309513061 電信甲 簡詠倫

1. Bayesian Linear Regression

- **Why we need the basis function for linear regression?**

在線性回歸模型中，我們試圖將輸入 x 做線性轉換來預測目標 t 值，但假設我們的最佳預測 t 的函數並不是對 x 的線性函數，我們便不能很好的預測 t 值。這時候應用 basis function 的好處就顯得重要。由於 basis function 並不是對 x 的線性函數，我們便能將輸入 x 代進合適的 basis function 後再對其做線性組合，便能有效降低錯誤，達到更好的擬合效果。

- **Prove that the predictive distribution**

Formulas in page 93.

$$p(x) = \mathcal{N}(x \mid \mu, \Lambda^{-1})$$

$$p(y \mid x) = \mathcal{N}(y \mid Ax+b, L^{-1})$$

$$\xrightarrow{\text{可得}} p(y) = \mathcal{N}(y \mid A\mu+b, L^{-1}+A\Lambda^{-1}A^T) = \int p(y \mid x)p(x)dx$$

又, $p(w) = \mathcal{N}(w \mid 0, \alpha^{-1}I)$, $p(t \mid x, w) = \mathcal{N}(t_n \mid y(x,w), \beta^{-1}I)$

且, $p(w \mid x, t) \propto p(t \mid x, w)\, p(w) \propto \exp(-\frac{1}{2}w^T\alpha w - \frac{1}{2}\beta(t_n - w^T\phi(x))^2)$

$\Longrightarrow$ 省略 $-\frac{1}{2}$, 並湊成高斯分佈的型式

$$\Rightarrow w^T \alpha w + \beta (t_n - w^T \phi(x_n))^2$$

$$= w^T \alpha w + \beta w^T \sum_{n=1}^{N} \phi(x_n)\phi(x_n)^T w - 2\beta w^T \sum_{n=1}^{N} \phi(x_n) t_n + const$$

$$= w^T \left(\alpha I + \beta \sum_{n=1}^{N} \phi(x_n)\phi(x_n)^T\right) w - 2 w^T \left(\beta \sum_{n=1}^{N} \phi(x_n) t_n\right) + const$$

$$\left(\because -\frac{1}{2}(x-\mu)^T \Sigma^{-1} (x-\mu) = -\frac{1}{2} x^T \Sigma^{-1} x + x^T \Sigma^{-1} \mu\right)$$

可推得 $\Rightarrow$ $p(w|X,t) = N(w|m, S^{-1})$,

其中 $S^{-1} = \alpha I + \beta \sum_{n=1}^{N} \phi(x_n)\phi(x_n)^T$

$m = S\beta \sum_{n=1}^{N} \phi(x_n) t_n$

再由 $p(t|x,X,t) = \int_{-\infty}^{\infty} p(t|x,w)\, p(w|X,t)\, dw$ 與 93頁公式對照

$p(y) = \int_{-\infty}^{\infty} p(y|x)\, p(x)\, dx$

$$\int_{-\infty}^{\infty} p(t|x,w)\, p(w|X,t)\, dw = \int_{-\infty}^{\infty} N(t_n | w^T\phi(x), \beta^{-1} I)\, N(w|m,S)\, dw$$

對照93頁公式即得: $m=\mu$, $S=\Lambda^{-1}$, $A=\phi(x)^T$, $b=0$, $\beta^{-1}=L^{-1}$

因此 $p(t|x,X,t) = N(t|m(x), s^2(x))$

其中 $m(x) = \beta \phi(x)^T S \sum_{n=1}^{N} \phi(x_n) t_n$

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

- **Could we use linear regression function for classification? Why or why not? Explain it!**

Linear regression 不太適合用來做分類。在分類問題中，大多偏向的是二元的結果（Yes or No）。至於線性迴歸不適合解決分類的原因舉個簡單的例子。在 Fig.1 中，算出大於 0.5 的值會被歸類在 Yes 區。但若出現一個新的

