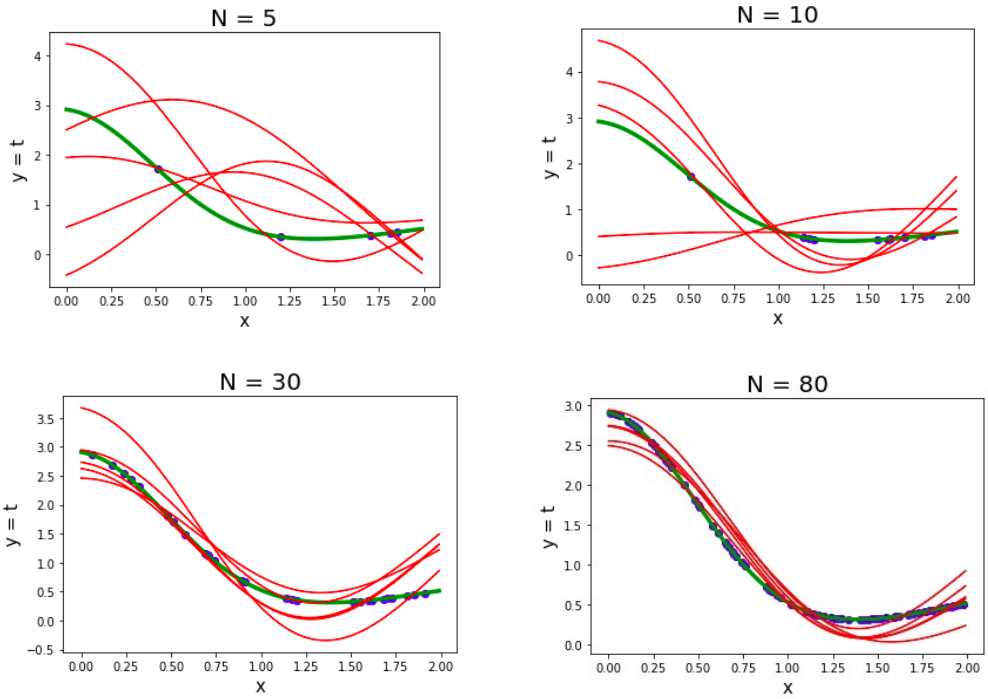
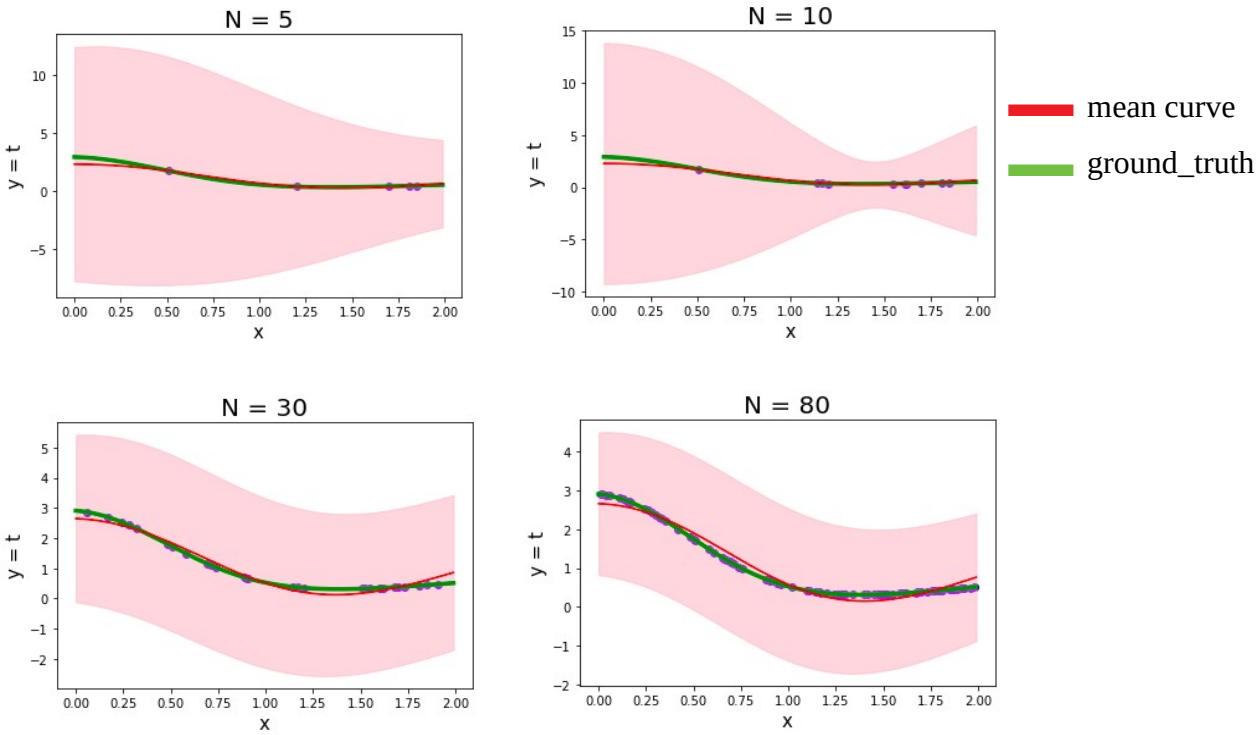
Mechine Learning 2020

