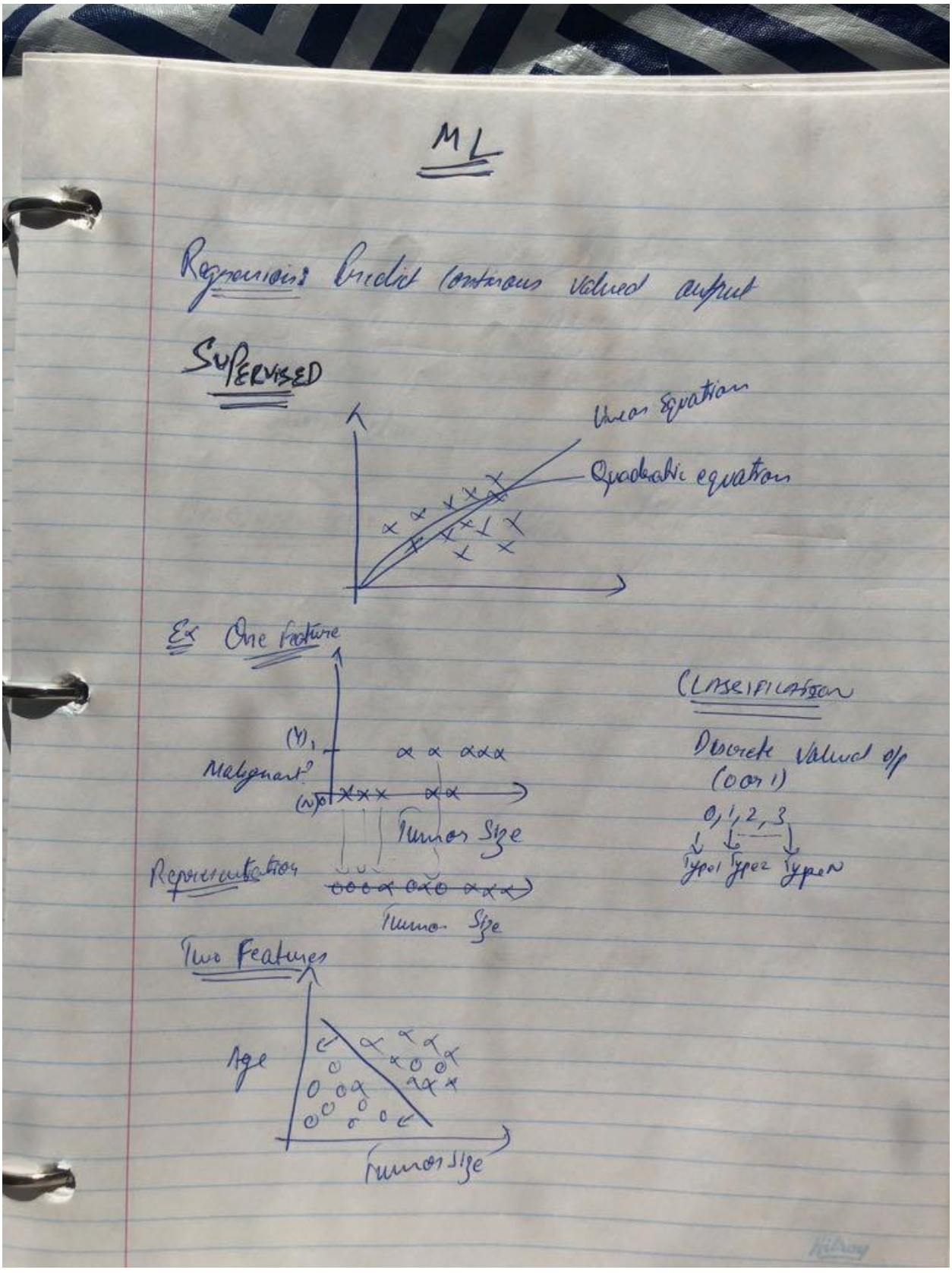

ML
Regression: Predict continuous valued output
SUPERVISED
Linear equation
Quadratic equation
Ex One feature
(Y)
Malignant?
(No)
Tumor Size
Representation
Tumor Size
Two Features
Age
Tumor Size
CLASSIFICATION
Discrete valued o/p
(0 or 1)
0, 1, 2, 3
Type1 Type2 Type3

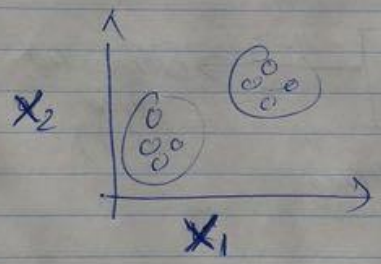
Problem 1: Items identical, how many you'll sell over 3 months

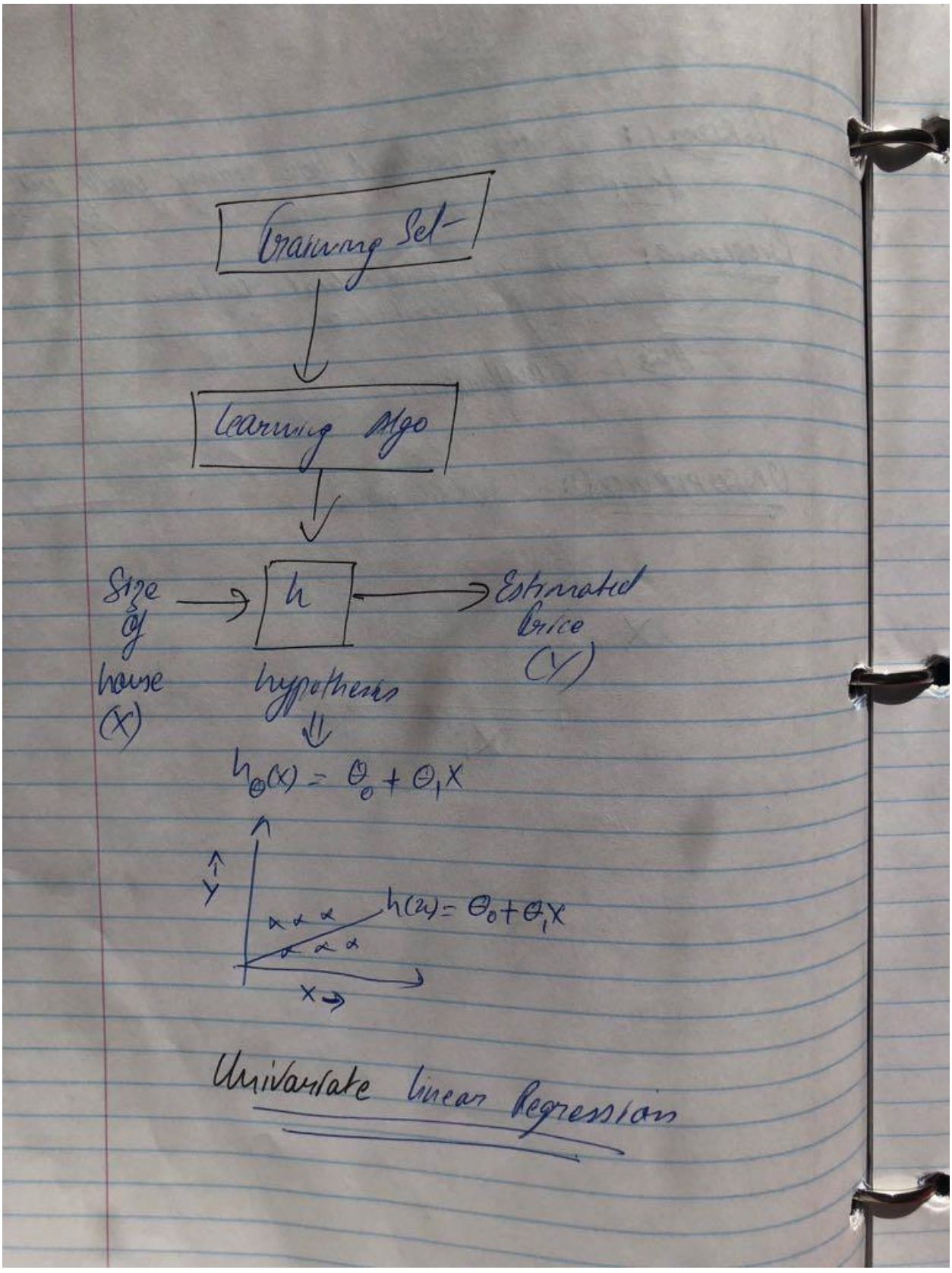
Ans: Regression

Problem 2: Examine individual customer accounts, & decide if it hacked or not?

Ans: classification

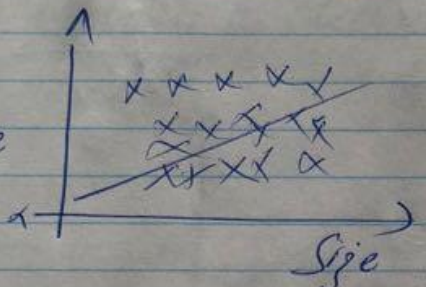
UNSUPERVISED: JUST DATA




Training Set
Learning Algo
Size of house (X)
h
Estimated price (Y)
hypothesis
$h_\theta(x) = \theta_0 + \theta_1 x$
Y
X
$h(x) = \theta_0 + \theta_1 x$
Univariate linear Regression

LINEAR REGRESSION

HOUSING PRICES [Regression Problem]



Size (in ft²) (x)	Price ($) (y)
2104	460
1234	52
⋮	⋮

$m = 47$

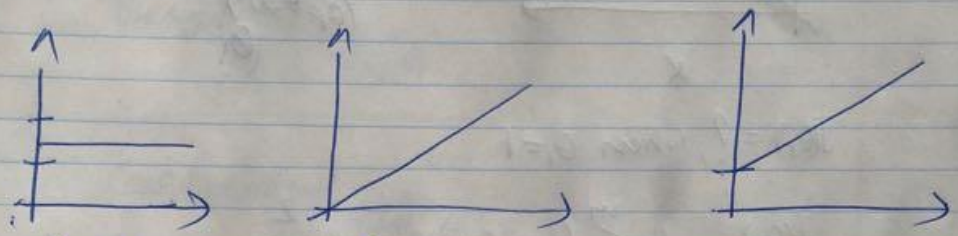
(x, y) - one Example

$(x^{(i)}, y^{(i)})$ - i^{th} training Example | $(x^{(1)}, y^{(1)}) = (2104, 460)$

Hypothesis:

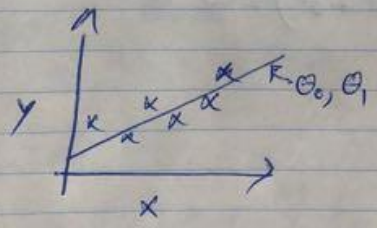
$$h_\theta(x) = \theta_0 + \theta_1 x$$

Q: how to choose θ_0, θ_1? [Parameters]



$\theta_0 = 1.5$, $\theta_1 = 0$ | $\theta_0 = 0$, $\theta_1 = 0.5$ | $\theta_0 = 1$, $\theta_1 = 0.5$

We have to come up values that corresponds to A Good Fit To Our Data.



MATHS:

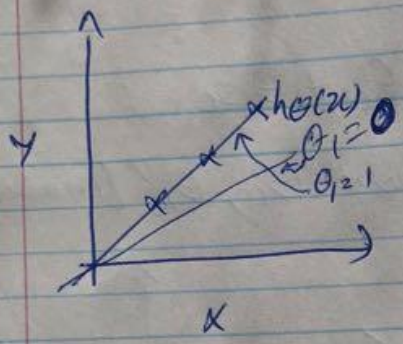
$$\underset{\theta_0, \theta_1}{\text{minimise}} \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)} \right)^2$$

(training Examples; $h_\theta(x^{(i)})$: our prediction; $y^{(i)}$: actual label)

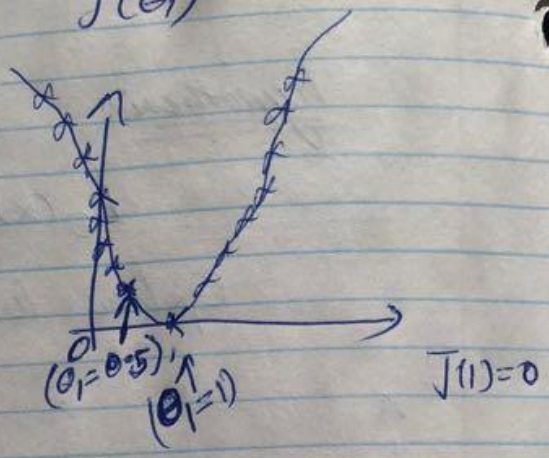
SQUARED ERROR FUNCTION / COST FUNCTION

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

$h_\theta(x)$



$J(\theta_1)$



$J(\theta_1) = ?$, when $\theta_1 = 1$

$$J(\theta_1) = \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)} \right)^2$$

$$= \frac{1}{2m} \sum_{i=1}^{m} \left(\theta_1 x^{(i)} - y^{(i)} \right)^2$$

$$= \frac{1}{2m} \left(0^2 + 0^2 + 0^2 \right) = 0^2$$

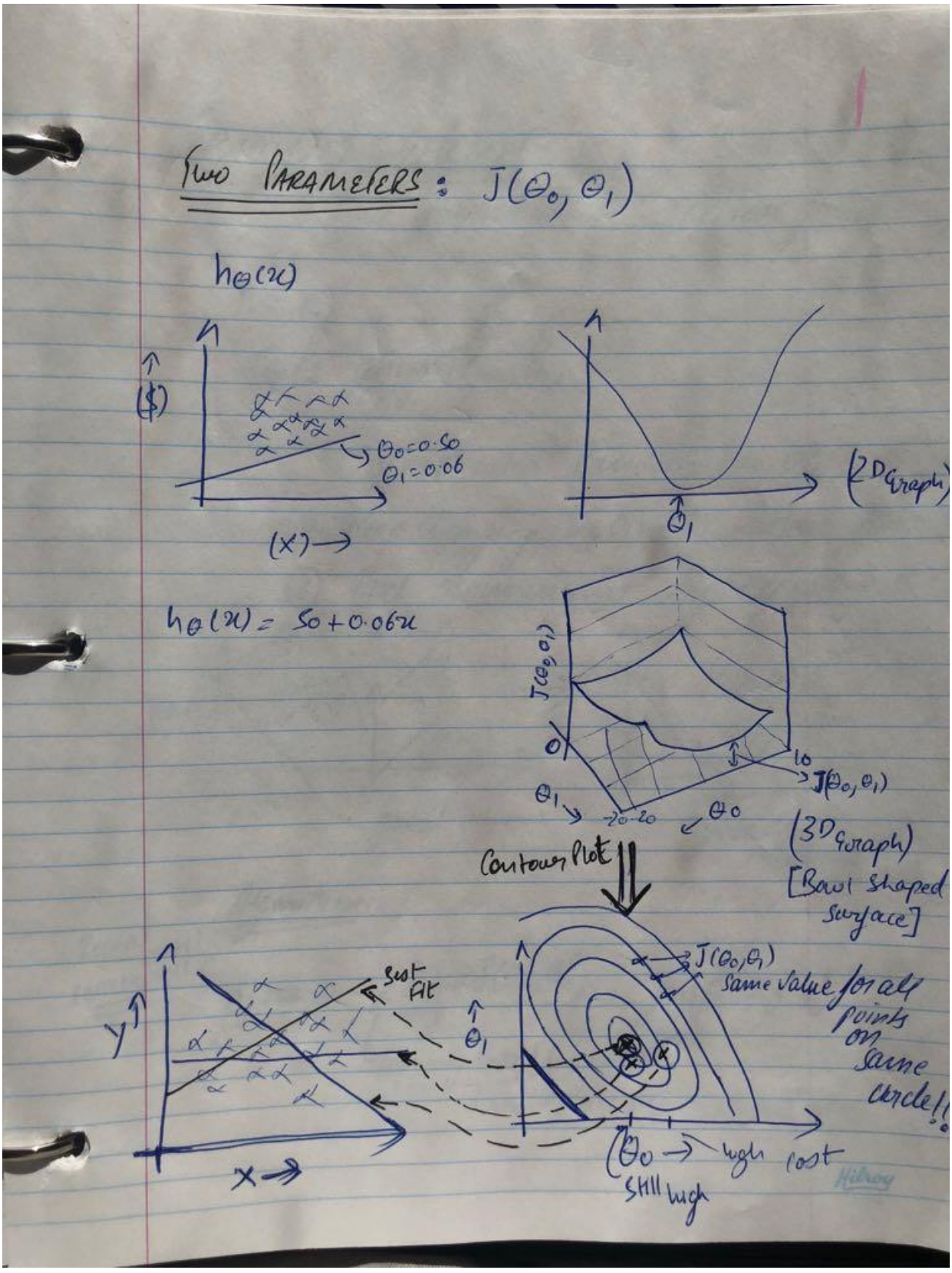
$\Rightarrow J(1) = 0$

$J(0.5) = ?$

$$J(0) = \frac{1}{2m} (1^2 + 2^2 + 3^2)$$

$$= \frac{1}{6} \cdot 14 = 2.3$$

If $\theta_1 = -ve$, $h(x) = -0.5$


Two PARAMETERS: $J(\theta_0, \theta_1)$
$h_\theta(x)$
($)
$\theta_0 = 0.50$
$\theta_1 = 0.06$
(X) →
(2D Graph)
$h_\theta(x) = 50 + 0.06x$
$J(\theta_0, \theta_1)$
θ_1
θ_0
(3D Graph)
[Bowl shaped surface]
Contour Plot
Best Fit
y
x
$J(\theta_0, \theta_1)$ Same value for all points on same circle!!
(θ_0 → high cost
Still high

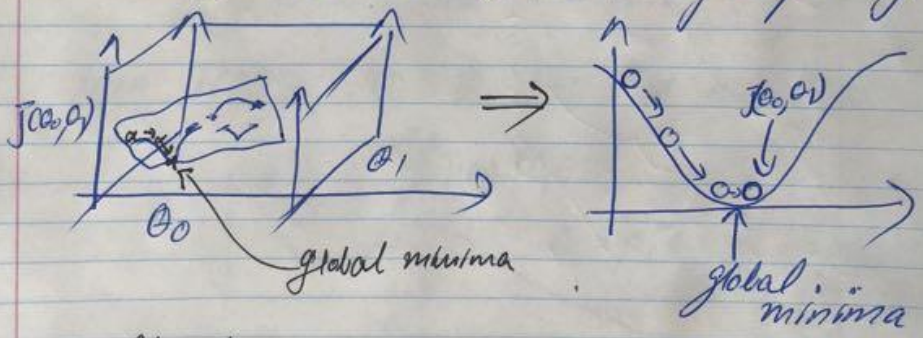
GRADIENT DESCENT

A way to minimise J() / cost function

$J(\theta_0, \theta_1)$

Goal: $\min_{\theta_0, \theta_1} J(\theta_0, \theta_1)$

1. Start with Random (θ_0, θ_1)
2. Keep changing (θ_0, θ_1) to reduce $J(\theta_0, \theta_1)$
3. Find minimum by repeating (2).



Algorithm:

Repeat until convergence {

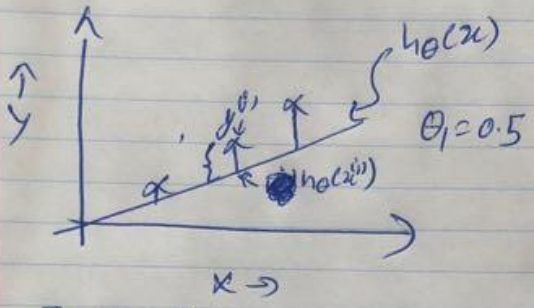
$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1) \quad \text{for } (j=0 \text{ \& } j=1)$$

}

α → learning rate [Any number] → how big of a step to take??

$h_\theta(x)$

(for fixed θ_1, this function of x)



$J(0.5) = ?$

$$= \frac{1}{2m}\left[(0.5-1)^2 + (1-2)^2 + (1.5-3)^2\right]$$

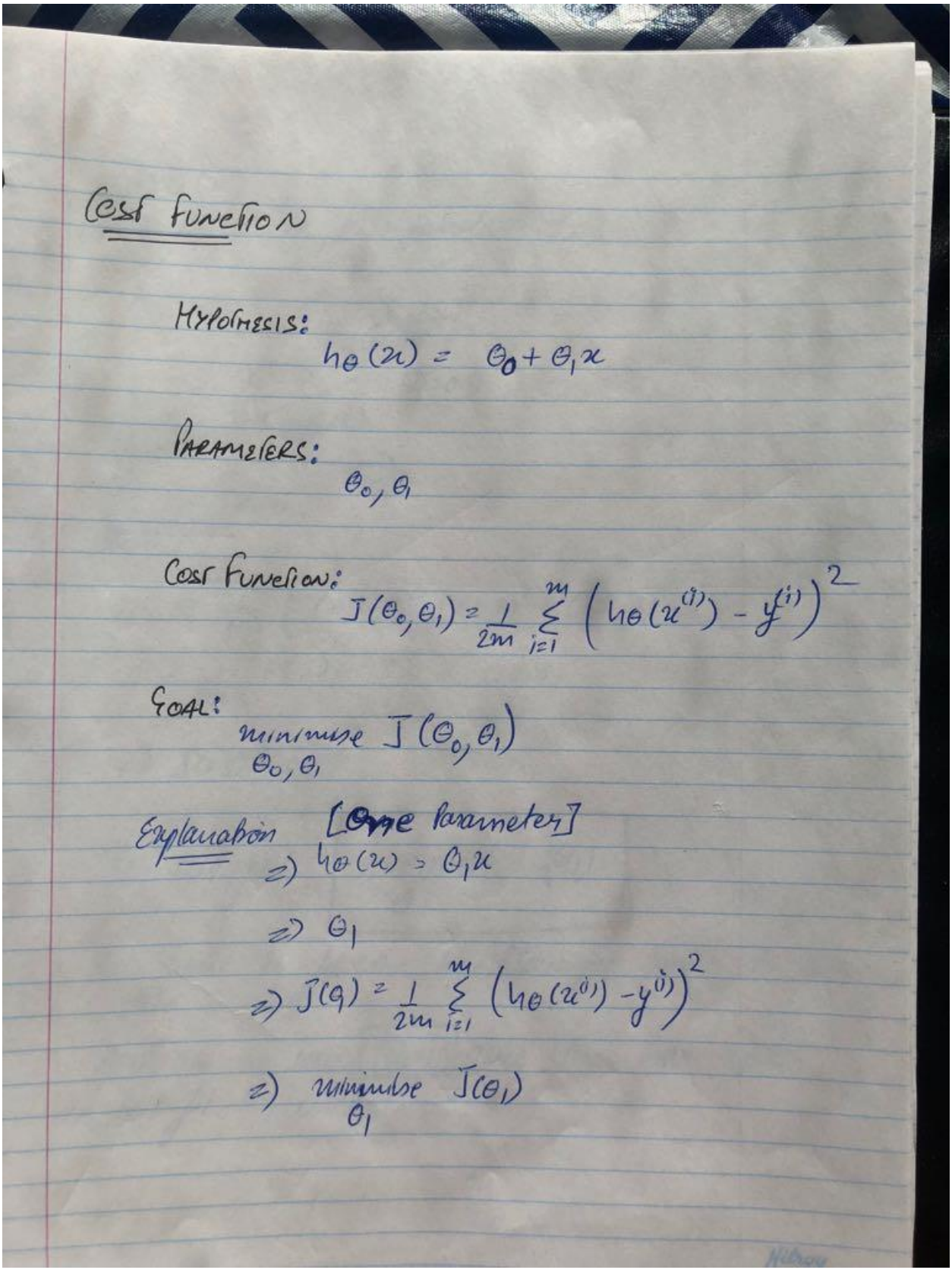
$$= \frac{1}{2\times 3}(3.5) = \frac{3.5}{6} = 0.58$$

$\Rightarrow J(0.5) = 0.58$

So, our goal $\boxed{\underset{\theta_1}{\text{minimize}}\ J(\theta_1)}$

We want to fit the line (Straight line) to our data

And, as we saw $\boxed{(\theta_1 = 1)}$ gave the line that best fit the data!