樣本如 Fig.2，則新的迴歸線會如藍線所示，即會容易造成判斷錯誤，因此

線性迴歸並不適合直接用來解決分類問題的。

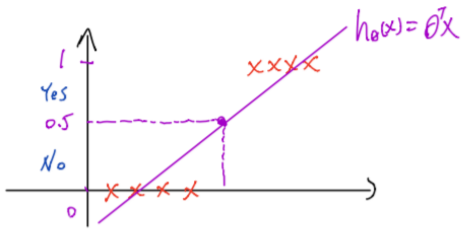


Fig.1

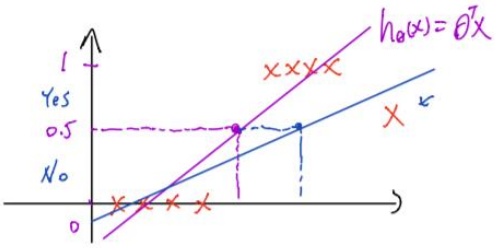


Fig.2

2. Linear Regression

1. Feature select

（a）

```
M=1
Training RMS = 0.06144009611792689
Valid RMS = 0.05158237423453033
```

```
M=2
Training RMS = 0.059005800736452216
Valid RMS = 0.054588785438852377
```

（b）我使用 L1 Regularization 的性質來決定最有貢獻的 feature。由於 L1 的性質能讓更多的參數變為 0，利用此特性可用來做 feature selection。我在誤差函數中加上 Regularization 項並套用在 gradient descent 中，來找出重要的參數。結果於 Fig.3。

```
weight_training3
```

```
[0.7152169704307553,
 0.017620798041575264,
 0.01747948606239043,
 0.007925576809003552,
 -0.0003576282911529555,
 0.012036189377204826,
 0.07240692108766486,
 0.011793848450350865]
```

Fig.3

由上圖參數結果對應回原資料，可知 CGPA 的權重參數是最大的，因此可判斷 CGPA 為最有貢獻的特徵。而 SOP 及 University 則相對貢獻較小。

2. Maximum likelihood approach

（a）

我選擇的是 Sigmoidal basis function，我認為 Polynomial basis 次方越高會造成模型複雜度的上升，就會需要更多的 Training data 來實作，故暫不考慮。而 Sigmoidal 及 Gaussian 的 data 有經過

Normalize，可提高模型精確度及收斂速度。而高斯由於是呈現鐘形分佈，較適合數值較集中於平均的資料。而 Sigmoidal function 根據資料大小將輸入壓在 0~1 之間，可能較能反映真實的情形，因此我選擇 Sigmoidal function。

（b）

Sigmoidal function 為：

$$\varphi_j(x) = \sigma(\frac{x - \mu_j}{s})$$

其中:

$$\sigma(a) = \frac{1}{1 + e^{-a}}$$

將輸出壓在(0,1)的範圍之間的非線性函數。我在這裡想利用上一大題的結論，將兩個權重較低的 features 去除，來降低模型的複雜度。而程式跑出來的結果如下。

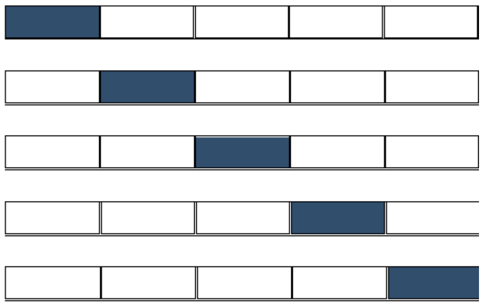
```
Sigmoid basis all features
Training RMS = 0.058375327550373876
Valid RMS = 0.06974456742184741

Sigmoid basis 5 features
Training RMS = 0.05849752556523983
Valid RMS = 0.07014837750511017
```

可得我雖然抽取掉兩個 features，但在誤差表現的部分相差不大且複雜度降低，因此我模型會使用 Sigmoidal function 當作基底函數，並只使用 5 個 features 來訓練。

