Machine Learning 2020 – HW2

A091501 張軼峯

1 Sequential Bayesian Learning

由 data.csv input $x = \{x_1, x_2, \dots, x_{100} | 0 \le x_i \le 2\}$ 和 target $t = \{t_1, t_2, \dots, t_{100}\}$

題目規定使用 Sigmoid Basis Function $\emptyset = [\emptyset_0, \dots, \emptyset_{M-1}]^T$, $\emptyset_j(x) = \sigma\left(\frac{x-\mu_j}{s}\right)$,

Sigmoid 之參數設定為$M = 3, s = 0.6, \mu_j = \frac{2j}{M}\ with\ j = 0, \dots, M-1.$

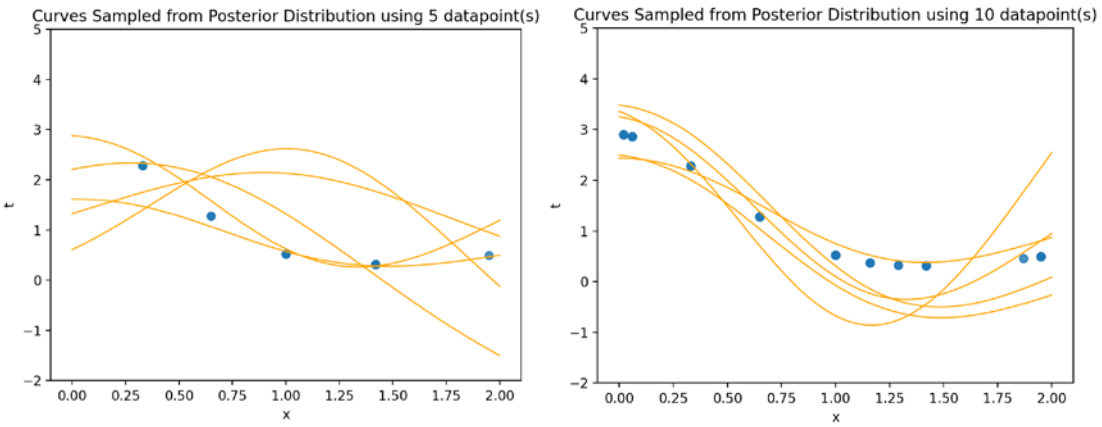
並實作$N = 5, 10, 30, 80.$

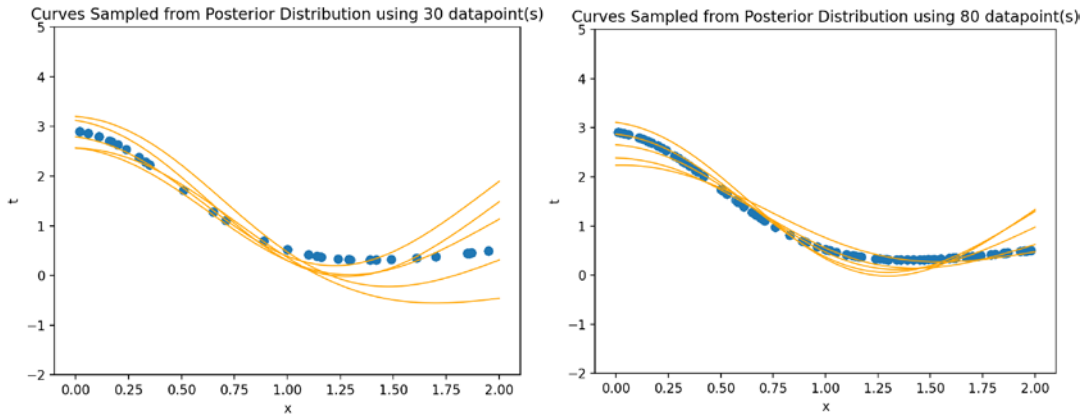
Bayesian Learning:
Conjugate prior assures that the posterior distribution has the same functional form as the prior. The posterior is computed and viewed as the prior for the next parameter updating.

Given:
Prior $p(w|\alpha) = N(0, \alpha^{-1}I) \rightarrow p(w) = N(w|m_0 = 0, S_0 = 10^{-6}I)$
Posterior $p(w|t) = N(w|m_N, S_N)$
由課本推導:
可計算出 posterior distribution 中 $m_N = \beta S_N \Phi^T t,\ \ {S_N}^{-1} = \alpha I + \beta \Phi^T \Phi$

1. Plot five curves sampled from the parameter posterior distribution and N data points.

從得到的 posterior distribution 隨機生成 5 個 weights，將 x 座標代入 basis function，由此 5 個 weights 可計算出 y 座標，即可畫出對應之 curve。





可以觀察到，隨著 datapoints 越多，由 posterior distribution sample 出來的 weight 所畫的 curve 就越收斂越貼近真實 data 的分布。且可觀察到 datapoints 數少時，sample 點越稀疏的地方，curve 越發散。

2. Plot the predictive distribution of target value t by showing the mean curve, the region of variance with one standard deviation on both sides of the mean curve and N data points.