Cost function

Hypothesis:

$$h_\theta(x) = \theta_0 + \theta_1 x$$

Parameters:

$$\theta_0, \theta_1$$

Cost Function:

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

Goal:

$$\underset{\theta_0, \theta_1}{\text{minimize}}\ J(\theta_0, \theta_1)$$

Explanation [One parameter]

$\Rightarrow h_\theta(x) = \theta_1 x$

$\Rightarrow \theta_1$

$\Rightarrow J(\theta_1) = \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)} \right)^2$

$\Rightarrow \underset{\theta_1}{\text{minimize}}\ J(\theta_1)$

Polynomial Regression

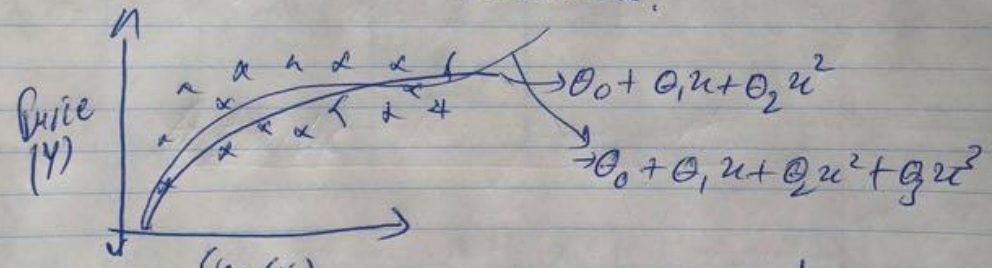
House Price Prediction

$h_\theta(x) = \theta_0 + \theta_1 \times \underbrace{\text{frontage}}_{x_1} + \theta_2 \times \underbrace{\text{depth}}_{x_2}$

New feature ⟹ Area

x = Frontage * depth

$h_\theta = \theta_0 + \theta_1 x$ ← land area.



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

$= \theta + \theta_1(\text{Size}) + \theta_2(\text{Size})^2 + \theta_3(\text{Size})^3$

$x_1 = \text{Size}$

$x_2 = (\text{Size})^2$

$x_3 = (\text{Size})^3$

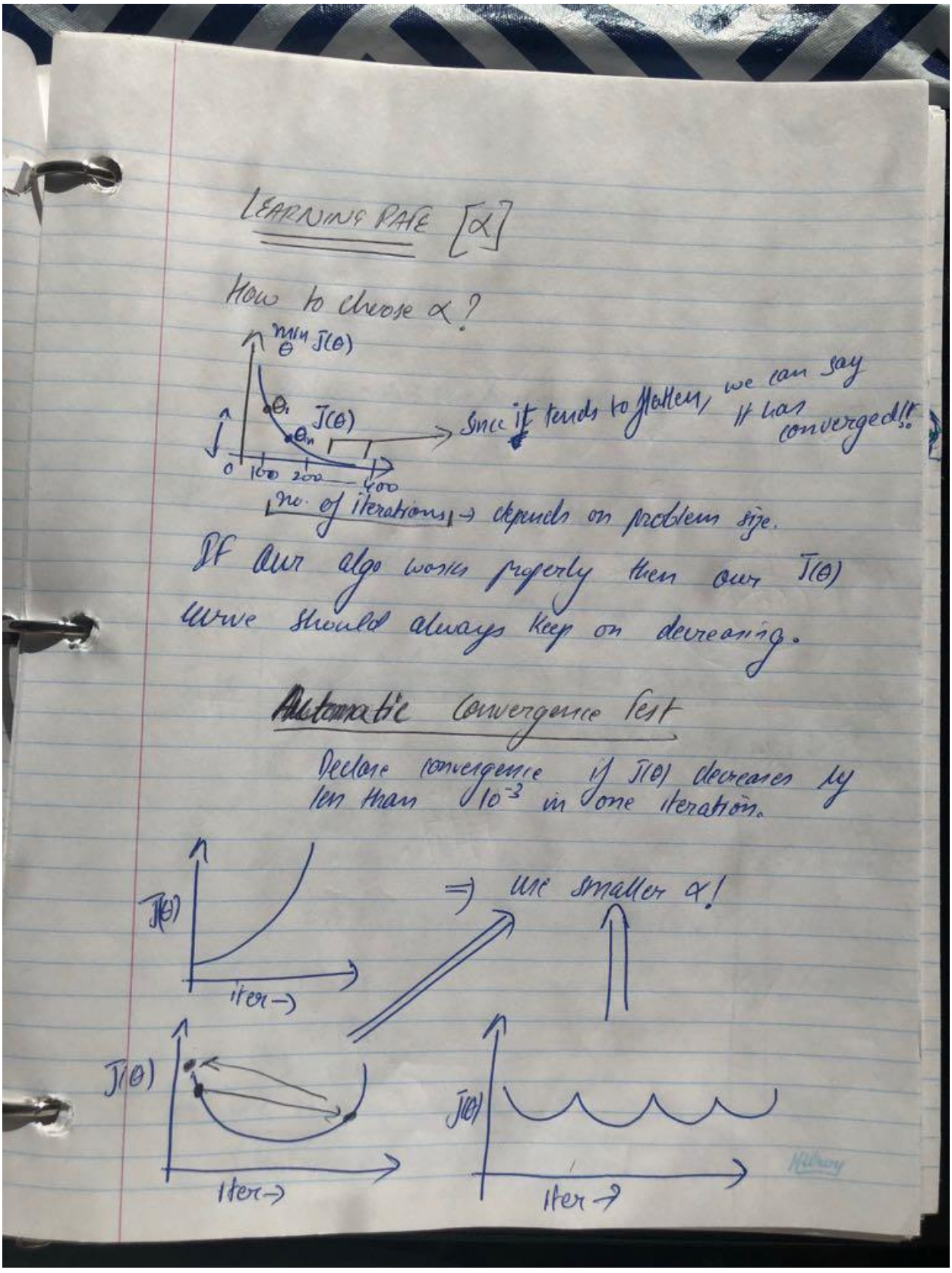
Size : 1-1000

Size^2 : 1-1000,000

Size^3 : 1-10^9

Or we choose something like:

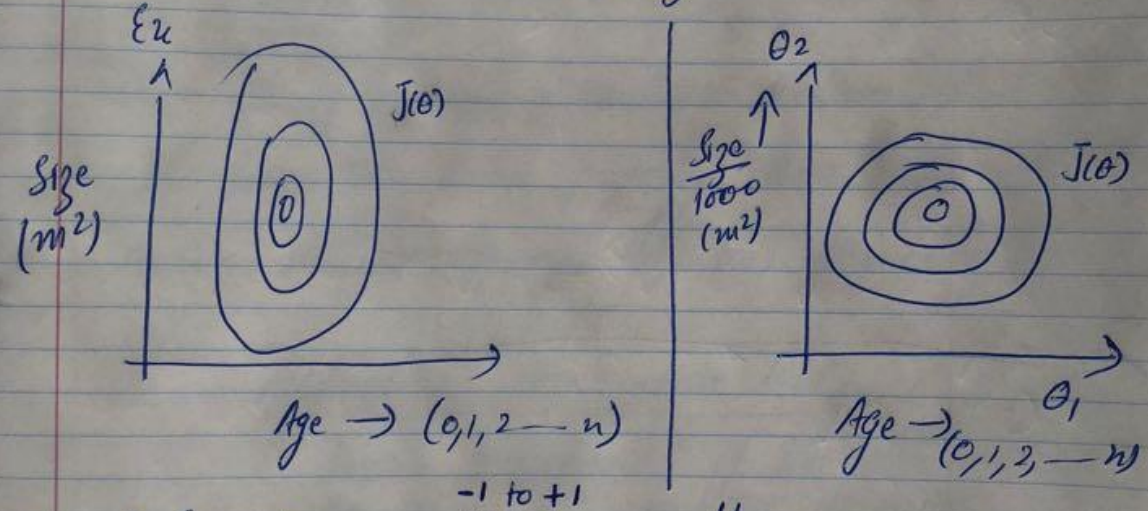
$$h_\theta(x) = \theta_0 + \theta_1(\text{Size}) + \theta_2\sqrt{\text{Size}}$$


LEARNING RATE [α]
How to choose α?
min J(θ)
θ
J(θ)
θ1
θn
0
100
200
400
Since it tends to flatten, we can say it has converged!!
no. of iterations → depends on problem size.
If our algo works properly then our J(θ) curve should always keep on decreasing.
Automatic convergence test
Declare convergence if J(θ) decreases by less than 10^-3 in one iteration.
J(θ)
iter →
⇒ use smaller α!
J(θ)
iter →
J(θ)
iter →

Feature Scaling

Make sure feature are on same scale so that gradient descent converges quickly.

If the scale is too big, then the contours graph will be very elongated causing it harder to converge.



⇒ Get feature to -1 to +1 range!! [(+3 to -3) is fine]

$[-1 \leq x_i \leq 1]$

One way:- Normalisation!! [Don't apply $x_0 = 1$ with it]

$$x_1 \leftarrow \frac{x_1 - \mu_1}{s_1}$$

(μ_1 ← mean of x_j; s_1 ↑ range $[(x_j)_{max} - (x_j)_{min}]$)

Gradient descent:

Repeat {

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$$

}

[Simultaneous update for every $j = 0, \dots, n$]

Algorithm

Repeat {

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

(Simultaneous update θ_j for $j = 0, 1, \dots n$)

}

$$\theta_0 = \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)}\right)^2 x_0^{(i)}$$

$$\theta_1 = \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)}\right)^2 x_1^{(i)}$$

$$\vdots$$

$$\theta_n = \theta_n - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)}\right)^2 x_n^{(i)}$$

$[\theta_0, \theta_1, \theta_2 \text{ —— }, \theta_n]$ $\begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ | \\ x_n \end{bmatrix}$

$(n+1) \times 1$

θ^T $\quad x$

$\Rightarrow \boxed{h_\theta(x) = \theta^T x}$

Multivariate Linear Regression

Hypothesis: ($x_0 = 1$ [assumption])

$$h_\theta(x) = \theta^T x = \theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \text{ —— } + \theta_n x_n$$

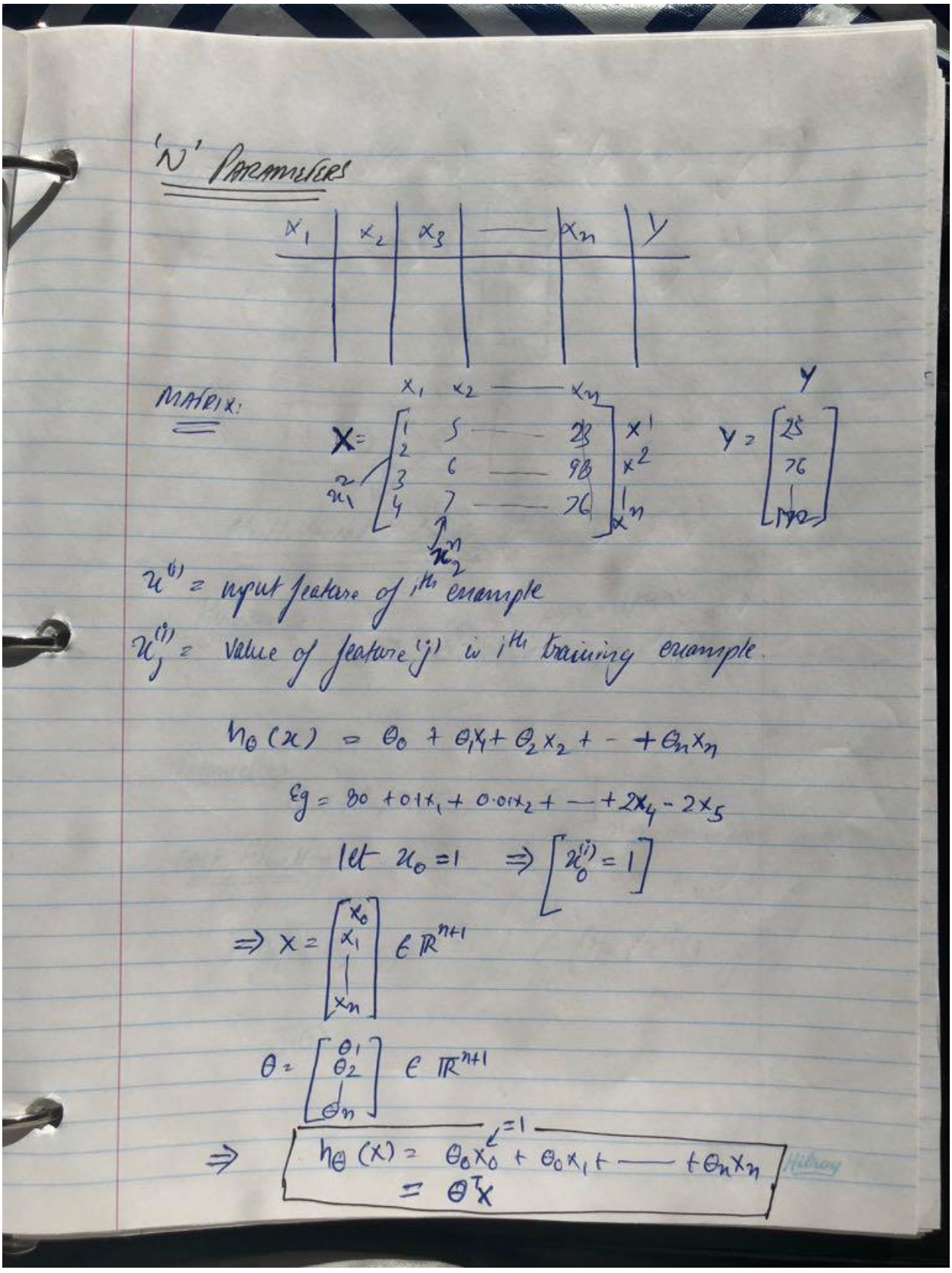
Parameters:

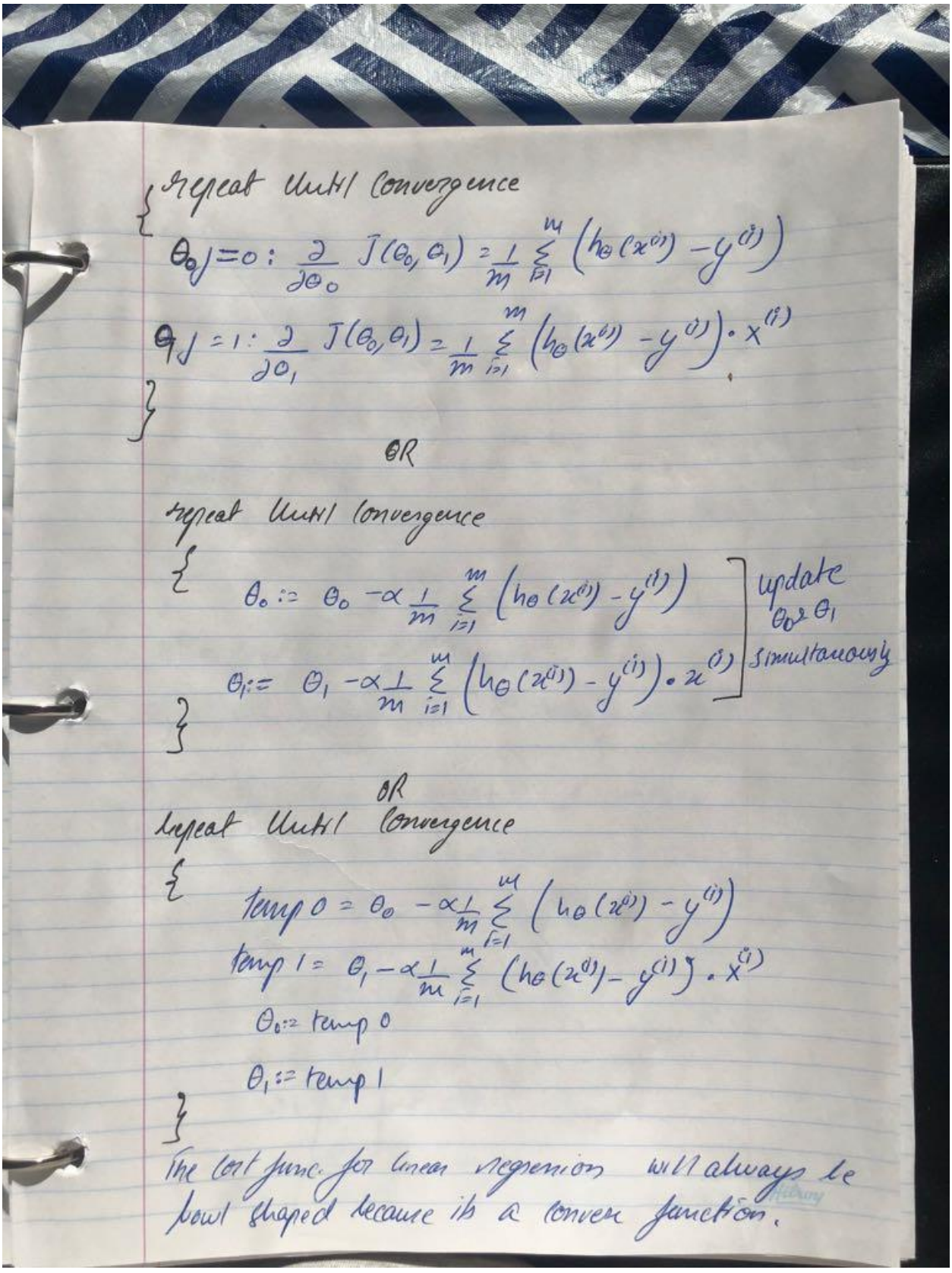
$\theta = [\theta_0, \theta_1, \theta_2 \text{ —— }, \theta_n]$ $\quad n+1$-dimension vector

Cost function:

$$J(\theta_0, \theta_1, \theta_2 \text{ —— }, \theta_n) = \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)}\right)^2$$

$J(\theta_0, \theta_1, \theta_2 \text{ —— }, \theta_n) \rightarrow J(\theta)$


'N' Parameters
x_1 x_2 x_3 —— x_n y
MATRIX:
x_1 x_2 —— x_n
x^1
x^2
x^n
y
x_1^2
x_2^n
$x^{(i)}$ = input feature of i^{th} example
$x_j^{(i)}$ = value of feature 'j' in i^{th} training example.
$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$
Eg = $80 + 0.1x_1 + 0.01x_2 + \cdots + 2x_4 - 2x_5$
let $x_0 = 1 \Rightarrow [x_0^{(i)} = 1]$
$\Rightarrow x = \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_n \end{bmatrix} \in \mathbb{R}^{n+1}$
$\theta = \begin{bmatrix} \theta_1 \\ \theta_2 \\ \vdots \\ \theta_n \end{bmatrix} \in \mathbb{R}^{n+1}$
$\Rightarrow$ $h_\theta(x) = \theta_0 x_0 + \theta_0 x_1 + \cdots + \theta_n x_n$ ($x_0 = 1$)
$= \theta^T x$



Repeat until convergence

$$\theta_0 := \theta_0 - \alpha \frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1) = \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)}\right)$$

$$\theta_1 := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_0, \theta_1) = \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)}\right) \cdot x^{(i)}$$

}

OR

repeat until convergence

{

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

$$\theta_1 := \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)}\right) \cdot x^{(i)}$$

] update θ_0 & θ_1 simultaneously

}

OR

repeat until convergence

{

$$temp\,0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)}\right)$$

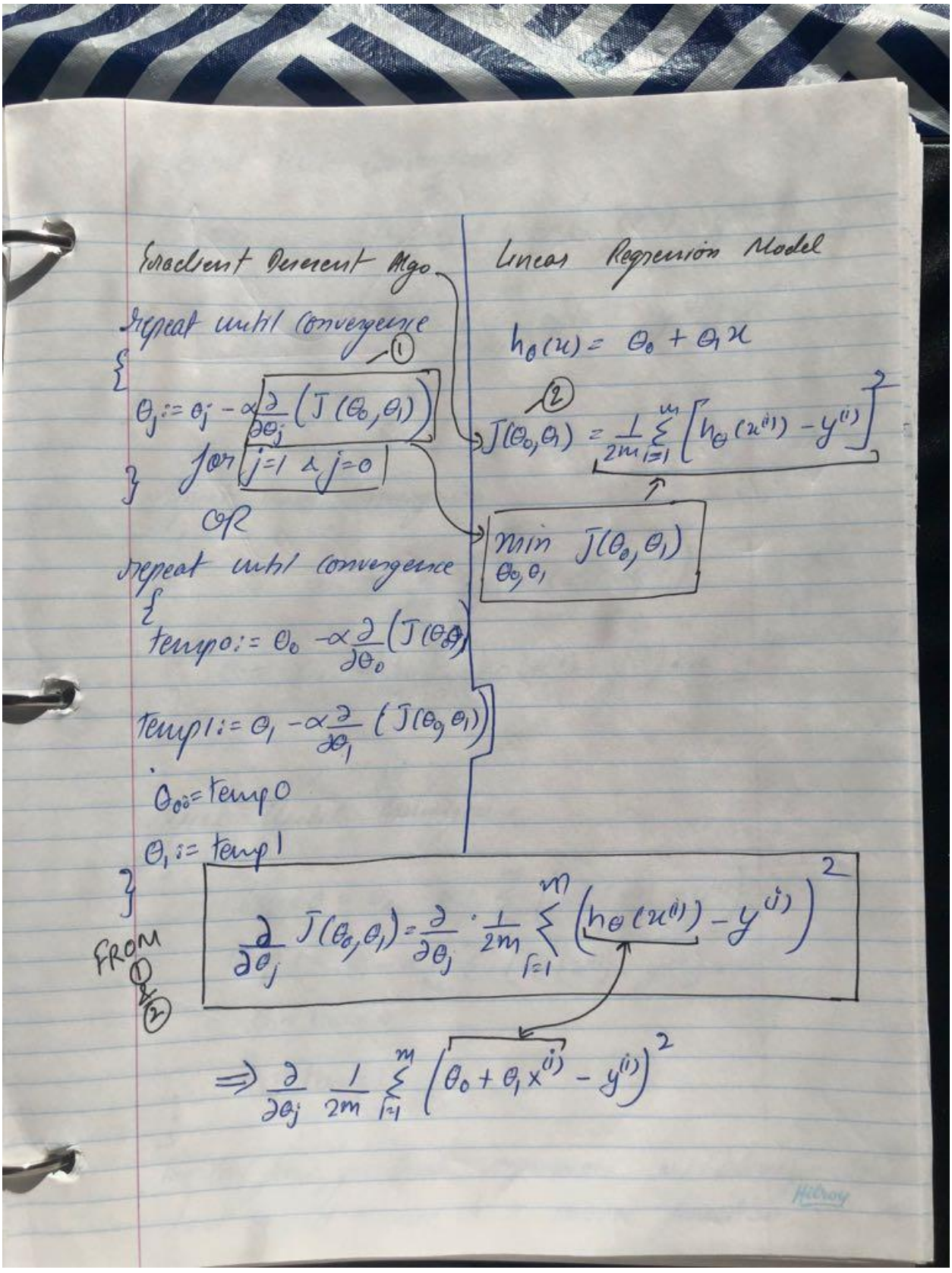
$$temp\,1 := \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)}\right) \cdot x^{(i)}$$