1. Sequential Bayesian Learning

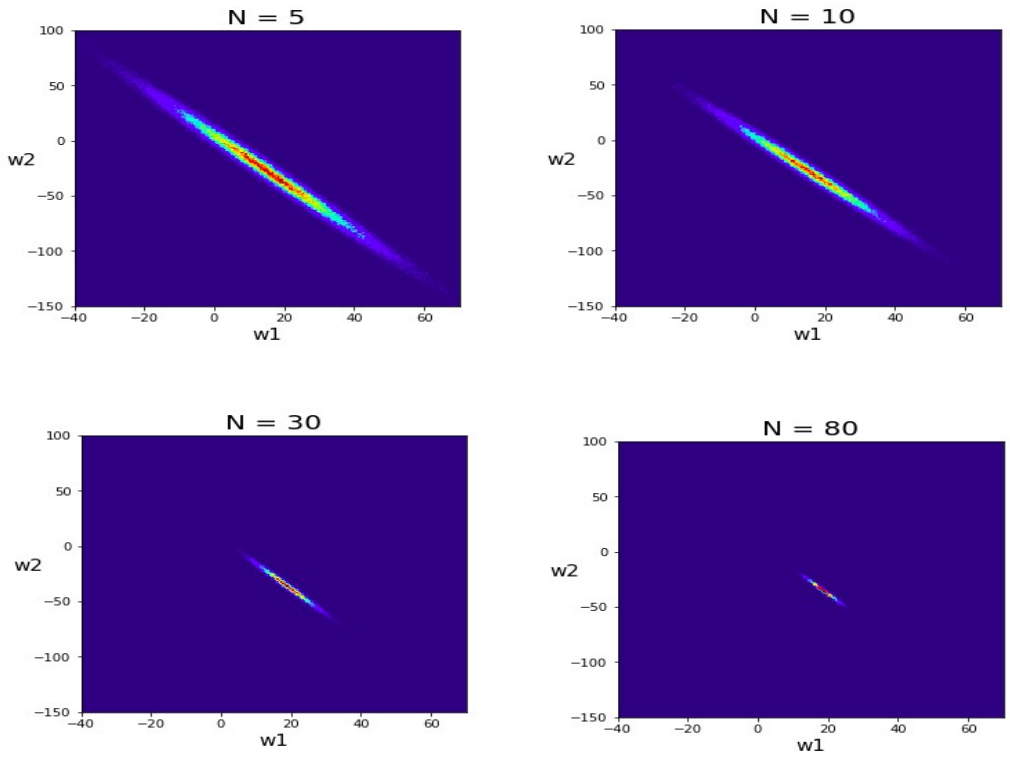
(1) Plot five curves sampled from the parameter posterior distribution and N data points



(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



(3) Plot the prior distributions by arbitrarily selecting two weights



(4) Make some discussion on the results of different N in 1, 2 and 3.

