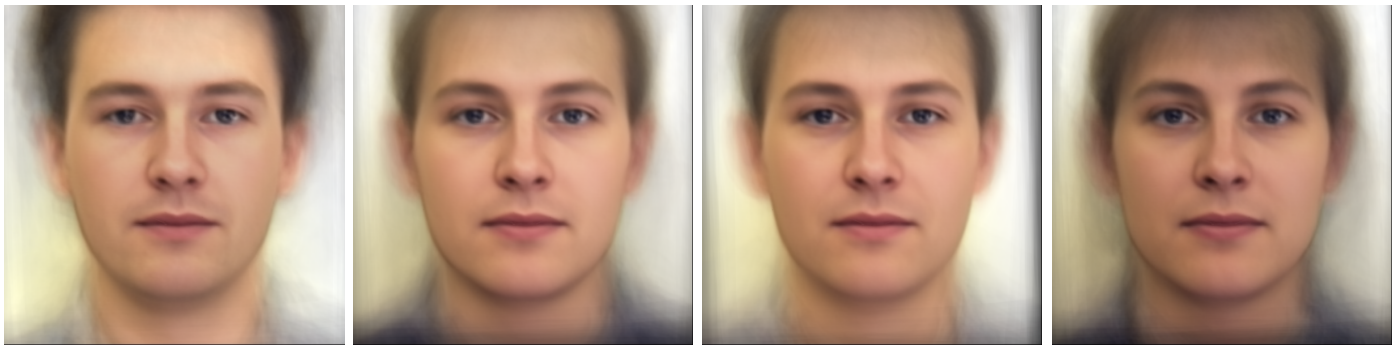
1.請畫出所有臉的平均。



2.請畫出前四個 Eigenfaces，也就是對應到前四大 Eigenvalues 的 Eigenvectors。 Collaborators: b03902086 陳柏屹

由左至右從最大的 Eigenvectors 開始排序。

3.請從數據集中挑出任意四個圖片，並用前四大 Eigenfaces 進行 reconstruction，並畫出結果。Collaborators: b03902086 陳柏屹



由左至右分別為 17, 70, 117, 170 號 reconstruct 出來的結果。

4.並寫出前四大 Eigenfaces 各自所佔的比重，請用百分比表示並四捨五入到小數點後一位。

依序為 4.1%, 3.0%, 2.4%, 2.2%

B. Visualization of Chinese Word Embedding

1.請說明你用哪一個 word2vec 套件，並針對你有調整的參數說明那個參數的意義。

我使用的套件是 gensim，
size = 128：詞向量的維度。
sindow = 5：考慮附近幾個詞。
min_count=1：出現幾次以上的詞才納入考慮。
workers=10：訓練時使用的執行緒數量。
sg=1：使用 skip-gram 訓練。

2.請在 Report 上放上你的 visualization 的結果。

我 visualize 了所有出現次數超過 2000 的詞（去掉標點與停用詞）。

3.請討論你從 visualization 的結果觀察到什麼。

看起來 train 得不錯，我圈起來的一些部分都滿有相關性的。但也有些詞附近都沒什麼東西、推測是跟他有關的詞彙出現次數不足所以在圖上看不到。另外有一些人命跟姓氏也聚在一起（出乎意料的許多人的名字出現超過2000次）。

C. Image Clustering

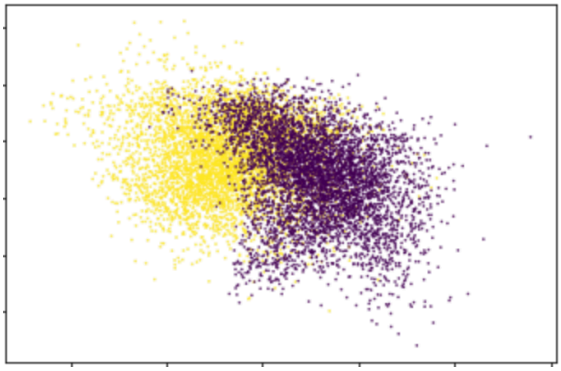
1.請比較至少兩種不同的 feature extraction 及其結果。

方法一：PCA only
對 images 做完 normalization後，直接使用 sklearn 的 PCA 降至 32 維。再用 sklearn 的 KMeans 分成兩群進行預測。

方法二：Autoencoder
建一個 128, 64, 32 units 的三層 DNN Encoder，再依序以 64, 128, 784 units decode，除最後一層用 tanh, activation='relu', optimizer=Adam, loss='mse', epochs=20。用 encoder 將 normalize 完的圖片降至 32 維後做 KMeans。

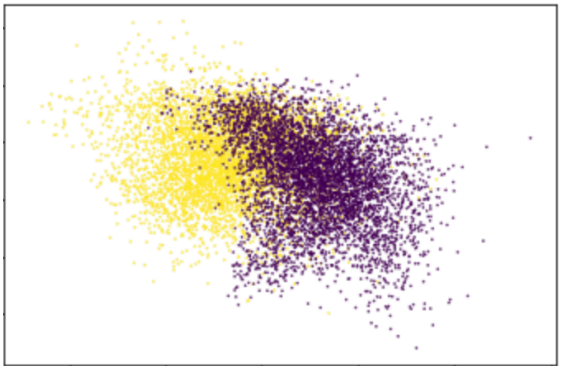
比較：PCA 與 Autoencoder 的 F1-Score 分別為 **0.03** 與 **0.99**。差異大到令人吃驚，且 Autoencoder 所花的時間其實也不長，在這個 work 上算上非常完美。推測關鍵是 PCA 僅能做到線性轉換，而 Autoencoder 可以非線性。

2.預測 visualization.npy 中的 label，在二維平面上視覺化 label 的分佈。



取 encode 後的第 2, 3 維當作 x,y 軸做視覺化。

3. visualization.npy 中前 5000 個 images 跟後 5000 個 images 來自不同 dataset。請根據這個資訊，在二維平面上視覺化 label 的分佈，接著比較和自己預測的 label 之間有何不同。



看起來分佈是非常相似，既然 autoencoder 能做到 0.99，這結果不意外。