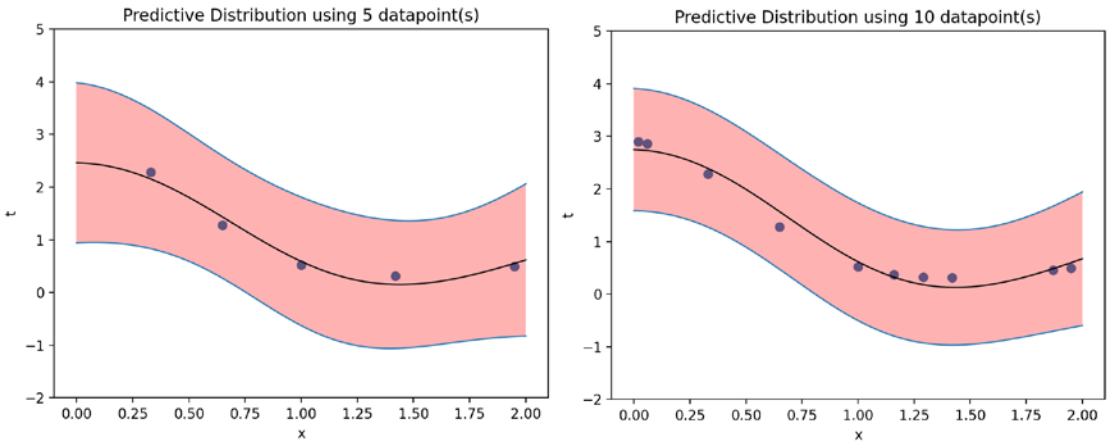
由 Predictive distribution

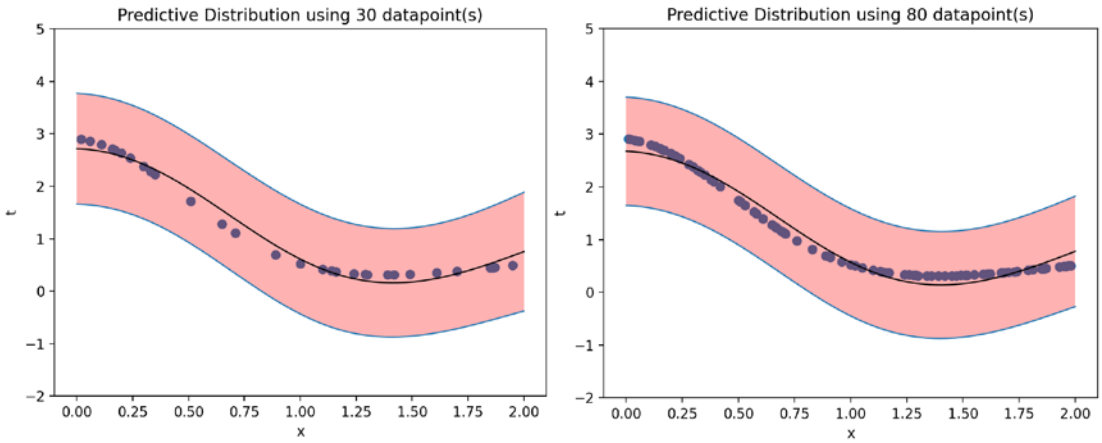
$$\mathrm{p(t|x,x,t)} = \int p(t|x,w)p(w|x,t)dw = N(t|m(x),S^2(x))$$

mean $\mathrm{m(x)} = \beta\phi(x)^T S \sum_{n=1}^{N} \phi(x_n)t_n \quad \Rightarrow \quad \mathrm{m(x)} = \phi(x_n)\cdot m_N$

variance $S^2(x) = \beta^{-1} + \phi(x)^T S_N \phi(x)$

standard deviation $\mathrm{std} = \mathrm{S(x)}$

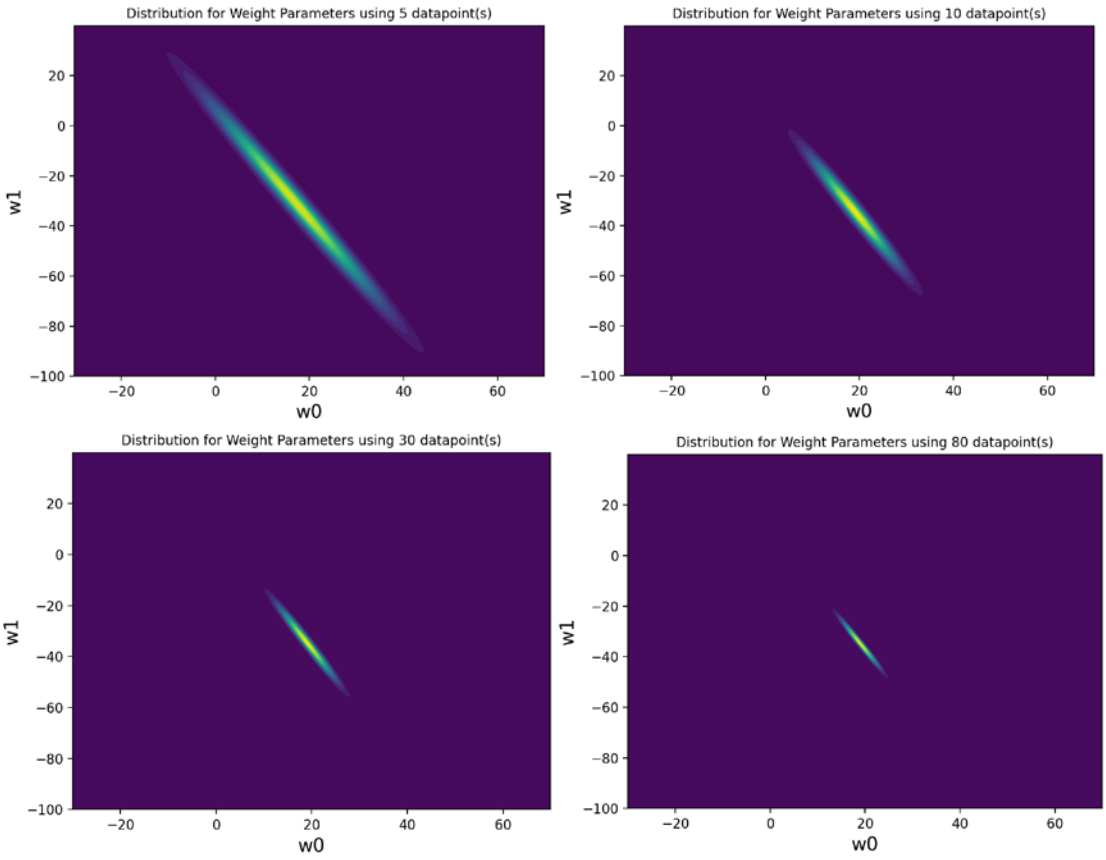




第二小題的結果與第一小題相似，隨著 datapoints 變多，一倍標準差也會越收斂。

3. Plot the prior distributions by arbitrarily selecting two weights.

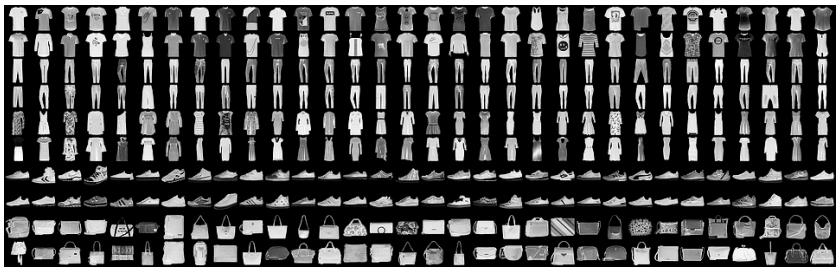
原始矩陣大小: m_N: $(M \times 1)$, S_N: $(M \times M)$，隨機取兩個 weights 畫出 prior distribution，此處作法為將m_N第三維設為 0，即可得到維度 2 的 prior。生成 mesh grid 代入 Gaussian 的 probability density function，即可計算出 mesh grid(x,y)座標所對應的 z 值。



隨著 datapoints 變多，prior distribution 分布的越窄，代表 variance 變小了。可學習的點數越多，可以學的越好，此直觀概念可以很貼切從這三個小題的實作中印證。

備註: 因在助教公布第二次作業補充資料前，已使用 scipy 寫好第一題，但第三小題來不及回頭改掉，還是使用 scipy 的 pdf 函式，接受助教斟酌扣分。這邊簡述如何不使用 scipy 更正作法，第一、二題可使用 `np.random.multivariate_normal` 代替 scipy 的 `multivariate_normal`。第二題會用到 rvs 函式隨機生成連續變數，可直接用一變數存取即可。而第三小題正確的實作方式已簡述如上。

2 Logistic Regression



Given Fashion_MNIST dataset, which contains 5 classes and 64 different images in each class. Normalize the data samples before training and randomly select 32 images as test data for each class.
Implement

- batch GD (batch gradient descent)
- SGD (stochastic gradient descent)
- mini-batch SGD (batch size = 32)
- Newton-Raphson algorithms

to construct a multiclass logistic regression model.

