ML Lecture 5: Logistic Regression

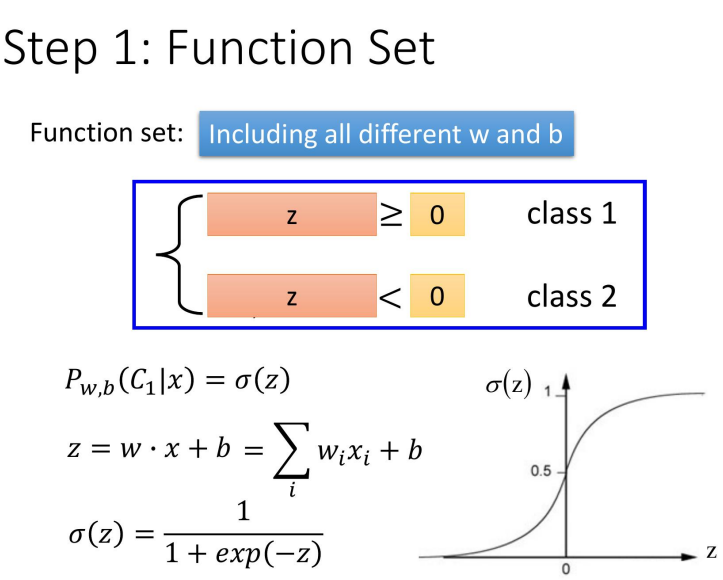
> 臺灣大學人工智慧中心 科技部人工智慧技術暨全幅健康照護聯合研究中心 http://ai.ntu.edu.tw

Step 1. Function Set

- **Posterior Probability：**

  > **$P_{w,b}(C_1|x) = \sigma(z)$，由 z 代入 $sigmodfunction$ 後得 $z = w * x + b$ ，z 由 w 和 b 所控制產生**

  ==> 所有 w 和 b 可產生的 $function$ 所成的集合，就是一個 **function set**



- 以**「圖像化」**表示「Logistic Regression」 這件事

  > Input x_1 到 x_I 分別乘上 weight w_1 到 w_I (內積)，再加上 bias, b，即為 z
  >
  > 通過 $sigmoidfunction$，output 的值是 **posterior probability**

- **比較**（Output 的值）
  - Logistic Regression：有通過 $sigmoidfunction$，output 的值介於 ***0~1***
  - Linear Regression：單純將 $feature * w + b$，output 可以是***任何值***

	Logistic Regression	***Linear Regression***
Step 1:	$f_{w,b}(x) = \sigma\left(\sum_i w_i x_i + b\right)$ Output: between 0 and 1	$f_{w,b}(x) = \sum_i w_i x_i + b$ Output: any value
Step 2:		
Step 3:		

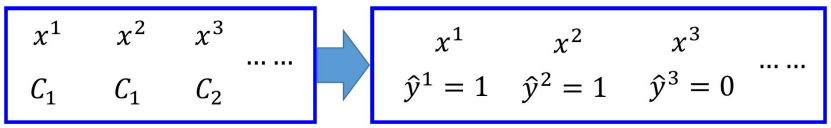
Step 2. Goodness of a Function

- **目標**：找出可以**最大化產生這 N 筆 training data 機率**的 w^*、b^*

  > w^*、$b^* = arg\ max_{w,b} L(w,b)$

- **轉化目標**：找出可以**最小化** $-lnL(w,b)$ 的 w^*、b^* (原因：簡化計算)

  > w^*、$b^* = arg\ max_{w,b} L(w,b) = arg\ min_{w,b} -lnL(w,b)$



$\hat{y}^n$: 1 for class 1, 0 for class 2

$$L(w,b) = f_{w,b}(x^1) f_{w,b}(x^2) \left(1 - f_{w,b}(x^3)\right) \cdots$$

$$w^*, b^* = arg \max_{w,b} L(w,b) \quad = \quad w^*, b^* = arg \min_{w,b} -lnL(w,b)$$

$$-lnL(w,b)$$

$$= -lnf_{w,b}(x^1) \rightarrow -[1\ lnf(x^1) + 0\ ln(1 - f(x^1))]$$

$$-lnf_{w,b}(x^2) \rightarrow -[1\ lnf(x^2) + 0\ ln(1 - f(x^2))]$$

$$-ln\left(1 - f_{w,b}(x^3)\right) \rightarrow -[0\ lnf(x^3) + 1\ ln(1 - f(x^3))]$$