第一題：

先算出 phi 矩陣 $\phi = [\phi_0, \ldots, \phi_{M-1}]^\top$ ，對於每個 phi $\phi_j(x) = \sigma(\frac{x-\mu_j}{s})$ ，接著利用以下公式算出 mN 以及 sN，公式如下：

$$m_N = S_N(S_0^{-1}m_0 + \beta\Phi^T t)$$
$$S_N^{-1} = S_0^{-1} + \beta\Phi^T\Phi$$
$$y(x, w) = w\Phi^T$$

得到 mN 跟 sN 後，便可以使用 normal distribution 去 sample 好幾組的 weight，並且畫出 curves

對於不同的 N 會發現當 N 越多時，畫出來的曲線 match ground truth 的程度越好。

第二題：

透過公式 $\sigma_N^2(\mathbf{x}) = \frac{1}{\beta} + \boldsymbol{\phi}(\mathbf{x})^{\mathrm{T}}\mathbf{S}_N\boldsymbol{\phi}(\mathbf{x})$ 可求出 phi 的 variance，並且將得到的 variance 開平方根得到 std_D，並根據 mean、mean – std_D、mean + std_D 劃出三條線，得到 predictive distribution，

Target value 的求得公式為 $p(t|\mathbf{x}, \mathbf{t}, \alpha, \beta) = \mathcal{N}(t|\mathbf{m}_N^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}), \sigma_N^2(\mathbf{x}))$

對於不同的 N 會發現當 N 越多時，它 predictive distribution 的範圍會變小，也就是 variance 會變小。

第三題：

這邊選的是 weight 1 和 weight 2 的 prior distribution，可以看到當 N 越大時，點會越集中、密集。

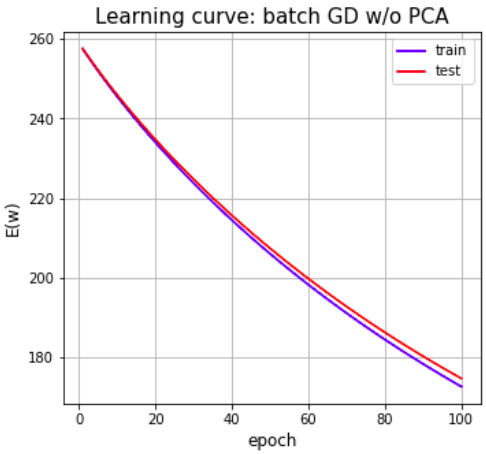
2. Logistic Regression

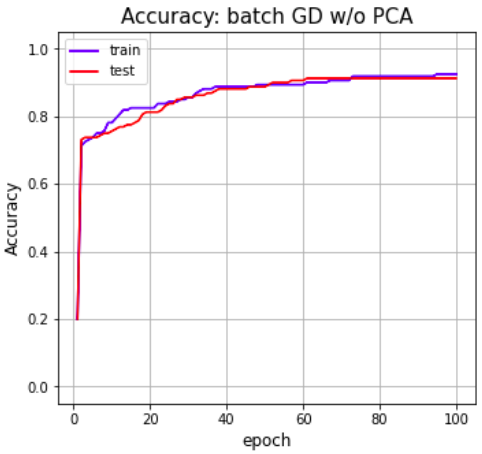
(1)

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.

