No.

Date

Contour plots

size(X,2)

size(X,1)

No.

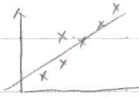
Date

Linear Regression

m = Number of training examples — (x, y) ← One training example — Training Set

x's = "input" variable / feature — $(x^{(i)}, y^{(i)})$ i-th training example

y's = "output" variable / target variable



$h(x)$

$h_\theta(x) = \theta_0 + \theta_1 x$ — θ_i's : parameters

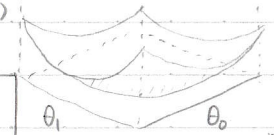
Learning Algorithm

$x \rightarrow [\,h\,] \rightarrow y$ — hypothesis

squared error function

minimize $\frac{1}{2m}\sum_{i=1}^{m}$ ← # training examples $(h_\theta(x^{(i)}) - y^{(i)})^2$ ⇒ cost function $J(\theta_0,\theta_1) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2$ — minimize $J(\theta_0,\theta_1)$ over θ_0,θ_1

- Hypothesis : $h_\theta(x) = \theta_0 + \theta_1 x$
- Parameter : θ_0, θ_1
- Cost function : $J(\theta_0,\theta_1) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2$
- Goal : minimize $J(\theta_0,\theta_1)$ over θ_0,θ_1



surf (X, Y, Z)

[X Y] = meshgrid(x, y)

Z = X .* Y

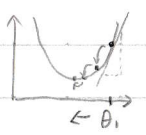
[Gradient descent algorithm]

learning rate

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0,\theta_1)$$

$$\frac{\partial}{\partial \theta_j} J(\theta_0,\theta_1) = \frac{\partial}{\partial \theta_j} \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2$$

$$= \frac{\partial}{\partial \theta_j} \frac{1}{2m}\sum_{i=1}^{m}(\theta_0 + \theta_1 x^{(i)} - y^{(i)})^2$$



$\theta_1 := \theta_1 - \alpha \frac{d}{d\theta_1} J(\theta_1)$ (>0)

$$\begin{cases} \theta_0 : j=0 & \frac{\partial}{\partial \theta_0} J(\theta_0,\theta_1) = \frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)}) \\ \theta_1 : j=1 & \frac{\partial}{\partial \theta_1} J(\theta_0,\theta_1) = \frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)}) \cdot x^{(i)} \end{cases}$$

"Batch" Gradient Descent : Each step of gradient descent uses all the training examples.

[Transpose] $B_{ij} = A_{ji}$

Review : Linear Algebra

[Matrix] Row × column

Row $\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}$ $\mathbb{R}^{2\times 3}$ $\mathbb{R}^{m\times n}$

Column

[Vector] An $n\times 1$ matrix

$y = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$ $y_i = i^{th}$ element

{ Upper case refers to MATRIX
Lower case refers to VECTOR

[Multiple features]

- n = number of features
- $x^{(i)}$ = input features of i^{th} training example
- $x_j^{(i)}$ = value of feature j in i^{th} training example

Multivariate Linear Regression

$$\begin{cases} h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta x_3 \\ \text{Define : } (x_0^{(i)} = 1) \\ x = \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_n \end{bmatrix} \in \mathbb{R}^{n+1} \quad \theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_n \end{bmatrix} \in \mathbb{R}^{n+1} \\ h_\theta(x) = \theta^T x = \theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 \end{cases}$$

Repeat {

$$\theta_j := \theta_j - \alpha \frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)}$$

$$\begin{cases} \theta_0 := \theta_0 - \alpha \frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)}) \cdot x_0^{(i)} \\ \theta_1 := \theta_1 - \alpha \frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)}) \cdot x_1^{(i)} \\ \theta_2 := \theta_2 - \alpha \frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)}) \cdot x_2^{(i)} \end{cases}$$

}

$\sum_{i=1}^{m}(X\theta - y) .* x_j$ → sum($\begin{bmatrix} \theta_0 x_0 - y_0 \\ \vdots \\ \theta_n x_n - y_n \end{bmatrix} .* \begin{bmatrix} x_2^{(0)} \\ x_2^{(1)} \\ \vdots \\ x_n^{(m)} \end{bmatrix}$) OR $\begin{bmatrix} x_2^{(0)} & x_2^{(1)} & \cdots & x_2^{(m)} \end{bmatrix} \begin{bmatrix} \theta_0 x_0 - y_0 \\ \vdots \\ \theta_n x_n - y_0 \end{bmatrix}$

$X' * (X \cdot \theta - y)$

$\begin{bmatrix} - & x_0 & - \\ - & x_1 & - \\ & \vdots & \\ - & x_n & - \end{bmatrix} \begin{bmatrix} \theta_0 x_0 - y_0 \\ \theta_1 x_1 - y_1 \\ \vdots \\ \theta_n x_n - y_0 \end{bmatrix}$

$\begin{bmatrix} \theta_0 x_0 - y_0 \\ \theta_1 x_1 - y_1 \\ \vdots \\ \theta_n x_n - y_n \end{bmatrix} \cdot x_2^{(i)}$ $\quad x_2^{(0)} \cdot (\theta_0 x_0 - y_0) + x_2^{(1)} \cdot (\theta_1 x_1 - y_1) + \cdots + x_n^{(1)} \cdot (\theta_n x_n - y_0)$

No.

Date

[Feature Scaling]

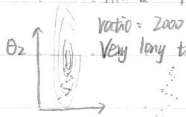
Idea: Make sure features are on a similar scale

E.g. x_1 = size (0~2000 feet^2)

x_2 = number of bedrooms (1~5)

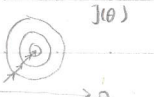
ratio = 2000 : 5

Very long time to find global minimum



$\Rightarrow$ $x_1 = \frac{\text{size}}{2000}$, $x_2 = \frac{\text{number of rooms}}{5}$

converge faster



Get every feature into approximately a $-1 \le x_i \le 1$ range

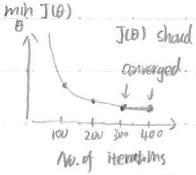
[Mean normalization]

Replace x_i with $x_i - \mu_i$ to make features have approximately zero mean (Do not apply to $x_0 = 1$)

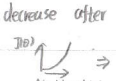
E.g. $x_1 = \frac{\text{size} - 1000}{2000}$, $x_2 = \frac{\#\text{bedroom} - 2}{5}$ $\begin{cases} -0.5 \le x_1 \le 0.5 \\ -0.5 \le x_2 \le 0.5 \end{cases}$

$x_i \leftarrow \frac{x_i - \mu_i}{s_i}$ ← average value of x_i ← range of the value (max − min)

[Making sure gradient descent is working correctly]



J(θ) should decrease after every iteration



$\Rightarrow$ Use smaller α

[Polynomial Regression] 多项式回归 { Quadratic function, Cubic function }

$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3$

$= \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3$

[Normal Equation]

Method to solve for θ analytically

$X = \begin{bmatrix} 1 & x_1^{(1)} & x_2^{(1)} & \cdots & x_n^{(1)} \\ 1 & x_1^{(2)} & x_2^{(2)} & \cdots & x_n^{(2)} \\ \vdots & & & & \\ 1 & x_1^{(m)} & x_2^{(m)} & \cdots & x_n^{(m)} \end{bmatrix}$ ← design matrix $\quad y = \begin{bmatrix} y^{(1)} \\ \vdots \\ y^{(m)} \end{bmatrix}$

