Date 1·28

Classification = Probabilistic Generative Model

· 找一个function，它的input是个object x，
output是这个object属於哪个class

for example.

1 金融：要不要借款給某一个人
Input：收入，工作，年紀，有无欠债
output：要不要借錢

2、醫療的診斷
Input：某一个人的症狀，年紀，性別，救醫歷史
output：生的是哪一种病

3、手寫文字辨識
Input：手寫一个字
output：辨识出是什麼

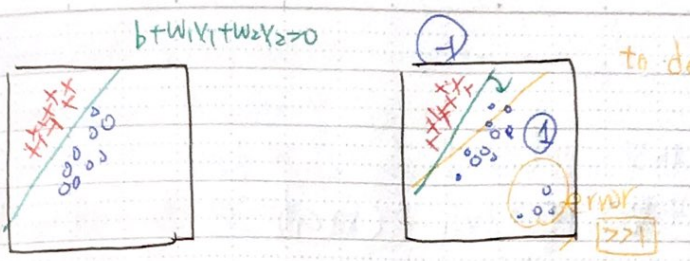
4、人臉辨識
Input：人臉
output：who is the person

· 一隻pokemon是由7个数字所組成的一个vector
↓
輸入一个function

Train data 400 ↓
Testing data 400 ↑

⇒ Classification as Regression
·[binary 當作example.)
Training x < class 1 ? Let class 1=1
class 2 ? class 2=-1
Testing = Close to 1 → class 1
-1 class 2.

Date



※ Regression 定义 function 好坏的方式对 classification 来说不適用

※ Multiple class

if now have class 1, class 2, class 3.

如果把這个當作 Regression 的問題來處理

沒有辦法得到好的結果

理想的作法

⇒ Classification 时 output 是離散的

Model $X \rightarrow f(X) \cdot \begin{cases} \text{class 1} & \text{if } f(X)>0 \\ \text{class 2} & \text{o.w.} \end{cases}$

Loss function

$L(f) = \sum_n \underbrace{\delta}_{\text{delta}}(f(x^n) \neq \hat{y}^n)$ // Index function 的概念

where $f(x^n) \neq \hat{y}^n \Rightarrow 1$

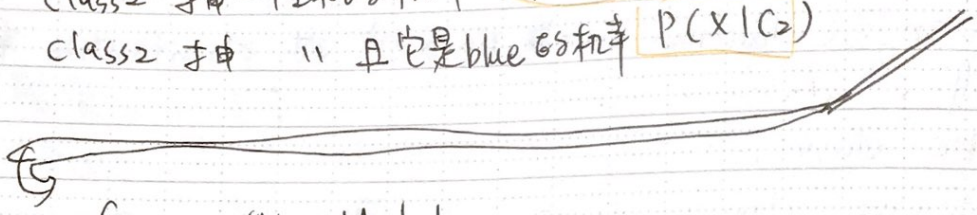
$f(x^n) = \hat{y}^n \Rightarrow 0$

Find the best function

example = Perceptron, SVM

Two class.

class1 抽一个球的机率 = $P(C_1)$
class1 抽一个球且它是蓝球的机率 $P(X|C_1)$ ⇒ 透过 training data 算出
class2 抽一个红球的机率 = $P(C_2)$
class2 抽 " 且它是blue的机率 $P(X|C_2)$



Generative Model.
算某一个 X 出现的机率

$$P(X) = P(C_1)P(X|C_1) + P(C_2)P(X|C_2)$$

⇒ can 自己产生 X

Prior

water & normal type with ID<400 ⇒ training, >400 ⇒ test
Train中 水 = 79
normal = 61

$$P(C_1) = \frac{79}{79+61}$$
$$P(C_2) = \frac{61}{79+61}$$

※每个 pokemon 都是用一堆 feature 来描述

Date

Gaussion Distribution. $f_{x\mu} = \frac{1}{(2\pi)^{\frac{D}{2}}} \frac{1}{|\Sigma|^{\frac{1}{2}}} \exp\left[-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)\right]$

input = vector x

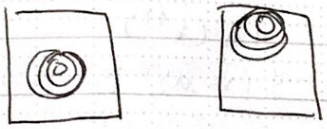
output = 机率密度 $\propto$ 机率

function 的形状由 μ 和 Σ 决定

μ ↓ mean　　Σ ↓ covariance matrix

同样的 Σ，不同的 μ

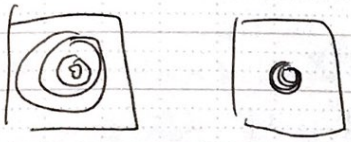
⇒ 机率密度(最高点的地方) 是一不样的



同样的 μ，不同的 Σ

⇒ 机率密度(分布的最高点) 是一样的

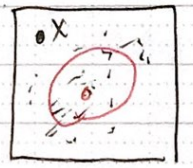
but 分散的程度 not same



Probability from class.