$$\vdots$$

- **計算**
  - 左式、右式同取 $-ln$ ，相乘變成相加
  - 符號轉換：$\hat{y}$ 的值代表說，現在 x 屬於哪一個 $class$
  - 就可以寫成 $\Sigma - \hat{y}^n ln f_{w,b}(x^n) + (1 - \hat{y}^n) ln(1 - f_{w,b}(x^n))$ Σ 後的式子，其實是兩個 Bernouli distribution 的 Cross Entropy

- $H(p,q) = -\Sigma p(x) ln(q(x))$

- **比較**（Minimize 的對象）
  - Logistic Regression：(function 的 output 與 targer) 的 **cross entropy**
  - Linear Regression： (function 的 output 減 targer) 的 **square error**

	Logistic Regression	***Linear Regression***
Step 1:	$f_{w,b}(x) = \sigma\left(\sum_i w_i x_i + b\right)$ Output: between 0 and 1	$f_{w,b}(x) = \sum_i w_i x_i + b$ Output: any value
Step 2:	Training data: $(x^n, \hat{y}^n)$ $\hat{y}^n$: 1 for class 1, 0 for class 2 $L(f) = \sum_n l(f(x^n), \hat{y}^n)$	Training data: $(x^n, \hat{y}^n)$ $\hat{y}^n$: a real number $L(f) = \frac{1}{2}\sum_n (f(x^n) - \hat{y}^n)^2$

Cross entropy:

$$l(f(x^n), \hat{y}^n) = -\left[\hat{y}^n ln f(x^n) + (1-\hat{y}^n) ln\left(1 - f(x^n)\right)\right]$$

Step 3. Find the best function

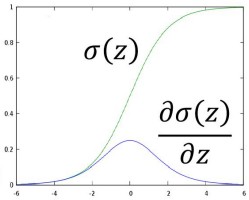
- 計算 $-lnL(w,b)$ 對 w_i 的微分

Step 3: Find the best function

$$\frac{-lnL(w,b)}{\partial w_i} = \sum_n -\left[\hat{y}^n \frac{ln f_{w,b}(x^n)}{\partial w_i} + (1-\hat{y}^n) \frac{ln\left(1 - f_{w,b}(x^n)\right)}{\partial w_i}\right]$$

$$\left(1 - f_{w,b}(x^n)\right) x_i^n$$

$$\frac{\partial ln f_{w,b}(x)}{\partial w_i} = \frac{\partial ln f_{w,b}(x)}{\partial z} \frac{\partial z}{\partial w_i} \qquad \frac{\partial z}{\partial w_i} = x_i$$



$$\frac{\partial ln\sigma(z)}{\partial z} = \frac{1}{\sigma(z)} \frac{\partial \sigma(z)}{\partial z} = \frac{1}{\sigma(z)} \sigma(z)\left(1 - \sigma(z)\right)$$

$$f_{w,b}(x) = \sigma(z) \qquad z = w \cdot x + b = \sum_i w_i x_i + b$$
$$= 1/1 + exp(-z)$$

Step 3: Find the best function

$$\frac{-lnL(w,b)}{\partial w_i} = \sum_n -\left[\hat{y}^n \frac{lnf_{w,b}(x^n)}{\partial w_i} + (1-\hat{y}^n)\frac{ln\left(1-f_{w,b}(x^n)\right)}{\partial w_i}\right]$$

$\left(1-f_{w,b}(x^n)\right)x_i^n$ $\quad -f_{w,b}(x^n)x_i^n$

$$\frac{\partial ln\left(1-f_{w,b}(x)\right)}{\partial w_i} = \frac{\partial ln\left(1-f_{w,b}(x)\right)}{\partial z}\frac{\partial z}{\partial w_i} \qquad \frac{\partial z}{\partial w_i} = x_i$$

$$\frac{\partial ln(1-\sigma(z))}{\partial z} = -\frac{1}{1-\sigma(z)}\frac{\partial \sigma(z)}{\partial z} = -\frac{1}{1-\sigma(z)}\sigma(z)(1-\sigma(z))$$

$$f_{w,b}(x) = \sigma(z) \qquad z = w\cdot x + b = \sum_i w_i x_i + b$$
$$= 1/1 + exp(-z)$$