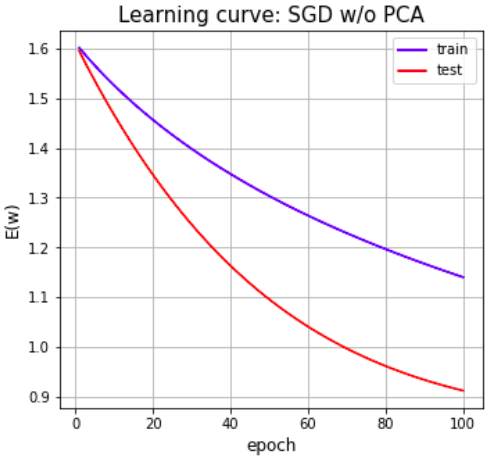
b) Show the classification results of training and test data.

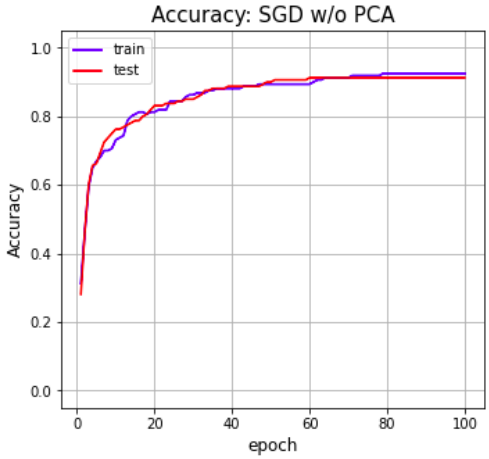
```
TYPE: batch GD w/o PCA
Training accuracy: 0.925
Test accuracy:     0.9125
```



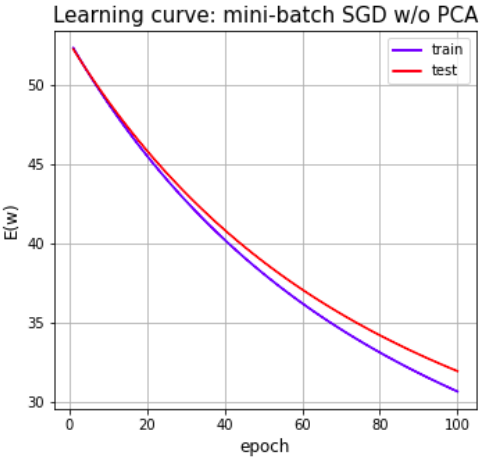


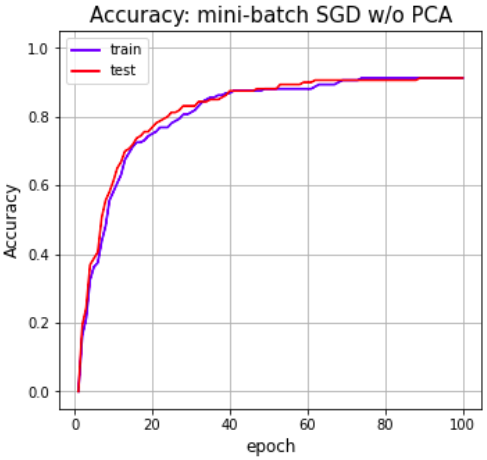

```
TYPE: SGD w/o PCA
Training accuracy: 0.925
Test accuracy:     0.9125
```






```
TYPE: mini-batch SGD w/o PCA
Training accuracy: 0.9125
Test accuracy:     0.9125
```