$\mathbb{R}^{m \times (n+1)}$ { m examples, n features }

Minimize the cost function

$\theta = (X^T X)^{-1} X^T y$

Feature scaling not necessary

What if $X^T X$ is non-invertible? (singular / degenerate) 奇异 退化

① Redundant features $x_1 = (3.28)^2 x_2$

② Too many features ($m \le n$) { Delete some feature, Regularization 正则化 }

[Gradient Descent vs Normal Equation]

Gradient Descent	Normal Equation
Need to choose α	No need to choose α
Need many iterations	Don't need to iterate
Work well even n is large	Need to compute $(X^T X)^{-1}$ $O(n^3)$, slow if n is very large
	$n \times n$

← ... n = 10000

[Vectorization]

$h_\theta(x) = \sum_{j=0}^{n} \theta_j x_j = \theta^T x$

Unvectorized implementation

```
for j = 1:n+1
    sum = sum + theta(j) * x(j)
```

Vectorized implementation

```
sum = theta' * x
```

[Optimization algorithms]

- Conjugate gradient
- BFGS
- L-BFGS

Adv: no need to manually pick α; faster than gradient descent

Disadv: More complex

[Advanced optimization function]

fminunc()

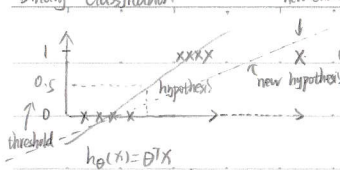
↗ function minimization unconstrained

No.

Date

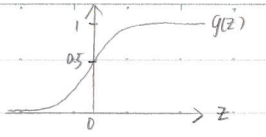
Binary Classification

new example



$h_\theta(x) = \theta^T x$

Case linear regression to change in straight line fit the data from the original one.



{ Sigmoid function, Logistic function }

[Logistic Regression] 罗辑斯回归

$0 \le h_\theta(x) \le 1$

$h_\theta(x) = g(\theta^T x)$ $\quad g(z) = \frac{1}{1+e^{-z}} = \frac{1}{1+e^{-\theta^T x}}$

$h_\theta(x)$ = estimated probability that $y=1$ on input x (given)

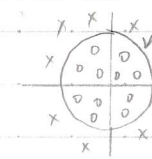
If $x = \begin{bmatrix} x_0 \\ x_1 \end{bmatrix} = \begin{bmatrix} 1 \\ \text{tumorSize} \end{bmatrix}$ $\Rightarrow h_\theta(x) = 0.7$

Tell the patient that 70% chance of tumor being bad (1)

$\Rightarrow h_\theta(x) = p(y=1 \mid x; \theta)$

"probability that $y=1$, given x, parameterized by θ"

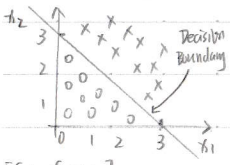
[Non-linear decision boundaries]



$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_2^2)$

Assume $\theta = [-1 \ 0 \ 0 \ 1 \ 1]^T$

[Decision Boundary] 决策边界 ← Is a property of hypothesis, not a property of data set



Assume $\theta = \begin{bmatrix} -3 \\ 1 \\ 1 \end{bmatrix}$

$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$

$\Rightarrow$ Predict "y=1" if $-3 + x_1 + x_2 \ge 0$

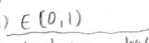
$\rightarrow -3 + x_1 + x_2 = 0$

$x_2 = -x_1 + 3$

Predict "y=1" if $-1 + x_1^2 + x_2^2 \ge 0$

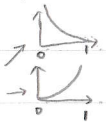
$x_1^2 + x_2^2 \ge 1$

Higher Order Polynomial

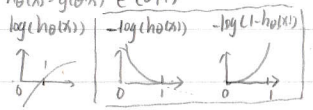


[Cost function]

$\text{Cost}(h_\theta(x), y) = \begin{cases} -\log(h_\theta(x)) & \text{if } y=1 \\ -\log(1-h_\theta(x)) & \text{if } y=0 \end{cases}$



$h_\theta(x) = g(\theta^T x) \in (0,1)$



Cost = 0 if $y=1$, $h_\theta(x) = 1$

But as $h_\theta(x) \to 0$, cost $\to \infty$

If $h_\theta(x) = 0$, predict $P(y=1 \mid x; \theta) = 0$ but $y=1$, penalize learning algorithm by a very larger cost

$\text{Cost}(h_\theta(x), y) = -y \cdot \log(h_\theta(x)) - (1-y) \cdot \log(1-h_\theta(x))$

$J(\theta) = \frac{1}{m} \sum_{i=1}^{m} \text{Cost}(h_\theta(x^{(i)}), y^{(i)})$

$\text{Cost}(h_\theta(x), y) = \begin{cases} -\log(h_\theta(x)) & \text{if } y=1 \\ -\log(1-h_\theta(x)) & \text{if } y=0 \end{cases}$

$J(\theta) = -\frac{1}{m} \left[\sum_{i=1}^{m} y^{(i)} \cdot \log(h_\theta(x^{(i)})) + (1-y^{(i)}) \cdot \log(1 - h_\theta(x^{(i)})) \right]$

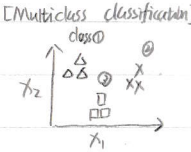
$\min_\theta J(\theta) \Rightarrow h_\theta(x) = \frac{1}{1+e^{-\theta^T x}}$ $\quad (P(y=1 \mid x; \theta))$

Want $\min_\theta J(\theta)$:

Repeat { $\theta_j := \theta_j - \alpha \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)}$ }

↳ $h_\theta(x) = \frac{1}{1+e^{-\theta^T x}}$

[Multiclass classification]



One-vs-all (One-vs-rest)

$h_\theta^{(1)}(x)$

$h_\theta^{(2)}(x)$

$h_\theta^{(3)}(x)$

$h_\theta^{(i)} = P(y=i \mid x; \theta)$ $\quad (i = 1, 2, 3)$

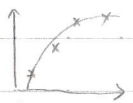
Train a logistic regression classifier $h_\theta^{(i)}(x)$ for each class i to predict the probability that $y=i$

On a new input x to make a prediction, pick the class i that maximizes $\max_i h_\theta^{(i)}(x)$

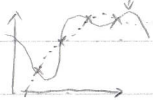
$\theta_0+\theta_1x_1+\theta_2x_2+\theta_3x_1^2+\theta_4x_2^2$

Regularization 正則化

overfitting



$\theta_0+\theta_1x+\theta_2x^2$



$\theta_0+\theta_1x+\theta_2x^2+\theta_3x^3+\theta_4x^4$

suppose we penalize and make θ_3, θ_4 really small.

$$\min_\theta \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)})-y^{(i)})^2+1000\theta_3^2+1000\theta_4^2 \quad \begin{cases}\theta_3\approx 0\\ \theta_4\approx 0\end{cases}$$

[Regularization]

Small values for parameters $\theta_0, \theta_1, \dots \theta_n$

- "Simpler" hypothesis
- Less prone to overfitting

regularization parameter

Trade offs between the cost & penality of huge θ

$$J(\theta)=\frac{1}{2m}\left[\sum_{i=1}^{m}(h_\theta(x^{(i)})-y^{(i)})^2+\lambda\cdot\sum_{j=1}^{n}\theta_j^2\right]$$

