ML HW2

1. Sequential Bayesian Learning

Bayesian Learning 會先利用 x,y 計算出 posterior 分佈，我們會得到 variance 跟 mean。根據公式如下，其中 S0 一開始設定爲 identity matrix

$$p(\boldsymbol{w} \mid \mathbf{t}) = \mathcal{N}(\boldsymbol{w} \mid \boldsymbol{m}_N, \boldsymbol{S}_N) \tag{3.49}$$

$$\boldsymbol{m}_N = \boldsymbol{S}_N(\boldsymbol{S}_0^{-1}\boldsymbol{m}_0 + \beta\boldsymbol{\Phi}^T\mathbf{t}) \tag{3.50}$$

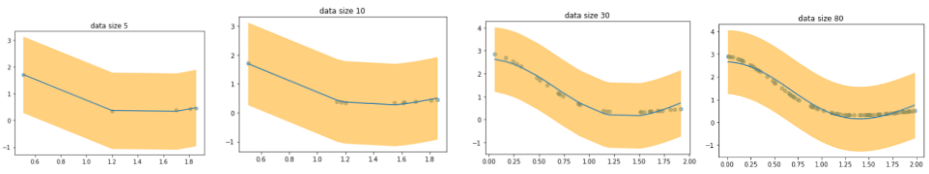
$$\boldsymbol{S}_N^{-1} = \boldsymbol{S}_0^{-1} + \beta\boldsymbol{\Phi}^T\boldsymbol{\Phi} \tag{3.51}$$

接下來利用 phi(x)，variance 和 mean 來預測，先用 multivariate 分佈來取得 weight，將 phi(x)*weight 得到預測答案。

Predictive distribution 利用 3.59 公式計算來得到方差，我們將方差開根號可以得到標準差，用來繪製此圖。

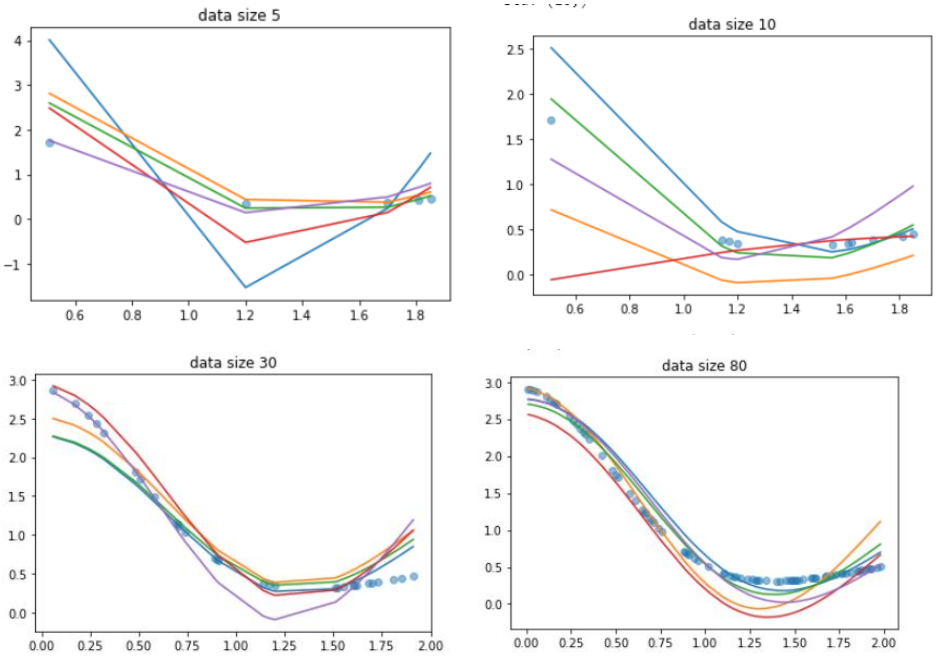
$$\sigma_N^2(\boldsymbol{x}) = \frac{1}{\beta} + \boldsymbol{\phi}(\boldsymbol{x})^T\boldsymbol{S}_N\boldsymbol{\phi}(\boldsymbol{x}) \tag{3.59}$$

可以從中觀察到資料增加時，標準差越來越小。



依序從 data 抽取 5, 10, 30 ,80 筆資料做訓練，可以從結果

圖看到一開始只有 5 跟 10 筆資料很難去 fit。Model 預測的五條線難以收斂，但到 data 有 30 到 80 筆時候可以看到資料開始慢慢收斂



2. Logistic Regression

我們使用圖片作爲分類任務訓練資料，資料匯入後會先將其坦平((28, 28) -> 784)後做一般化並將 dataset 隨機選取各類 32 張作爲測試資料其餘作爲訓練資料。

我們可以利用 softmax 來取得預測結果。

$$p(\mathcal{C}_k \mid \boldsymbol{\phi}) = y_k(\boldsymbol{\phi}) = \frac{\exp(a_k)}{\sum_j \exp(a_k)} \tag{4.104}$$

計算 loss 利用 cross entropy 來計算

$$E(\boldsymbol{w}_1,\ldots,\boldsymbol{w}_K) = -\ln p(\boldsymbol{T} \mid \boldsymbol{w}_1,\ldots,\boldsymbol{w}_K) = -\sum_{n=1}^{N}\sum_{k=1}^{K} t_{nk}\ln y_{nk} \tag{4.108}$$

將此公式求導，得到公式 4.109，我們將用此公式來更新 weight。

$$\nabla_{\boldsymbol{w}_j} E(\boldsymbol{w}_1,\ldots,\boldsymbol{w}_K) = \sum_{n=1}^{N}(y_{nj} - t_{nj})\boldsymbol{\phi}_n \tag{4.109}$$