- 計算結果：

> $\Sigma_n - (\hat{y}^n - f_{w,b}(x^n))x_i^n$
>
> $w_i^n \leftarrow w_i - \eta\Sigma_n - (\hat{y}^n - f_{w,b}(x^n))x_i^n$

下面的式子代表 w 的 update 取決於三件事情

1. η（Learning rate）：自己設定的
2. x_i ：來自於 data
3. $\hat{y}^n$：目標、$f_{w,b}(x^n)$ 現在 model 的 output $\hat{y}^n - f_{w,b}(x^n)$ 代表現在 function 的 output 跟理想目標的差距有多大，越遠，update 量就越大

Step 3: Find the best function

$$\frac{-lnL(w,b)}{\partial w_i} = \sum_n -\left[\hat{y}^n \frac{lnf_{w,b}(x^n)}{\partial w_i} + (1-\hat{y}^n)\frac{ln\left(1-f_{w,b}(x^n)\right)}{\partial w_i}\right]$$

$\left(1-f_{w,b}(x^n)\right)x_i^n$ $\quad -f_{w,b}(x^n)x_i^n$

$$= \sum_n -\left[\hat{y}^n\left(1-f_{w,b}(x^n)\right)x_i^n - (1-\hat{y}^n)f_{w,b}(x^n)x_i^n\right]$$

$$= \sum_n -\left[\hat{y}^n - \hat{y}^n f_{w,b}(x^n) - f_{w,b}(x^n) + \hat{y}^n f_{w,b}(x^n)\right]x_i^n$$

$$= \sum_n -\left(\hat{y}^n - f_{w,b}(x^n)\right)x_i^n$$

Larger difference, larger update

$$w_i \leftarrow w_i - \eta\sum_n -\left(\hat{y}^n - f_{w,b}(x^n)\right)x_i^n$$

- **比較**（Logistic Regression 跟 Linear Regression 做 Gradient Descent 時參數 update 的方式）
  - 相同：update 的式子
  - 不同：$\hat{y}^n$

> Logistic Regression 的 $\hat{y}^n$ 一定是 0 或 1
>
> Linear Regression 的 $\hat{y}^n$ 可以是任何實數

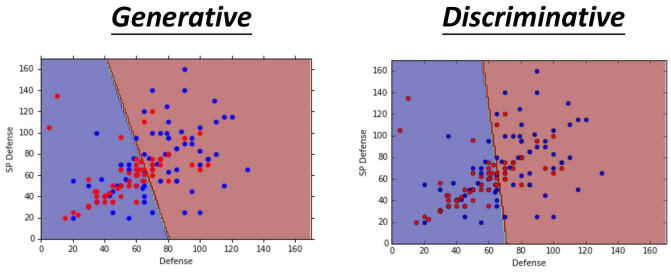
	Logistic Regression	***Linear Regression***
Step 1:	$f_{w,b}(x) = \sigma\left(\sum_i w_i x_i + b\right)$ Output: between 0 and 1	$f_{w,b}(x) = \sum_i w_i x_i + b$ Output: any value
Step 2:	Training data: $(x^n, \hat{y}^n)$ $\hat{y}^n$: 1 for class 1, 0 for class 2 $L(f) = \sum_n l(f(x^n), \hat{y}^n)$	Training data: $(x^n, \hat{y}^n)$ $\hat{y}^n$: a real number $L(f) = \frac{1}{2}\sum_n (f(x^n) - \hat{y}^n)^2$
Step 3:	Logistic regression: $w_i \leftarrow w_i - \eta \sum_n -\left(\hat{y}^n - f_{w,b}(x^n)\right) x_i^n$ Linear regression: $w_i \leftarrow w_i - \eta \sum_n -\left(\hat{y}^n - f_{w,b}(x^n)\right) x_i^n$	

Discriminative vs. Generative

- 定義
- **Discriminative** 的方法：如 Logistic Regression **Generative** 的方法：如使用 Gaussian 描述 Posterior probability
  - **相同**：function set、model（皆為 $P(C_1|x) = \sigma(w \cdot x + b)$）