- Assume point sampled from Gaussion Distribution.
- Find the Probability for new point x

now know $\mu = \begin{bmatrix} 1.5 \\ 1.3 \end{bmatrix}$ $\Sigma = \begin{bmatrix} 8.4 & 3.2 \\ 3.2 & 9.2 \end{bmatrix}$



If x 越靠近中心点(μ) = sample 的机率越大

越远

how to find μ and Σ

⇒ Maximum Likelihood

#Maximum likelihood

☆ The Gaussian 在任何的一組 μ and Σ 都可以生成所需的东

⇒ 只是有些地方机率很高，有些地方机率很低

But 沒有一个地方的机率是 0

⇒ 其 likelihood 不一样

⇒ likelihood of a Gaussion with μ and Σ

= the probability of the Gaussion samples. $x^1, \dots x^{79}$

$L(\mu, \Sigma) = f_{\mu,\Sigma}(x^1) f_{\mu,\Sigma}(x^2) \cdots f_{\mu}(x^{79})$

$f_{\mu,\Sigma}(x) = \frac{1}{(2\pi)^{\frac{D}{2}}} \frac{1}{|\Sigma|^{\frac{1}{2}}} \exp\left(-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)\right]$

$\mu^*, \Sigma^* = \arg \max_{\mu,\Sigma} L(\mu, \Sigma)$

$\mu^* = \frac{1}{79} \sum_{n=1}^{79} x^n$ (or 把 f 对 μ^* 取微分，找 $=0$ 的解)

- $\Sigma^* = \frac{1}{79} \sum_{n=1}^{79} (x^n - \mu^*)(x^n - \mu^*)^T$ → Transform (or f 对 Σ^* 微分，再对 $x-\mu$ 的)

#Now can do classification.

$$P(C_1|x) = \frac{P(x|C_1)P(C_1)}{P(x|C_1)P(C_1) + P(x|C_2)P(C_2)}$$

$P(x|C_1) = f_{\mu^1,\Sigma^1}(x)$

$\mu^1 = \begin{bmatrix} 75 \\ 71.3 \end{bmatrix}$ $\Sigma^1 = \begin{bmatrix} 814 & 321 \\ 321 & 929 \end{bmatrix}$

$P(x|C_2) = f_{\mu^2,\Sigma^2}(x)$

$\mu^2 = \begin{bmatrix} 57.6 \\ 59.8 \end{bmatrix}$ $\Sigma^2 = \begin{bmatrix} 841 & 432 \\ 432 & 685 \end{bmatrix}$

$P(C_1) = 0.56$, $P(C_2) = 0.44$

If $P(C_1|x) > 0.5$ ⇒ $x \in$ class1 (水系)

小結：if we want to do pokemon 屬性分類

- 我們假設一个机率 model < required probability

每一个 class 自己的 distribution

每一个 class 自己的机率，Use Gaussion

Date

比較常見的作法：不同的 class 可以 share 同一個 covariance matrix
⇒ less 參數 ⇒ 不太會 overfitting

If now covariance matrix 的 feature size 很大時
⇒ 它的 size 增長很快

If now have different Σ
⇒ model 參數太多.
⇒ 可能 overfitting

Modifying Model (by Bishop ch 4.2.2)

- Maximum likelihood

water = (μ^1, Σ)

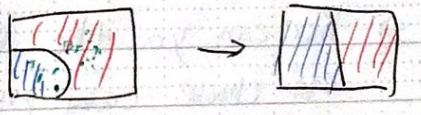
normal = (μ^2, Σ)