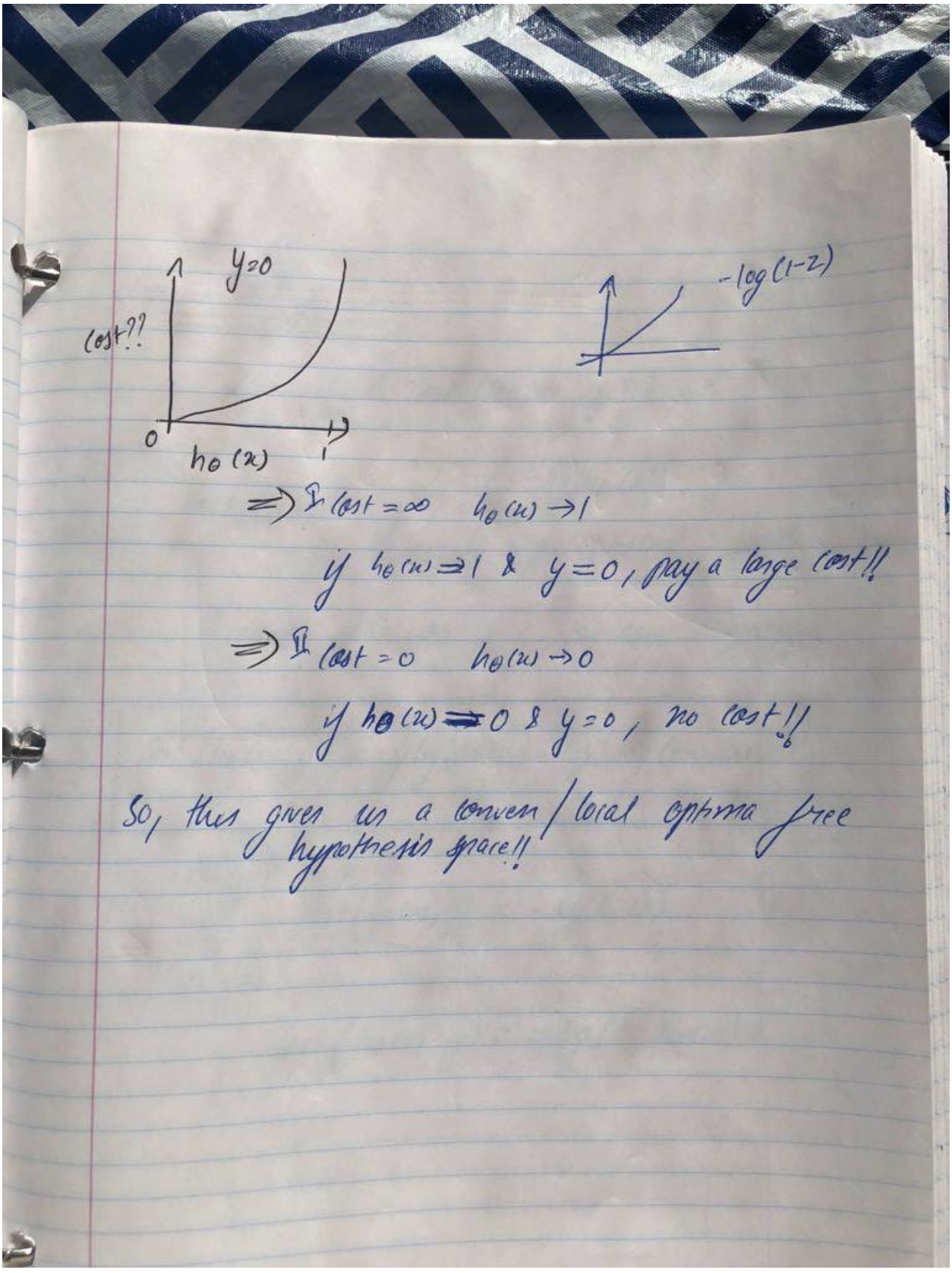
$\theta_0 := temp\,0$

$\theta_1 := temp\,1$

}

The cost func. for linear regression will always be bowl shaped because it's a convex function.


Gradient Descent Algo
Repeat until convergence
{
$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} (J(\theta_0, \theta_1))$ ①
} for $j=1$ & $j=0$
OR
Repeat until convergence
{
$temp0 := \theta_0 - \alpha \frac{\partial}{\partial \theta_0} (J(\theta_0, \theta_1))$
$temp1 := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} (J(\theta_0, \theta_1))$
$\theta_0 := temp0$
$\theta_1 := temp1$
}
Linear Regression Model
$h_\theta(x) = \theta_0 + \theta_1 x$
② $J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} \left[h_\theta(x^{(i)}) - y^{(i)} \right]^2$
$\min_{\theta_0, \theta_1} J(\theta_0, \theta_1)$
FROM ① & ②
$\frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1) = \frac{\partial}{\partial \theta_j} \cdot \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)} \right)^2$
$\Rightarrow \frac{\partial}{\partial \theta_j} \frac{1}{2m} \sum_{i=1}^{m} \left(\theta_0 + \theta_1 x^{(i)} - y^{(i)} \right)^2$



$\Rightarrow$ I cost $= \infty$ $h_\theta(x) \to 1$

if $h_\theta(x) \Rightarrow 1$ & $y = 0$, pay a large cost!!

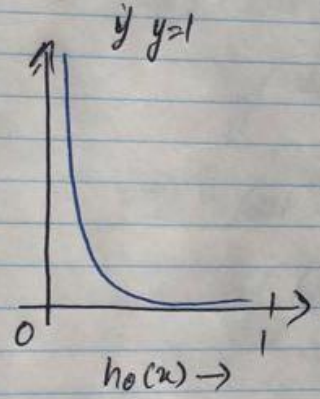
$\Rightarrow$ II cost $= 0$ $h_\theta(x) \to 0$

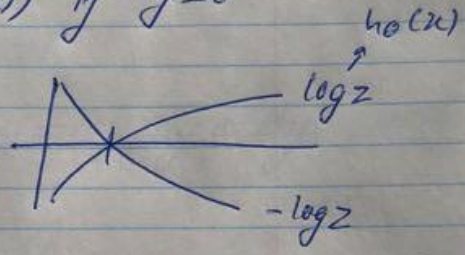
if $h_\theta(x) \Rightarrow 0$ & $y = 0$, no cost!!

So, thus gives us a convex/local optima free hypothesis space!!

Logistic Regression Cost Function

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





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

But as
$h_\theta(x) \rightarrow 0$
$\text{Cost} \rightarrow \infty$

⇒ captures intuition that if $h_\theta(x) = 0$
predict $P(y=1|x;\theta) = 0$
but $y=1$ for that x, then
we'll penalize learning algorithm by a
very large cost.

Example

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

m examples; $x \in \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_n \end{bmatrix}$ $x_0 = 1$, $y \in \{0, 1\}$

Hypothesis

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

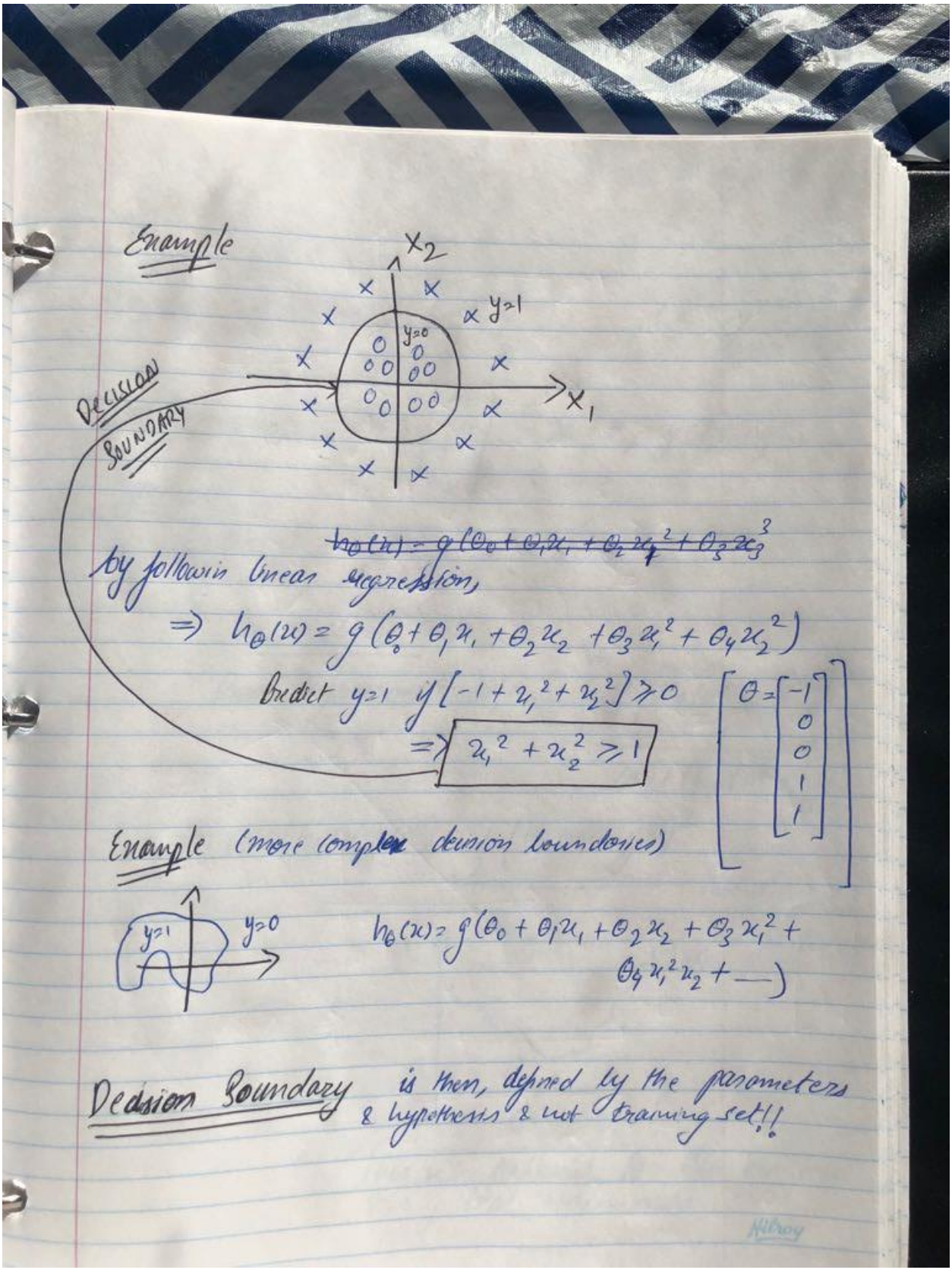
How to choose θ??

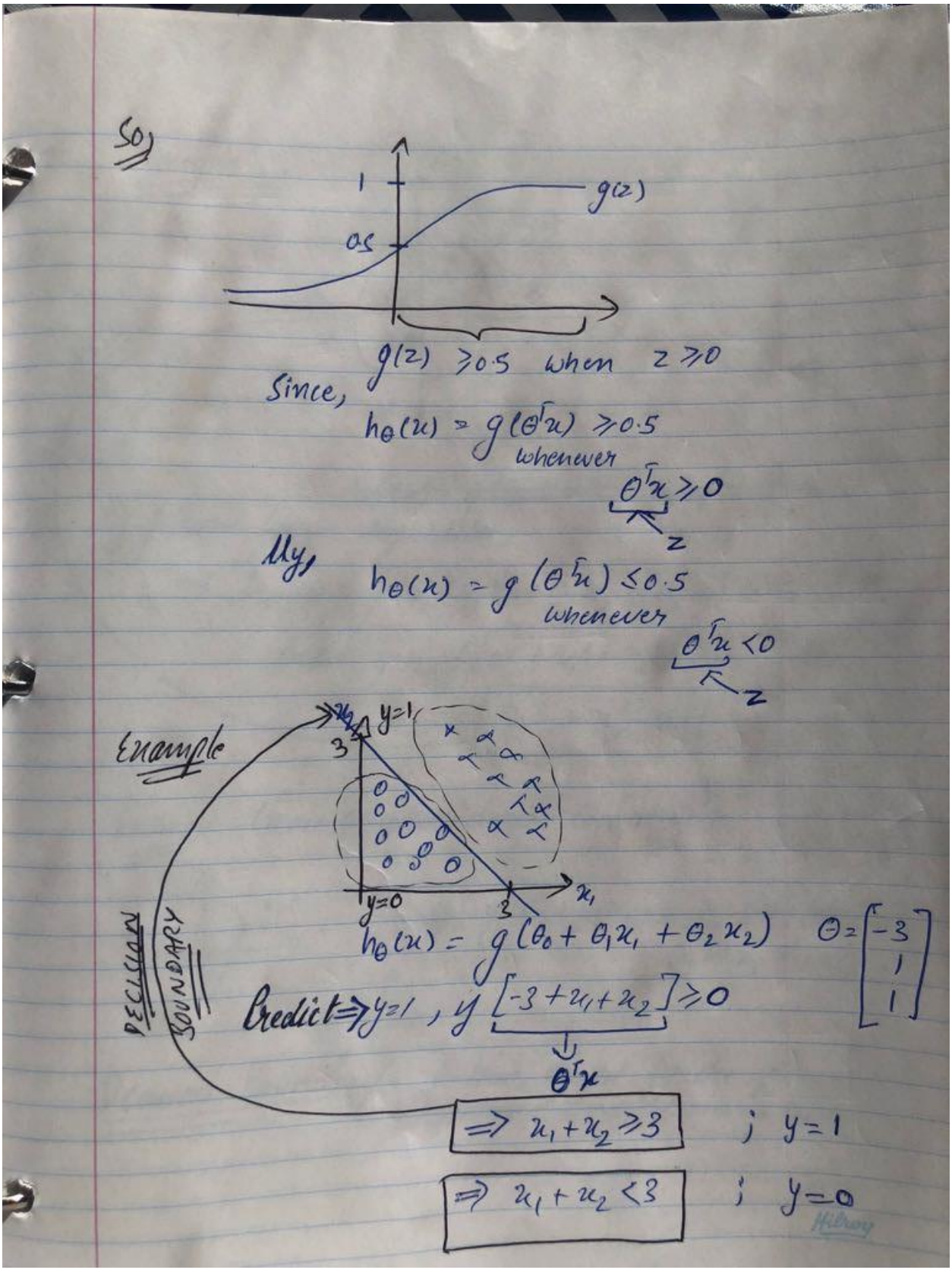
Linear Regression :=

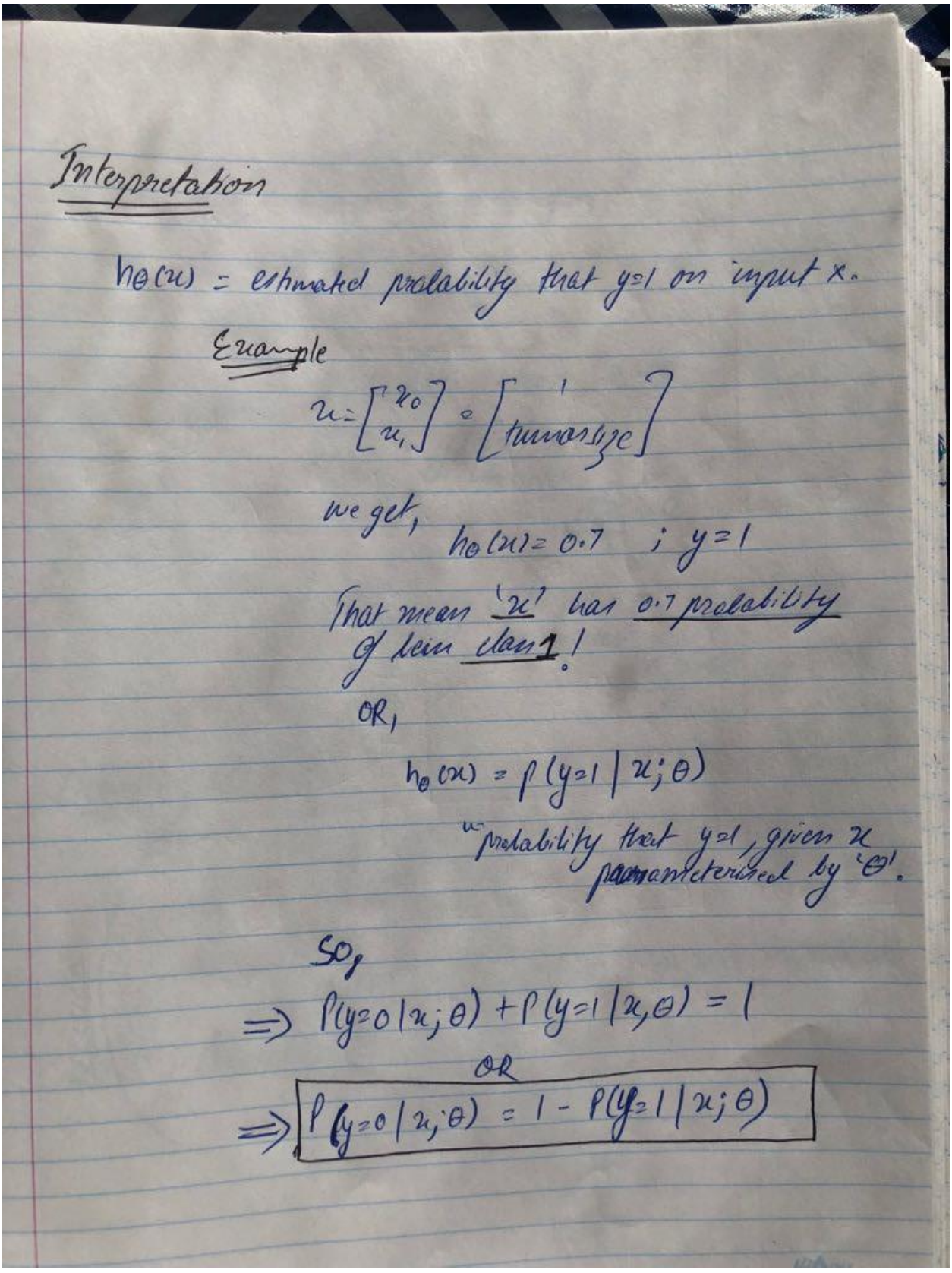
$$J(\theta) = \frac{1}{m} \sum_{i=1}^{m} \underbrace{\frac{1}{2}\left(h_\theta(x^{(i)}) - y^{(i)}\right)^2}_{Cost(h_\theta(x^{(i)}), y)}$$

$$\Rightarrow Cost(h_\theta(x^{(i)}), y^{(i)}) = \underbrace{\frac{1}{2}\left(h_\theta(x^{(i)}) - y^{(i)}\right)^2}_{\text{This is a non convex function!!}}$$

This is a problem! As there are too many local minimas.


Example
x_2
x_1
$y=1$
$y=0$
Decision Boundary
~~$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_1^2 + \theta_3 x_3^3$~~
by following linear regression,
$\Rightarrow h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_2^2)$
Predict $y=1$ if $[-1 + x_1^2 + x_2^2] \geq 0$
$\Rightarrow x_1^2 + x_2^2 \geq 1$
$\theta = \begin{bmatrix} -1 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix}$
Example (more complex decision boundaries)
$y=1$
$y=0$
$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_1^2 x_2 + \ldots)$
Decision Boundary is then, defined by the parameters & hypothesis & not training set!!


So,
1
0.5
g(z)
g(z) ≥ 0.5 when z ≥ 0
Since, $h_\theta(x) = g(\theta^T x) \geq 0.5$ whenever $\theta^T x \geq 0$
z
Ny, $h_\theta(x) = g(\theta^T x) \leq 0.5$ whenever $\theta^T x < 0$
z
Example
y=1
3
y=0
3
x_1
x_2
DECISION BOUNDARY
$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$
$\theta = \begin{bmatrix} -3 \\ 1 \\ 1 \end{bmatrix}$
Predict ⇒ y=1, if $[-3 + x_1 + x_2] \geq 0$
$\theta^T x$
$\Rightarrow x_1 + x_2 \geq 3$; y=1
$\Rightarrow x_1 + x_2 < 3$; y=0



Interpretation

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

Example

$$x = \begin{bmatrix} x_0 \\ x_1 \end{bmatrix} = \begin{bmatrix} 1 \\ \text{tumorsize} \end{bmatrix}$$

we get, $h_\theta(x) = 0.7 \quad ; y=1$

That mean 'x' has 0.7 probability of been class 1!

OR,

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

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

So,

$$\Rightarrow P(y=0 \mid x;\theta) + P(y=1 \mid x,\theta) = 1$$

OR

$$\Rightarrow \boxed{P(y=0 \mid x,\theta) = 1 - P(y=1 \mid x;\theta)}$$

Want $0 \leq h_\theta(x) \leq 1$

We can make

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

to

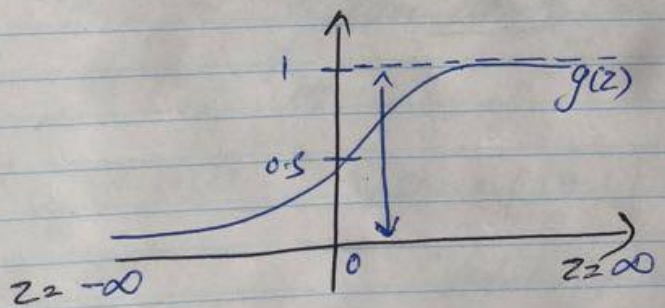
Hypothesis

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

Sigmoid function

$$g(z) = \frac{1}{1+e^{-z}}$$

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

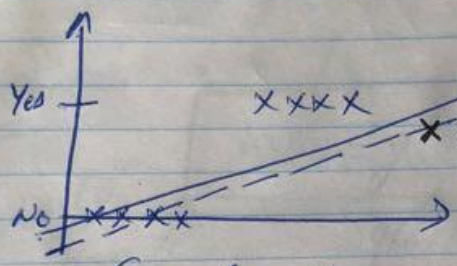


So, when z reaches $+ve/-ve\ \infty$, $g(z)$ reaches 1/0

Hence $h_\theta(x) = g(z)$ will always be less than 1 & more than 0.

LOGISTIC REGRESSION
(SIGMOID FUNCTION)

Example1



if an example was here, it caused linear regression to change its line, giving us worse hypotheses.!!