$\lambda\cdot(\theta_1^2+\theta_2^2+\theta_3^2+\cdots\theta_n^2)$

求偏导时只有 θ_j^2 有用到，其它全为常数

$\frac{\partial}{\partial\theta_j}\left(\frac{1}{2m}\cdot\lambda\cdot\theta_j^2\right)=\frac{\lambda}{m}\theta_j$

[Regularized Linear Regression]

Repeat {

$$\theta_0:=\theta_0-\alpha\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)})-y^{(i)})x_0^{(i)}$$

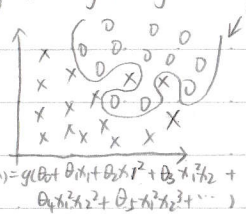
$$\theta_j:=\theta_j-\alpha\left[\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)})-y^{(i)})x_j^{(i)}+\frac{\lambda}{m}\theta_j\right] \Rightarrow \theta_j:=\theta_j(1-\alpha\frac{\lambda}{m})-\alpha\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)})-y^{(i)})\cdot x_j^{(i)}$$

Normal equation $X=\mathbb{R}^{m\times(n+1)}$

正则化可以解决奇异矩阵不可逆的情况

$$\theta=(X^TX+\lambda\underbrace{\begin{bmatrix}0&&\\&1&\\&&\ddots\end{bmatrix}}_{\mathbb{R}^{(n+1)(n+1)}})^{-1}X^Ty$$

[Regularized Logistic regression]



overfitting

$h_\theta(x)=g(\theta_0+\theta_1x_1+\theta_2x_1^2+\theta_3x_1^2x_2+\theta_4x_1^2x_2^2+\theta_5x_1^2x_2^3+\cdots)$

$$J(\theta)=-\left[\frac{1}{m}\sum_{i=1}^{m}y^{(i)}\log h_\theta(x^{(i)})+(1-y^{(i)})\log(1-h_\theta(x^{(i)}))\right]+\frac{\lambda}{2m}\sum_{j=1}^{n}\theta_j^2$$

$$\begin{cases}\theta_0:=\theta_0-\alpha\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)})-y^{(i)})x_0^{(i)}\\ \theta_j:=\theta_j-\alpha\left[\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)})-y^{(i)})x_j^{(i)}+\frac{\lambda}{m}\theta_j\right]\end{cases} \quad h_\theta(x)=\frac{1}{1+e^{-\theta^Tx}}$$

Advanced optimization.

```
function [jVal, gradient] = costFunction(theta)
  jVal = [code to compute J(theta)];
```

$$J(\theta)=\left[-\frac{1}{m}\sum_{i=1}^{m}y^{(i)}\log(h_\theta(x^{(i)})+(1-y^{(i)})\log(-h_\theta(x^{(i)}))\right]+\frac{\lambda}{2m}\sum_{j=1}^{n}\theta_j^2$$

θ_0 gradient(1) $=\frac{\partial}{\partial\theta_0}J(\theta)=\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)})-y^{(i)})x_0^{(i)}$

θ_1 gradient(2) $=\frac{\partial}{\partial\theta_1}J(\theta)=\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)})-y^{(i)})x_1^{(i)}+\frac{\lambda}{m}\theta_1$

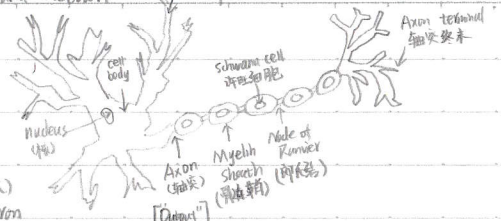
θ_2 gradient(3) $=\frac{\partial}{\partial\theta_2}J(\theta)=\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)})-y^{(i)})x_2^{(i)}+\frac{\lambda}{m}\theta_2$

...

nctool

Neural Network

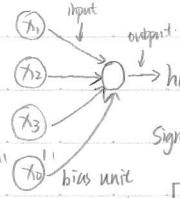
Dendrite (樹突) ["Input"]



(神經元) Neuron

[Computational Unit]

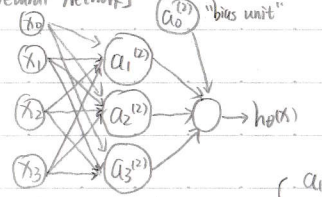
Neuron model: Logistic unit



$h_\theta(x)=\frac{1}{1+e^{-\theta^Tx}}$

$g(z)=\frac{1}{1+e^{-z}}$ Sigmoid (logistic) activation function

[Neural Network]



$a_i^{(j)}$: "activation" of unit i in layer j

$\Theta^{(j)}$: matrix of weights controlling function mapping from layer j to layer $j+1$

$x=\begin{bmatrix}x_0\\x_1\\x_2\\x_3\end{bmatrix}$ $\theta=\begin{bmatrix}\theta_0\\\theta_1\\\theta_2\\\theta_3\end{bmatrix}$ ← "weight" (parameters)

E.g. $a_1^{(2)}$:= the activation of the first unit in layer 2

"Activation" means the value that is computed by and is output by a specific

Layer 1 Input Layer | Layer 2 "Hidden" Layer | Layer 3 Output Layer

$$\begin{cases}a_1^{(2)}=g(\Theta_{10}^{(1)}x_0+\Theta_{11}^{(1)}x_1+\Theta_{12}^{(1)}x_2+\Theta_{13}^{(1)}x_3)\\ a_2^{(2)}=g(\Theta_{20}^{(1)}x_0+\Theta_{21}^{(1)}x_1+\Theta_{22}^{(1)}x_2+\Theta_{23}^{(1)}x_3)\\ a_3^{(2)}=g(\Theta_{30}^{(1)}x_0+\Theta_{31}^{(1)}x_1+\Theta_{32}^{(1)}x_2+\Theta_{33}^{(1)}x_3)\end{cases} \quad \Theta^{(1)}\in\mathbb{R}^{3\times4}$$

$$\begin{bmatrix}\Theta_{10}&\Theta_{11}&\Theta_{12}&\Theta_{13}\\ \Theta_{20}&\Theta_{21}&\Theta_{22}&\Theta_{23}\\ \Theta_{30}&\Theta_{31}&\Theta_{32}&\Theta_{33}\end{bmatrix}\begin{bmatrix}x_0\\x_1\\x_2\\x_3\end{bmatrix}=\begin{bmatrix}a_1^{(2)}\\a_2^{(2)}\\a_3^{(2)}\end{bmatrix}$$

$h_\Theta(x)=a_1^{(3)}=g(\Theta_{10}^{(2)}a_0^{(2)}+\Theta_{11}^{(2)}a_1^{(2)}+\Theta_{12}^{(2)}a_2^{(2)}+\Theta_{13}^{(2)}a_3^{(2)})$

3 4

⇒ If network has S_j units in layer j, S_{j+1} units in layer $j+1$, then $\Theta^{(j)}$ will be $\mathbb{R}^{S_{j+1}\times(S_j+1)}$

1 → 2 → 3

$\mathbb{R}^4$ $\Theta^{3\times4}$ $\mathbb{R}^3$ $\Theta^{1\times3}$ $\mathbb{R}^1$

Assum $x=a^{(1)}$

[Forward propagation: Vectorized Implementation]

$$x=\begin{bmatrix}x_0\\x_1\\x_2\\x_3\end{bmatrix} \quad z^{(2)}=\begin{bmatrix}z_1^{(2)}\\z_2^{(2)}\\z_3^{(2)}\end{bmatrix}=\begin{bmatrix}\Theta_{10}^{(1)}x_0+\Theta_{11}^{(1)}x_1+\Theta_{12}^{(1)}x_2+\Theta_{13}^{(1)}x_3\\ \Theta_{20}^{(1)}x_0+\Theta_{21}^{(1)}x_1+\Theta_{22}^{(1)}x_2+\Theta_{23}^{(1)}x_3\\ \Theta_{30}^{(1)}x_0+\Theta_{31}^{(1)}x_1+\Theta_{32}^{(1)}x_2+\Theta_{33}^{(1)}x_3\end{bmatrix}$$

$z^{(2)}=\Theta^{(1)}x=\Theta^{(1)}a^{(1)}$

$a^{(2)}=g(z^{(2)})$

$a^{(1)}=\begin{bmatrix}x_0\\x_1\\x_2\\x_3\end{bmatrix}$ $a^{(2)}=\begin{bmatrix}a_0\\a_1\\a_2\\a_3\end{bmatrix}$ Add. $a_0=1$

Because we start with the activation of the input-units and then we sort of forward-propagate that to the hidden layer and compute the activations...