Algorithms	Batch size	Iterations in one epoch
batch GD	N	1
SGD	1	N
mini-batch SGD	B	N/B
Newton-Raphson	N	1

N = number of training data, B = batch size

Softmax function $\mathrm{p}(C_k|\phi_n) = \frac{\exp(a_{nk})}{\sum_j \exp(a_{nj})} = y_k(\phi_n) \triangleq y_{nk}$

Cross Entropy as error function $\mathrm{E(w)} = -\sum_{n=1}^{N}\sum_{k=1}^{K} t_{nk} \log y_{nk}$

1. Set the initial weight vector to be a zero vector. Implement batch GD, SGD, mini-batch SGD (batch size = 32) and Newton-Raphson algorithms to construct a multiclass logistic regression.

   (a) Plot the learning curves of E(w) and the accuracy of classification versus the number of epochs until convergence for training data as well as test data.
   (b) Show the classification results of training and test data.

此題為典型的 Multiclass Logistic Regression，使用 Softmax transformation

$$\mathrm{p}(C_k|\phi_n) = y_k(\phi_n) = \frac{\exp(a_{nk})}{\sum_j \exp(a_{nj})} \quad \text{, where } a_{nk} = {w_k}^T \phi_n.$$

$$\mathrm{E(w)} = -\sum_{n=1}^{N}\sum_{k=1}^{K} t_{nk} \log y_{nk}$$

$$\nabla \mathrm{E(w)} = \sum_{n=1}^{N} (y_n - t_n)\phi_n = \Phi^T(y - t)$$

並用以下方式更新 weights

$$w^{(\tau+1)} = w^{(\tau)} - \eta \nabla E(w) = w^{(\tau)} + \eta \phi_n t_n$$

而 batch GD, SGD, mini-batch SGD 的不同在於 batch 大小的不同。

其中，Newton-Raphson 更新 weight 的方法為

$$w^{(new)} = w^{(old)} - H^{-1}\nabla E(w)$$

where

Hessian matrix $\mathrm{H} = \nabla\nabla\mathrm{E(w)} = \sum_{n=1}^{N} y_n(1 - y_n)\phi_n {\phi_n}^T = \Phi^T R \Phi$

Weighting matrix $R_{n\times n} = [R_{nn}], \quad R_{nn} = y_n(1 - y_n)$

$$\nabla_{wk}\nabla_{wj}\mathrm{E(w)} = \sum_{n=1}^{N} y_{nk}\left(I_{kj} - y_{nj}\right)\phi_n {\phi_n}^T$$

$$w^{(new)} = w^{(old)} - H^{-1}\nabla E(w) = w^{(old)} - (\Phi^T R \Phi)^{-1}\Phi^T(y - t)$$

為了使不同方法比較時有共同標準，固定 Epoch: 100, Learning Rate: 10^{-5}，進行第一組實驗。

實驗(一)

Epoch: 100, Learning Rate: 10^{-5}	Batch GD	SGD	Mini-batch SGD	Newton-Raphson
Training Accuracy	0.9438	0.9938	0.9438	0.9313
Test Accuracy	0.8875	0.9	0.8875	0.8813

實驗(二)

Epoch: 200, Learning Rate: 10^{-6}	Batch GD	SGD	Mini-batch SGD	Newton-Raphson
Training Accuracy	0.9125	0.9875	0.8875	0.9625
Test Accuracy	0.90625	0.8875	0.85	0.9438

- Batch GD 以整個 training set 來計算當前梯度，優點為較準確，因為同時考慮了整個 training data，然而當 data set 越大，計算將顯得耗時。由下圖也可觀察到，Batch GD 的 Cross entropy 曲線與其它方法相較之下為較平滑的曲線。因此，在一定範圍內，增加 batch size 的大小，有助於收斂 training 時的穩定性。

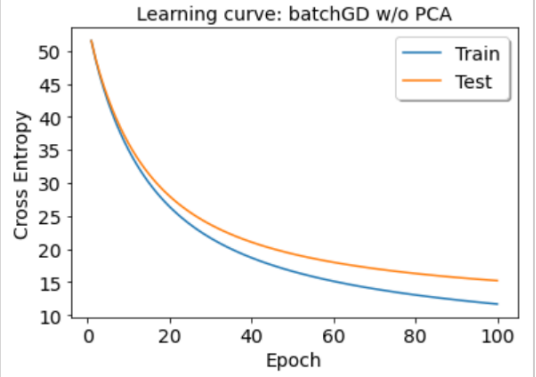
- SGD 則相反，每一次計算 gradient 只用一個樣本，優點為計算快速，然而可能較不準，如實驗一的 SGD Cross entropy 曲線發生震盪的行為，需將 learning rate 調小，如實驗二做圖。

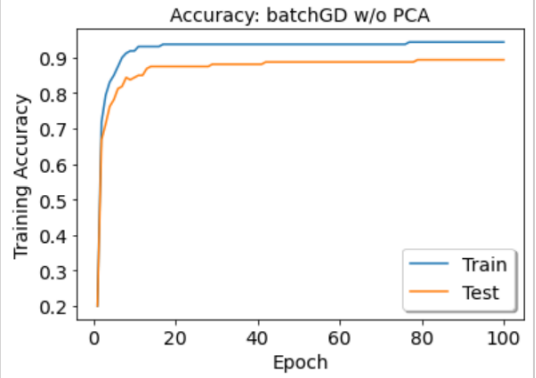
- Mini-batch SGD 則為上述兩者的折衷，綜合了兩者的優點，也是較常使用的方法。也有許多文獻探討過 batch size 與 learning rate 之間的關係，通常而言增加 learning rate，batch size 也需隨著增加，有助於穩定地收斂。

- Newton-Raphson 的概念為對 cost function 求解(use a local quadratic approximation to the log likelihood function)，而非尋找 local minimum(or maximum)。因此 Newton-Raphson 法為對 f'(x)做微分，即二次微分，換言之，cost function 必須選擇可二次微分之 function。也因 Newton-Raphson 需要做二次微分，若 data 的維度很高，計算時的時間複雜度也會很高。由實驗(一) Newton-Raphson 的圖可發現，因使用了較複雜的 model，出現了 overfitting 的現象。