Threshold classifier output $h_\theta(x)$ at 0.5:

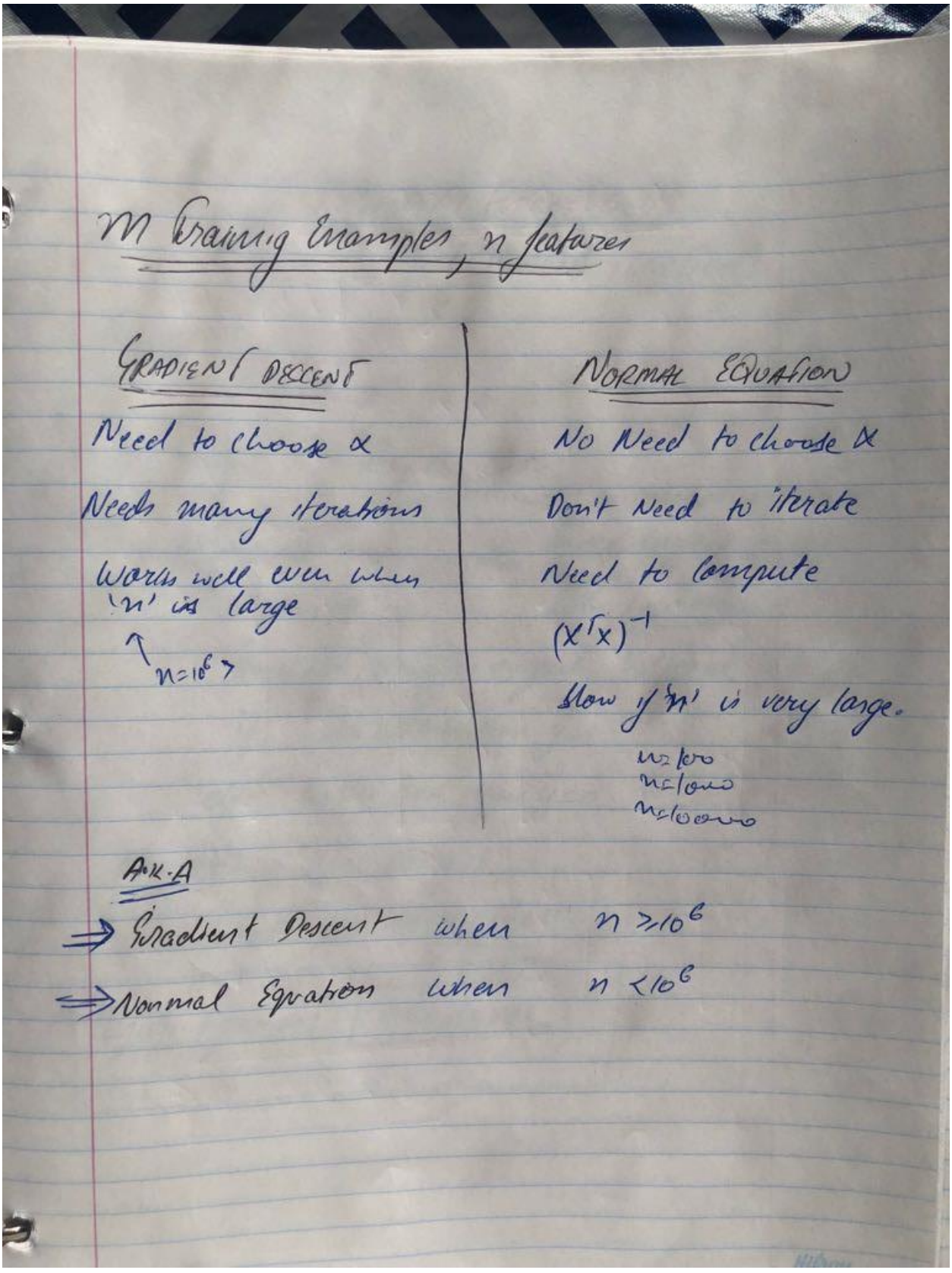
if, $h_\theta(x) \geq 0.5$ then $y=1$

else, $h_\theta(x) \leq 0.5$ then $y=0$

Example2 $y=0$ or $y=1$

$h_\theta(x)$ can be >1 or <0!!

Logistic Regression: $0 \leq h_\theta(x) \leq 1$



m Training Examples, n features

GRADIENT DESCENT	NORMAL EQUATION
Need to choose α	No Need to choose α
Needs many iterations	Don't Need to iterate
Works well even when 'n' is large ($n=10^6$)	Need to compute $(X^T X)^{-1}$
	Slow if 'n' is very large. ($n=100$, $n=1000$, $n=10000$)

A.K.A

⇒ Gradient Descent when $n \geq 10^6$

⇒ Normal Equation when $n < 10^6$

NORMAL EQUATION

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

Set $A = X^T X$

$$(X^T X)^{-1} = A^{-1}$$

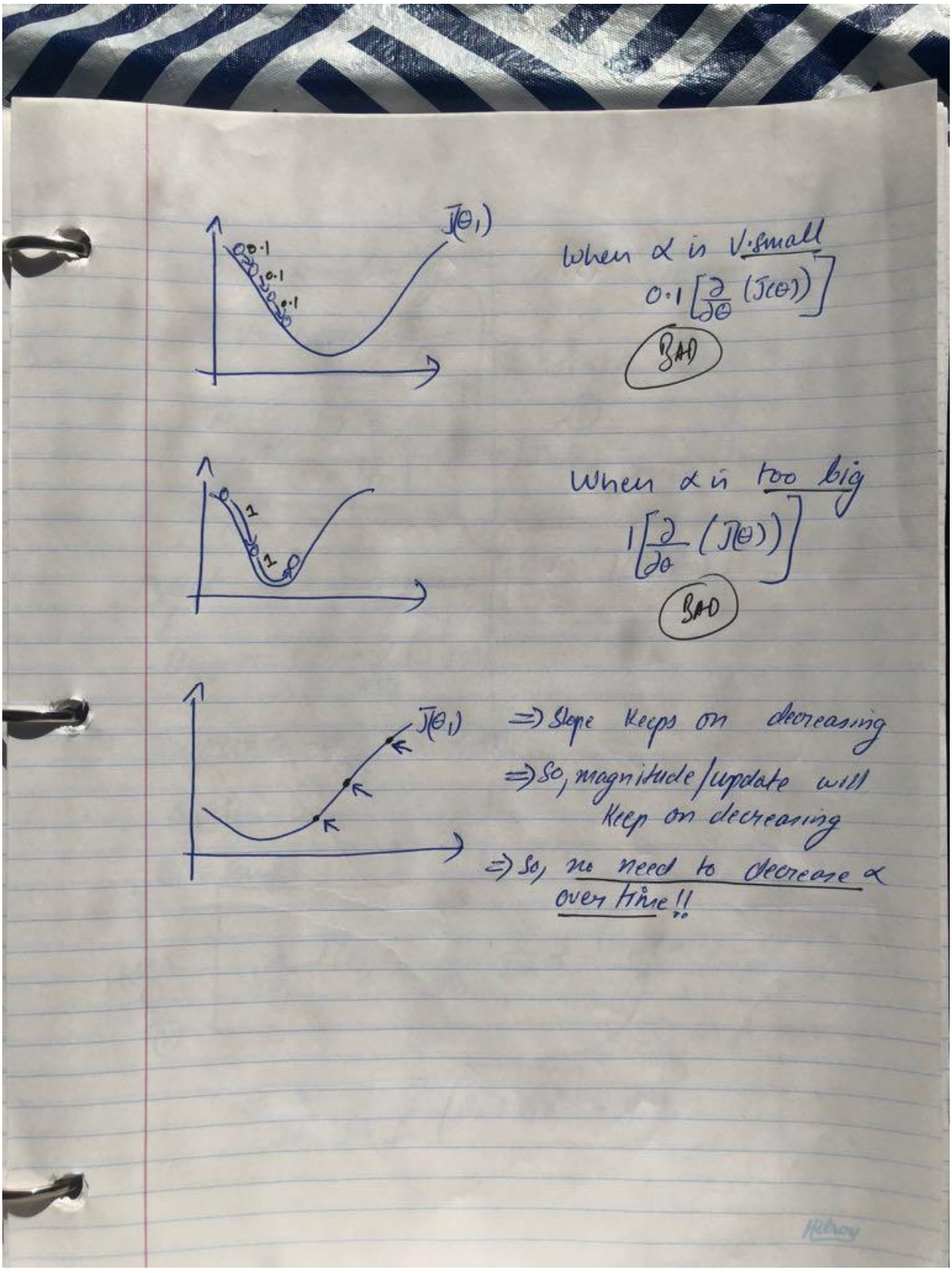
m examples $(x^{(1)}, y^{(1)}), \dots, (x^{(m)}, y^{(m)})$; n features

$$x^{(i)} = \begin{bmatrix} x_0^{(i)} \\ x_1^{(i)} \\ x_2^{(i)} \\ \vdots \\ x_n^{(i)} \end{bmatrix} \in \mathbb{R}^{n+1} \qquad X = \begin{bmatrix} — (x^{(1)})^T — \\ — (x^{(2)})^T — \\ \vdots \\ — (x^{(m)})^T — \end{bmatrix}$$

(design Matrix)

Eg $x^{(i)} = \begin{bmatrix} 1 \\ x_1^{(i)} \end{bmatrix}$ $\qquad X = \begin{bmatrix} 1 & x_1^{(1)} \\ 1 & x_2^{(1)} \\ 1 & x_m^{(1)} \end{bmatrix}$ $m \times 2$ $\qquad y = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(m)} \end{bmatrix}$

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


J(θ1)
0.1
0.1
0.1
When α is V.small
0.1 [∂/∂θ (J(θ))]
BAD
When α is too big
1 [∂/∂θ (J(θ))]
BAD
J(θ1)
⇒ Slope keeps on decreasing
⇒ So, magnitude/update will keep on decreasing
⇒ So, no need to decrease α over time!!

Correct Implementation

{
$temp0 := \theta_0 - \alpha \frac{\partial}{\partial \theta_0} (J(\theta_0, \theta_1))$

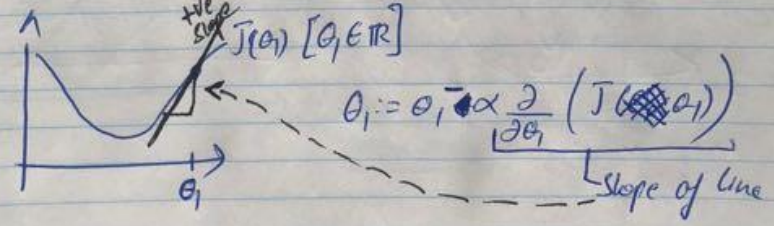
$temp1 := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_0, \theta_1)$

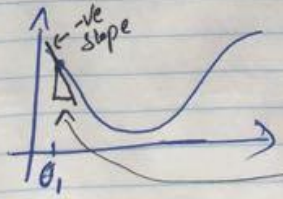
$\theta_0 := temp0$
$\theta_1 := temp1$
] → update only after computing $J(\theta_0, \theta_1)$

} Repeat Until Convergence

Simultaneous Update
of
Gradient Descent

$\min_{\theta_1} J(\theta_1)$ $\theta_1 \in \mathbb{R}$ [all real numbers]





$\frac{\partial}{\partial \theta_1} J(\theta_1) < 0$

$\theta_1 := \theta_1 - \alpha [-ve]$

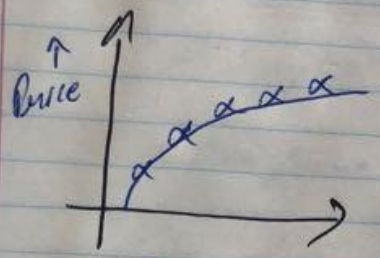
$+$

inc θ

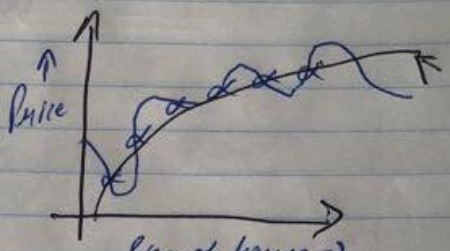
BAD decision.

REGULARISATION

Intuition



$\theta_0 + \theta_1 x + \theta_2 x^2$



$\theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$

if they $\theta_{3,4} \approx 0$

Suppose we penalise & make θ_3, θ_4 really small

$$\rightarrow \min_{\theta} \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)}\right)^2$$ [SQUARED ERROR COST FUNCTION]

$+1000\theta_3^2 + 1000\theta_4^2$

To make it small, θ_3 & θ_4 has to be small!

So, as we minimising will end up, $\theta_3 \approx 0$, $\theta_4 \approx 0$

A quadratic function that is good!
By adding the 2 new terms!!

Addressing Overfitting

Reduce Number of features

- manually
- model selection algorithms!

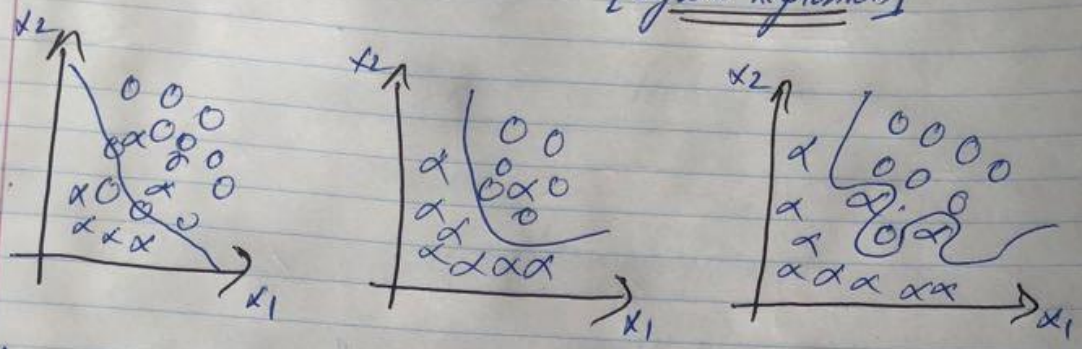
What if we throw useful feature away?

REGULARIZATION

- Keep all feature, but reduce magnitude/ values of parameter θ_j
- Works well when we have alot of features each of which contributes a bit to predicting y.

OVERFITTING

[Logistic Regression]



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

g = Sigmoid function

"underfit"

$g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_2^2 + \theta_5 x_1 x_2)$

"Just Right"

$g(\theta + \theta_1 x_1 + \cdots)$

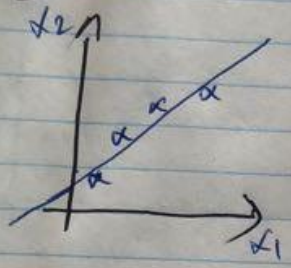
"overfit"

Same reasoning as linear regression!

OVERFITTING

[Linear regression]

Example

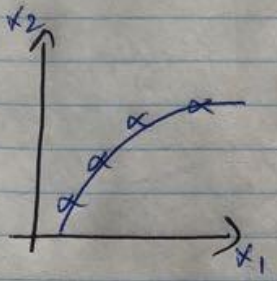


$\theta_0 + \theta_1 x$

"Underfit"

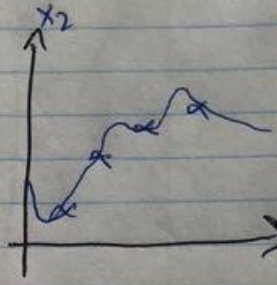
"High bias"

Poor fit



$\theta_0 + \theta_1 x + \theta_2 x^2$

"Just Right"



$\theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$

"overfitting"

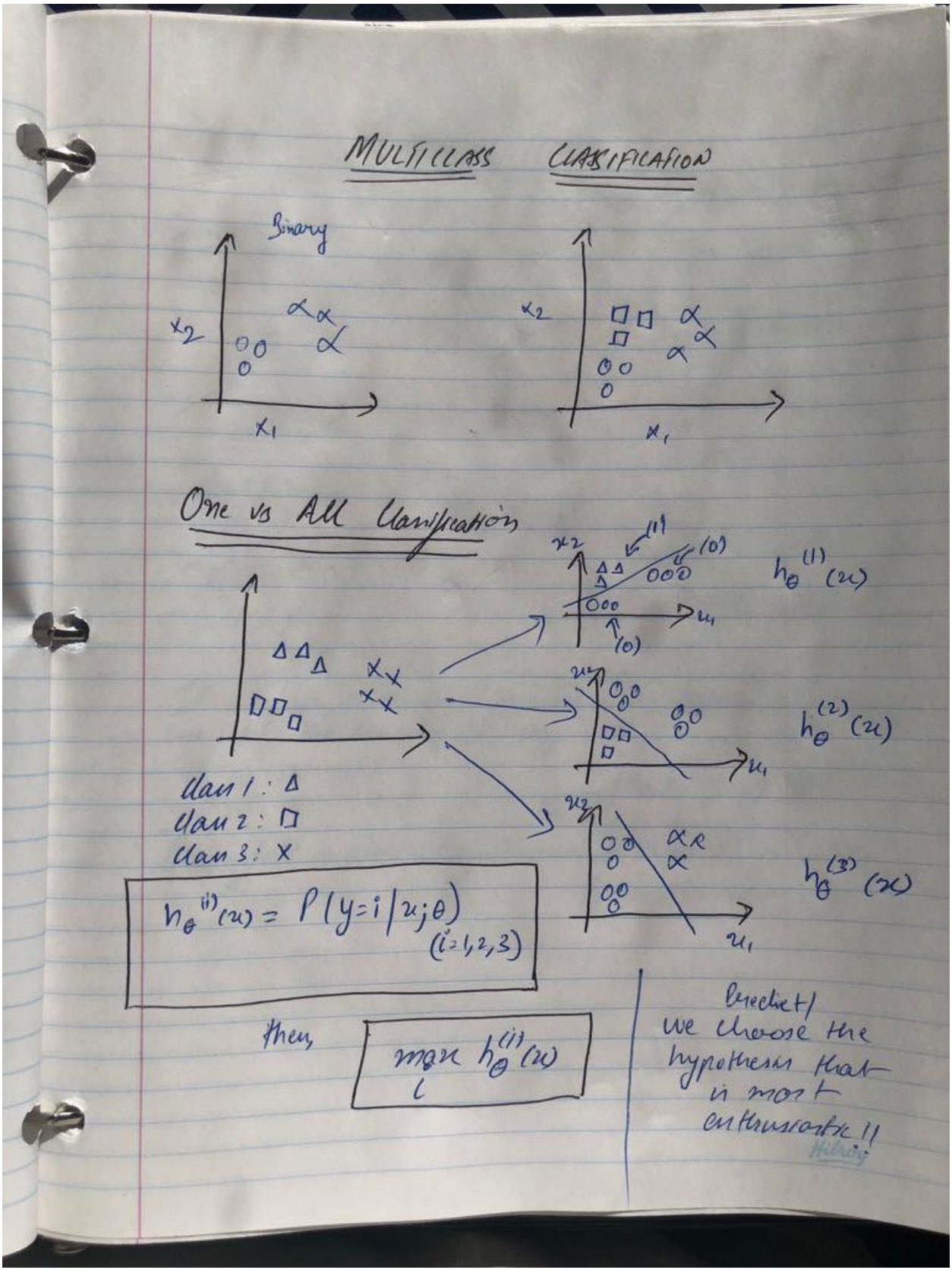
"High variance"

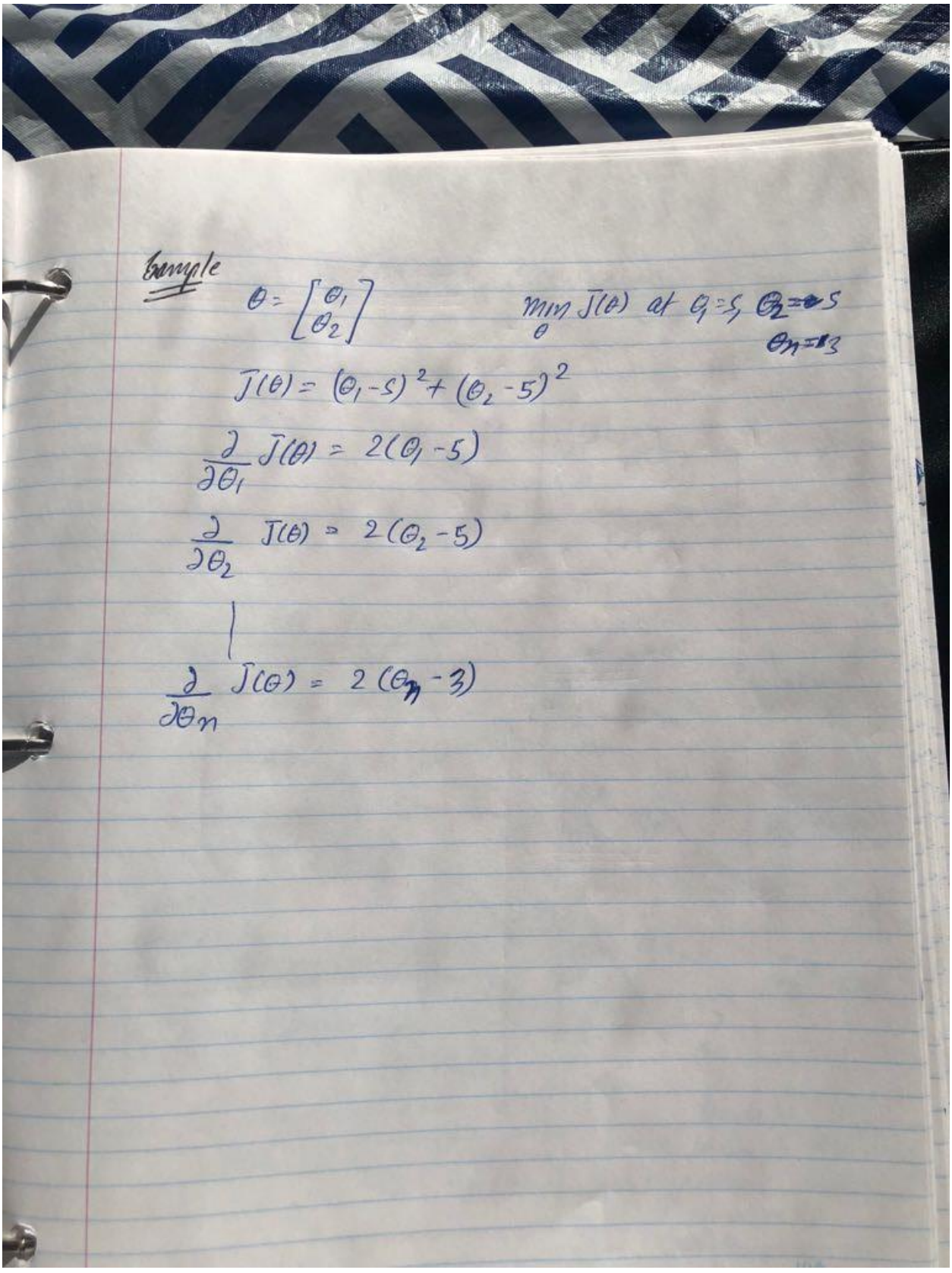
If we have too many features, the learned hypothesis may fit the training set very well

$$J(\theta) = \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)} \right)^2 \approx 0,$$

But, fail to generalise to new examples!

(predict prices on new examples)


MULTICLASS CLASSIFICATION
Binary
One vs All Classification
Class 1: Δ
Class 2: □
Class 3: X
$h_\theta^{(i)}(x) = P(y=i|x;\theta)$ $(i=1,2,3)$
then $\max_i h_\theta^{(i)}(x)$
Predict! We choose the hypothesis that is most confidence!!