$h_\Theta(x)\to\begin{cases}\text{Add } a_0^{(2)}=1 & a^{(2)}\in\mathbb{R}^4\\ z^{(3)}=\Theta^{(2)}a^{(2)}\\ h_\Theta(x)=a^{(3)}=g(z^{(3)})\end{cases}$

$h_\Theta(x)$

$a^{(2)}=g(\Theta^{(1)}a^{(1)})$ $a^{(3)}=g(\Theta^{(2)}a^{(2)})$

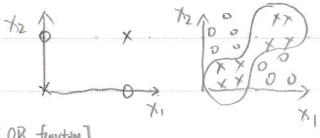
$\mathbb{R}^3=\mathbb{R}^{3\times4}\times\mathbb{R}^4$ $\mathbb{R}^1=\mathbb{R}^{1\times3}\times\mathbb{R}^3$

No.

Date

Non-linear classification example : XOR / NXOR

x_1, x_2 are binary $(0,1)$

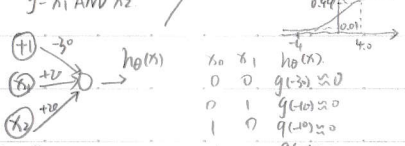


[Simpler example : AND]

$x_1, x_2 \in \{0,1\}$

$y = x_1$ AND x_2

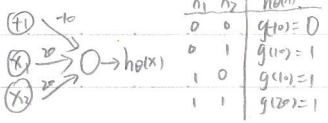
$h_\theta(x) = g(-30 + 20x_1 + 20x_2)$



x_0	x_1	$h_\theta(x)$
0	0	$g(-30) \approx 0$
0	1	$g(-10) \approx 0$
1	0	$g(-10) \approx 0$
1	1	$g(10) \approx 1$

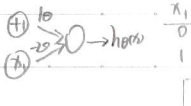
$\theta_1^{(1)}$ $\theta_2^{(1)}$ $\theta_3^{(1)}$

[OR function]



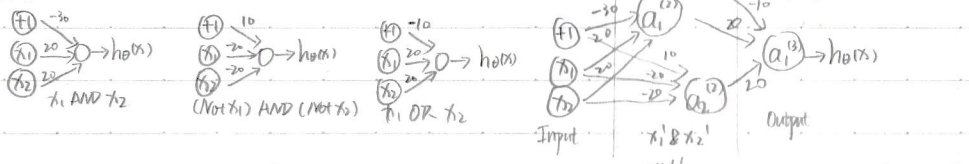
x_1	x_2	$h_\theta(x)$
0	0	$g(-10) \approx 0$
0	1	$g(10) \approx 1$
1	0	$g(10) \approx 1$
1	1	$g(30) \approx 1$

[NOT function]



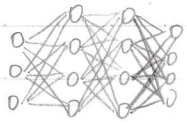
x_1	$h_\theta(x)$
0	$g(10) \approx 1$
1	$g(-10) \approx 0$

[Together]



x_1	x_2	$a_1^{(2)}$	$a_2^{(2)}$	$a_1^{(3)}$
0	0	0	1	1
0	1	0	0	0
1	0	0	0	0
1	1	1	1	1

[Multiple output units : One vs all]



$h_\theta(x) \in \mathbb{R}^4$

$y^{(i)}$ one of $\begin{bmatrix}1\\0\\0\\0\end{bmatrix}\begin{bmatrix}0\\1\\0\\0\end{bmatrix}\begin{bmatrix}0\\0\\1\\0\end{bmatrix}\begin{bmatrix}0\\0\\0\\1\end{bmatrix}$

Training set : $(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \cdots, (x^{(m)}, y^{(m)})$

[Multi-class classification]

$y \in \mathbb{R}^K$ E.g. $\begin{bmatrix}1\\0\\0\\0\end{bmatrix}$

K output units

$h_\theta(x) = \mathbb{R}^K$

$S_L = K$

[Binary classification]

$y = 0$ or 1

1 output unit

$h_\theta(x) \in \mathbb{R}$

$S_L = 1$ $K = 1$

L = total no. of layers in network $(L=4)$

S_L = no. of units (not counting bias unit) in layer L

$S_1 = 3$ $S_3 = 4$

$S_2 = 4$ $S_4 = 4$

[Cost function]

$h_\theta(x) \in \mathbb{R}^K$ $(h_\theta(x))_i = i^{th}$ output

$$J(\theta) = -\frac{1}{m}\left[\sum_{i=1}^{m}\sum_{k=1}^{K} y_k^{(i)} \log (h_\theta(x^{(i)}))_k + (1-y_k^{(i)}) \log (1-(h_\theta(x^{(i)}))_k)\right] + \frac{\lambda}{2m}\sum_{l=1}^{L-1}\sum_{i=1}^{S_l}\sum_{j=1}^{S_{l+1}} (\theta_{ji}^{(l)})^2$$

[Gradient computation]

Given one training example (x,y):

forward propagation

$a^{(1)} = x$

$z^{(2)} = \theta^{(1)} a^{(1)}$

$a^{(2)} = g(z^{(2)})$ (add $a_0^{(2)}$)

$z^{(3)} = \theta^{(2)} a^{(2)}$

$a^{(3)} = g(z^{(3)})$ (add $a_0^{(3)}$)

$z^{(4)} = \theta^{(3)} a^{(3)}$

$a^{(4)} = h_\theta(x) = g(z^{(4)})$

[Backpropagation Algorithm]

$\delta_j^{(l)}$ = "error" of node j in layer l

$\delta_j^{(4)} = a_j^{(4)} - y_j$ $\quad \delta^{(4)} = a^{(4)} - y$ (vectorized)

$\delta^{(3)} = (\theta^{(3)})^T \delta^{(4)} .* g'(z^{(3)}) \rightarrow a^{(3)} .* (1-a^{(3)})$

$\delta^{(2)} = (\theta^{(2)})^T \delta^{(3)} .* g'(z^{(2)})$

$\frac{\partial}{\partial \theta_{ij}^{(l)}} J(\theta) = a_j^{(l)} \delta_i^{(l+1)}$ (ignoring λ)

Campus

9-4

9-5 Gradient Checking

9-6 Random Initialization

100,0000

Ten million

No.