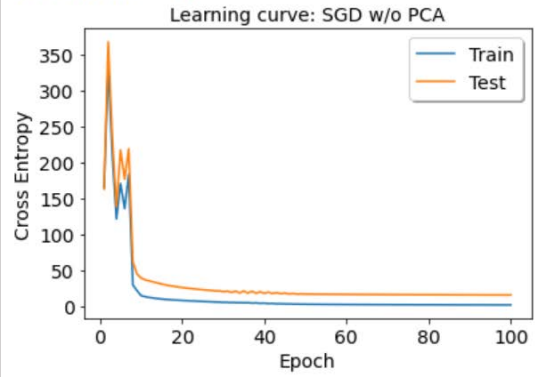
實驗(一)

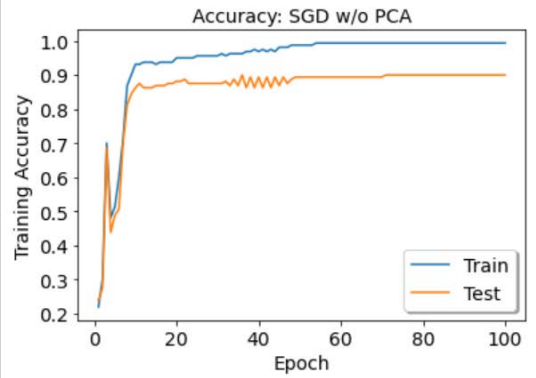
```
TYPE: batchGD w/o PCA
epoch:  100  lr:  0.0001
Training Accuracy:  0.94375
Test Accuracy:  0.89375
```



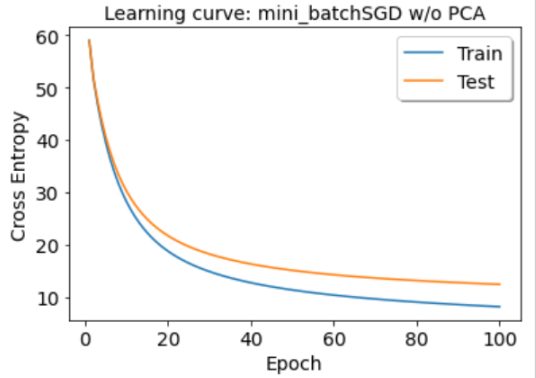


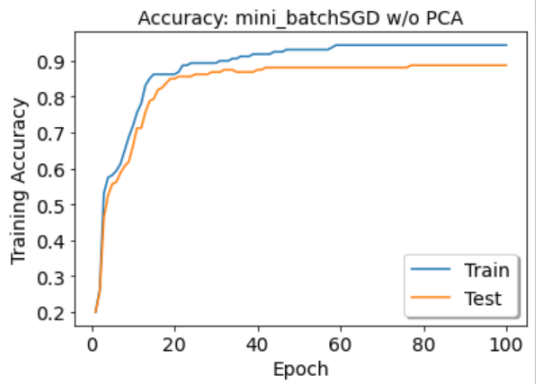

```
TYPE: SGD w/o PCA
epoch:  100  lr:  0.0001
Training Accuracy:  0.99375
Test Accuracy:  0.9
```



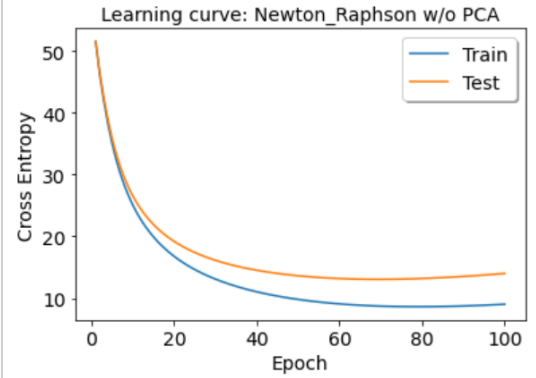


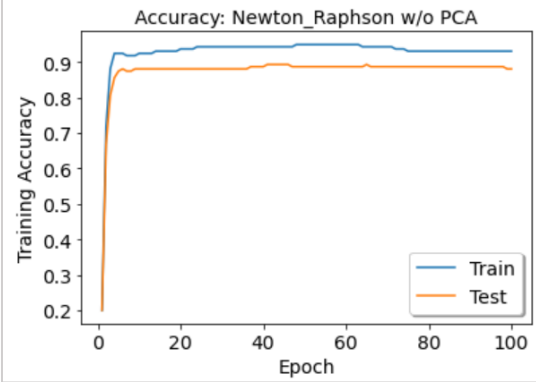

```
TYPE: mini_batchSGD w/o PCA
epoch:  100  lr:  0.0001
Training Accuracy:  0.94375
Test Accuracy:  0.8875
```





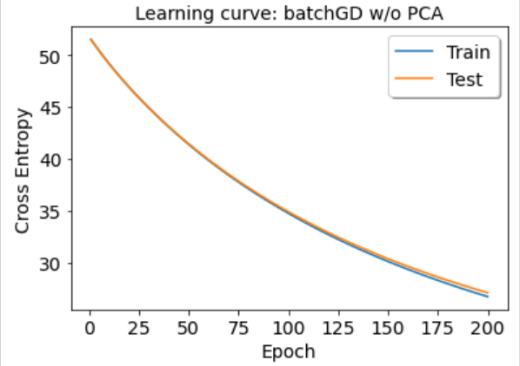

```
TYPE: Newton_Raphson w/o PCA
epoch:  100  lr:  0.0001
Training Accuracy:  0.93125
Test Accuracy:  0.88125
```

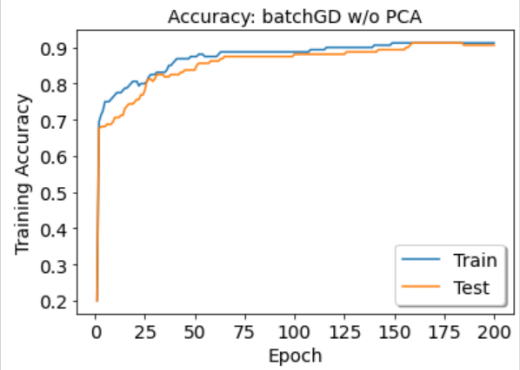




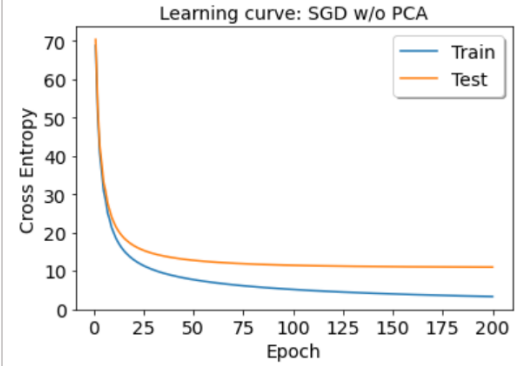
實驗(二)