Example

$\theta = \begin{bmatrix} \theta_1 \\ \theta_2 \end{bmatrix}$ $\min_{\theta} J(\theta)$ at $\theta_1 = 5$, $\theta_2 = 5$ $\theta_n = 3$

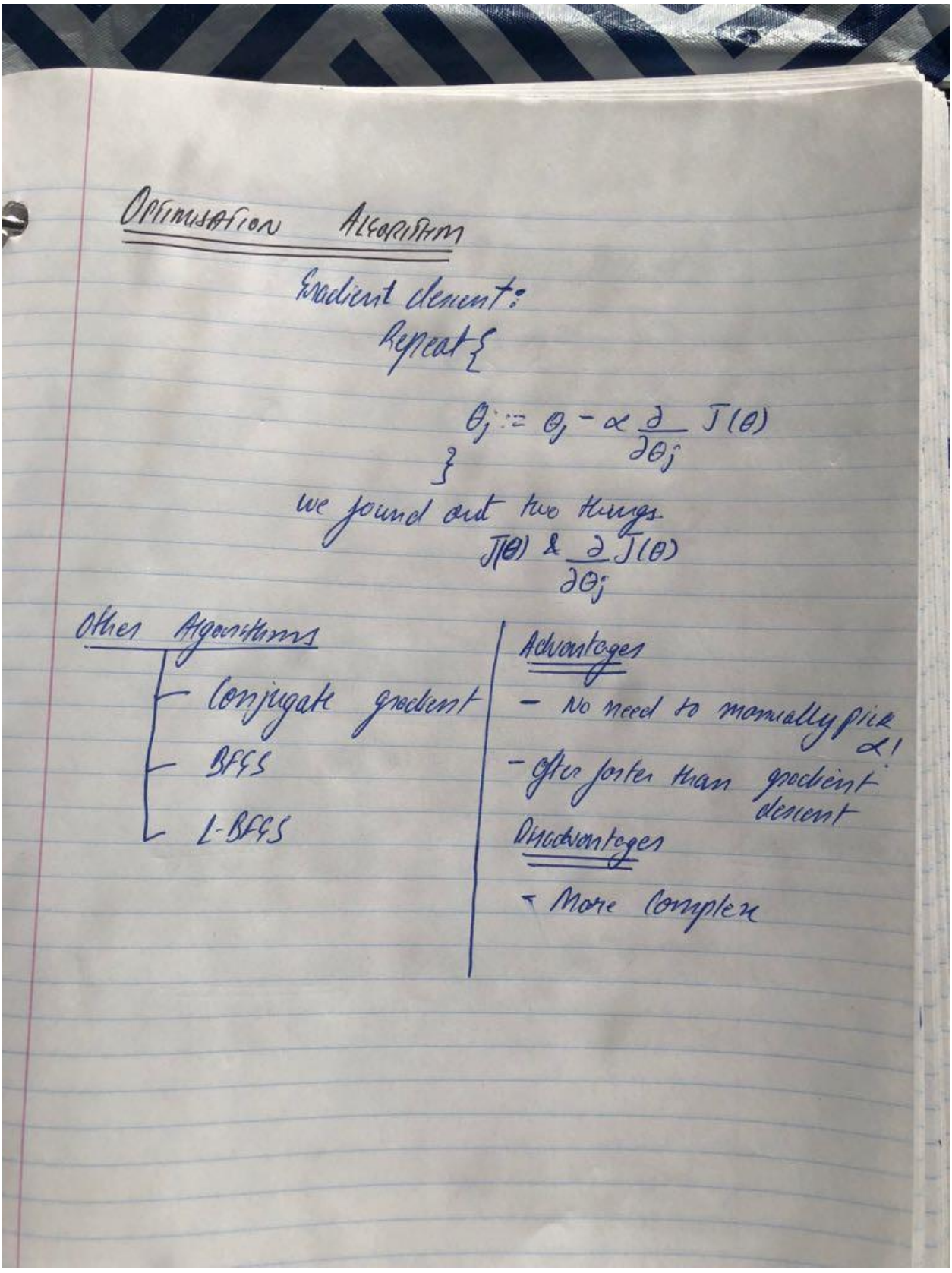
$J(\theta) = (\theta_1 - 5)^2 + (\theta_2 - 5)^2$

$\frac{\partial}{\partial \theta_1} J(\theta) = 2(\theta_1 - 5)$

$\frac{\partial}{\partial \theta_2} J(\theta) = 2(\theta_2 - 5)$

|

$\frac{\partial}{\partial \theta_n} J(\theta) = 2(\theta_n - 3)$



Optimisation Algorithm

Gradient descent:

Repeat {

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$$

}

we found out two things.

$J(\theta)$ & $\frac{\partial}{\partial \theta_j} J(\theta)$

Other Algorithms

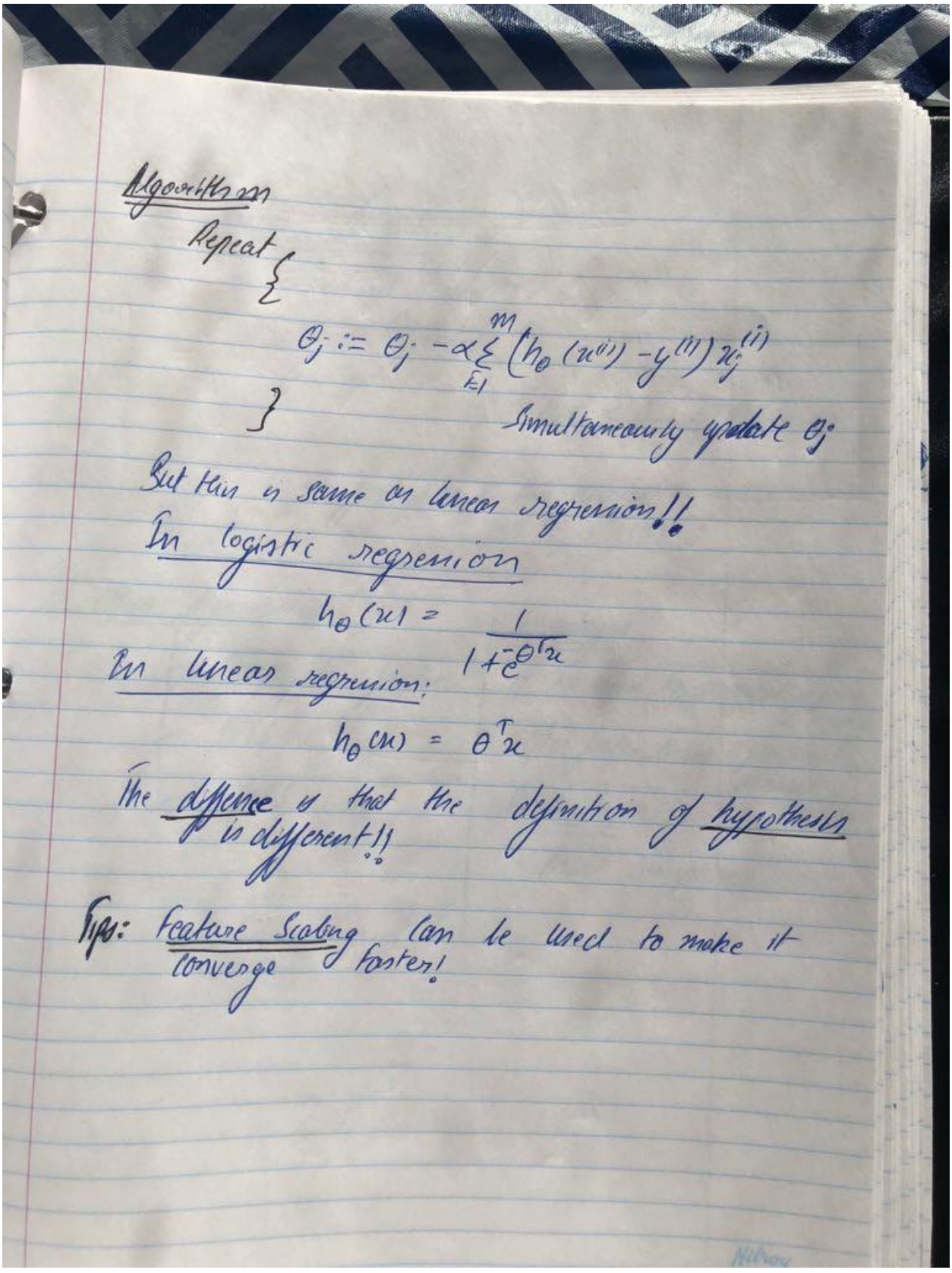
- Conjugate gradient
- BFGS
- L-BFGS

Advantages

- No need to manually pick α!
- Often faster than gradient descent

Disadvantages

- More complex



Algorithm

Repeat {

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

} Simultaneously update θ_j

But this is same as linear regression!!

In logistic regression

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

In linear regression:

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

The difference is that the definition of hypothesis is different!!

Tips: Feature Scaling can be used to make it converge faster!

Cost function of logistic Regression

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

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

To get parameters θ:

$$\min_\theta J(\theta)$$ → How? Gradient Descent!!

Prediction:

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

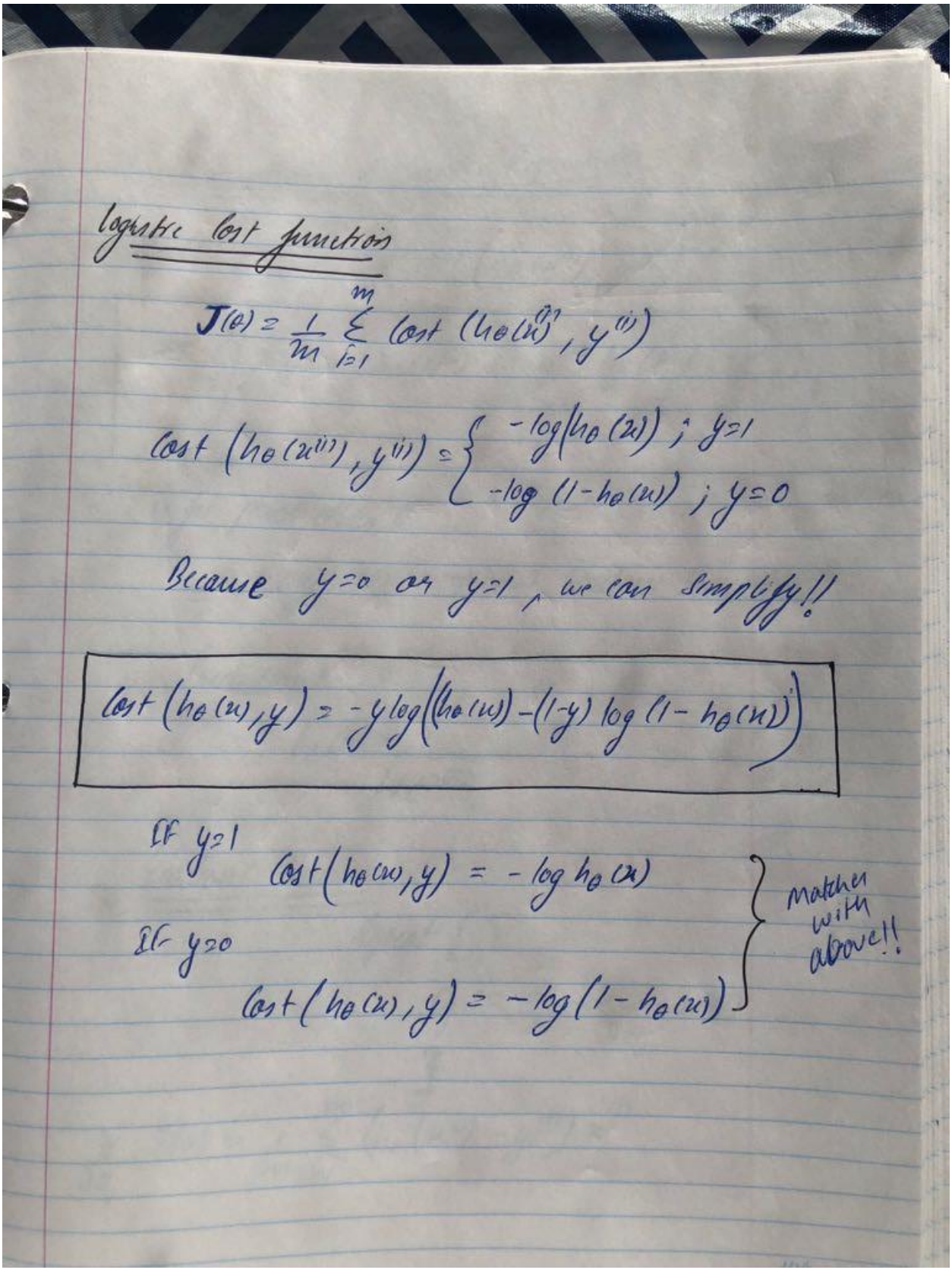
Gradient Descent

Repeat {

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$$

}

$$\frac{\partial}{\partial \theta_j} J(\theta) = \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)}\right) x_j^{(i)}$$



Logistic cost function

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

$$\text{cost}\left(h_\theta(x^{(i)}), y^{(i)}\right) = \begin{cases} -\log(h_\theta(x)) & ; y=1 \\ -\log(1-h_\theta(x)) & ; y=0 \end{cases}$$

Because $y=0$ or $y=1$, we can simplify!!

$$\text{cost}(h_\theta(x), y) = -y \log\left((h_\theta(x)) - (1-y) \log(1-h_\theta(x))\right)$$

If $y=1$

$$\text{cost}(h_\theta(x), y) = -\log h_\theta(x)$$

If $y=0$

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

} matches with above!!

So, Anyways.

Discrete Random Variable

- A discrete variable, that can have / be present as a possible value in a instance of a random number!!

Continous Random Variable

- Random Variable, in a continous set of numbers can have multiple possible values.

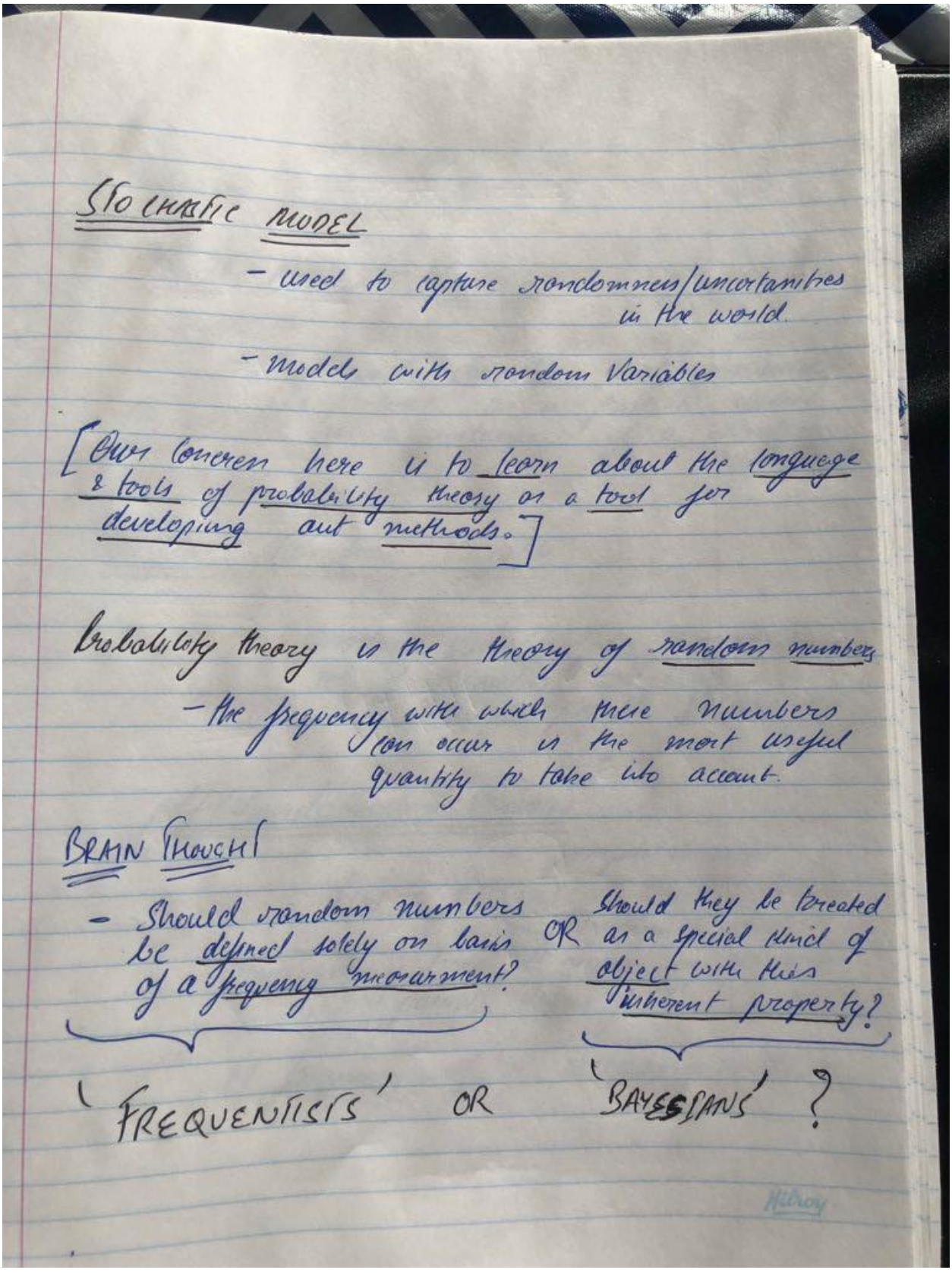
pdf (discrete Random Variable)

$$P(x) = P(X = x)$$

(x: regular variable; X: random variable)

How many time 'x' occurs in all the possible values of 'X'.

i.e. frequency of 'x' occuring in 'X'.



STOCHASTIC MODEL

- used to capture randomness/uncertainties in the world.
- models with random variables

[Our concern here is to learn about the language & tools of probability theory as a tool for developing our methods.]

Probability theory is the theory of random numbers

- the frequency with which these numbers can occur is the most useful quantity to take into account.

BRAIN STORM!

- Should random numbers be defined solely on basis of a frequency measurement? OR Should they be treated as a special kind of object with this inherent property?

'FREQUENTISTS' OR 'BAYESIANS'?

PROBABILITY THEORY

Random Variable?

- Something that a different value (x) each instant (X)

Discrete Variable

- Same value (a) all the time.

Probability Density Function

The complete knowledge of a random variable

Example: ~~XXXX~~

Let, $X \in x_i$ $\left[\begin{matrix} x = \{0,1,2,\dots\} \\ x_i \in \mathbb{R} \end{matrix}\right]$

then,

$pdf(x_i) = ?$ $\boxed{\frac{x_i}{X}}$

i.e. how likely each value is for random variable x.

Regularised logistic Regression

$$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_1^2 + \theta_3 x_1^2 x_2 + \theta_4 x_1^2 x_2^2 + \theta_5 x_1^2 x_2^3 + \dots)$$