$L(\mu^1, \mu^2, \Sigma) = f_{\mu^1,\Sigma}(x^1) \times \cdots f_{\mu^1,\Sigma}(x^{79}) \times f_{\mu^2,\Sigma}(x^{80}) \times \cdots f_{\mu^2,\Sigma}(x^{140})$

- if μ^1, μ^2 same, Σ^1 Σ^2 not same

$\Rightarrow \Sigma = \frac{79}{140}\Sigma^1 + \frac{61}{140}\Sigma^2$

if now 共同 covariance
⇒ The boundary is linear
(原本是曲線) : Gaussion



All = total : hp, att, dev, sp_de, speed
54% accuracy → 79 accuracy

Three Step

回顧機率的 model，we know that machine learning 有 3 step

now 用機率 model 也是 3 step

1. Function set (Model)

$$f(x)$$

$$X \rightarrow \boxed{P(C_1|X) = \frac{P(X|C_1)P(C_1)}{P(X|C_1)P(C_1)+P(X|C_2)P(C_2)}}$$

$$\boxed{f(x) = \begin{cases} \text{class 1} & \text{if } P(C_1|X) > 0.5 \\ \text{class 2} & \text{O.W.} \end{cases}}$$

① probability distribution ⇒ 決定 model 的參數

② required probability

will get 不同的 probability distribution

choose different Σ, μ

2. Goodness of a function

⇒ evaluate function set 每個 function 的好壞

⇒ 找一個 probability distribution，可以最大化這個 data 的 likelihood

i.e. $\mu^*, \Sigma^* = \arg \max_{\mu,\Sigma} L(\mu, \Sigma)$

3. Find the best function

Probability distribution

⇒ You can use the distribution you like

if choose easy probability model = 參數少，bias 大，variance 小

complex = 大，小 大

⇒ Use data set 去決定 which is better

$$P(X|C_1) = P(x_1|C_1)\,P(x_2|C_1)\cdots P(x_K|C_1)\cdots$$

X 是 by 一組 feature 決定 $[x_1, \cdots x_K]$

and $x_1 \cdots x_K$ is independent

(它從機率 model 產生出的機率 is independent)

1 維的 Gaussian

⇒ covariance matrix ⇒ diagonal

⇒ 不是對角線的地方=0

只有對角線的地方有值

⇒ 可以 decrease 參數量

⇒ can get easy model

⇒ 結果表現不太好

⇒ still need covariance

for binary feature (是 or 不是)
⇒ use Bernoulli distribution

If now assume all dimension are independent
⇒ 不 model feature & feature 間 covariance 的关係
⇒ using Naive Bayes Classifier
(强不强 depend on 假設是不是精準的)

Posterior Probability

$$P(C_1|X) = \frac{P(X|C_1)P(C_1)}{P(X|C_1)P(C_1) + P(X|C_2)P(C_2)}$$

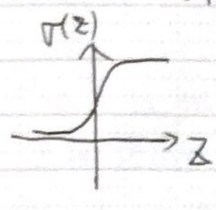
$$= \frac{1}{1 + \frac{P(X|C_2)P(C_2)}{P(X|C_1)P(C_1)}}$$

$$= \frac{1}{1+\exp(-z)} \quad \text{w/ } z = \ln \frac{P(X|C_1)P(C_1)}{P(X|C_2)P(C_2)}$$

$$= \sigma(z)$$

sigmoid function

=



$z \to \infty, \sigma(z) \doteq 1$

$\to -\infty, \sigma(z) \doteq 0$

$$z = \ln \frac{P(X|C_1)P(C_1)}{P(X|C_2)P(C_2)}$$

$$= \ln \frac{P(X|C_1)}{P(X|C_2)} + \ln \boxed{\frac{P(C_1)}{P(C_2)}} \rightarrow \frac{\frac{N_1}{N_1+N_2}}{\frac{N_2}{N_1+N_2}} = \frac{N_1}{N_2}$$

$$= \ln\left(\frac{|\Sigma^2|^{\frac{1}{2}}}{|\Sigma^1|^{\frac{1}{2}}} \exp\left\{-\frac{1}{2}\left[(x-\mu^1)^T(\Sigma^1)^{-1}(x-\mu^1) - (x-\mu^2)^T(\Sigma^2)^{-1}(x-\mu^2)\right]\right\}\right) + \ln\frac{N_1}{N_2}$$