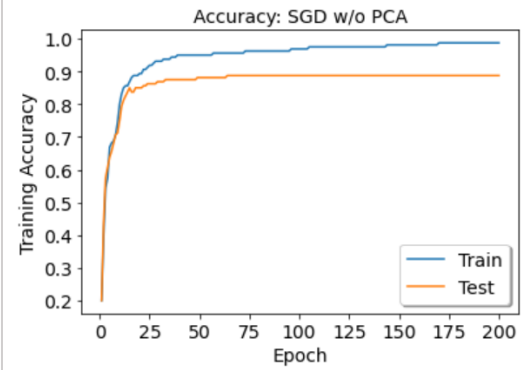
TYPE: batchGD w/o PCA
epoch: 200 lr: 1e-05
Training Accuracy: 0.9125
Test Accuracy: 0.90625



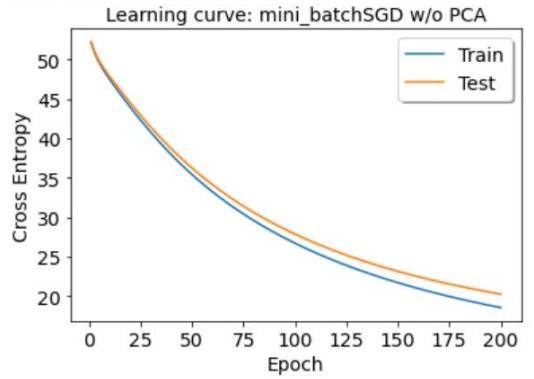


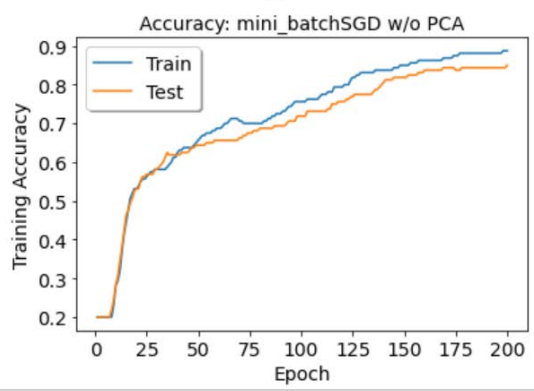
TYPE: SGD w/o PCA
epoch: 200 lr: 1e-05
Training Accuracy: 0.9875
Test Accuracy: 0.8875



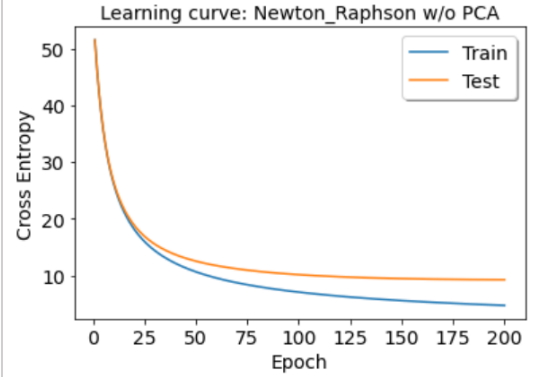


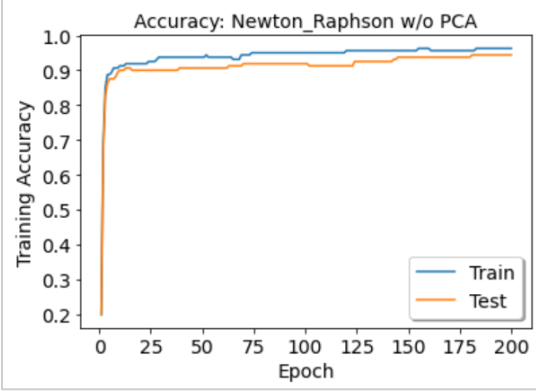
TYPE: mini_batchSGD w/o PCA
epoch: 200 lr: 1e-05
Training Accuracy: 0.8875
Test Accuracy: 0.85





TYPE: Newton_Raphson w/o PCA
epoch: 200 lr: 1e-05
Training Accuracy: 0.9625
Test Accuracy: 0.94375





2. Use principal component analysis (PCA) to reduce the dimension of images to d = 2, 5, 10.

(a) Repeat 1 by using PCA to reduce the dimension of images to d.

首先，Standardize data (training data – mean of training data)，接著計算此 training data 的 covariance matrix，並對 covariance matrix 做 eigen decomposition，得到 eigenvalue 與 eigenvector。此時取前 d 個最大的 eigenvalue 所對應之 eigenvector，將每一 training data 與此 eigenvector 的集合做內積，即可將 data 降至 d 維。

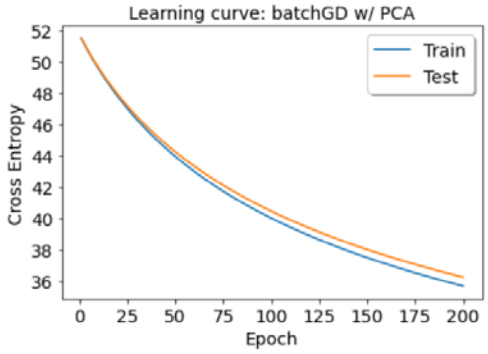
通常 PCA 的目的為避免 Curse of dimension，通過降維(dimension reduction)，能夠適當保存資料的特性，減少資料的維度，並同時維持訓練的 accuracy。

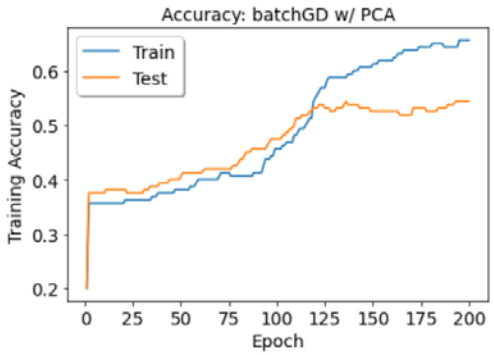
可以觀察到，因為降到 d = 2 維後，input data 的維度降得太低 (複雜度降低)，SGD 更容易出現震盪的情形(一樣可透過降低 learning rate 來解決)。另外，也可以觀察到，維度降低後，各 model 的 accuracy 都降低不少，只有 Newton-Raphson 可維持較準確率。原因如上 Newton-Raphson 方法實作部分所述。

隨著 d 提高，各 model 的 accuracy 也慢慢回升，當 d = 10 時，accuracy 已經和沒有降維之前相當。顯示由 784 維降低至 10 維已可得到相同的訓練結果，有效提升訓練的效率。