Cost function

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

$\theta_1, \theta_2 \dots \theta_n$ don't become large

Gradient Descent

Repeat {

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

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

$j = 0, 1, 2, 3, \dots, n$

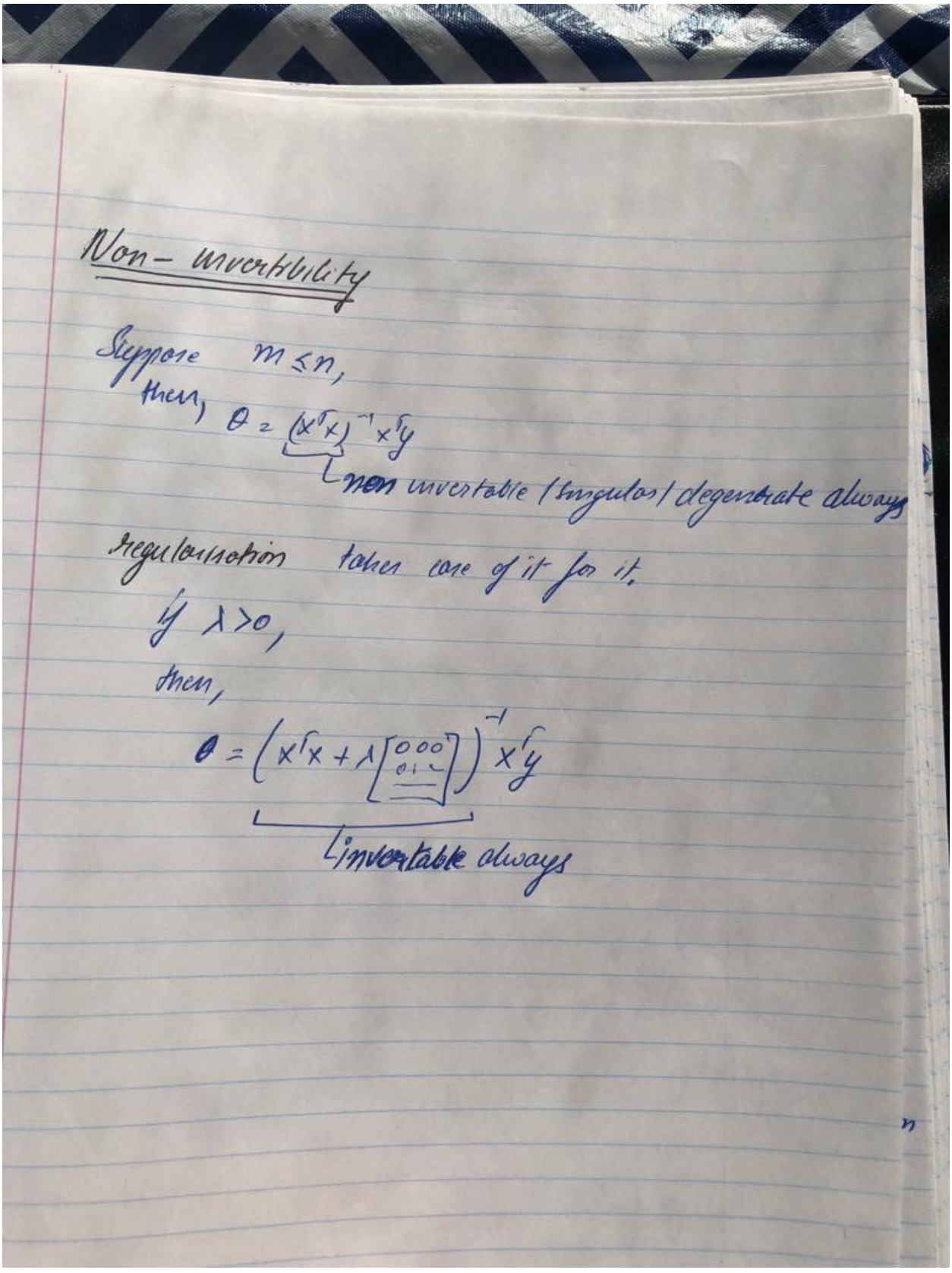
$\theta_1, \theta_2, \dots, \theta_n$

}

updated equation

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

Hidroy



Non-invertibility

Suppose $m \leq n$,

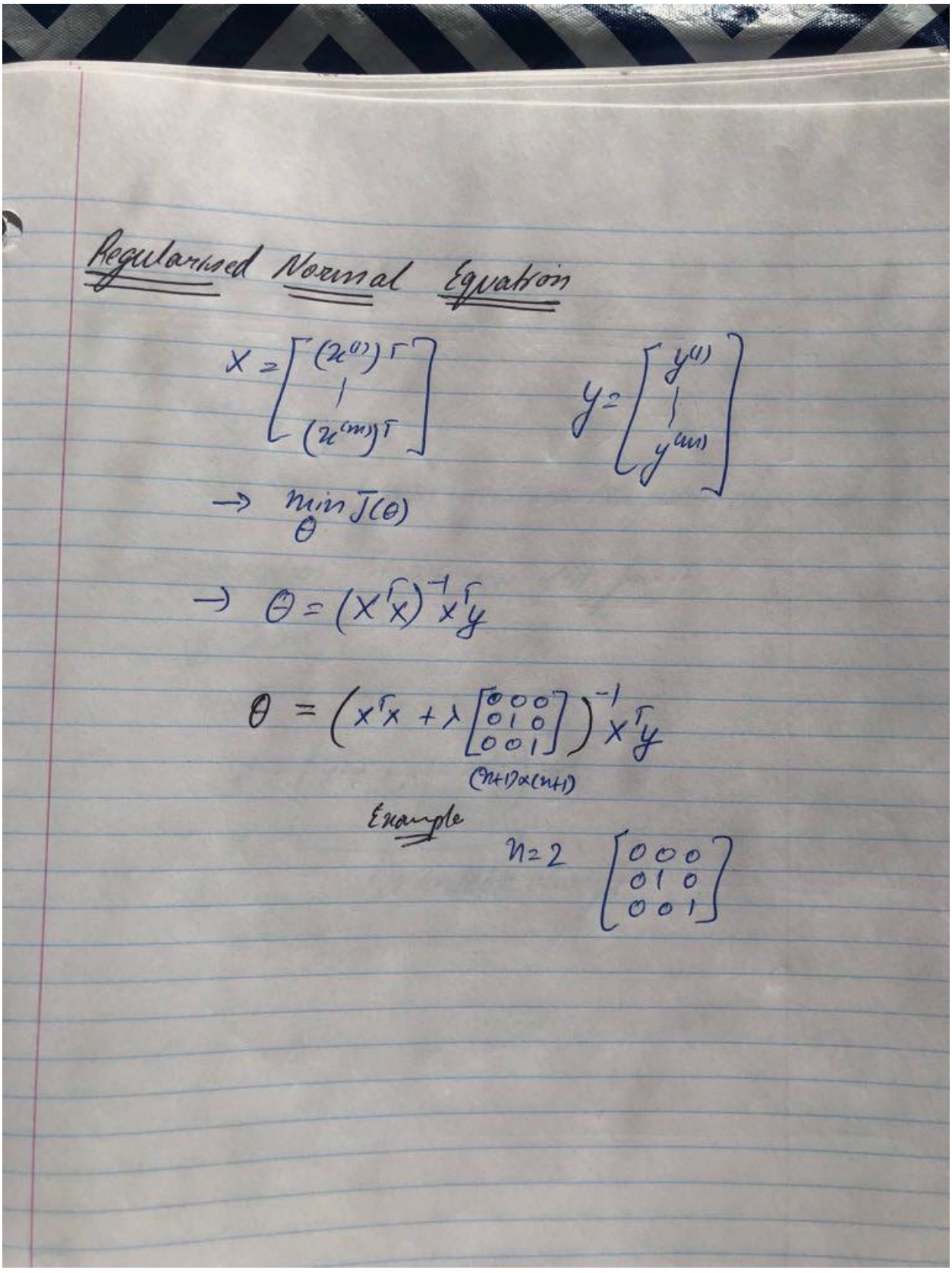
then, $\theta = \underbrace{(X^T X)^{-1}}_{\text{non invertable / singular / degenerate always}} X^T y$

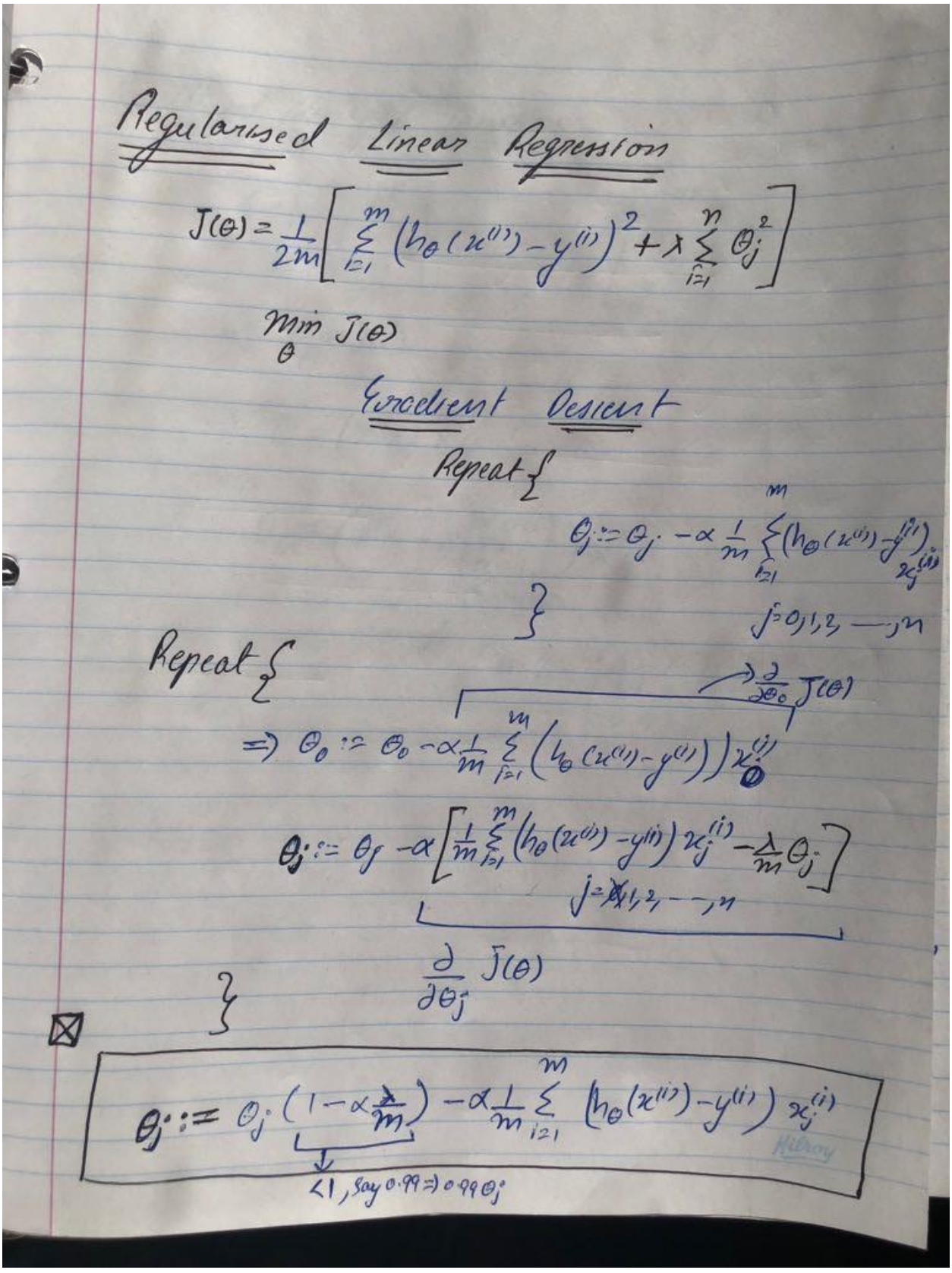
Regularisation takes care of it for it.

if $\lambda > 0$,

then,

$$\theta = \underbrace{\left(X^T X + \lambda \begin{bmatrix} 0 & & \\ & 0 & \\ & & 1 \end{bmatrix}\right)^{-1}}_{\text{invertable always}} X^T y$$


Regularised Normal Equation
$X = \begin{bmatrix} (x^{(1)})^T \\ | \\ (x^{(m)})^T \end{bmatrix}$
$y = \begin{bmatrix} y^{(1)} \\ | \\ y^{(m)} \end{bmatrix}$
$\rightarrow \min_\theta J(\theta)$
$\rightarrow \theta = (X^T X)^{-1} X^T y$
$\theta = \left(X^T X + \lambda \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}\right)^{-1} X^T y$
$(n+1) \times (n+1)$
Example
$n = 2$
$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$


Regularised Linear Regression
$J(\theta) = \frac{1}{2m}\left[\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2 + \lambda\sum_{j=1}^{n}\theta_j^2\right]$
$\min_\theta J(\theta)$
Gradient Descent
Repeat {
$\theta_j := \theta_j - \alpha\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)}$
} $j = 0, 1, 2, \ldots, n$
Repeat {
$\frac{\partial}{\partial\theta_0} J(\theta)$
$\Rightarrow \theta_0 := \theta_0 - \alpha\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_0^{(i)}$
$\theta_j := \theta_j - \alpha\left[\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)} - \frac{\lambda}{m}\theta_j\right]$
$j = 1, 2, \ldots, n$
$\frac{\partial}{\partial\theta_j} J(\theta)$
}
$\theta_j := \theta_j\left(1 - \alpha\frac{\lambda}{m}\right) - \alpha\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)}$
< 1, say $0.99 \Rightarrow 0.99\theta_j$

If (λ) is very large? (too large for our problems)
Say $\lambda = 10^{10}$

We would end up penalising $\theta_1, \theta_2, \theta_3, \theta_4$ heavily
then they will be all close to zero
which means we are getting rid of them
which then given a straight line / it
fails to fit the data, because it has
a too high bias!

So, some care should be taken to choose (λ)

Regularised cost function

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

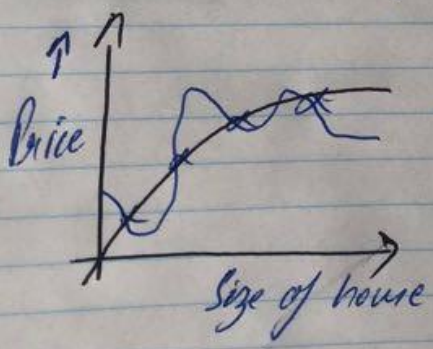
Goal 2

regularisation parameter

Regularisation parameter (λ)

Controls the trade-off b/w two different goals.

- Goal 1: Fit the training set well
- Goal 2: Keep parameter small, to get simple hypothesis, to prevent overfitting



Regularisation in brief

Small values for parameter $\theta_0, \theta_1, \theta_2, \ldots \theta_n$

- "Simpler" hypothesis
- less prone to overfitting!

We basically made the parameters $\theta_0 \ldots \theta_n$ small that their effect on the overall cost function is also very small.

Which then helps it generalise better.

Example

Housing:

- Features: $x_1, x_2, x_3, \ldots, x_{100}$
- Parameters: $\theta_0, \theta_1, \theta_2, \theta_3 \ldots, \theta_{100}$

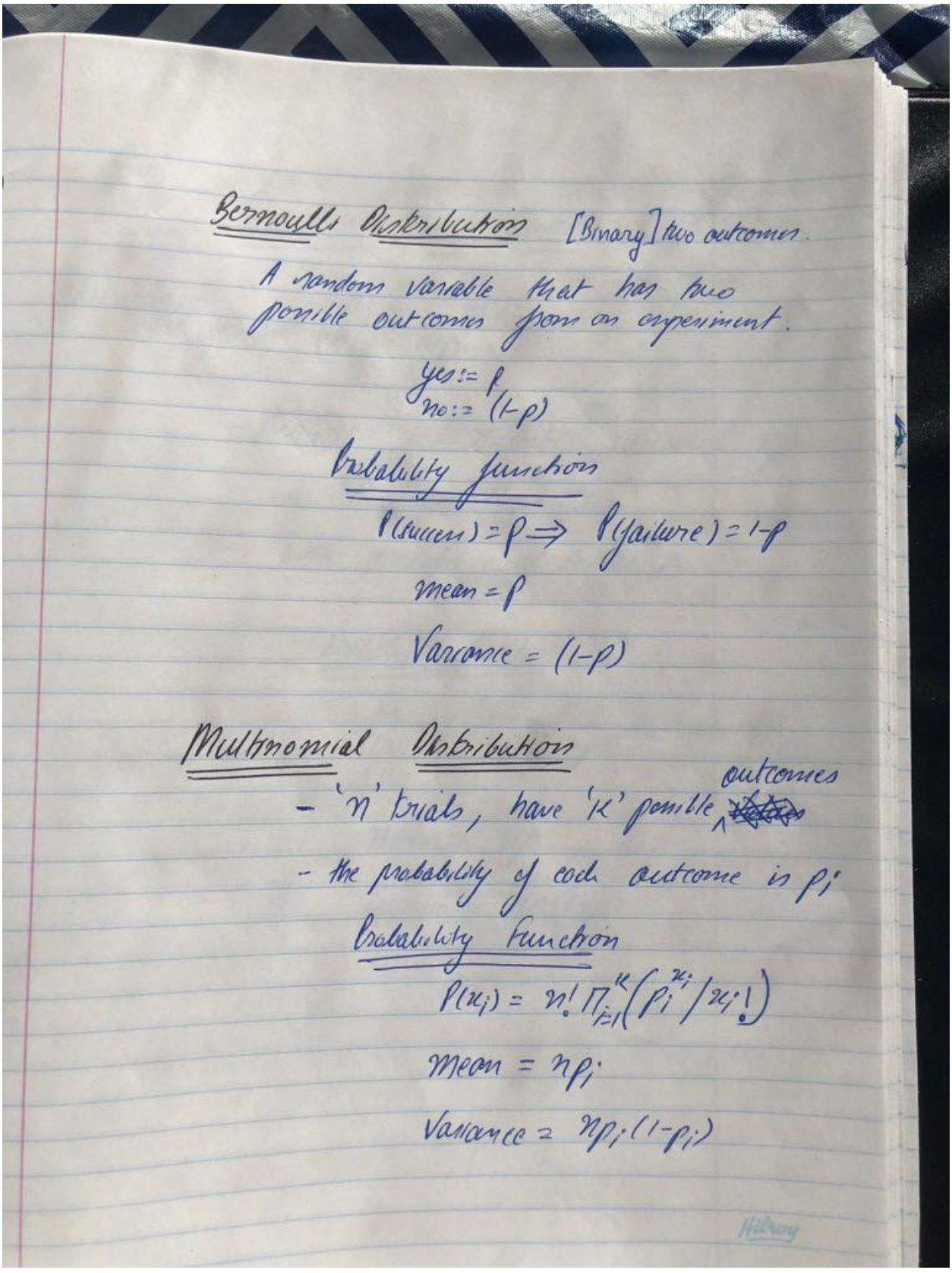
Which parameter less relevant?

We don't know!!

So we use regularisation.

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

$\theta_0, \theta_1, \theta_2, \theta_3 \ldots, \theta_{100}$ don't take θ_0



Bernoulli Distribution [Binary] two outcomes.

A random variable that has two possible outcomes from an experiment.

yes := p
no := (1-p)

Probability function

P(success) = p ⇒ P(failure) = 1-p

mean = p

Variance = (1-p)

Multinomial Distribution

- 'n' trials, have 'k' possible outcomes
- the probability of each outcome is p_i

Probability function

$$P(x_i) = n! \prod_{i=1}^{k} \left(p_i^{x_i} / x_i! \right)$$

mean = np_i

Variance = $np_i(1-p_i)$

What if we don't know the distribution function?
- Estimate them from data!

Sample mean [Unbiased]

$$\bar{x} = \frac{1}{n} \sum_{i=1}^{n} x_i$$

Sample Variance [Biased]

$$s_1^2 = \frac{1}{n} \sum_{i=1}^{n} (\bar{x} - x_i)^2$$

These are the appropriate maximum likelihood estimates of these parameters.

Unbiased Sample Variance

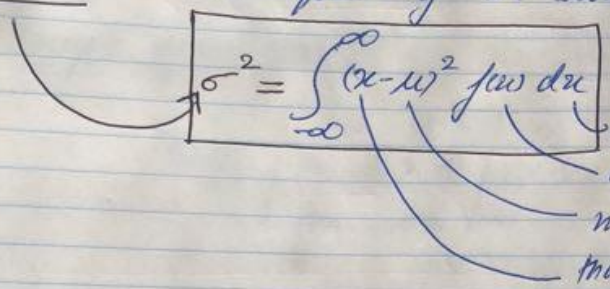
$$s_2^2 = \frac{1}{n-1} \sum_{i=1}^{n} (\bar{x} - x_i)^2$$

useful for gaussian distributed data as all higher moment than variance is zero.

There are infinite pdfs available

a.k.a. Standard deviation (std).

Variance is the square of the above std!



Variance can also be called the second moment & mean the first moment.

⇒ We thus need more than two moments to characterise a probability function uniquely.

So let's say, we are trying to find the (n^{th}) moment out of all moments.

$$m^n = \int_{-\infty}^{\infty} (x-\mu)^n f(x) dx$$

The Variance is the second moment about the mean.

Higher moments define more characteristics like third moment relate to quantity called Skewness.

& fourth is Kurtosis.

Example (mean isn't the only interesting thing)

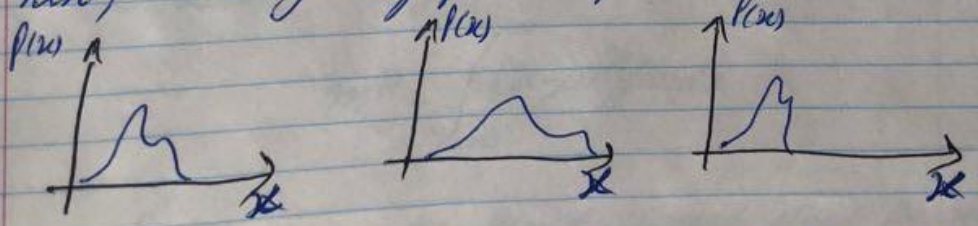
Ask what median is?

Median is some value for which it is equally likely to find a value lower or higher than itself!!

$$\int_{-\infty}^{\text{median}(x)} p(x)\,dx = \int_{\text{median}(x)}^{-\infty} p(x)\,dx$$

Thanks to maths mean = median in a symmetric distribution!

Also, how big the graph is (Spread)



gives us an intuition of how far the values are.

But what is $p(x)$?

- we usually don't know.,

If we knew what $p(x)$ is, then we could stop Right here but we don't know what it is!!

That is why we try to approximate the value of 'μ'.

Example:

$$\hat{\mu} = \frac{1}{N} \sum_{i=1}^{N} x_i p_i$$

- Estimate of the mean!
- probability of the value
- Value
- Sum of all such values
- make it easy mathematically

BRAIN THOUGHT

The mean of a distribution is not the only interesting quantity that characterizes a distribution.

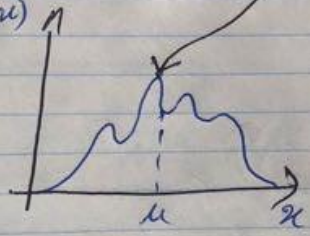
But, think about it!

it would be difficult to store that many values

So logically we try to take the probability that represents the function the most

i.e. we take the most frequent random variable / value.

or graphically. Peak! it is the largest peak value of the distribution.



$$\Rightarrow \boxed{p^{max} = \text{argmax}_x\, p(x)}$$

More commonly, we ask the average value which is found by calculating the mean!

$$\Rightarrow \boxed{\mu = \int_{-\infty}^{\infty} x p(x) dx}$$

The calculation of adding all the numbers together with their frequency

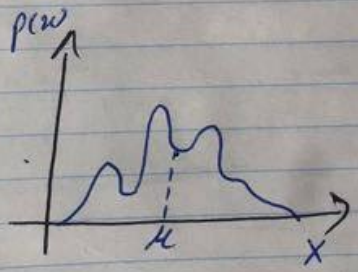
STATIC PROBABILITY DENSITY FUNCTION

- only consider independent random values

- State pdf() describes all that we know about the corresponding random variable.

Example

A pdf : p(x) has values.

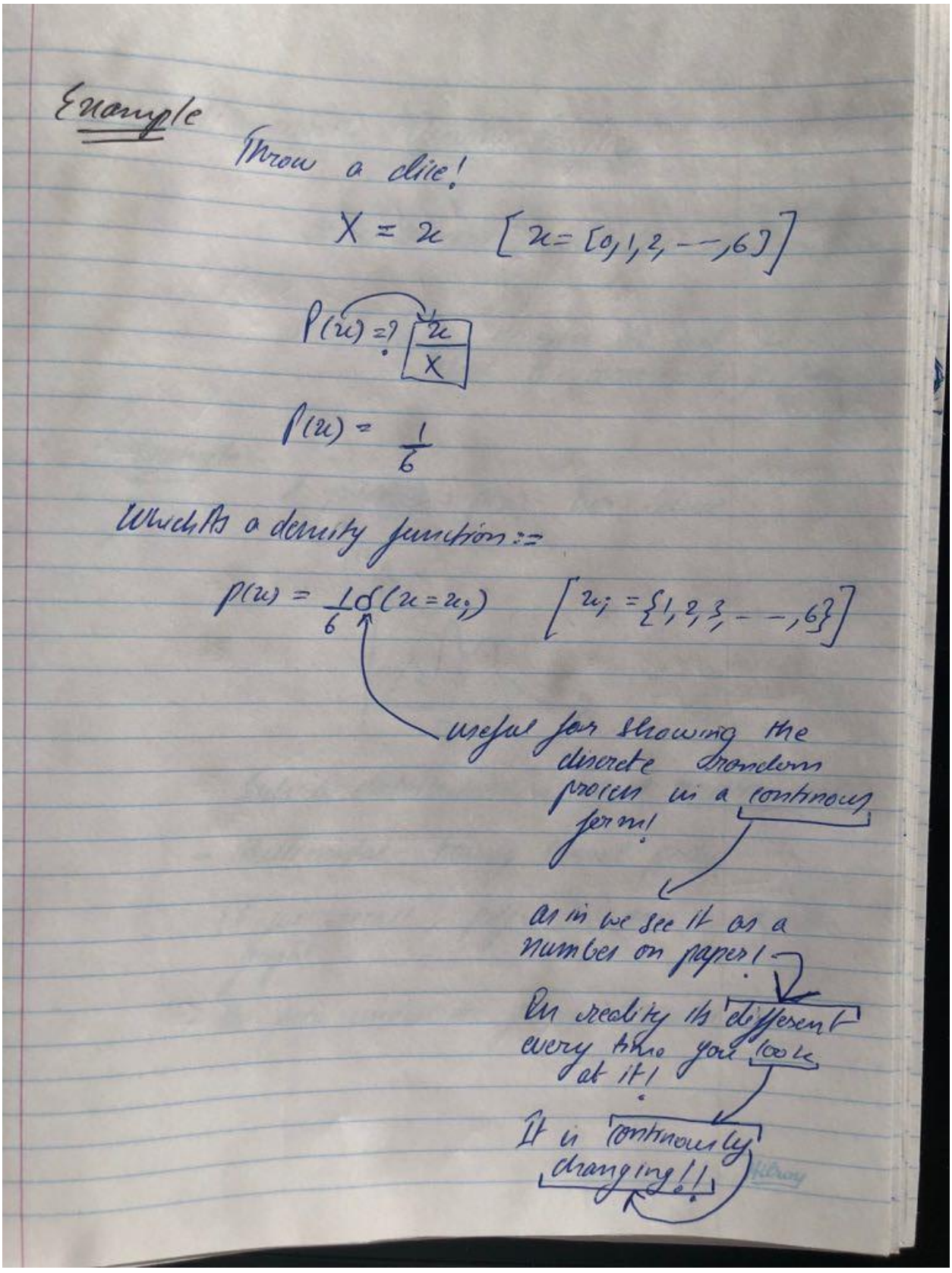


Such a distribution is called multimodal!