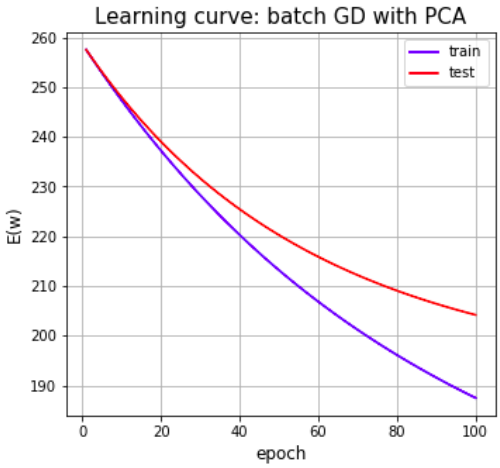


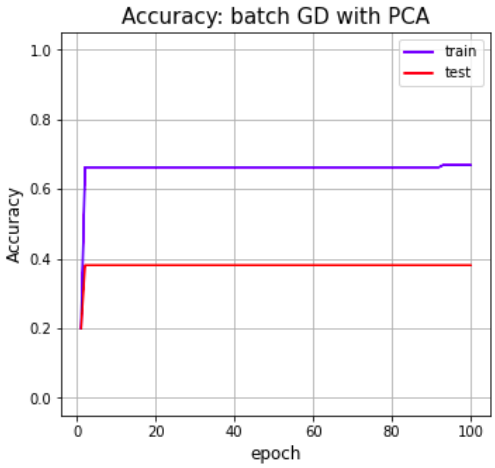


(2)

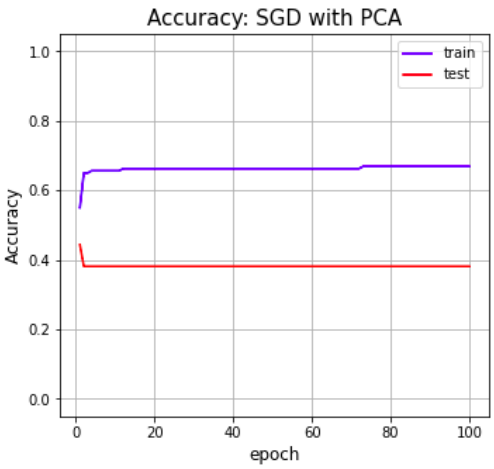
a) Repeat 1 by using PCA to reduce the dimension of images to d.

```
-----------------------------------PCA to dimension 2---------------------------------
TYPE: batch GD with PCA
Training accuracy: 0.66875
Test accuracy:     0.38125
```

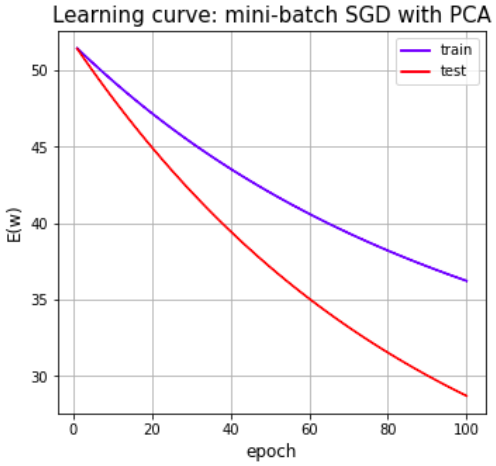


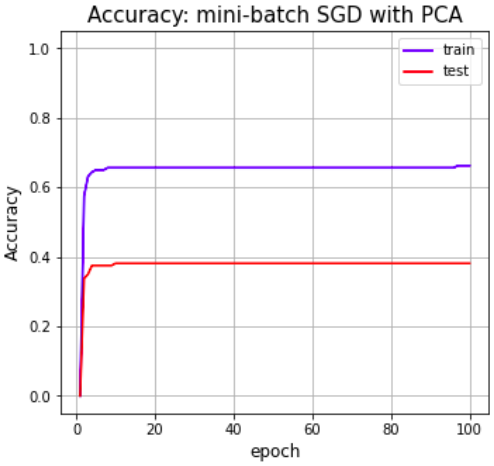



```
-----------------------------------PCA to dimension 2---------------------------------
TYPE: SGD with PCA
Training accuracy: 0.66875
Test accuracy:     0.38125
```