    不同：兩者對 probability distribution 做不同的假設
    - Discriminative：沒有做任何假設
    - Generative：會假設機率分佈是 Gaussian, Bernoulli, Naive Bayes... 等等
- 例子分別應用兩者的結果
  - 防禦力&特殊防禦力的例子，藍色是水系的寶可夢，紅色是一般系的寶可夢，都使用 7 個 feature
  - Generative model 可獲得 73% 的正確率 Discriminative model 可獲得 79% 的正確率

Generative v.s. Discriminative



All: hp, att, sp att, de, sp de, speed

73% accuracy 79% accuracy

- 討論：**何時 Generative model 的表現較比 Discriminative model 好**
  - 資料量大小：
    - Discriminative model 因爲不做任何假設，故 performance 受資料影響很大 Generative model 會做假設（如同自行腦補），資料量很少時，較有優勢
    - 資料量小：Discriminative model 誤差較大，Generative model 表現可能較好 資料量大：Discriminative model 誤差較小，表現較有可能優於 Generative model
  - Noise 存在：
    - 資料有 noise 時，因為 label 本身就有些問題，故一些假設可能可以把有問題的 data 忽略掉 Generative model 的表現可能較 Discriminative 好
  - 分割資料來源：
    - Discriminative model 直接假設一個 posterior probability Generative model 可將 formulation 拆成 **prior** 跟 **class-dependent** 的 probability 兩項 而這兩項可以來自**不同的資料來源**
    - 舉例： 語音辨識使用 NN，是 discriminative 的方法； 但是整個語音辨識系統，是 generative 的 system。

      > prior 的部分使用文字的 data 處理，class-dependent 的部分，需要聲音和文字的配合。

Generative v.s. Discriminative

- Usually people believe discriminative model is better
- Benefit of generative model
  - With the assumption of probability distribution
    - less training data is needed
    - more robust to the noise
  - Priors and class-dependent probabilities can be estimated from different sources.

Process of Multi-class Classification

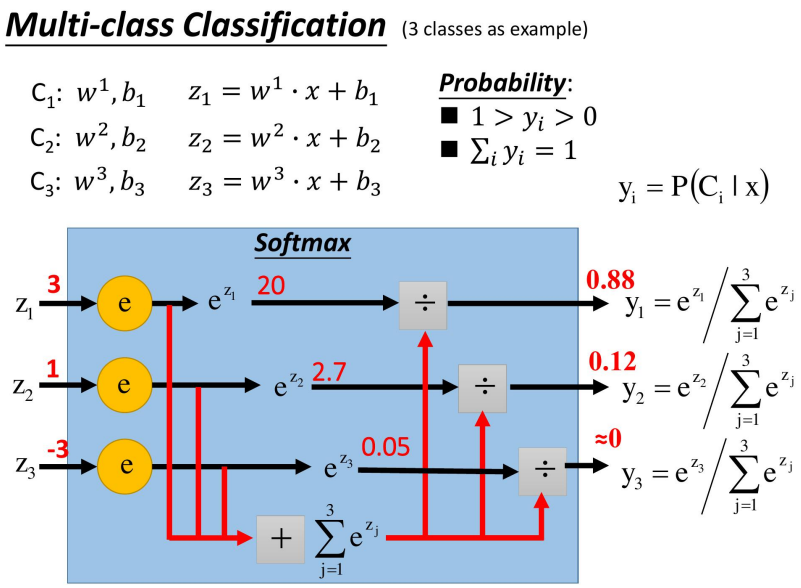
- 定義
  - 三個類別：C_1, C_2, C_3
  - 每個類別相對應的 weight, bias：w^1, w^2, w^3 (vector)，b_1, b_2, b_3 (scalar)
  - 要分類的對象：x
- 步驟
  1. 將 x 乘上 weight 加上 bias 得到 z

     > $z_1 = w^1 * x + b_1$ ex. $z_1 = 3$
     >
     > $z_2 = w^2 * x + b_2$ ex. $z_2 = 1$
     >
     > $z_3 = w^3 * x + b_3$ ex. $z_3 = -3$

  2. 將 z 丟入 Softmax function

     > 取 exponential 得 e^{z1}, e^{z2}, e^{z3} ，相加得 total sum = $\Sigma_{j=1}^{3} e^{z_j}$