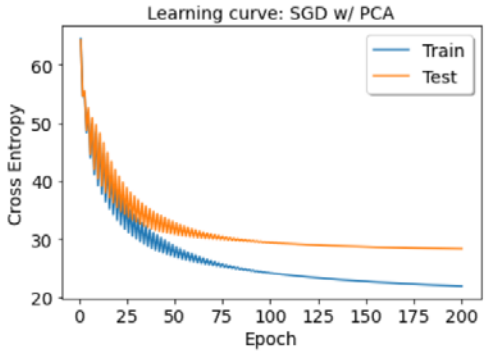
d = 2

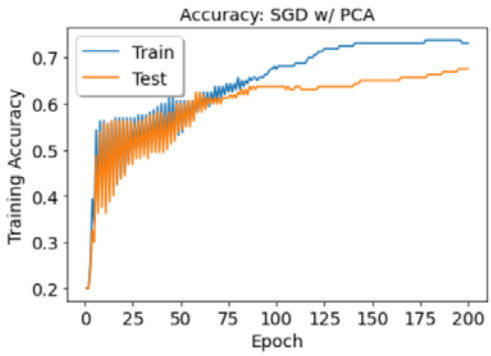
```
TYPE: batchGD w/ PCA
epoch:  200  lr:  1e-05
Training Accuracy:  0.65625
Testing Accuracy:  0.54375
```



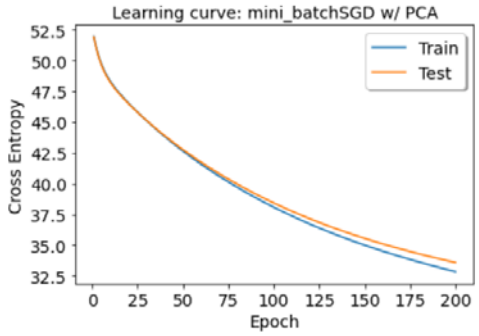


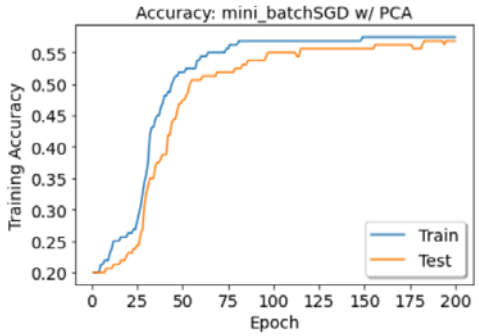

```
TYPE: SGD w/ PCA
epoch:  200  lr:  1e-05
Training Accuracy:  0.73125
Testing Accuracy:  0.675
```



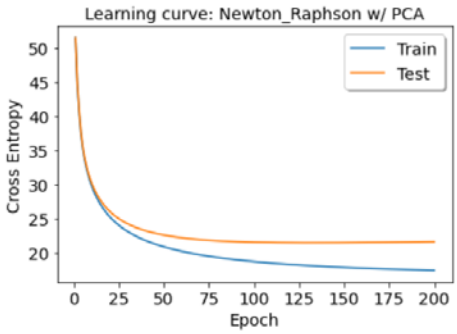


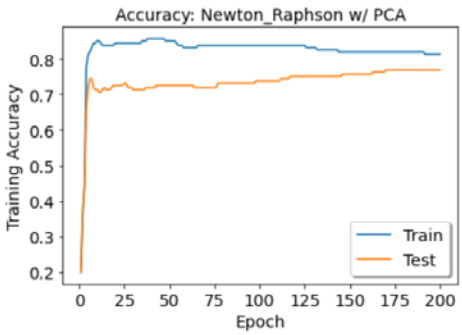

```
TYPE: mini_batchSGD w/ PCA
epoch:  200  lr:  1e-05
Training Accuracy:  0.575
Testing Accuracy:  0.56875
```





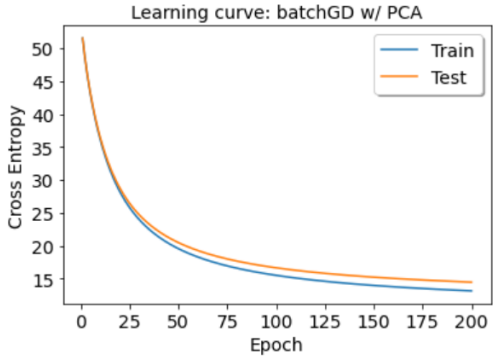

```
TYPE: Newton_Raphson w/ PCA
epoch:  200  lr:  1e-05
Training Accuracy:  0.8125
Testing Accuracy:  0.76875
```

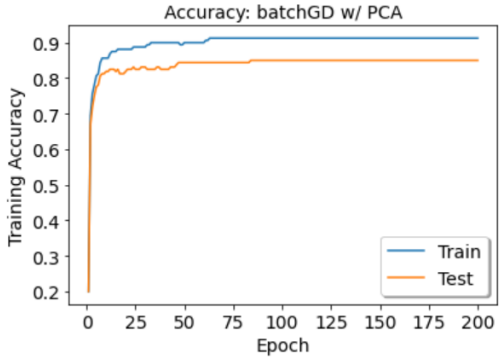




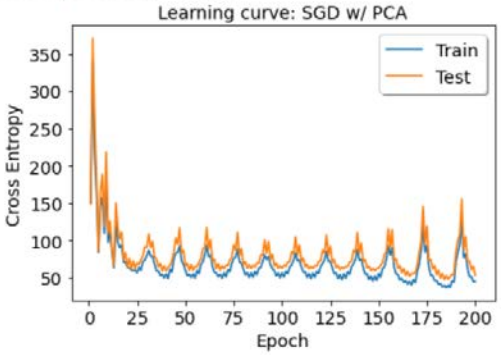
d = 5

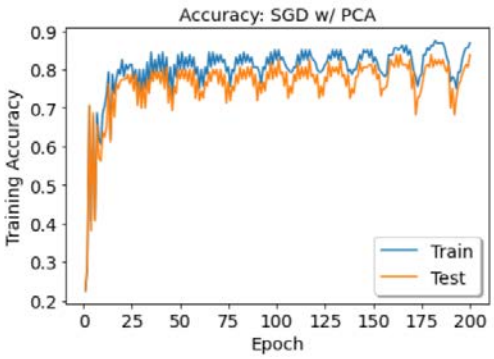
```
TYPE: batchGD w/ PCA
epoch:  200  lr:  0.0001
Training Accuracy:  0.9125
Testing Accuracy:  0.85
```



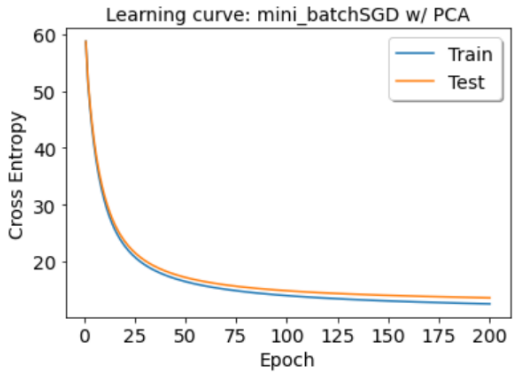


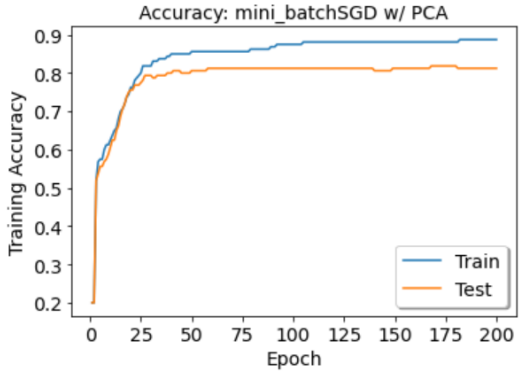

```
TYPE: SGD w/ PCA
epoch:  200  lr:  0.0001
Training Accuracy:  0.86875
Testing Accuracy:  0.8375
```