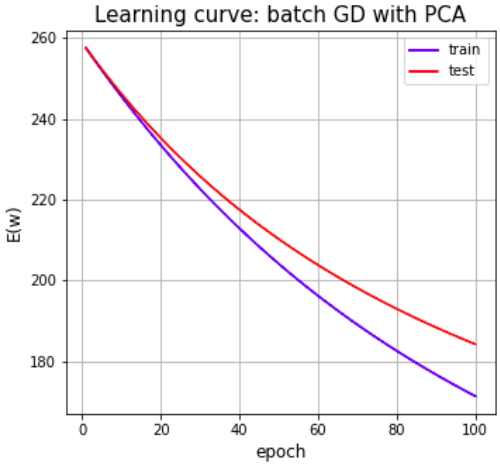


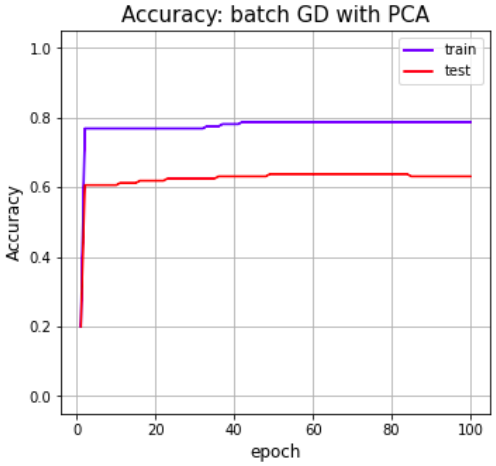

```
-----------------------------------PCA to dimension 2---------------------------------
TYPE: mini-batch SGD with PCA
Training accuracy: 0.6625
Test accuracy:     0.38125
```



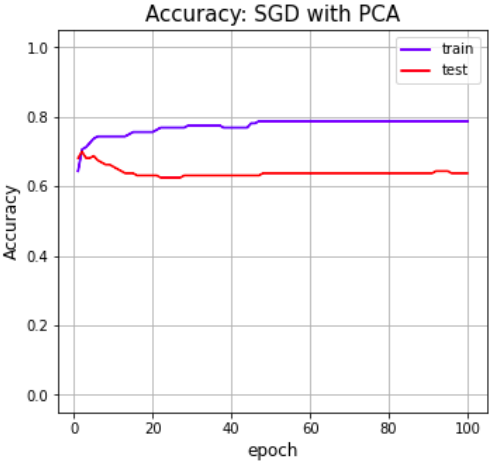


```
-----------------------------------PCA to dimension 5-----------------------------------
TYPE: batch GD with PCA
Training accuracy: 0.7875
Test accuracy:     0.63125
```

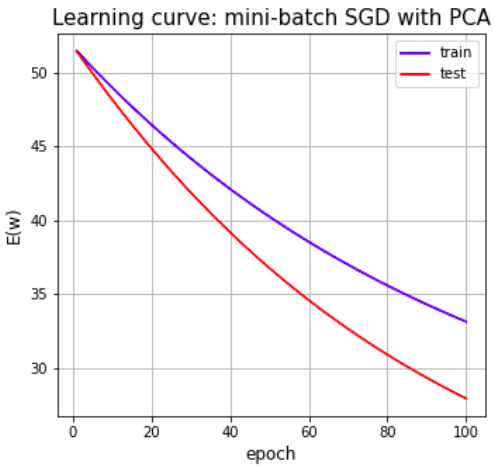


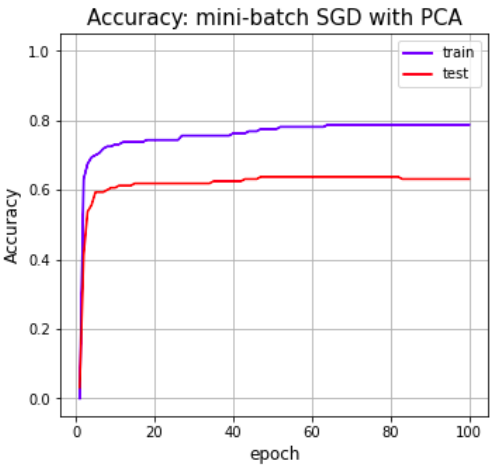



```
-----------------------------------PCA to dimension 5-----------------------------------
TYPE: SGD with PCA
Training accuracy: 0.7875
Test accuracy:     0.6375
```