$$= \ln\frac{|\Sigma^2|^{\frac{1}{2}}}{|\Sigma^1|^{\frac{1}{2}}} - \frac{1}{2}x^T(\Sigma^1)^{-1}x + (\mu^1)^T(\Sigma^1)^{-1}x - \frac{1}{2}(\mu^1)^T(\Sigma^1)^{-1}\mu^1$$
$$+ \frac{1}{2}x^T(\Sigma^2)^{-1}x - (\mu^2)^T(\Sigma^2)^{-1}x + \frac{1}{2}(\mu^2)^T(\Sigma^2)^{-1}\mu^2 + \ln\frac{N_1}{N_2}$$

$\Sigma_1 = \Sigma_2 = \Sigma$

$$= \underbrace{(\mu^1-\mu^2)^T\Sigma^{-1}}_{w^T} x \; \underbrace{-\frac{1}{2}(\mu^1)^T(\Sigma^1)^{-1}\mu^1 + \frac{1}{2}(\mu^2)^T(\Sigma^2)^{-1}\mu^2 + \ln\frac{N_1}{N_2}}_{b}$$

Date

$P(C_1|X) = \sigma(z)$

$= \sigma(wx + b)$

⇒ We know that $\Sigma_1 = \Sigma_2 = \Sigma$ (share Σ)

the boundary is linear

⇒ In generative model, we ~~asst~~ estimate N_1 N_2, μ^1, μ^2, Σ

Then we have w and b

⇒ 直接把 w 和 b 找出來

⇒ Logistic Regression

Date

Logistic Regression

→ 找 w & b 決定的 model

	Logistic Regression	Linear Regression
step 1	$f_{w,b}(x) = \sigma(\sum_i w_i x_i + b)$ output = between 0 & 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 class 2 $L(f) = \sum_n C(f(x^n), \hat{y}^n)$	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	$w_i = w_i - \eta \sum_n -(\hat{y}^n - f_{w,b}(x^n)) x_i^n$ same	

- Cross entropy $C(f(x^n), \hat{y}^n) = -[\hat{y}^n \ln(x^n) + (1-\hat{y}^n)\ln(1-f(x^n))$

  $f(x^n)$ is a bernoulli distribution

  $\hat{y}^n$

  ⇒ want 這兩個的 distribution 越接近越好

step2:

Training data = $\{(x^1, \hat{y}^1=1), (x^2, \hat{y}^2=1), (x^3, \hat{y}^3=0), \dots, (x^N, \hat{y}^N=1)\}$

Posterior probability

Assume the data is generated based on $f_{w,b}(x) = P_{w,b}(C_1|x)$

(i.e 主面這組 training data 是由 $P_{w,b}(C_1|x)$ 產生百分)

Now given w and b, what is the probability?

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

Most likely $w^* b^*$ (最大化這個機率的 w^*, b^*)

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

$-\ln L(w,b)$

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

$= -[\hat{y}^1 \ln f_{w,b}(x^1) + (1-\hat{y}^1)\ln(1-f_{w,b}(x^1))]$

$-[\hat{y}^2 \ln f_{w,b}(x^2) + (1-\hat{y}^2)\ln(1-f_{w,b}(x^2))]$

$-[\hat{y}^3 \ln f_{w,b}(x^3) + (1-\hat{y}^3)\ln(1-f_{w,b}(x^3))]$

$\cdots$

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

2個 bernoulli distribution 的 cross entropy

Date

2 distribution

Distribution P		Distribution q
$P(x=1) = \hat{y}^n$	cross entropy	$q(x=1) = f(x^n)$
$P(x=0) = 1-\hat{y}^n$		$q(x=0) = 1-f(x^n)$

$H(P,q) = -\sum_x P(x) \ln(q(x))$

※ cross entropy：這兩個 distribution 有多接近

if 兩個 distribution same 其 cross entropy = 0

$\Rightarrow$ $\hat{y}^n * \ln(f(x^n)) + (1-\hat{y}^n) * \ln(1-f(x^n))$ is the cross entropy

