More Notes on Lab 2

1. Perceptron Learning Algorithm (PLA)

Step 1	Given training dataset:
	$D = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$ // N training examples
Step 2	Initialize all weights w_i to random values
	// $\mathbf{w} = (w_0, w_1, \ldots, w_M)$; M+1 attributes
Step 3	WHILE not all examples correctly predicted DO
Step 4	FOR each training example $\mathbf{x}_k \in D$
	If sign $(\mathbf{w} \bullet \mathbf{x}_k) \neq y_k$ then $\mathbf{w} \leftarrow \mathbf{w} + y_k\, \mathbf{x}_k$

2. Stochastic Logistic Regression Algorithm (Stochastic Logistic Regression)

Step 1	Input training data set:
	$D = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$ // N training examples
Step 2	Initialize $\mathbf{w}$ and choose a learning rate η
	// $\mathbf{w} = (w_0, w_1, \ldots, w_M)$; initialize all weights w_i to random values
Step 3	UNTIL a termination condition is met, DO
Step 4	FOR each training example $\mathbf{x}_k \in D$
	// Update the weight vector for each training example
	$\mathbf{w} \leftarrow \mathbf{w} + \eta(y_k - \widehat{y_k})\mathbf{x}_k$
	// where $\widehat{y_k} = a = \sigma(n) = \frac{1}{1+e^{-n}}$, and
	// $n = w_0 x_0 + w_1 x_1 + \ldots + w_M x_M = [w_0, w_1, \ldots, w_M]\begin{bmatrix} x_0 \equiv 1 \\ x_1 \\ \vdots \\ x_M \end{bmatrix}$

3. **Stochastic Logistic Regression** 與 PLA 之間的差異是：
   (a) 在 Step 2 中，**Stochastic Logistic Regression** 除了要給定隨機初使值 $\mathbf{w} = (w_0, w_1, \ldots, w_M)$，還需要給定學習率$\eta$參數值，一般設定學習率 $0 < \eta < 1$。
   (b) 在 Step 3 中，**Stochastic Logistic Regression** 停止條件可以考慮以下情況而停止:
      (i) Epoch 超過最大給定的世代數，例如設定最大世代數時為 1000，則程式在執行 1000 次的世代回圈後停止；此條件保證程式會停止，如果給定最大的世代數很大，程式也有可能找到不錯的 $\mathbf{w} = (w_0, w_1, \ldots, w_M)$，即很好的決策邊界線 (Decision boundary)，但也不敢斷定是否真的很好。
      (ii) 另外停止條件可以設定某個誤差度量足夠小而停止，如果由此條件停止，我們知道找到的 $\mathbf{w} = (w_0, w_1, \ldots, w_M)$具有足夠小的誤差而停止，比(i)最大世代數的停止條件更有信心，即決策邊界線(Decision boundary)是不錯的。誤差度量的選擇，可參考第(e)項說明。

(iii) 程式中建議使用兩個停止條件，即超過最大世代數或誤差度量足夠小而停止

(c) 在 Step 4 中， **Stochastic Logistic Regression** 依據 Gradient Descent Algorithm 的概念，導出 $\mathbf{w}$ 的更新方式：$\mathbf{w} \leftarrow \mathbf{w} + \eta(y - \hat{y})\mathbf{x}$，其中訓練資料 y 的值是 0 或 1，而 $\hat{y} = a = \sigma(n) = \frac{1}{1+e^{-n}}$，其值在 0 到 1 之間，直接更新 $\mathbf{w}$。保持 $\hat{y}$ 在 0 到 1 之間，不要轉換 $\hat{y}$ 為 0 或 1。

(d) 何時轉換 $\hat{y}$ 為 0 或 1?

在預測測試資料的類別標籤時，需要將概率 $\hat{y}$ 轉換為類別標籤。因此，如果 $\hat{y} \geq 0.5$，則預測為 1；否則預測為 0。以 Lab 2 為例，只有 Case 4 有測試資料，因此一旦程式停止訓練，$\mathbf{w} = (w_0, w_1, w_2)$已定案，將各筆測試資料 $\mathbf{x} = (x_0 \equiv 1, x_1, x_2)$，代入 $n = w_0 x_0 + w_1 x_1 + w_2 x_2$，求出 $\hat{y} = a = \sigma(n) = \frac{1}{1+e^{-n}}$，將 $\hat{y}$ 轉換為類別標籤 0 或 1。

(e) 誤差度量的選擇

(i) **Logistic Regression** 的誤差函數為 Cross-entropy loss function:

$E(\hat{y}) = E(w_0, w_1, w_2) = -(y \ln \hat{y} + (1 - y) \ln (1 - \hat{y}))$，

可採用一個世代中，其資料集合的平均誤差小於某個給定的容忍值 τ，即

$\frac{1}{N}\sum_{k=1}^{N} -(y_k \ln \widehat{y_k} + (1 - y_k) \ln (1 - \widehat{y_k})) < \tau$。

(ii) Cross-entropy loss function 的原始概念為 Max $\hat{y}^{y}(1-\hat{y})^{(1-y)}$，若設 $J(\hat{y}) = J(w_0, w_1, w_2) = \hat{y}^{y}(1-\hat{y})^{(1-y)}$，則可採用一個世代中，其資料集合的平均 J 函數值大於某個給定的門檻值 γ，即$\frac{1}{N}\sum_{k=1}^{N} (\widehat{y_k}^{y_k}(1-\widehat{y_k})^{(1-y_k)}) > \gamma$。

此時，我們希望每個$\widehat{y_k}$接近 1。

$$\left(\begin{array}{l} \text{If } y = 1\text{: } P(y|\mathbf{x}) = \hat{y}^{1}(1-\hat{y})^{(1-1)} = \hat{y}\,(1-\hat{y})^{0} = \hat{y}\text{：希望 } \hat{y} \text{ 接近 } 1 \\ \text{If } y = 0\text{: } P(y|\mathbf{x}) = \hat{y}^{0}(1-\hat{y})^{(1-0)} = 1(1-\hat{y})^{1} = 1-\hat{y}\text{：還是希望 } \hat{y} \text{ 接近 } 1 \end{array}\right)$$

我們因此可以設定$\frac{1}{N}\sum_{k=1}^{N} (\widehat{y_k}^{y}(1-\widehat{y_k})^{(1-y)}) > \gamma$，例如令$\gamma = 0.9$。

(iii) 其他誤差度量，也可以採用。例如：

- Mean Squared Error (MSE) = $\frac{1}{N}\sum_{k=1}^{N}(y_k - \widehat{y_k})^2 < \tau$
- Root Mean Squared Error (RMSE) = $\sqrt{\text{MSE}} < \tau$
- Mean Absolute Error (MAE) = $\frac{1}{N}\sum_{k=1}^{N}|y_k - \widehat{y_k}| < \tau$