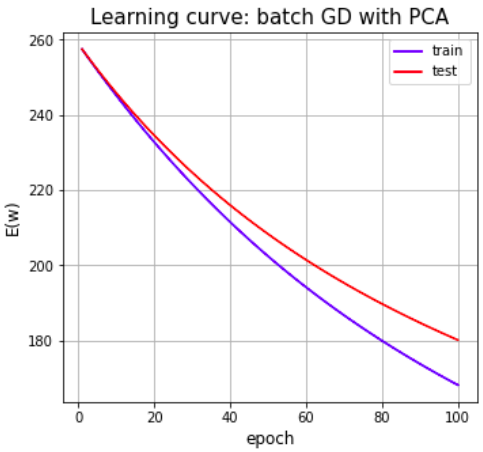


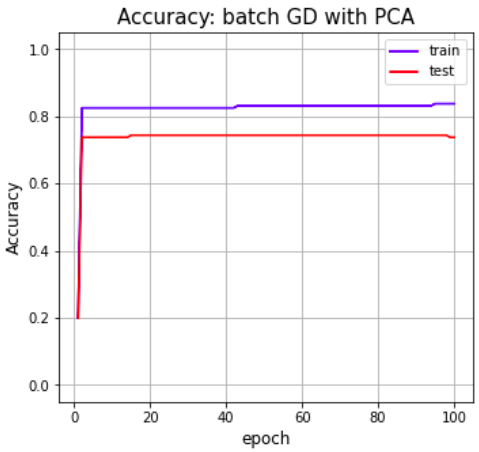

```
-----------------------------------PCA to dimension 5-----------------------------------
TYPE: mini-batch SGD with PCA
Training accuracy: 0.7875
Test accuracy:     0.63125
```



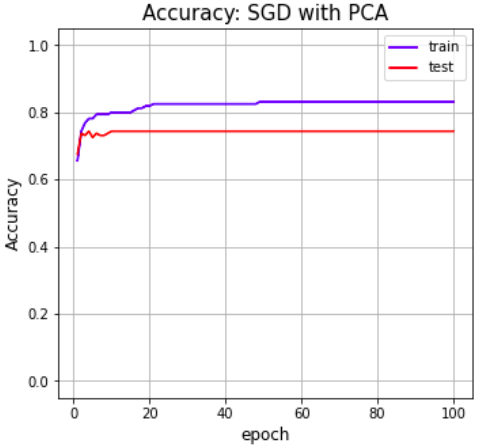


```
-----------------------------------PCA to dimension 10-----------------------------------
TYPE: batch GD with PCA
Training accuracy: 0.8375
Test accuracy:     0.7375
```

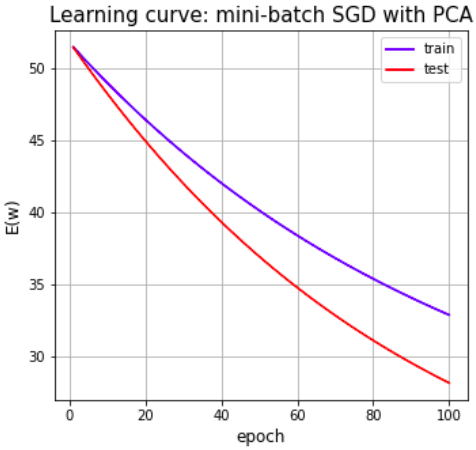


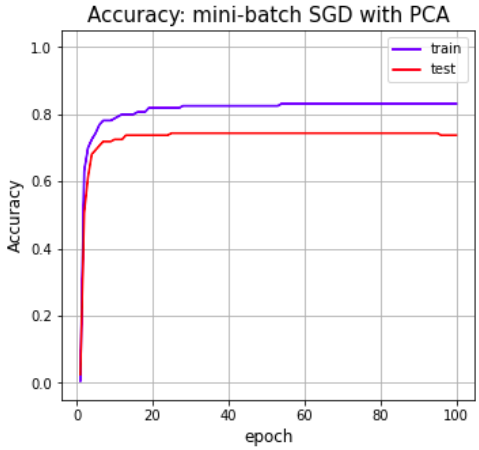



```
-----------------------------------PCA to dimension 10-----------------------------------
TYPE: SGD with PCA
Training accuracy: 0.83125
Test accuracy:     0.74375
```