- Multimodal - having several peaks!

If we recall, pdf() is area under the graph!

- the area under the graph must be 1!


Example
Throw a dice!
X = x [x = [0, 1, 2, --, 6]]
P(x) = ? x / X
P(x) = 1/6
Which is a density function :=
P(x) = 1/6 δ(x = x_i) [x_i = {1, 2, 3, - -, 6}]
useful for showing the discrete random process in a continuous form!
as in we see it as a number on paper!
In reality it's different every time you look at it!
It is continuously changing!!

We don't want to take infinite considerations
So, we limit it!

$$P(a < x < b) = \int_{x=a}^{b} p(x)\,dx$$

Thankfully, we have a mathematical notation!

DELTA-FUNCTION: δ-function: write discrete random process in a continuous form!

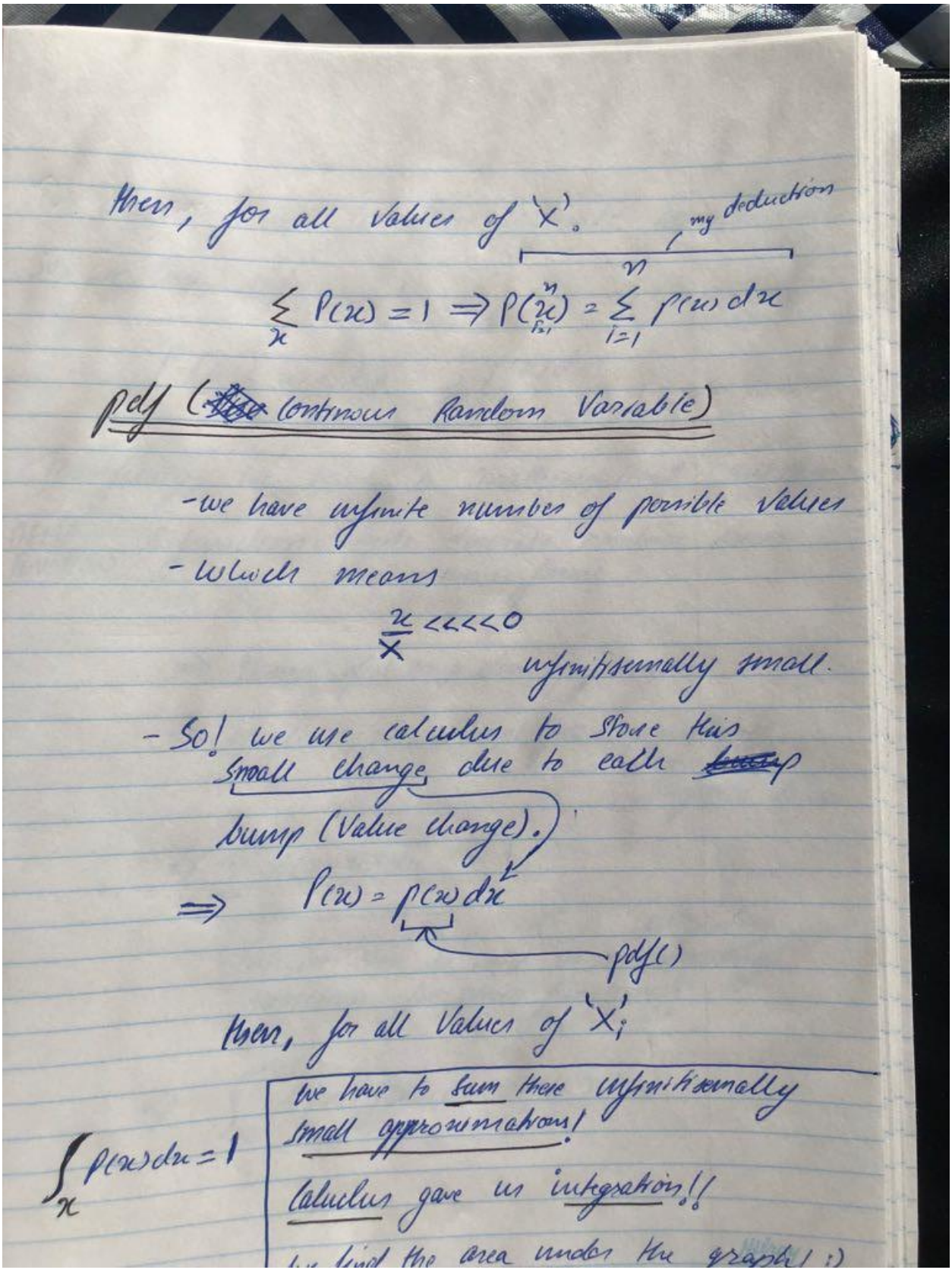
$\Rightarrow$ Think of it as a density function.

$\Rightarrow$ Which is zero except for its arguments. (infinite values)

$$\Rightarrow \int_{-\infty}^{\infty} \delta(x = x_1)\,dx = 1$$

This can then be used as an integrating function for other functions!!

$$\Rightarrow \int_{-\infty}^{\infty} \delta(x_1)\, f(x)\,dx = f(x_1)$$



Then, for all values of 'x'. (my deduction)

$$\sum_{x} P(x) = 1 \Rightarrow P(\overset{n}{x}) = \sum_{i=1}^{n} p(x)\,dx$$

pdf (continuous Random Variable)

- we have infinite number of possible values
- which means

$$\frac{x}{X} << << 0$$

infinitesimally small.

- So! we use calculus to store this small change due to each bump (Value change).

$$\Rightarrow P(x) = p(x)\,dx$$

pdf()

Then, for all values of 'X';

$$\int_{x} p(x)\,dx = 1$$

we have to sum these infinitesimally small approximations!

calculus gave us integration!!

to find the area under the graph! :)

use log to reduce computation!

$$l(w) = \log L(w) = \sum_{i}^{m} \log\left(p(w; y^{(i)}, x^{(i)})\right)$$

Since, log function increases monotonically,

$$\max(L) = \max(l)$$

Thus, maximum (log-) likelihood

can thus be := $\boxed{w^{MLE} = \arg\max_w l(w)}$

Gradient Descent

$$l(w, \sigma) = \log \prod_{i=1}^{m} \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(y^{(i)} - w^T x^{(i)})^2}{2\sigma^2}\right)$$

$$= \sum_{i=1}^{m} \left(\log \frac{1}{\sqrt{2\pi}\sigma} - \frac{(y^{(i)} - w^T x^{(i)})^2}{2\sigma}\right)$$

OR

$$l(w, \sigma) = -\frac{m}{2} \log 2\pi\sigma - \sum_{i=1}^{m} \frac{(y^{(i)} - w^T x^{(i)})^2}{2\sigma^2}$$

maximise this

But, one data point isn't enough to make good predictions! we need more.

let's say, we have two data points.

$$(x^{(1)}, y^{(1)}) \& (x^{(2)}, y^{(2)})$$

Well, now we need the joint distribution function.
But let's just assume that each feature is independent of each other.

$$P(Y_1 = y^{(1)}, Y_2 = y^{(2)} | x^{(1)}, x^{(2)}; w_0, w_1, \sigma) =$$

$$= P(Y_1 = y^{(1)} | x^{(1)}, x^{(2)}; w_0, w_1, \sigma) P(Y_2 = y^{(2)} | x^{(1)}, x^{(2)}; w_0, w_1, \sigma)$$

If we have 'm' samples then

$$P(Y_1 = y^{(1)}, Y_2 = y^{(2)}, \dots, Y_m = y^{(m)} | x_1, x_2, \dots, x_m; w) = \boxed{\prod_i^m p(y_i | x_i; w)}$$

We can Now Add Specific observations (training data) into resulting function.

Likelihood function → We then have a function that is a function of parameters

$$\rightarrow \boxed{L(w) = \prod_i^m p(w; y^{(i)}, x^{(i)})}$$

MAXIMUM LIKELIHOOD ESTIMATE

" Given a parameterized hypothesis function $p(y|x;w)$, we will choose as parameters the values which make the data y most likely under assumptions. "

Consider only one dimension with only one variable. x

$$p(Y_1 = y^{(1)} | x^{(1)}, w_0, w_1, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(- \frac{(y^{(1)} - w_0 - w_1 x^{(1)})^2}{2\sigma^2} \right)$$

Now ask which parameter to choose ~~so that the~~ in order to maximize the probability that $y = y^{(1)}$

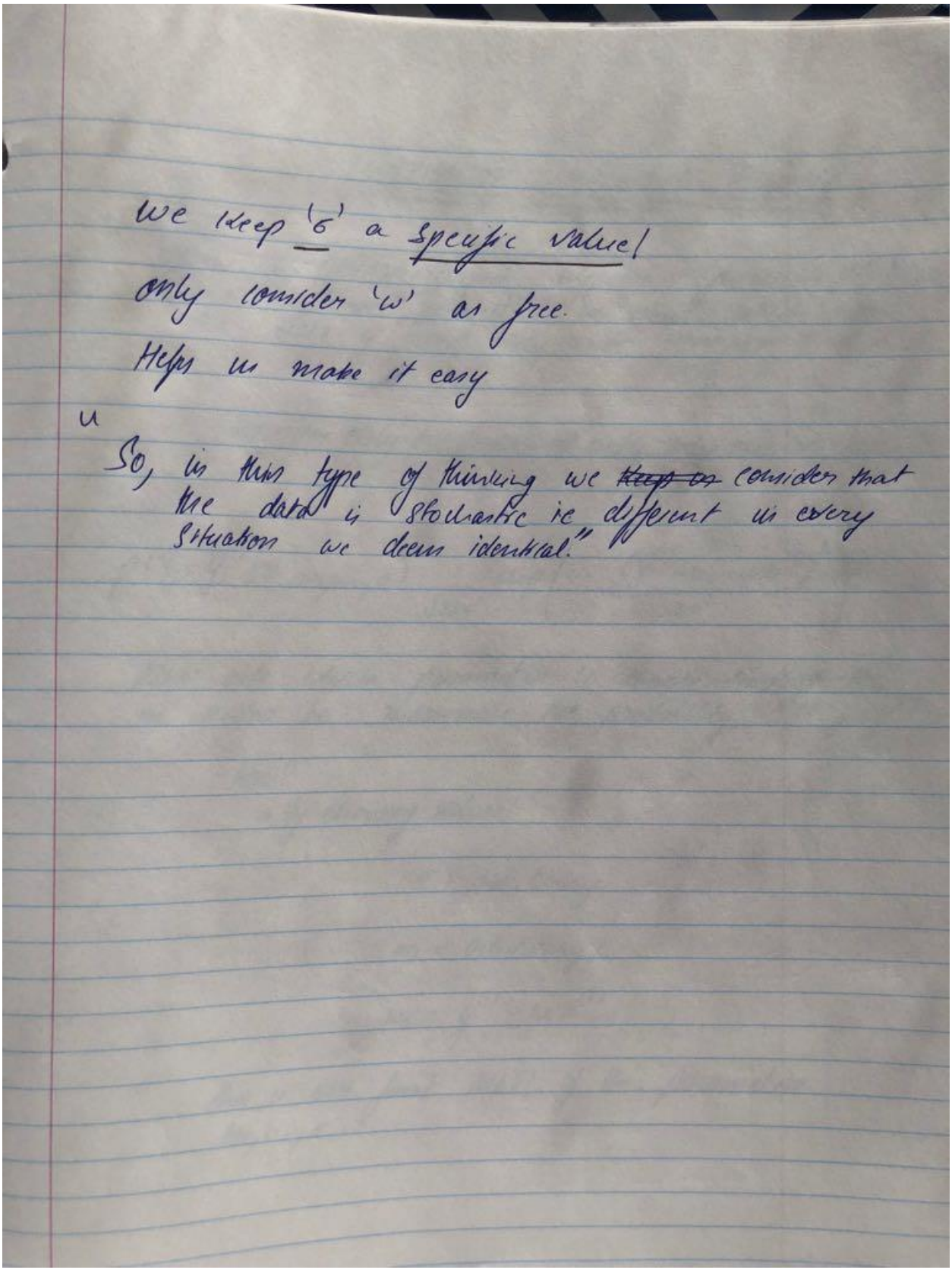
- How?
  - By choosing values!

σ = arbitrary

w_1 = arbitrary

$w_0 = y^{(1)} - w_1 x^{(1)}$

This is our first MLE! of the parameters w_0, w_1, σ.



we keep 'b' a specific value!
only consider 'w' as free.
Helps us make it easy

u

So, in this type of thinking we ~~keep as~~ consider that the data is stochastic ie different in every situation we deem identical."

Revising our hypothesis according to the ~~central limit~~ theory stochastic nature of data.

Conditional Density Function

$$p(\hat{y}, x|w) = p(\hat{y}|x;w)p(x)$$

Assume feature value are equally likely

So that $p(x)$ is constant ∝

we only have to focus on $p(\hat{y}|x;w)$

So

previously, hypothesis = $\hat{y}(x;w)$

new hypothesis = $\hat{y}(x;w)$ X additive gaussian noise

$$p(\hat{y}|x;w,\sigma) = N(\mu = w^T x, \sigma)$$

$$= \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(\hat{y}-w^T x)^2}{2\sigma^2}\right)$$

i.e. Gaussian distributed with mean $\mu = \hat{y}(x)$ that depends linearly on the value of x.

So, a linear regression

$\hat{y}(x;w) = w_0 + w_1 x$ ← how long motor is turned on.
↑ one variable

Bilinear Hypothesis

$\hat{y}(x;w) = w_0 + w_1 x_1 x_2$ ← power setting
↑ two variables

Sometimes we don't know one of these features. These features are therefore unknown knowledge rather than noise? Basically, the point is we don't care if the noise is either

- a randomness that comes from irreducible indeterminacy, i.e. true randomness in the world that can't be penetrated further for knowledge.

OR

- noise might represent epistemological limitations such as lack of knowledge of hidden process or limitations in observing states directly.

PROBABILISTIC MODELS

A simple stochastic generalisation of linear regression example to introduce formalism.

Example from Robotics

└ model how for robot moves

- When wheel are turning after given number of seconds after activating the corresponding motor with certain power.

Then input would be $(x^{(i)}, y^{(i)})$

$x^{(i)}$ is time we let motors run

$y^{(i)}$ distance robot traveled

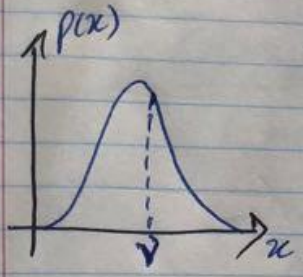
y has to be provided by teacher. We use ultrasonic sensor to automate the collection of data. (how close the robot is to wall)

we don't need the teacher later on!

Chi-Square Distribution

- Sum of squares of normally distributed random numbers.

Probability density function



$$p(x) = \frac{x^{(v-2)/2} e^{-x/2}}{2^{v/2} \Gamma(v/2)}$$

scipy.stats.gamma

Mean = v

Variance = $2v$

Multivariate Normal Distribution

- Several random variables $x_1, x_2, x_3 \text{---}, x_n$

$$p(x_1, x_2, \text{---}, x_n) = p(x) = \frac{1}{(\sqrt{2\pi})^n \sqrt{(\det(\Sigma))}} \exp\left(\frac{-1}{2}(x-\mu)^T \Sigma^{-1} (x-\mu)\right)$$

mean = $\vec{\mu}$

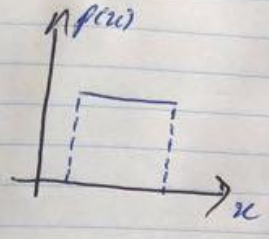
Variance $\Rightarrow \Sigma = [cov[x_i, x_j]]_{i=1,2,\text{---},n,\ k=1,2,\text{---},k}$

covariance matrix

Hilroy

Uniform Distribution

- Equally distributed random numbers in the interval $a \leq x \leq b$



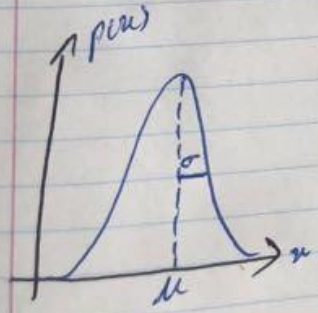
Probability density function

$$p(x) = \frac{1}{b-a}$$

$$\text{mean} = \frac{a+b}{2}$$

$$\text{Variance} = \frac{(b-a)^2}{2}$$

Gaussian Distribution



Probability density function

$$p(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-(x-\mu)^2}{2\sigma^2}}$$

$$\text{mean} = \mu$$

$$\text{Variance} = \sigma^2$$

Binomial Distribution (k=2)

Binomial coefficient := $\binom{n}{x} = \frac{n!}{x!(n-x)!}$

- Describes the number of 'successes' in 'n' Bernoulli trials, with probability of success p.

Probability function

$$f(x) = \binom{n}{x} p^x (1-p)^{n-x}$$

Mean := np

Variance := $np(1-p)$

GENERATIVE MODELS

- High dimensional data
- Jerit learn kai about nature of specific task &
- predict using this knowledge

- learning about classes & representing them in neural form is called representational learning.

- Representational models can be used to 'generate' examples of class objects & this model is therefore also called Generative model.

$$p(x|y;\theta)$$

Example :- learn chair, learn table
(knowledge)
↓
use to predict.

What is 'l'?

l^1 is regularisation with a bias in the parameters.

that we know

- Assume they should be normally distributed around zero.

Values of parameters them,

$$w^{MAP} = \text{argmax}_w \; p(X, y|w) N(0, \sigma^2)$$

Take log!

$$w^{MAP} = \text{argmax}_w \log\left(p(X, y|w)\right) + \alpha \|w\|^2$$

$$\alpha = \frac{1}{2\sigma^2}$$

Thus

LASSO $\rightarrow$ L^2 regularisation for Gaussian prior.

Ridge $\rightarrow$ L^1 for isotropic laplace distribution

forces more weights toward zero than L^2!!

- which can then be used to find the most likely value for the parameters!
- partition function doesn't depend on w. (the denominator)

Thus,

$$w^{MAP} = \text{argmax}_w \; p(x, y|w)\, p(w)$$

- w^{MAP}: Max aposteriori
- argmax: choose a 'w' so that it returns max. value
- $p(x, y|w)$: likelihood of that value
- $p(w)$: prior distribution

- Treat probability function as function of parameters.
- For uniform $p(w) = \text{const}$

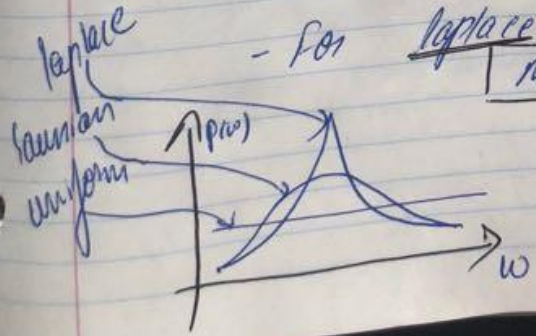
$$w^{MAP} = w^{MLE}$$

- For Gaussian distributed

$$MAP = MLE + L^2$$

- For laplace

$$MAP = MLE + L'$$



The main problem,

- that we don't know the distribution of apriori

- what if feature data is not given?

$$p(y, x|w) = p(y|x;w)\, p(x|w)$$

($p(y|x;w)$: conditional density; $p(x|w)$: marginal density)

- Now choose the feature data uniformly.
- So, that the marginal distribution is a constant

$$p(x|w) = \text{constant}$$

Hence,

$$p(y, x|w) \propto p(y|x;w)$$

Use Bayes theorem,

$$p(w|x,y) = \frac{p(y, x|w)\, p(w)}{\int_{w' \in W} p(x, y|w')\, p(w')\, dw'}$$

(denominator: partitioning factor; W: domain of possible parameter value)

MAXIMUM A POSTERIORI

Assume we know $p(w|x,y)$, ~~we can then~~ the most probable value, of given data set.

$$w^{MAP} = \text{argmax}_w \, p(w|x,y)$$

- MLE is work horse of probabilistic supervised learning.
- In probabilistic case, parameter should be selected based on data.