（c）

我使用 5-fold cross validation 進行驗證。



500 筆資料分割成 5 份，每次便取 100 筆當作 Validation data

	取全部 features 訓練			取 5 個 features 訓練		
	Training	Validation		Training	Validation	
Fold1	0.0602639	0.0621191		0.0605878	0.0617394	
Fold2	0.0623260	0.0538624	Under-fitting	0.0624706	0.0544065	Under-fitting
Fold3	0.0604093	0.0620714		0.0609324	0.0606973	
Fold4	0.0601389	0.0629296		0.0603373	0.0632350	
Fold5	0.0583753	0.0697445	Over-fitting	0.0584975	0.0701483	Over-fitting
平均	0.0603027	0.0621454		0.0605651	0.0620453	

RMS						
原本	0.0583753	0.069744		0.0584975	0.0701483	

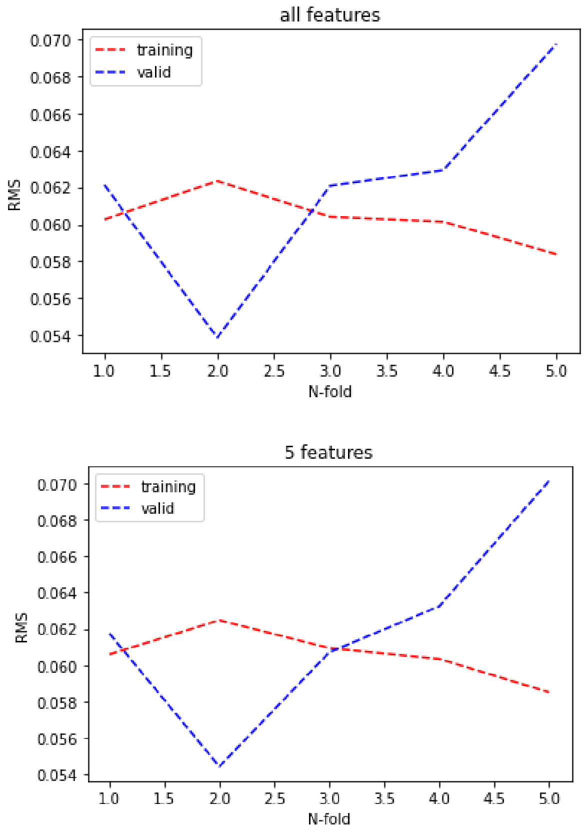


Fig.4

本來以為 5 個 fold 的數據會差不多，但其中還是有些不同，甚至有 over-fitting 或 under-fitting 的現象。但平均起來的表現比原本切割資料的方式好一些。因此可以說資料切分的方式跟選擇模型都佔了訓練結果好壞的重要因素。

3. Maximum a posterior approach

（a）

MAP 和 ML 最大的不同是，MAP 需要考慮模型參數的事前機率，而在 ML 中我們是不考慮事前機率的，專注在求得最大 Likelihood 上。而若在 MAP 中我們的事前機率假定為高斯分佈，則錯誤函數為：

$$E_D(w) + \lambda E_W(w)$$

$$E_W(w) = \frac{1}{2} w^T w$$

$$E_D(w) = \frac{1}{2} \sum_{n=1}^{N} \{t_n - w^T \varphi(x_n)\}^2$$

$$\frac{1}{2} \sum_{n=1}^{N} \{t_n - w^T \varphi(x_n)\}^2 + \frac{\lambda}{2} w^T w$$

$$\boxed{\mathrm{w} = (\lambda I + \varphi^T \varphi)^{-1} \varphi^T t}$$

由於多了 lambda 項，能讓訓練出的模型更加平滑，減少 over-fitting 或 under-fitting 的產生。

（b）

我將只使用 5 個 features，並用 Sigmoidal function 當作基底函數的模型代入，並用不同的 lambda 值來觀察結果。如下圖：

```
lambda = 0
Training RMS = 0.060095462716651955
Valid RMS = 0.0637923030156136

lambda = 0.01
Training RMS = 0.06009549453581747
Valid RMS = 0.06379436516936092

lambda = 0.1
Training RMS = 0.060098488713130685
Valid RMS = 0.0638131403743131

lambda = 1
Training RMS = 0.060292080359305715
Valid RMS = 0.06399822861255183

lambda = 10
Training RMS = 0.06326414251877281
Valid RMS = 0.06478048329340885

lambda = 30
Training RMS = 0.06921288207599371
Valid RMS = 0.06719470721965735
```