Date

[Backpropagation Algorithm]

Training Set $\{(x^{(1)}, y^{(1)}), \cdots, (x^{(m)}, y^{(m)})\}$

Set $\Delta_{ij}^{(l)} = 0$ (for all i,j)

For $i = 1$ to m

Set $a^{(1)} = x^{(i)}$

Perform forward propagation to compute $a^{(l)}$ for $l = 2, 3, \cdots, L$

Using $y^{(i)}$, compute $\delta^{(L)} = a^{(L)} - y^{(i)}$

Compute $\delta^{(L-1)}, \delta^{(L-2)}, \cdots, \delta^{(2)}$

$\Delta_{ij}^{(l)} = \Delta_{ij}^{(l)} + a_j^{(l)} \delta_i^{(l+1)}$

$$\begin{cases} D_{ij}^{(l)} := \frac{1}{m}\Delta_{ij}^{(l)} + \lambda\theta_{ij}^{(l)} & j \neq 0 \\ D_{ij}^{(l)} := \frac{1}{m}\Delta_{ij}^{(l)} & j = 0 \end{cases}$$

$\frac{\partial}{\partial \theta_{ij}^{(l)}} J(\theta) = D_{ij}^{(l)}$

$X(i,:)$ - sheet

KOKUYO

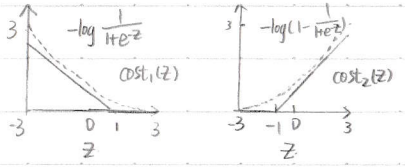
No.

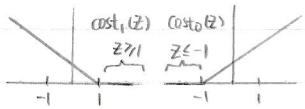
Date

Support Vector Machine

Replace the cost function

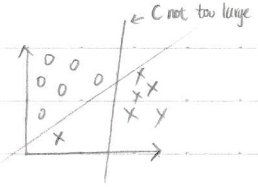
$$\min_\theta C\sum_{i=1}^{m}\left[y^{(i)}\,\text{cost}_1(\theta^T x^{(i)}) + (1-y^{(i)})\,\text{cost}_0(\theta^T x^{(i)})\right] + \frac{1}{2}\sum_{j=1}^{n}\theta_j^2$$



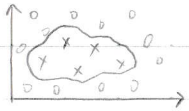


If $y=1$ we want: $\theta^T x \geq 1$ (not just ≥ 0)

If $y=0$ we want: $\theta^T x \leq -1$ (not just < 0)



[Non-Linear Decision Boundary]



Predict $y=1$ if $\theta_0+\theta_1x_1+\theta_2x_2+\theta_3x_1x_2+\theta_4x_1^2+\theta_5x_2^2+\cdots \geq 0$

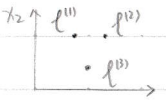
$$h_\theta(x) = \begin{cases} 1 & \text{if } \theta_0+\theta_1x_1+\cdots \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

$\Rightarrow \theta_0+\theta_1f_1+\theta_2f_2+\theta_3f_3+\cdots$

[Kernel]

Given x, compute new feature depending on proximity to landmarks $l^{(1)}, l^{(2)}, l^{(3)}$

$K(x, l^{(i)})$



Given x: $f_1 = \text{similarity}(x, l^{(1)}) = \exp\left(-\frac{\|x-l^{(1)}\|^2}{2\sigma^2}\right)$

$f_2 = \text{similarity}(x, l^{(2)}) = \exp\left(-\frac{\|x-l^{(2)}\|^2}{2\sigma^2}\right)$

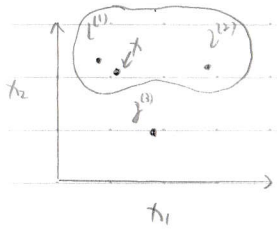
$\uparrow$ ...

Kernel function (Gaussian Kernels)

$f_1 = \text{similarity}(x, l^{(1)}) = \exp\left(-\frac{\|x-l^{(1)}\|^2}{2\sigma^2}\right) = \exp$

$$\Downarrow \begin{cases} \text{If } x \approx l^{(1)} \rightarrow f_1 \approx \exp\left(-\frac{0^2}{2\sigma^2}\right) \approx 1 \\ \text{If } x \text{ far from } l^{(1)} \rightarrow f_1 \approx \exp\left(-\frac{(\text{large number})^2}{2\sigma^2}\right) \approx 0 \end{cases}$$

离的越近，值为"1"



Predict "1" when $\theta_0+\theta_1f_1+\theta_2f_2+\theta_3f_3 \geq 0$

Assume $\theta_0=-0.5$, $\theta_1=1$ $\theta_2=1$ $\theta_3=0$

x: $\theta_0+\theta_1\times 1+\theta_2\times 0+\theta_3\times 0 = -0.5+1 = 0.5 \geq 0$

No.

Date

[SVM with Kernels]

Hypothesis: Given x, compute features $f \in \mathbb{R}^{m+1}$

Predict "$y=1$" if $\theta^T f \geq 0$

Training (features: $n=m$, examples)

$$\min_\theta C \sum_{i=1}^{m} y^{(i)} \text{cost}_1(\theta^T f^{(i)}) + (1-y^{(i)}) \text{cost}_0(\theta^T f^{(i)}) + \frac{1}{2} \sum_{j=1}^{n} \theta_j^2$$

$C (= \frac{1}{\lambda})$
- Large C: Low bias, high variance
- Small C: Higher bias, low variance

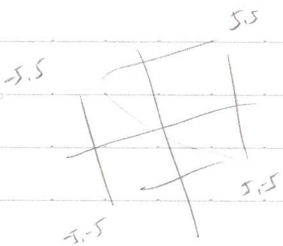
σ^2
- Large σ^2: Features f_i vary more smoothly. Higher bias, lower variance
- Small σ^2: Features f_i vary less smoothly. Low bias, higher variance

Use software package (e.g. liblinear, libsvm) to solve for parameters θ.

Need to specify:
- Choice of parameter C.
- Choice of kernel (similarity function):
  - No kernel ("linear kernel") → Predict "$y=1$" if $\theta^T x \geq 0$
  - Gaussian kernel: $f_i = \exp\left(-\frac{\|x - l^{(i)}\|^2}{2\sigma^2}\right)$, where $l^{(i)} = x^{(i)}$. Need to choose σ^2.
    - ↑ function $f = \text{kernel}(x_1, x_2)$
  - Do perform feature scaling

No.

Date



(5-y) ×11 + 5 + x

1 GEs (NAND2)

12.08

32 nm → 17686.8

Size

538.199

$53_{16} \times CA_{16}$

$83_{10} \times 202_{10}$

$x^8 + x^4 + x^3 + x + 1$

8 7 6 5 4 3 2 1 0

100011011 283

S: 0 0 0 0 0 0 0 1

r: 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 1   1

11   2

111   3

4

0 0 0 1 1 1 1 1   5

0 1 1 0 0 0 1 1   A

0 1 1 1 1 1 0 0   B

7 C

76543210

$53_{16} = 01010011_2$

$CA_{16} = 11001010_2$

$(x^6 + x^4 + x + 1)(x^7 + x^6 + x^3 + x)$

$= ~~x^{13}~~ + ~~x^{12}~~ + ~~x^9~~ + ~~x^7~~ + ~~x^{11}~~ + ~~x^{10}~~ + ~~x^7~~ + x^5$

$~~x^8~~ + ~~x^7~~ + x^4 + x^2 + ~~x^7~~ + ~~x^6~~ + x^3 + x$

$+x$ $+x^2$

$x^{13} + x^{12} + x^{11} + x^{10} + x^9 + x^8 + x^6 + x^5 + x^4 + x^3$

76543210

$0000,0000,0101,0011_2 \cdot 11001010_2$

001010011 0

101001 1000

101001100000

10100110000000

11111101111110 mod 100011011 (11B)

xor 100011011

→ 011100000011110

100011011

→ 0110110101110

100011011

→ 0101011110110

100011011

→ 00100011010

100011011

→ 000000001

01010011

00000000

→ 01010011

xor 10100110

→ 11110101

xor 0100

No.