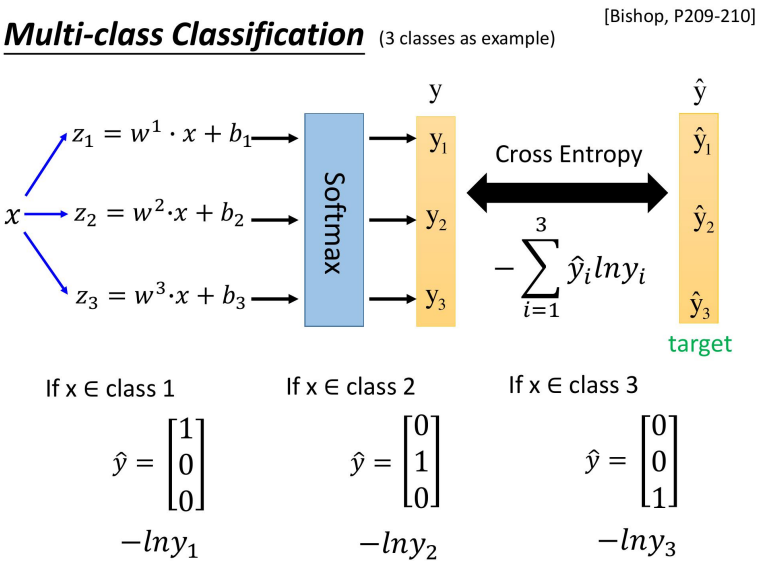
     > 各項除以 total sum (做 normalization)，得 output, $y = (y_1, y_2, y_3)$
     >
     > $y_1 = s^{z_1} / \Sigma_{j=1}^{3} e^{z_j}$ ex. 計算得 $y_1 = 0.88$
     >
     > $y_2 = s^{z_2} / \Sigma_{j=1}^{3} e^{z_j}$ ex. 計算得 $y_2 = 0.12$
     >
     > $y_3 = s^{z_3} / \Sigma_{j=1}^{3} e^{z_j}$ ex. 計算得 $y_3 = 0$



3. Minimize Cross Entropy
   - 計算 y 跟目標函數 $\hat{y}$ 之間的 cross entropy：$-\Sigma_{i=1}^{3}\hat{y}lny_i$

     $$\hat{y}_1 = \begin{bmatrix}1\\0\\0\end{bmatrix}\ \hat{y}_2 = \begin{bmatrix}0\\1\\0\end{bmatrix}\ \hat{y}_3 = \begin{bmatrix}0\\0\\1\end{bmatrix}$$

   - 列出 maximum likelihood 的 function，經過整理即可得 minimize cross entropy



- Softmax
  - 對最大值做強化（取 exponential 時使大值跟小值間的差距更大）
  - 經過 Softmax function 後，output 值 (y) 介於 0~1 之間
  - y_i 即為第 i 個 class (z_i)的 posterior probability

    > $y_1 = 0.88$ 代表 x 屬於 class1 的機率是 88% $y_2 = 0.12$ 代表 x 屬於 class2 的機率是 12%
    >
    > $y_3 = 0$ 代表 x 屬於 class3 的機率趨近於 0

- Softmax 中為何使用 exponential

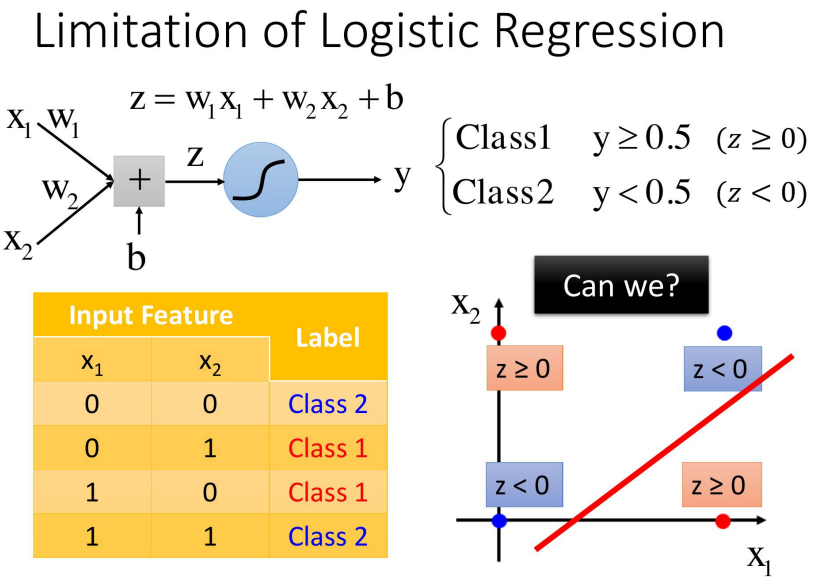
  > 可以參考 Bishop 教科書的推導，也可搜尋 "Maximum Entropy" 獲得更多資訊