Step 3: Find the best function

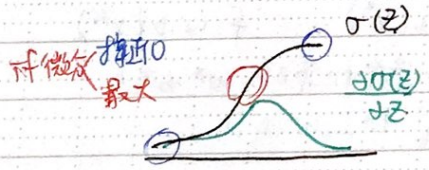
$$-\frac{\partial \ln L(w,b)}{\partial w_i} = \sum_n -[\hat{y}^n \underbrace{\frac{\ln f_{w,b}(x^n)}{\partial w_i}}_{[1-f_{w,b}(x)]\cdot x_i^n} + (1-\hat{y}^n) \underbrace{\frac{\ln(1-f_{w,b}(x^n))}{\partial w_i}}_{-f_{w,b}(x)\cdot x_i^n}]$$

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

$f_{w,b}(x) = \sigma(z) = \frac{1}{1+\exp(-z)}$

$z = w\cdot x + b = \sum_i w_i x_i + b$

$$\begin{cases} \frac{\partial z}{\partial w_i} = x_i \\ \frac{\partial \ln f_{w,b}(x)}{\partial z} = \frac{\partial \ln \sigma(z)}{\partial z} = \frac{1}{\sigma(z)}\cdot\frac{\partial \sigma(z)}{\partial z} = \frac{1}{\sigma(z)}\cdot\sigma(z)(1-\sigma(z)) = 1-\sigma(z) \end{cases}$$



$$\frac{\partial \ln(1-f_{w,b}(x^n))}{\partial w_i} = \frac{\partial \ln(1-f_{w,b}(x^n))}{\partial z}\frac{\partial z}{\partial w_i}$$

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

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

$$= -\frac{1}{1-\sigma(z)}\cdot\sigma(z)(1-\sigma(z))\cdot x_i = -\sigma(z)\cdot x_i$$

Date

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

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

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

If now use gradient descent to update

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

w 的 update 取決 3 件事

① η = learning rate = 自己調的

x_i : 來自 data

$\hat{y}^n - f_{w,b}(x^n)$: 現在 f 的 output 和 理想的目標差距有多大

目標　現在 model 的 output

Logistic Regression + Square Error

Step 1 : $f_{w,b}(x) = \sigma\left(\sum_i w_i x_i + b\right)$

Step 2 : Training data $(x^n, \hat{y}^n)$　$\hat{y}^n = \begin{cases} 1 & \text{class 1} \\ 0 & \text{class 2} \end{cases}$

$L(f) = \frac{1}{2}\sum_n \left(f_{w,b}(x^n) - \hat{y}^n\right)^2$

Step 3 : $\frac{\partial \left(f_{w,b}(x) - \hat{y}\right)^2}{\partial w_i} = 2\left(f_{w,b}(x) - \hat{y}\right) \cdot \frac{\partial f_{w,b}(x)}{\partial z} \cdot \frac{\partial z}{\partial w_i}$

$= 2\left(f_{w,b}(x) - \hat{y}\right) \cdot f_{w,b}(x)\left(1 - f_{w,b}(x)\right) \cdot x_i$

If now $\hat{y}^n = 0$　If $f_{w,b}(x^n) = 1$　far from the target → $\frac{\partial L}{\partial w_i} \approx 0$

" $= 0$　close to "　→ $\frac{\partial L}{\partial w_i} \approx 0$

$\hat{y}^n = 1$　$f(x^n) = 1$　close　≈ 0

≈ 0　fare　≈ 0

☆ If now is cross entropy , far from 目標，微分值越大 幾次 update 越快

square error " " 很小 慢

now we don't know gradient 小的時候(微分小) 是 close to / far from 目標

Logistic regression 称為 discriminative
use Gaussion 來描述 posterior probability 稱為 Generative

Discriminative vs Generative

實際上這个 model., function set is same

if we 把 covariance matrix 設成是 share

$\sigma(w \cdot x + b) = P(C_1|x)$

If we use Logistic regression

→ 可以直接找出 w & b (use gradient descent)

⇒ 沒有對 probability distribution 有任何假設/描述

If now is generative model

→ 会先算 $\mu^1, \mu^2, \Sigma^{-1}$ (Σ 的 inverse)

$$w^T = (\mu^1 - \mu^2)^T \Sigma^{-1}$$

$$b = -\frac{1}{2}(\mu^1)^T(\Sigma^1)^{-1}\mu^1 + \frac{1}{2}(\mu^2)^T(\Sigma^2)^{-1}\mu^2 + \ln\frac{N_1}{N_2}$$

⇒ 对 probability distribution 假設 it is gaussion, bernoulli, No Naive Bayes

⇒ 由於 we use 不同的假設

所以 根據同一組 training data 找出的參數会不一樣

(i.e 找到的 w & b is different)

⇒ which is better. (w & b)

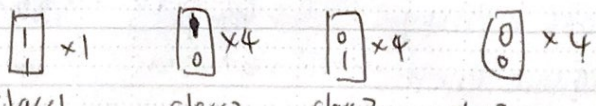
⇒ discriminative is better

→ why?