Date

deta

0x80 = 1000 ,0000

0x1B = 0001 ,1011

polynomial basis

p=1 q=1

p=p^(q<<1)^0

0000 0000
0000 0010
0000 0001
p=0000 0011

q=q<<1

0000 0001
0000 0010
0000 0011
0000 1111
q= 1111 1111

gf(2^8)
gf(2^4)
3d(

8bit × 8bit
(4bit × 4bit)

$\frac{d\tilde{u}}{dt}$

$\frac{d}{dt}y_1 + a\frac{d}{dt}y_1 + b = 0$

λ Lambda φ phi

δ Delta

125%

No.

Date

1. Galois Field]

$gf(p^n) = \{0, 1, 2, \cdots, p-1\} \cup \{p, p+1, p+2, \cdots, p^2-1\} \cup \{p^2, p^2+1, p^2+2, \cdots, p^3-1\} \cup \{p^{n-1}, p^{n-1}+1, p^{n-1}+2, \cdots, p^n-1\}$

gf(5) = (0, 1, 2, 3, 4)

$gf(2^3) = \{0, 1, 2, 2+1, 2^2, 2^2+1, 2^2+2, 2^2+2+1\}$
$= (0, 1, 2, 3, 4, 5, 6, 7)$

$84 = 2^6+2^4+2^2 = 01010100$
$13 = 2^3+2^2+2^0 = 00001101$

```
  01010100
X 00001101
----------
  01010100
 00000000
+01010100
 00000000
----------
 01011000
+10110001
----------
 00000001
```

10110011

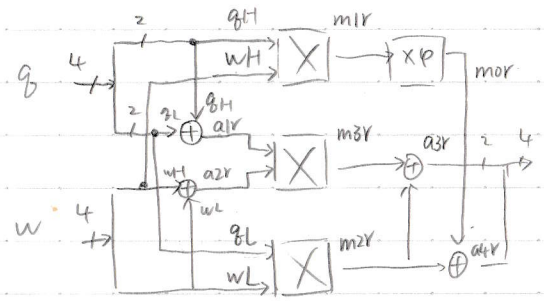
100

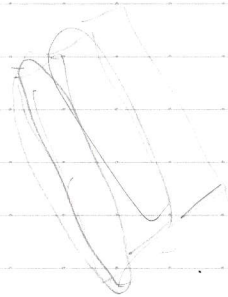
x^2+x+1 × 0111 = 0010

$(x^2+x+1)(x) = x^3+x^2+x$ = 0 1 1 1 0

$(qH+qL)(wH+wL)$

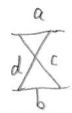
$qHwH + qHwL + qLwH + qLwL$ ✓ ✓ ✓





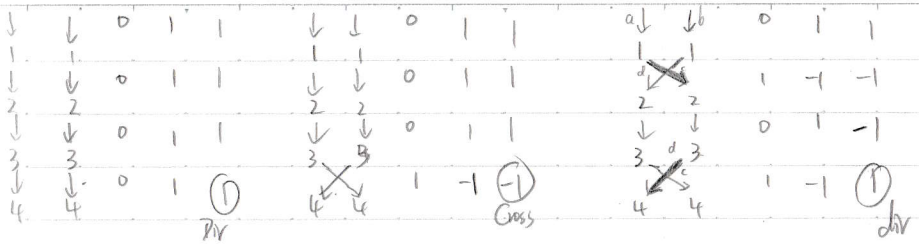
a

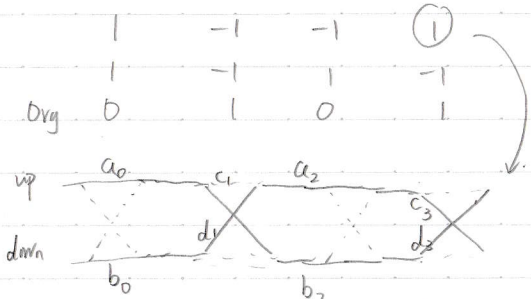
max(a(:,1))

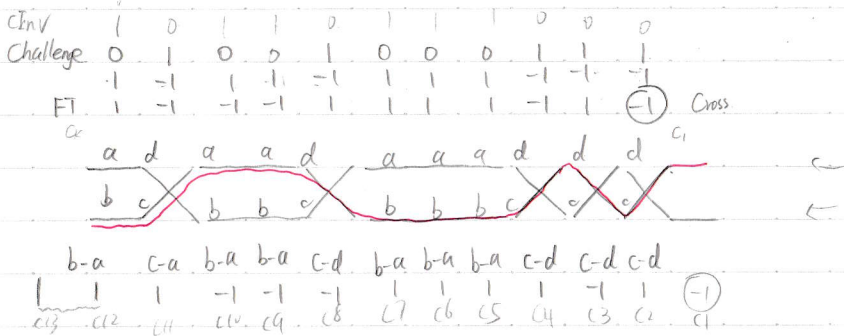


$\delta_0 = a - b$

$\delta_1 = c - d$







dir_path

cro_path

down_delay = (dir_path.* c_inv + cro_path.* c).* [1 FT(c)(1:end-1)]

$\frac{\delta_1^0+\delta_2^0-\delta_2^1}{2} + \frac{\delta_1^0-\delta_1^1}{2} = \delta_1^0$

$\frac{\delta_1^0+\delta_1^1+\delta_2^0-\delta_2^1}{2} - \frac{\delta_1^0-\delta_1^1}{2} = \delta_1^1$

$-\frac{\delta_1^0+\delta_1^1+\delta_2^0-\delta_2^1}{2} + \frac{\delta_1^0-\delta_1^1}{2} = -\delta_1^1$

$-\frac{\delta_1^0+\delta_1^1}{2} + \frac{\delta_1^0-\delta_1^1}{2} = -\delta_1^0$

No.

Date

$\frac{a}{b}$ $\overset{c}{\times}$ d

$\delta_0 = b - a$ $\delta_1 = c - d$

000 / 111 ①	$\frac{a}{b}$ = =	$\delta_0^1 + \delta_0^2 + \delta_0^3$	$w_4 + w_3 + w_2 + w_1 = \delta_0^1 + \delta_0^2 + \delta_0^3$
001 / 11(-1) ②	= = ×	$\delta_0^1 + \delta_0^2 + \delta_1^3$	$w_4 + w_3 + w_2 - w_1 = \delta_0^1 + \delta_0^2 + \delta_1^3$
010 / 1(-1)1 ③	= × =	$\delta_0^1 + \delta_1^2 - \delta_0^3$	$w_4 + w_3 - w_2 - w_1 = \delta_0^1 + \delta_1^2 - \delta_0^3$
011 / 1(-1)(-1) ④	= × ×	$\delta_0^1 + \delta_1^2 - \delta_1^3$	$w_4 + w_3 - w_2 + w_1 = \delta_0^1 + \delta_1^2 - \delta_1^3$
100 / (-1)11 ⑤	× = =	$\delta_1^1 - \delta_0^2 - \delta_0^3$	$w_4 - w_3 - w_2 - w_1 = \delta_1^1 - \delta_0^2 - \delta_0^3$
101 / (-1)1(-1) ⑥	× = ×	$\delta_1^1 - \delta_0^2 - \delta_1^3$	$w_1 = \frac{\delta_0^3 - \delta_1^3}{2}$
110 / (-1)(-1)1 ⑦	× × =	$\delta_1^1 - \delta_1^2 + \delta_0^3$	$w_2 = \frac{\delta_0^2 - \delta_1^2 + \delta_1^3 + \delta_0^3}{2}$
111 / (-1)(-1)(-1) ⑧	× × ×	$\delta_1^1 - \delta_1^2 + \delta_1^3$	$w_3 = \frac{\delta_0^1 + \delta_0^2 + \delta_1^2 - \delta_1^1}{2}$

$w_4 = \frac{\delta_0^1 + \delta_1^1}{2}$

No.