Limitation of Logistic Regression

- Logistic Regression 有時無法直接對 data 做分類， 因為兩個 class 之間的 boundary 是一直線，無法好好地將資料分割
- 舉例：
  - 假設，如左下表格

    > class1 有兩筆 data：(0,1)、(1,0) class2 有兩筆 data：(0,0)、(1,1)

  - 如右下圖，我們無法以 Logistic Regression 好好地將紅色、藍色分成兩邊 因爲其 boundary 是一直線

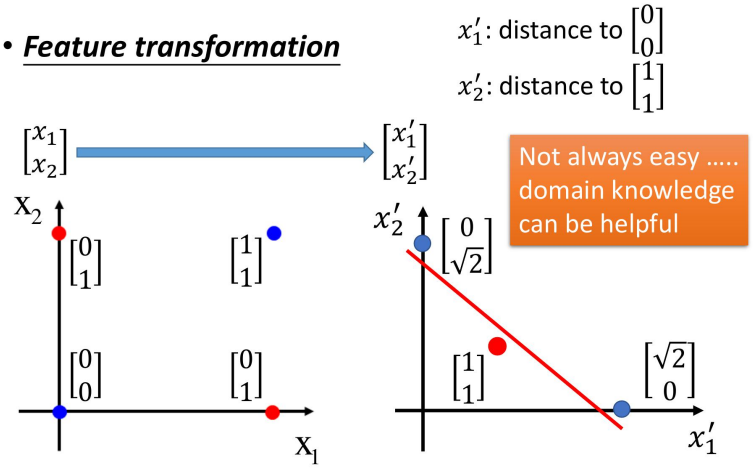


- 解決：**Feature Transformation**
  - 將原來的 x_1, x_2 做一些轉化後，找到一個較好的 feature space，轉化成 $x_1^{'}, x_2^{'}$ 讓 Logistic Regression 可以處理
  - 舉例：

    > 定義 $x_1^{'}$ 是某一點到 (0,0) 的距離，$x_2^{'}$ 是某一點到 (1,1) 的距離 轉換後，如右下圖，紅色的點重疊在一起，而 Logistic Regression 可找到 boundary 分開

Limitation of Logistic Regression

• ***Feature transformation***

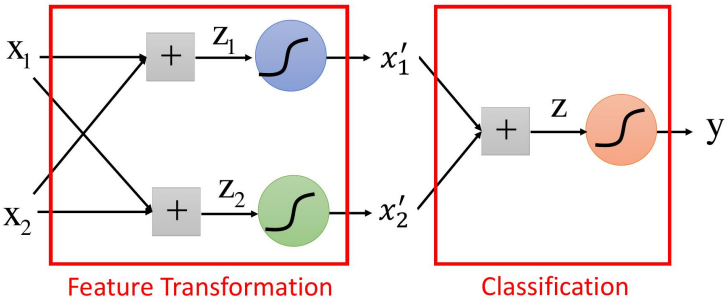


- 如何讓機器自己產生 Feature Transformation
  - 將許多個 Logistic Regression **串接**起來，如下圖

> 前面兩個 Logistic Regression (藍色, 綠色)就是在做 Feature Transformation 轉換後，再由紅色的 Logistic Regression 做分類

Limitation of Logistic Regression

• Cascading logistic regression models



(ignore bias in this figure)

- **Neuron & Neural Network**
  - Neuron：我們將每一個 Logistic Regression 稱作 Neuron
  - Neural network：這些 Logistic Regression 串接起來就稱作 Neural network (類神經網路)
  - This is Deep Learning!