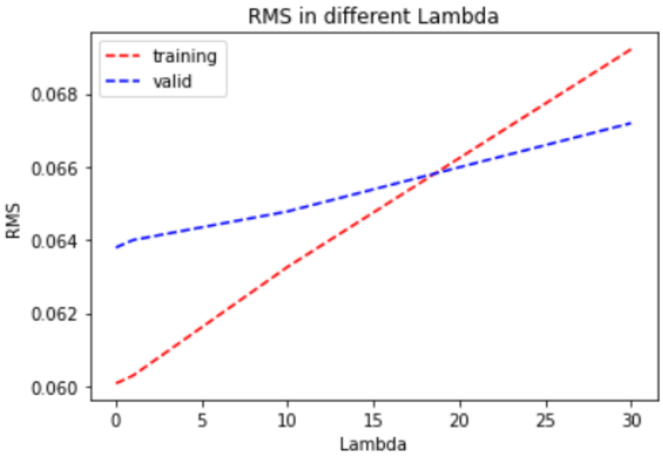


Fig.5

接下來再做 5 fold cross validation 做驗證，結果如下：

```
Lambda = 0的5-fold RMS
train
[0.059828999316874826, 0.061695330066706126, 0.05964977832661263, 0.06236580421502309, 0.05926635415112024]

平均為:0.06056125321526738

valid
[0.06490819324157625, 0.0577676221872536, 0.06591242907352043, 0.05406594515507924, 0.06770384727058375]

平均為:0.062071607385602655

Lambda = 0.01的5-fold RMS
train
[0.05982904038688345, 0.06169536072831166, 0.0596498117295086, 0.06236584097969546, 0.05926640988010301]

平均為:0.060561292740900434

valid
[0.06490111469888883, 0.05777735525512537, 0.06591500451164162, 0.05406472647817993, 0.06768534529811117]

平均為:0.062068709248389385

Lambda = 0.1的5-fold RMS
train
[0.05983288417868764, 0.06169824058366792, 0.059652942343190345, 0.062369283079489406, 0.05927158881164662]

平均為:0.06056498779933638


valid
[0.0648418836875566, 0.0578651670425736, 0.06593910011960782, 0.05405743716841653, 0.0675308933691945]

平均為:0.06204689627746981

Lambda = 1的5-fold RMS
train
[0.06007153288780299, 0.06188191920123788, 0.05984951153170849, 0.06258321153114908, 0.05957448900039866]

平均為:0.06079213283045942

valid
[0.06453454872675643, 0.058725692286042665, 0.06621174558788705, 0.054201888704445075, 0.06675669106173579]

平均為:0.0620861132733734

Lambda = 10的5-fold RMS
train
[0.06336510996626028, 0.06463383057542135, 0.0627004325619815, 0.06549256895469721, 0.06284502323866585]


平均為:0.06380739305940523

valid
[0.0656246185290917, 0.06466755459205595, 0.06744760124692407, 0.05699828236868015, 0.06842260446729395]

平均為:0.06463213224080916

Lambda = 30的5-fold RMS
train
[0.06941344616240071, 0.07014616151697248, 0.06852033359989111, 0.07106460571479949, 0.06843508194863143]

平均為:0.06951592578853905

valid
[0.0711406176934409, 0.0747247251548205, 0.06927977523752203, 0.06260586750048192, 0.07266554887245255]

平均為:0.07008330689174358
```

由結果也可大致觀察出 over-fitting 的狀況趨緩，但整體的錯誤率

是提高的。

（c）

由結果發現，給定合適的 lambda 值，MAP 確實能減少 over-fitting 等現象，和第一題論點一致。但比起 ML，MAP 的誤差也是相對較高的。因此兩種方式的選擇皆會對模型產生重大的影響，如何取決也是我們重要的課題之一。