Date

$\frac{\delta_1^0 - \delta_1^1}{2}$ $\frac{\delta_1^0 + \delta_1^1}{2}$

0 $\Delta t = \delta_1^0$

1 1

1

-1 1 $\Delta t = \delta_1^1$

C1 C2 1

$\frac{\delta_1^0 - \delta_1^1}{2}$ $\frac{\delta_1^0 + \delta_1^1 + \delta_2^0 - \delta_2^1}{2}$ $\frac{\delta_2^0 + \delta_2^1}{2}$

0(1) 0(1)	1	$\frac{\delta_1^0 - \delta_1^1}{2} + \frac{\delta_1^0 + \delta_1^1 + \delta_2^0 - \delta_2^1}{2} + \frac{\delta_2^0 + \delta_2^1}{2} = \delta_1^0 + \delta_2^0$	$\frac{\delta_1^0 + \delta_2^0}{2}$
→ 1 1			
0(1) 1(-1)	1	$\frac{-\delta_1^0 + \delta_1^1}{2} + \frac{-\delta_1^0 + \delta_1^1 + \delta_2^0 - \delta_2^1}{2} + \frac{\delta_2^0 + \delta_2^1}{2} = -\delta_1^0 + \delta_2^1$	$\frac{\delta_1^0 + \delta_2^1}{2}$ X
-1 -1			
1(-1) 0 1(1)	1	$\frac{-\delta_1^1 + \delta_1^1}{2} + \frac{-\delta_1^1 + \delta_1^1 + \delta_2^0 - \delta_2^1}{2} + \frac{\delta_2^0 + \delta_2^1}{2} = \delta_1^1 + \delta_2^0$	$\frac{\delta_1^1 - \delta_2^0}{2}$ X
-1 1			
1(-1) 1(-1)	1	$\frac{-\delta_1^1 + \delta_1^1}{2} + \frac{-\delta_1^1 - \delta_1^1 + \delta_2^0 - \delta_2^1}{2} + \frac{\delta_2^0 + \delta_2^1}{2} = -\delta_1^1 + \delta_2^1$	$\frac{\delta_1^1 - \delta_2^1}{2}$ X X
1 -1			

1	1	1	...
1	-1	1	$\delta_1^1 + \delta_2^0$
-1	-1	1	
-1	1	1	

No.

Date

16-bit

Sub

32-bit C ↙ 8-bit × 4 A B C D

32-bit }

1) 1→1 ✓

2)

4) 1→1 ✓

5) Sy A B CD → B A D C

1) Seed → ~~32~~ 16 challenge → 16_response - multi-bit

2) 16_res → sub →

3) Sub both

4) Permutation

1

No.

Date

Campus

12.08

d-c
b-a

up
bot
>

bot-up > 0 → 1
bot-up < 0 → 0

a
up c
bot d
b

X = Y ?

1-2C

Q

Δt: delay-bot > 0
D
Y=1
<0
Y=0

D: delay-bot > 0
1=0
Y=1

C= 0 1 0 0 0 1 1 0 bit
up
bot
b-a d-c a-b a-b a-b c-d d-c a-b
+ + − − − − + −
1-2C 1 −1 1 1 1 −1 −1 1
ft(c) 1 −1 −1 −1 −1 1 −1 −1
1 1 −1 −1 −1 −1 1 −1 (-1)

1 0 1 0 1 1 0 0
up
bot
d-c a-b c-d b-a d-c c-d b-a b-a
+ − − + + − + +
1 −1 1 −1 1 −1 −1 1 1
ft(c) −1 −1 1 1 −1 1 1 1
1 −1 −1 1 1 −1 1 1 (1)

up
bit

sign(delay)

sign(bit × delay)

$\delta_0 = b-a$
$\delta_1 = d-c$

KOKUYO

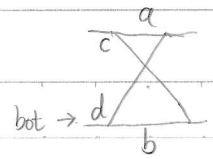
No.