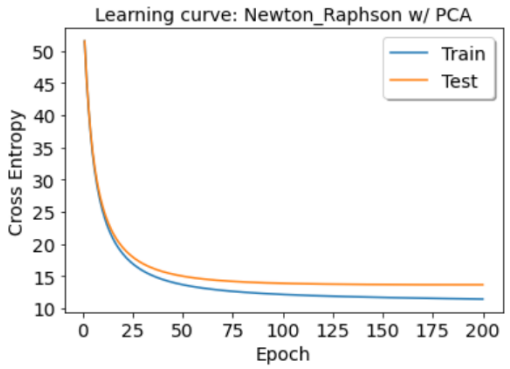


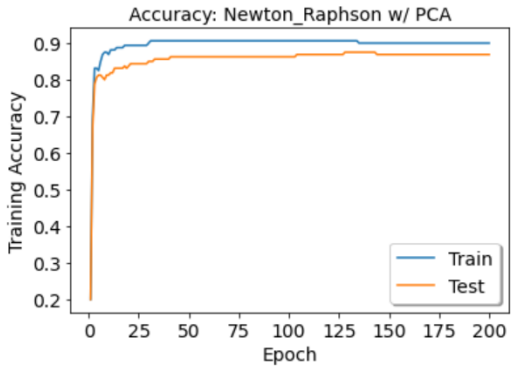

```
TYPE: mini_batchSGD w/ PCA
epoch:  200  lr:  0.0001
Training Accuracy:  0.8875
Testing Accuracy:  0.8125
```





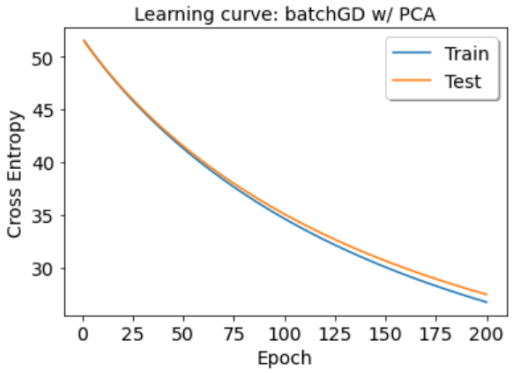

```
TYPE: Newton_Raphson w/ PCA
epoch:  200  lr:  0.0001
Training Accuracy:  0.9
Testing Accuracy:  0.86875
```

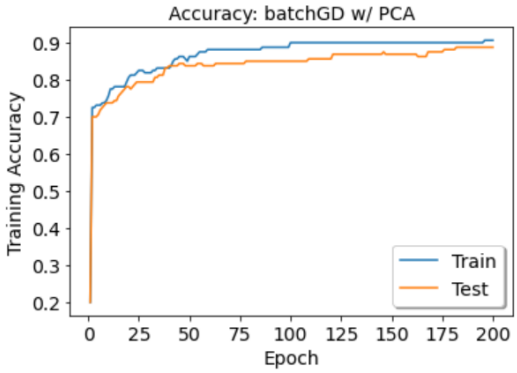




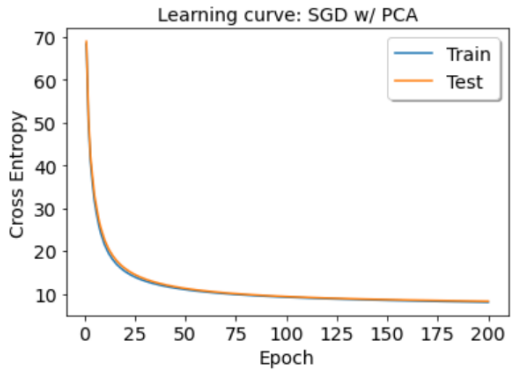
d = 10

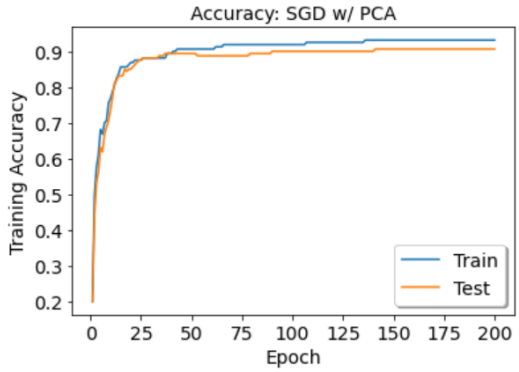
```
TYPE: batchGD w/ PCA
epoch:  200  lr:  1e-05
Training Accuracy:  0.90625
Testing Accuracy:  0.8875
```



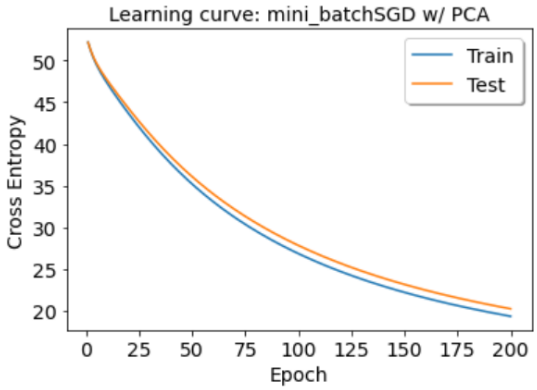


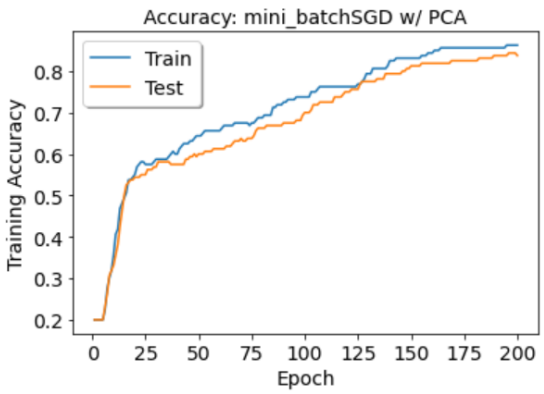

```
TYPE: SGD w/ PCA
epoch:  200  lr:  1e-05
Training Accuracy:  0.93125
Testing Accuracy:  0.90625
```