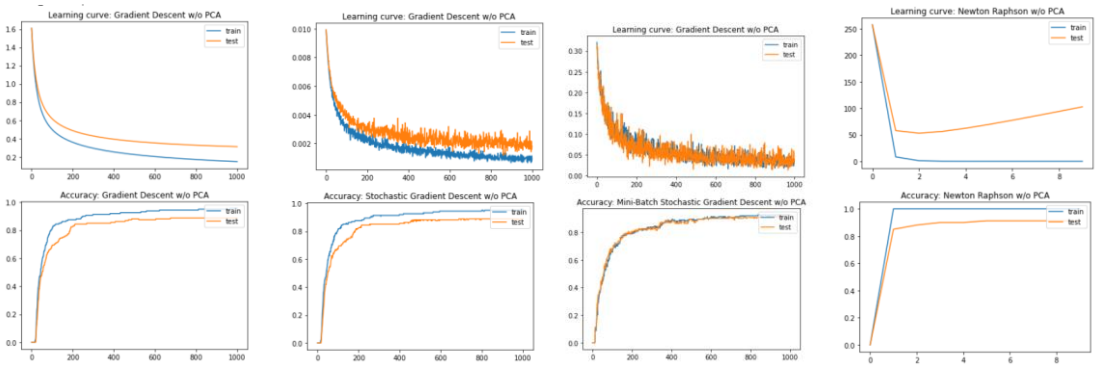
Newton-Raphson 利用求導一次的 E 以及求導兩次的海珊矩陣來更新 weight

$$\boldsymbol{w}^{新} = \boldsymbol{w}^{旧} - \boldsymbol{H}^{-1}\nabla E(\boldsymbol{w}) \tag{4.92}$$

$$\nabla E(\boldsymbol{w}) = \sum_{n=1}^{N}(y_n - t_n)\boldsymbol{\phi}_n = \boldsymbol{\Phi}^T(\mathbf{y} - \mathbf{t}) \tag{4.96}$$

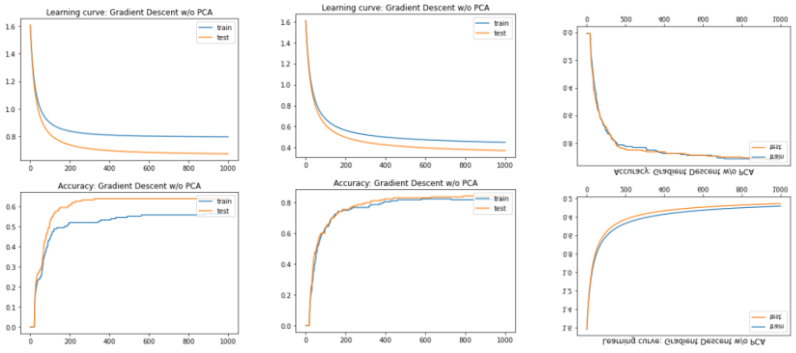
$$\boldsymbol{H} = \nabla\nabla E(\boldsymbol{w}) = \sum_{n=1}^{N} y_n(1 - y_n)\boldsymbol{\phi}_n\boldsymbol{\phi}_n^T = \boldsymbol{\Phi}^T\boldsymbol{R}\boldsymbol{\Phi} \tag{4.97}$$

Result: 這四張圖分別代表 batch gradient, stochastic gradient, min-batch stochastic gradient, newton Raphson。有一點比較奇怪的是 batch gradient 跟 stochastic gradient 我得出來的 accuracy 是一樣的，我認爲可能是資料量過小。我有發現做過一般化後訓練資料準確度會從 0.8 提升到 0.9。



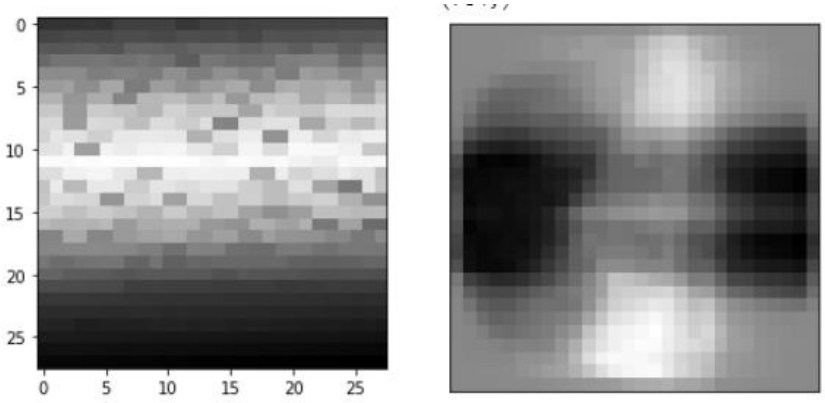
PCA：利用 training data 計算出 eignvectors，並利用 top d eigenvalue 對應的 eigenvectors ，將 training data 及 test data 投影上去以降至 d 維。

Result: 這三張圖是使用 batch gradient 分別做 pca2, 5, 10 的結果，可以發現如果將維度降爲 2 維準確度只有 0.6，大概維度爲 10 準確度比較接近原始資料而且訓練速度快很多。



在 plot eigenvectors corresponding to top d eigenvalues 中我有拿我的方法跟 sklearn 比較，如下。這邊可能還要研究一

下到底是哪裏出問題，因爲準確度是差不多的可能是方法不同。



3. Decision regions and data points