Date

$\delta_0^{0/1}$ C_1 C_2 C_3 C_4

$\delta_0^0 = b - a$

$\delta_0^1 = d - c$



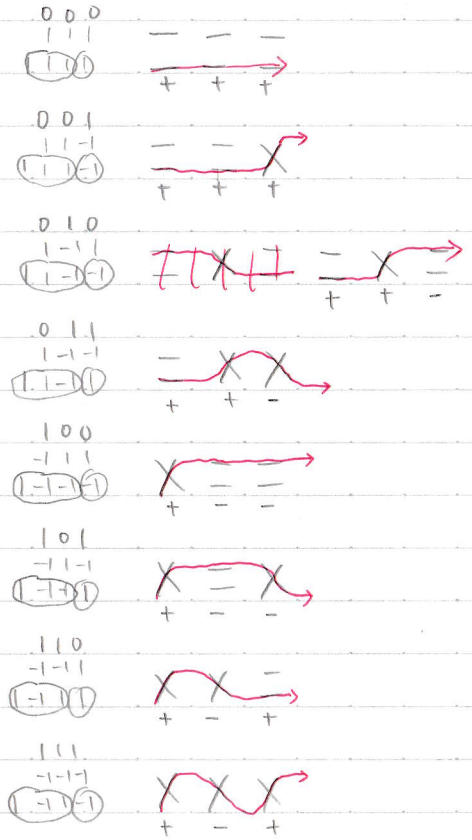
$f_t(C) = [f_1, f_2, f_3, f_4]$

$[1, f_1, f_2, f_3, f_4] \cdot [w_1 + w_2 + w_3 + w_4] \cdot f_4$

$[f_4, f_1 f_4, f_2 f_4, f_3 f_4, f_4^2] \cdot [w_1 + w_2 + w_3 + w_4]$

$\Rightarrow [f_4, f_1 f_4, f_2 f_4, f_3 f_4, 1] \cdot [w_1 + w_2 + w_3 + w_4]$

feature transformation



1 $\delta_1^0 + \delta_2^0 + \delta_3^0 = w_1 + w_2 + w_3 + w_4$

2 $\delta_1^0 + \delta_2^0 + \delta_3^1 = w_1 + w_2 + w_3 - w_4$

3 $\delta_1^0 + \delta_2^1 - \delta_3^0 = w_1 + w_2 - w_3 - w_4$

4 $\delta_1^0 + \delta_2^1 - \delta_3^1 = w_1 + w_2 - w_3 + w_4$

5 $\delta_1^1 - \delta_2^0 - \delta_3^0 = w_1 - w_2 - w_3 - w_4$

6 $\delta_1^1 - \delta_2^0 - \delta_3^1 = w_1 - w_2 - w_3 + w_4$

7 $\delta_1^1 - \delta_2^1 + \delta_2^0 = w_1 - w_2 + w_3 + w_4$

8 $\delta_1^1 - \delta_2^1 + \delta_3^1 = w_1 - w_2 + w_3 - w_4$

① $\delta_1^0 + \delta_2^0 + \delta_3^0 = w_1 + w_2 + w_3 + w_4$

② $\delta_1^0 + \delta_2^0 + \delta_3^1 = w_1 + w_2 + w_3 - w_4$

$\Rightarrow w_4 = \frac{\delta_3^0 - \delta_3^1}{2}$

② $\delta_1^0 + \delta_2^1 - \delta_3^0 = w_1 + w_2 - w_3 - w_4$

⑤ $\delta_1^1 - \delta_2^0 - \delta_3^0 = w_1 - w_2 - w_3 - w_4$

$\Rightarrow w_2 = \frac{\delta_1^0 - \delta_1^1 + \delta_2^1 + \delta_2^0}{2}$

② $\delta_1^0 + \delta_2^0 + \delta_3^1 = w_1 + w_2 + w_3 - w_4$

③ $\delta_1^0 + \delta_2^1 - \delta_3^0 = w_1 + w_2 - w_3 - w_4$

$\Rightarrow w_3 = \frac{\delta_2^0 - \delta_2^1 + \delta_3^1 + \delta_3^0}{2}$

⑤ $\Rightarrow w_1 = \frac{\delta_1^0 + \delta_1^1}{2}$

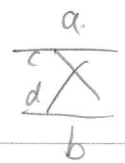
Campus

KOKUYO

No.

Date

3-bit



0 0 0 — — —

bot. → b-a b-a b-a

0 0 1 — — —

→ b-a b-a d-c

0 1 0 — — —

→ b-a d-c a-b

0 1 1 — — —

b-a d-c c-d

1 0 0 → → →

d-c a-b a-b

1 0 1 → →

d-c a-b c-d

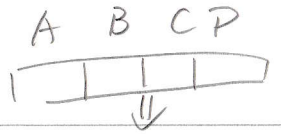
1 1 0 — —

1 1 1

15000

End Note

16



No. 6353

Date

$w^1 = \frac{\delta_1^0 - \delta_1^1}{2}$

$w^2 = \frac{\delta_1^0 + \delta_1^1 + \delta_2^0 - \delta_2^1}{2}$

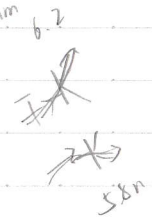
$w^3 = \frac{\delta_2^0 + \delta_2^1}{2}$

$\vec{\Phi}_j = \Pi_{i=j}^{2} (1-2C_i)$ for $j = 1, \ldots, 2$

0 0	→ 1, 1, 1	$\frac{\delta_1^0 - \delta_1^1}{2} + \frac{\delta_1^0 + \delta_1^1 + \delta_2^0 - \delta_2^1}{2} + \frac{\delta_2^0 + \delta_2^1}{2} = \delta_1^0 + \delta_2^0$
0 1	→ -1, -1, 1	$\frac{-\delta_1^0 + \delta_1^1}{2} + \frac{-\delta_1^0 + \delta_1^1 - \delta_2^0 + \delta_2^1}{2} + \frac{\delta_2^0 + \delta_2^1}{2} = -\delta_1^0 + \delta_2^1$
1 0	→ -1 1, 1	$\frac{\delta_1^0 + \delta_1^1}{2} + \frac{\delta_1^0 + \delta_1^1 + \delta_2^0 - \delta_2^1}{2} + \frac{\delta_2^0 - \delta_2^1}{2} = \delta_1^1 + \delta_2^0$
1 1	→ 1 -1 1	$\frac{\delta_1^0 - \delta_1^1}{2} + \frac{\delta_1^0 - \delta_1^1 + \delta_2^0 + \delta_2^1}{2} + \frac{\delta_2^0 + \delta_2^1}{2} = -\delta_1^1 + \delta_2^1$

00		$\delta_1^0 + \delta_2^0$		$\delta_1^0 + \delta_2^0$	0 0	1 1
01		$\delta_1^0 + \delta_2^0$		$-\delta_1^0 + \delta_2^0$	0 1	1 -1
10		$\delta_1^1 - \delta_2^0$		$\delta_1^1 + \delta_2^0$	1 0	-1 1
11		$\delta_1^1 - \delta_2^1$		$-\delta_1^1 + \delta_2^0$	1 1	-1 -1

Map seed → 16 bits.



5th → 1th

No.

Date

Thesis

3 { Chapter 1 motivation → 3 Page Stating actual problem PUF can be modeled by ML Security application
— structure

clear { 2 Background
— PUF description
— ML alg LR (Example) ↙ detailed NN, SVM
— design part { per / sub

3 Design (cheap cost)
- goal
- experiment (setup) visio (figures)
- ↑ what have been done

LESR
Arbiter

{ ① original
sub Input / Output
4X per Input
Combination of sub & per (Input)

5000 samples
(At least 8 bits)

security and reliability
[0,]%
~~Error~~

Seed
figures

4 Evaluation

65nm Cost arbiter + sub / per benfit
arbiter + hash function

5 Conclusion
future

EndNote for referencing

env
access to CRPs
other

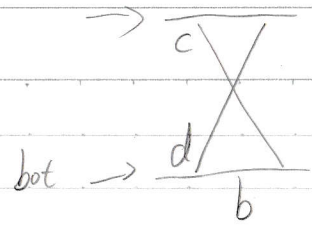
~~In design goal~~
threat model

Switch

a b c d
b a d c

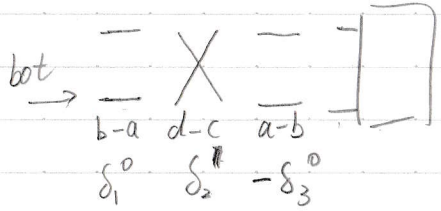
No.

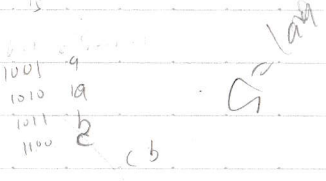
Date

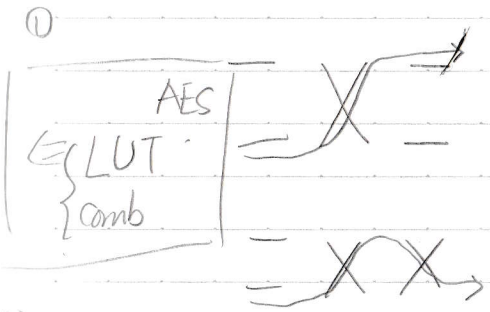


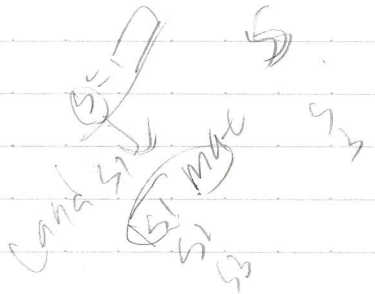
$\delta_1^0 = b - a$

$\delta_1' = d - c$









permutation

per + Seed + sub

sub + Seed + per

Two
① hash function
② Multiply the prediction of all bits
③ Area (GE)