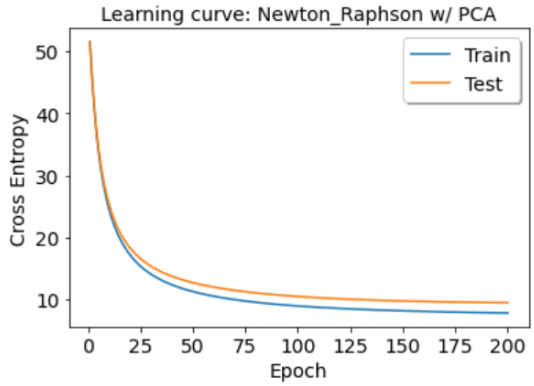


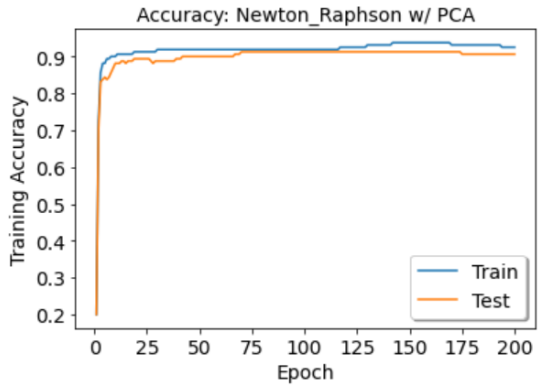

```
TYPE: mini_batchSGD w/ PCA
epoch:  200  lr:  1e-05
Training Accuracy:  0.8625
Testing Accuracy:  0.8375
```





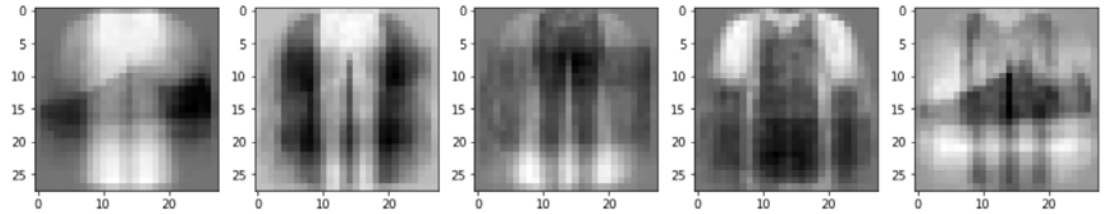

```
TYPE: Newton_Raphson w/ PCA
epoch:  200  lr:  1e-05
Training Accuracy:  0.925
Testing Accuracy:  0.90625
```

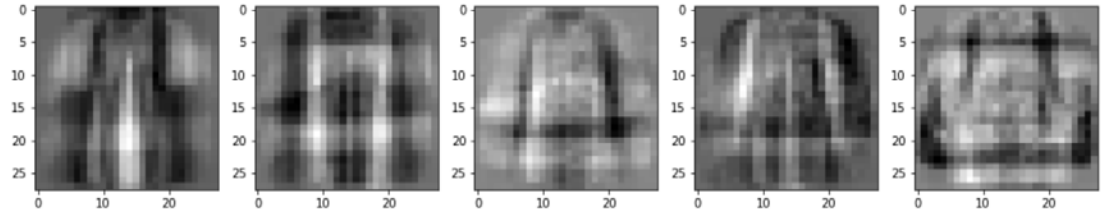




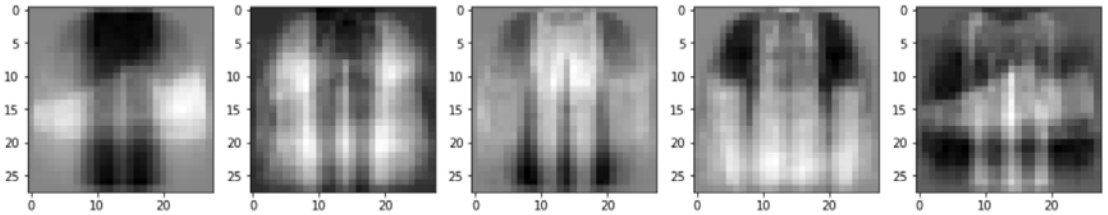
(b) Plot d eigenvectors corresponding to top d eigenvalues.

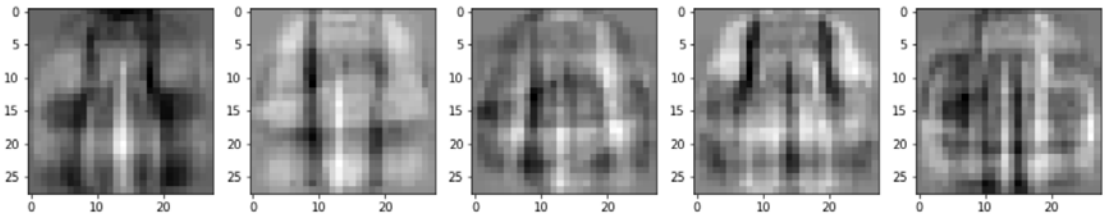
依 eigenvalue 大小取前 d = 10 個 eigenvector，並 reshape 回圖片大小輸出。此 10 個 Eigenvector 的意義為讓資料投影至此方向時，會有最大(此為前 10 大)變異量的投影軸，因此越前面的影像會越清晰，因我們將整個 training data 取 covariance，因此圖片上會有 5 個 classes 的形狀。若印出對應 eigenvalue 最小的 eigenvector，會是一片灰色。





p.s. 也會出現圖案黑白顏色相反的圖，此處還不太清楚為何。

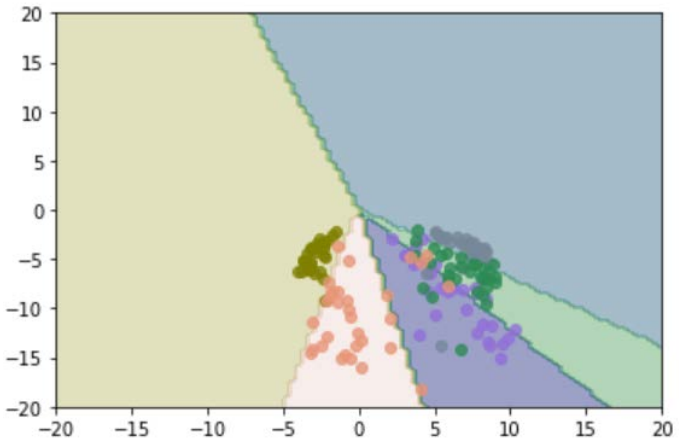




3. What do the decision regions and data points look like on the vector space?

(a) Plot the decision regions and data points of the images on the span of top 2 eigenvectors by using PCA to reduce the dimension of images to 2.

將資料降至 d = 2 維，利用 mesh grid 得出(x, y)座標，代入訓練好的 model 預測出每一點的 class 預測值，因此可分為 5 個顏色。下圖為將座標點代入 SGD 且 lr = 10E-6, epoch = 200，的 model 之作圖結果。可以看到五個分類的 data 大致皆落在各自的 decision regions 內。



(b) Repeat 3(a) by changing the order from M = 1 to M = 2.

可能因 Model 的 accuracy 很低，因此 decision region 只能分成 3~4 個 region。