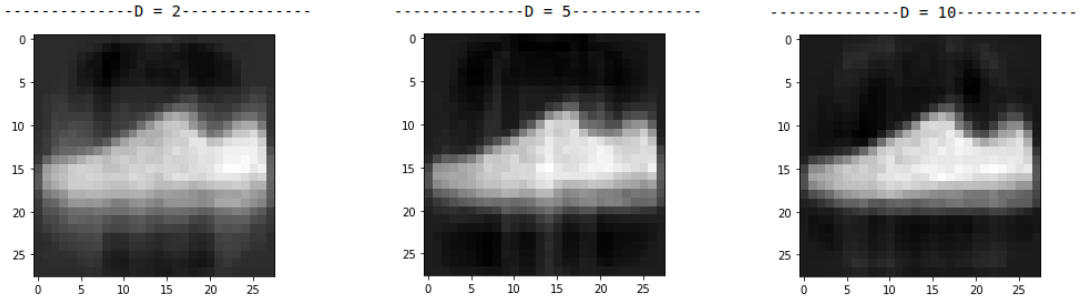



```
-----------------------------------PCA to dimension 10-----------------------------------
TYPE: mini-batch SGD with PCA
Training accuracy: 0.83125
Test accuracy:     0.7375
```





b) Plot d eigenvectors corresponding to top d eigenvalues.



(4) Make some discussion on the results of 1, 2 and 3.

第一題:

Gradient Descent 的公式為 $\mathbf{w}^{(\tau+1)} = \mathbf{w}^{(\tau)} + \eta(t_n - \mathbf{w}^{(\tau)\mathrm{T}}\boldsymbol{\phi}_n)\boldsymbol{\phi}_n$

先讀取圖片(H=28, W=28)，並將圖片 reshape 成 (1,784)。而 training 時，將最剛開始的 weight 初始成一個零矩陣，接著透過算出 a $\mathrm{a} = \Phi^T x$ 再透過算出 y $\mathbf{y}(\Phi) = \frac{\exp a}{\sum_j \exp(a_j)}$

最後透過公式算出新 $\mathrm{w}^{(\tau+1)}$的並繼續做下一次的運算，不停的迭代 100 次。

Error 則是由公式 $E(\mathbf{w}) = -\sum_{m=1}^{N}\sum_{k=1}^{K} t_{nk}\log \check{y}_{nk}$. 算出。

Accuracy 則是比較 y 和 t 的分類結果。

從 Error 的結果可看出，使用 SGD 的方式計算出來的 error 最小，因為 data 是一筆一筆丟進去 train 的，而第二好的則是 mini-batch SGD，因為 data 是 32 筆分批 train 的，最差的則是 GD，因為 data 是全部一起 train。把 data 分批進去 train 的好處是當做到一次 epoch 時，weight 已經迭代很多次了，所以計算出來的 error 會更小，缺點則是需要花費更多的時間去做運算。

第二題:

由結果可看出，當 data 經過 PCA 後，依然是 SGD 的 Error 最小，接著是 mini-batch SGD，最後才是 GD。而下降 data 的維度也會影響其 Accuracy，從結果可以看出下降越多維度，保留的特徵訊息越少，計算出來的 Accuracy 也會越小，Test 和 Train 的都是。

而降維後的 data 再恢復成原始的維度，會因為降維是不可逆的訊息壓縮，所以恢復後的圖會原圖來的模糊，而又因下降的維度差異，模糊的程度也會有所不同。從結果可以看出，下降維度越多，恢復後的圖越模糊。