Example:

每一筆 data 有兩个 feature, 共 13 筆 data

Training data



Testing data  class 1 ?
class 2 ?

Date

⇒ How about Naive Bayes?

$P(X|C_i) = P(x_1|C_i)P(x_2|C_i)$

↳ P of X 從某一个 class 產生出來的机率

$P(C_1) = \frac{1}{13}$ $P(C_2) = \frac{12}{13}$

$P(X_1=1|C_1) = 1$ $P(X_2=1|C_1) = 1$

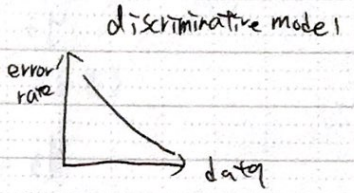
$P(X_1=1|C_2) = \frac{1}{3}$ $P(X_2=1|C_2) = \frac{1}{3}$

⇒ $P(C_1|X) = \frac{P(X|C_1)P(C_1)}{P(X|C_1)P(C_1)+P(X|C_2)P(C_2)}$

$= \frac{1 \times 1 \times \frac{1}{13}}{\frac{1}{13} + \frac{1}{3} \times \frac{1}{3} \times \frac{12}{13}}$

< 0.5

$\in$ class 2



⇒ If now training data 很少，use generative model

※ generative model 有自己的假設

有時会无視 data，遵從內心的假設

⇒ if label 有問題，is noise

⇒ use generative model，可以对比有問題的忽視掉

※ 做 discriminate 時應該上是假設了 posterior probability

然后 find posterior 內的參數

but if now do generative，会把 formulation 拆成

$<$ prior

class-dependent 的 probability

(文字) ⇒ 可以把 prior estimator 更精確

⇒ 可以有不同的來源

(voice & 文字)

forexample：語音辨識現在都是用 neural network

⇒ is a discriminative 方法

but 其整个 system 是一个 generative 的 system

全部 use DNN 可能沒有那麼精確，∵ still need to 算 prior 的 probability

↳ 某句話被說出來的机率 = if we want estimator，並不須要有 audio 的 data

only need 足够多文字 in 網路，then 可以算其机率

Date

Multi-class classification

- $C_1: w^1, b_1$ $\quad z_1 = w^1 x + b_1$
  $C_2: w^2, b_2$ $\quad z_2 = w^2 x + b_2$
  $C_3: w^3, b_3$ $\quad z_3 = w^3 x + b_3$

  可以是 $-\infty \sim +\infty$ 的任何值

對最大值做強化

Softmax function (類似作 normalization)

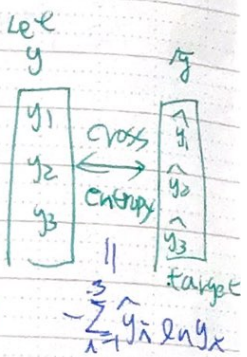
⇒ 把 z_1, z_2, z_3 取 exponential the 把 them summation 起來 $\sum_{j=1}^{3} e^{z_j}$

⇒ then $y_1 = \frac{e^{z_1}}{\sum_{j=1}^{3} e^{z_j}}$ $\quad y_1 = P(C_1|x)$

$y_2 = \frac{e^{z_2}}{\sum_{j=1}^{3} e^{z_j}}$ $\quad y_2 = P(C_2|x)$

$y_3 = \frac{e^{z_3}}{\sum_{j=1}^{3} e^{z_j}}$ $\quad y_3 = P(C_3|x)$

⇒ $y_x = P(C_x|x)$



⇒ ① $0 < y_x < 1$

② $\sum_{j=1}^{n} y_x = 1$

P 209~210

(Pf:

假設 3 個 class 都是 Gaussian distribution

and 共用同一個 covariance matrix

⋮

we will get softmax 的 function

google = maximum entropy

↳ 和 Logistic Regression Same

Date

If $x \in$ class 1 $\quad \hat{y} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$

$\in$ class 2 $\quad \hat{y} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$

$\in$ class 3 $\quad \hat{y} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$

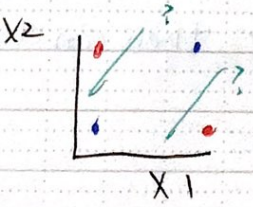
⇒ cross entropy 這個 function (minimize cross entropy)
其且 maximize likelihood
多個 class is same
→ 把 max likelihood 的那個 function 列出來
then 整理 will get minimize cross entropy

We want to Use logistic regression 分類

⇒ We can't
the output we hope $\begin{cases} \text{class 1} & y \geq 0.5 \\ \text{class 2} & y < 0.5 \end{cases}$

⇒ 因為 logistic regression 兩個 class 之間的 boundary 是一條直線

⇒ We can't 把紅色 put 一直，blue put 一直



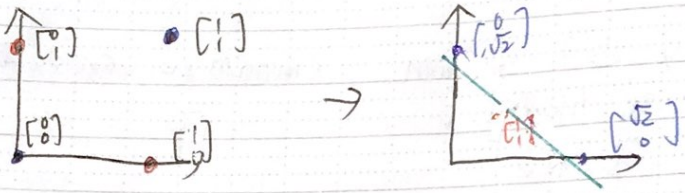
⇒ 所
We still want to use logistic regression
→ We can use Feature Transformation
做一些轉換去找一個比較好的 feature space
→ 其可以套 logistic regression 處理

Date

$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \longrightarrow \begin{bmatrix} x_1' \\ x_2' \end{bmatrix}$

Let x_1' be the distance to $\begin{bmatrix} 0 \\ 0 \end{bmatrix}$

x_2' be the distance to $\begin{bmatrix} 1 \\ 1 \end{bmatrix}$



$\begin{bmatrix} 0 \\ 0 \end{bmatrix} \to \begin{bmatrix} 0 \\ \sqrt{2} \end{bmatrix}$

$\begin{bmatrix} 1 \\ 1 \end{bmatrix} \to \begin{bmatrix} \sqrt{2} \\ 0 \end{bmatrix}$

but not always easy to find a good feature transformation

⇒ we hope this transformation 是由机器自己产生的

⇒ 把很多 Logistic Regression Cascade (接) 起来

feature transform

$x_1 \times w + x_2 \times w \to z_1$ 的 sigmoid function → output 为 x_1'

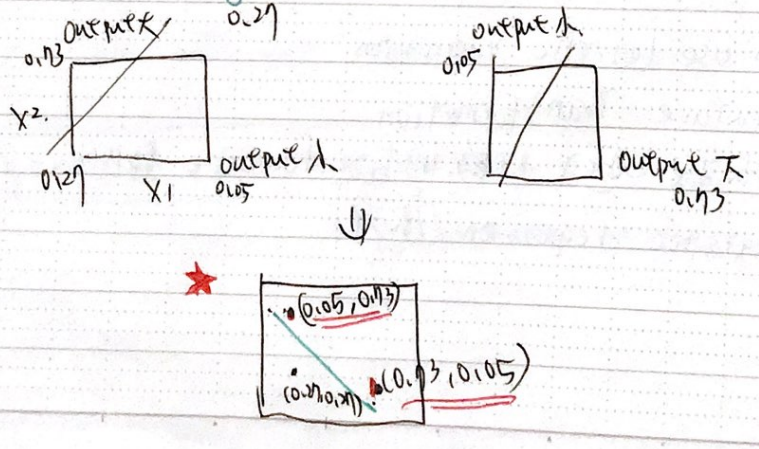
另一个 logistic regression model

$x_1 \times w + x_2 \times w \to z_2$ ⟶ sigmoid function → output 为 x_2'

↳ 接个 logistic regression 的 model → then can classification

(input x_1', x_2')

→ y.



★ 把每个 Logistic Regression 叫作一个 Neuron

把这些 Logistic Regression 串起来所成的 network 就叫作 neural network (类神经网路)