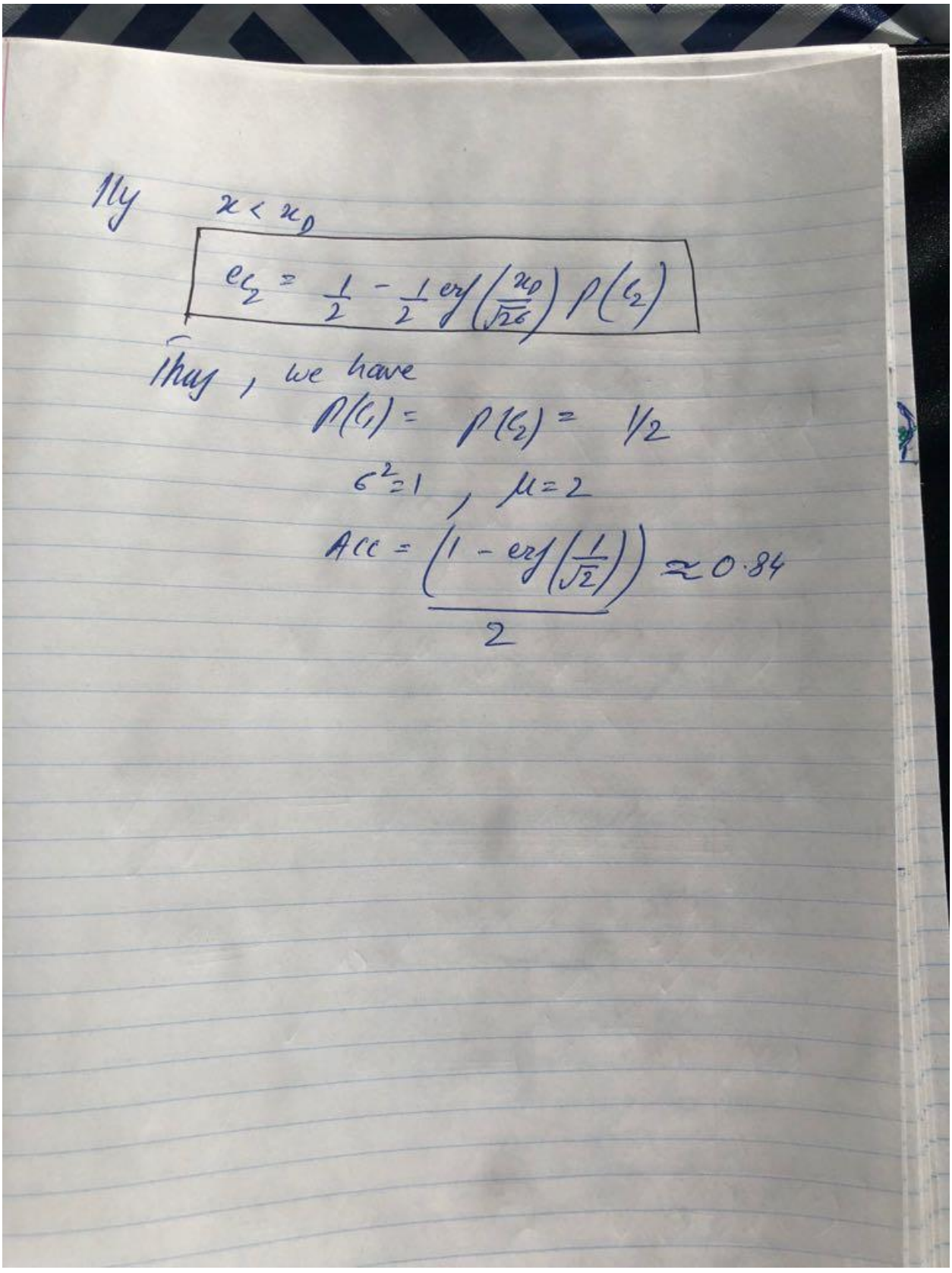
$$p(w|x,y)$$

Assume, we know the conditional probability

we can then choose a parameter value we like.

$$w^{MAP} = \text{argmax}_w \, p(w|x,y)$$

w^{MAP}: Value w, which is the most probable value


Ily $x < x_0$
$e_{C_2} = \frac{1}{2} - \frac{1}{2} erf\left(\frac{x_0}{\sqrt{2}\sigma}\right) P(C_2)$
Thus, we have
$P(C_1) = P(C_2) = 1/2$
$\sigma^2 = 1$, $\mu = 2$
$Acc = \frac{\left(1 - erf\left(\frac{1}{\sqrt{2}}\right)\right)}{2} \approx 0.84$

Take $\mu = 2$ & $\sigma = 1$

The bayesian decision point is where the two posteriors are equal.

$$e^{\frac{-x_0^2}{2\sigma^2}} p(C_1) = e^{\frac{-(x_0-\mu)^2}{2\sigma^2}} p(C_2)$$

$p(C_1)$ & $p(C_2)$ are marginal class probabilities

$$\boxed{x_0 = \frac{1}{2}\mu + \frac{\sigma^2}{\mu}\left(p(C_2) - p(C_1)\right)}$$

To find the error, find the area under the graph with wrong classification

Example: we predict C_1 wrongly for $x > x_0$ in the following fraction of cases

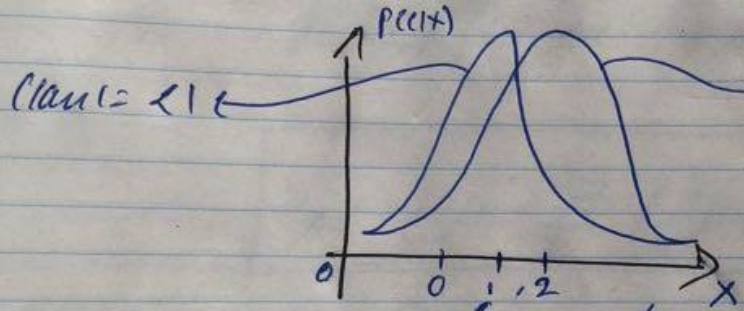
$$e_{C_1} = \frac{1}{\sqrt{2\pi}\sigma}\int_{x_0}^{\infty} e^{\frac{-x^2}{2\sigma^2}} dx \; p(C_1)$$

$$\Rightarrow \boxed{e_{C_1} = \frac{1}{2} - \frac{1}{2}\,\mathrm{erf}\left(\frac{x_0}{\sqrt{2}\sigma}\right) p(C_1)}$$

Supervised Gaussian Generative Models

– 1D Gaussian Example

~~Linear discriminant analysis~~



Gaussian distribution of feature 'x' of two classes.

- the features are overlapping i.e. the classes aren't fully distributed by this feature.

⇒ we can only classify values less than 1 as c1 & > 1 as c2

$$p(x|y=c_1) = \frac{1}{\sqrt{2\pi}\sigma} e^{\frac{-x^2}{2\sigma^2}}$$

$$p(x|y=c_2) = \frac{1}{\sqrt{2\pi}\sigma} e^{\frac{(x-\mu)^2}{2\sigma^2}}$$

In order to make a discriminitive model from generative model, we need class priors.

relative

what is the ~~Relevant~~ frequency of the classes is.

then,

$$p(y|x;\theta) = \frac{p(x|y;\theta)\,p(y)}{p(x)}$$

- the bayesian way of deciding which class has the target posterior probability

$$\underset{(y)}{\text{argmax}}\; p(y|x;\theta) = \underset{(y)}{\text{argmax}}\; \frac{p(y|x;\theta)\,p(y)}{p(x)}$$

Since, denominator has no 'y' we get ⇒

$$= \underset{(y)}{\text{argmax}}\; p(x|y;\theta)\,p(y)$$

11y - Binary Classification

$$\underset{(y)}{\text{argmax}}\; p(y|x;\theta) = \underset{(y)}{\text{argmax}}\; p(x|y=0,\theta)\,p(y=0),\; p(x|y=1,\theta)\,p(y=1)$$

Bayesian Decision Point / Dividing Hyperplane

$$P(y=1 \mid x^t) = P(y=0 \mid x^t) = 0.5 \rightarrow x^t w^T x^t = 0$$

- Values higher than this would result in positive output.

Sigmoid function

$$h(x;w) = \frac{1}{1+\exp(-w^T x)}$$

$$\frac{\partial h}{\partial w} = \frac{\partial}{\partial w}\frac{1}{1+e^{-wx}}$$

$$= \frac{1}{(1+e^{-wx})^2} e^{-wx}(-x)$$

$$= \frac{1}{(1+e^{-wx})}\left(1 - \frac{1}{(1+e^{-wx})}\right)(-x)$$

$$= -h(1-h)x$$

using this in ② & putting it in ①

$$y\left(\frac{1}{h} - (1-y)\frac{1}{1-h}\right)h(1-h) = y(1-h) - (y-h)h$$

$$= y - yh - h + yh$$

$$= y - h$$

Given learning rule,

$$w_j \leftarrow w_j + \alpha\left(y^{(i)} - h(x^{(i)};w)\right)x_j^{(i)}$$

Same as linear regression, but hypothesis is different

Hilary

Logistic Regression

$$p(y=1 \mid x;w) = h(x;w)$$

$$p(y=0 \mid x;w) = 1 - h(x;w)$$

($h(x;w)$: hypothesis)

Combined,

$$p(y \mid x;w) = (h(x;w))^{y} (1-h(x;w))^{1-y}$$

log-likelihood

$$l(w) = \sum_{i=1}^{m} y^{(i)} \log(h(x;w)) + (1-y^{(i)}) \log(1-h(x;w))$$

- To find maximum,

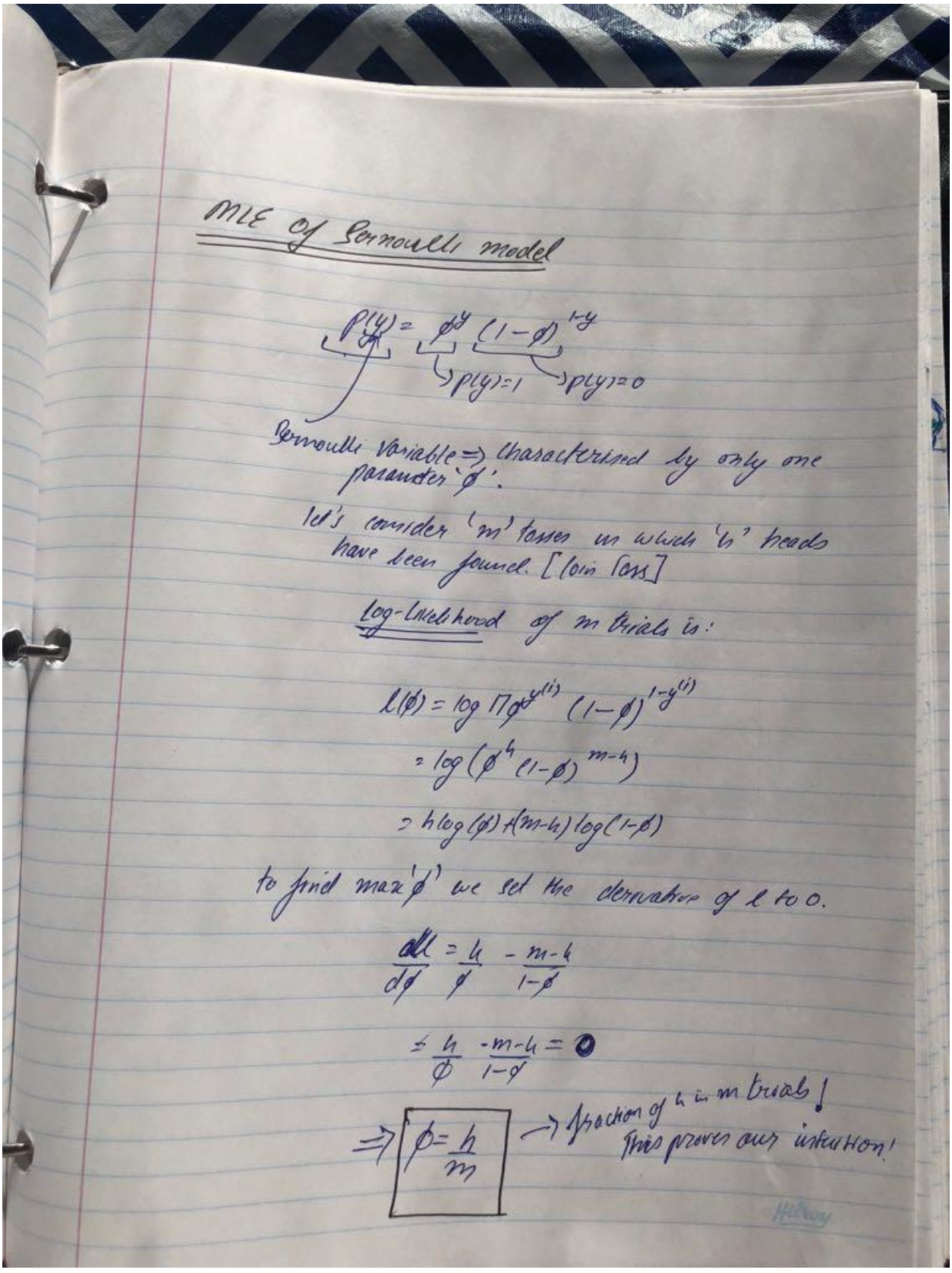
Gradient Ascent!

$$w \leftarrow w + \alpha \nabla_w l(w) \quad -(1)$$

We calculate partial derivative of log-likelihood function w.r.t. parameters.

$$\frac{\partial l(w)}{\partial w_j} = \left(y \frac{1}{h} - (1-y) \frac{1}{1-h} \right) \frac{\partial h(w)}{\partial w_j} \quad -(2)$$

Let's try to apply it!



MLE of Bernoulli model

$$p(y) = \phi^{y}(1-\phi)^{1-y}$$

$p(y)=1$ $\quad$ $p(y)=0$

Bernoulli Variable $\Rightarrow$ characterised by only one parameter 'ϕ'.

Let's consider 'm' tosses in which 'h' heads have been found. [Coin Toss]

Log-Likelihood of m trials is:

$$\ell(\phi) = \log \prod \phi^{y^{(i)}}(1-\phi)^{1-y^{(i)}}$$

$$= \log\left(\phi^{h}(1-\phi)^{m-h}\right)$$

$$= h\log(\phi) + (m-h)\log(1-\phi)$$

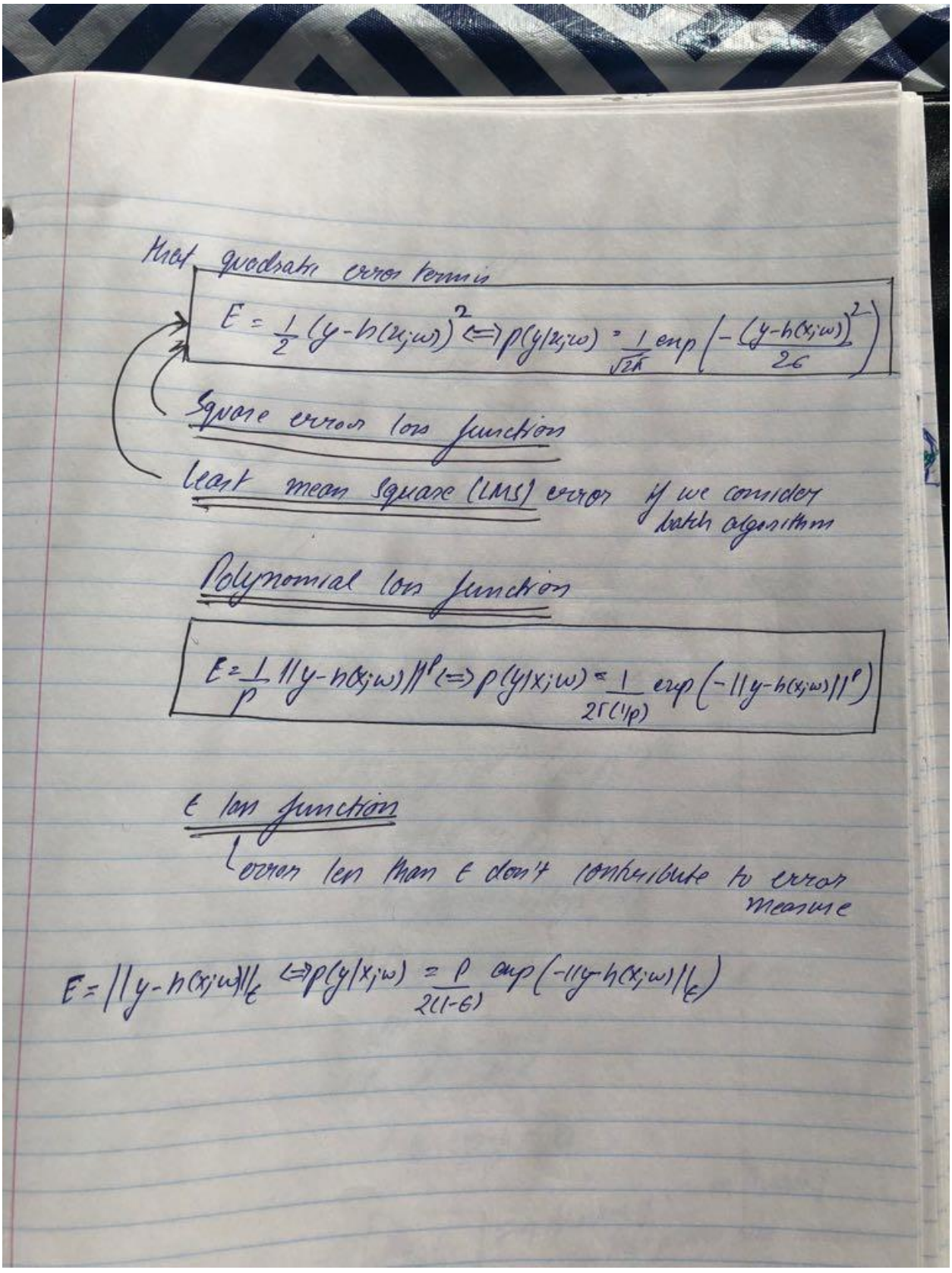
to find max 'ϕ' we set the derivative of ℓ to 0.

$$\frac{d\ell}{d\phi} = \frac{h}{\phi} - \frac{m-h}{1-\phi}$$

$$= \frac{h}{\phi} - \frac{m-h}{1-\phi} = 0$$

$$\Rightarrow \boxed{\phi = \frac{h}{m}}$$ $\rightarrow$ fraction of h in m trials!

This proves our intuition!

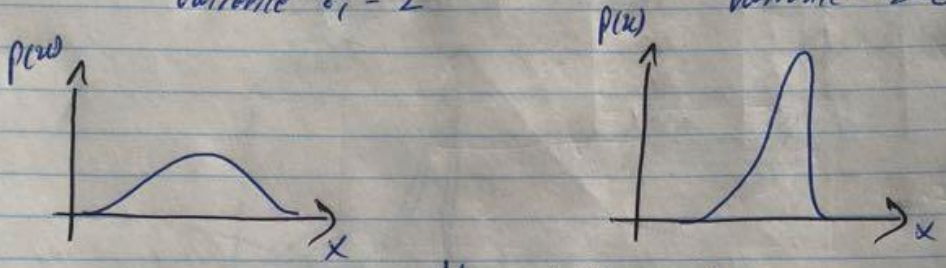

Most quadratic error term is
$E = \frac{1}{2}(y - h(x;w))^2 \Leftrightarrow p(y|x;w) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{(y-h(x;w))^2}{2\sigma}\right)$
Square error loss function
least mean square (LMS) error if we consider batch algorithm
Polynomial loss function
$E = \frac{1}{p}\|y - h(x;w)\|^p \Leftrightarrow p(y|x;w) = \frac{1}{2\Gamma(1/p)} \exp\left(-\|y - h(x;w)\|^p\right)$
ϵ loss function
error less than ϵ don't contribute to error measure
$E = \|y - h(x;w)\|_\epsilon \Leftrightarrow p(y|x;w) = \frac{p}{2(1-\epsilon)} \exp\left(-\|y - h(x;w)\|_\epsilon\right)$

10

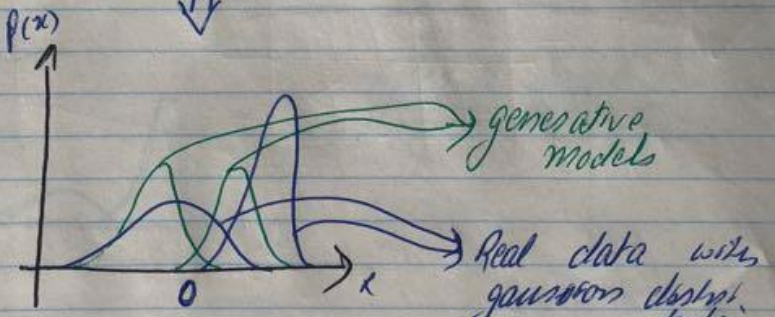
Example Consider two gaussian distributions ~~having same likelihood~~, generate data from both having equal likelihood to be from either.

data1 mean = $\mu_1 = -1$ data2 mean = $\mu_2 = 4$

Variance $\sigma_1 = 2$ Variance = $\sigma_2 = 0.5$



Equal likelihood



Initial condition

generative models

Real data with gaussian distributions.

We could do this, choose the hypothesis to match the actual data-generating system (world).

E-Step: Make assumption of label from data

M-Step: Use this hypothesis to update the parameters of the model to maximize the probability of observations.

Hilroy

9

Algorithm: (Estimation, maximisation)

① Initialize parameters ϕ, μ, Σ randomly!

② Repeat until convergence!

{

Estimation Step:

For each data point 'i' & class 'j'
"Soft-classify" data as

$$w_j^{(i)} = p(z^{(i)} = j / x^{(i)}; \phi, \mu, \Sigma)$$

Maximisation Step:

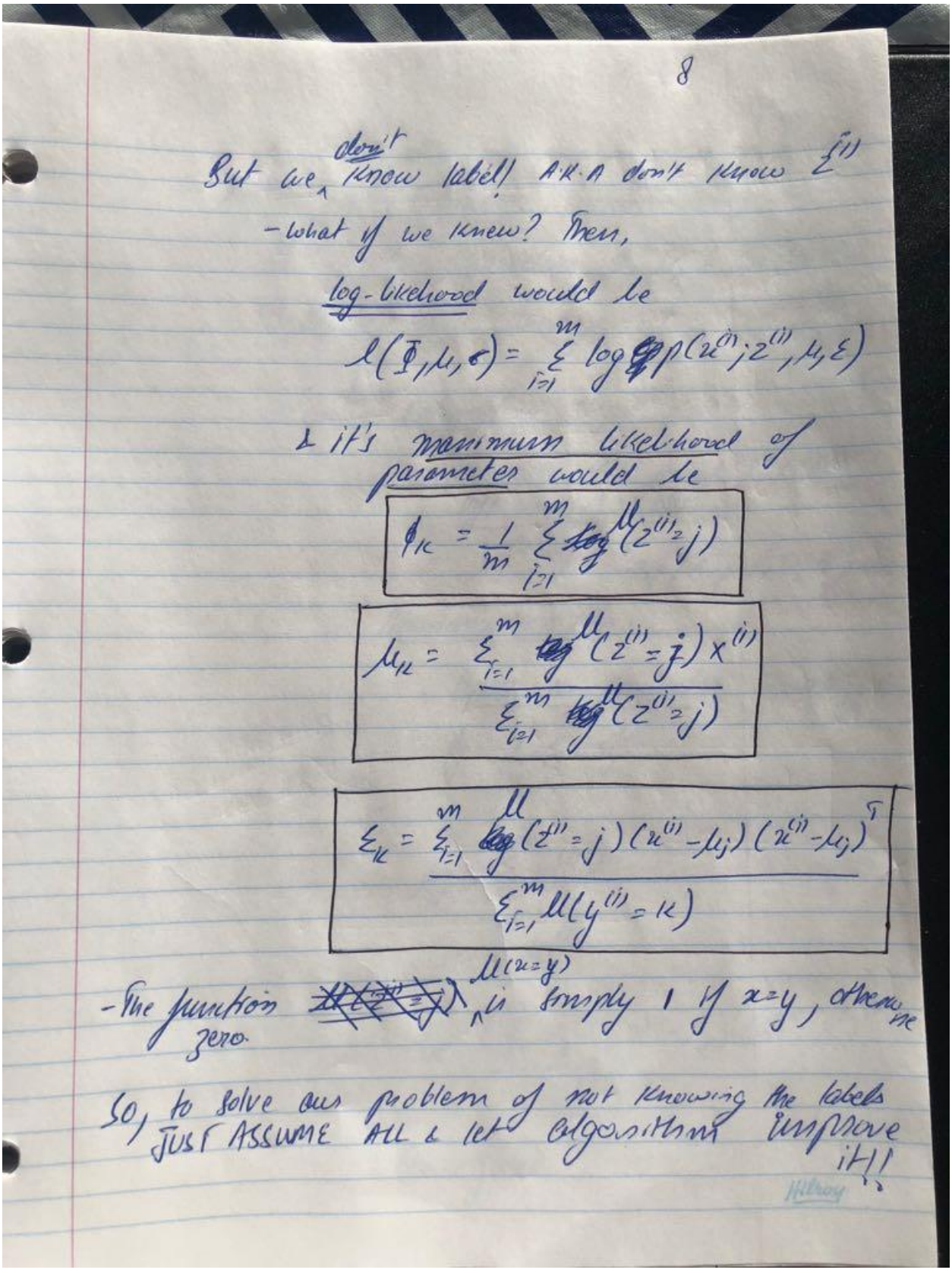
Update the parameter according to

$$\phi_j = \frac{1}{m} \sum_{i=1}^{m} w_j^{(i)}$$

$$\mu_j = \frac{\sum_{i=1}^{m} w_j^{(i)} x^{(i)}}{\sum_{i=1}^{m} w_j^{(i)}}$$

$$\Sigma_k = \frac{\sum_{i=1}^{m} w_j^{(i)} (x^{(i)} - \mu_j)(x^{(i)} - \mu_j)^T}{\sum_{i=1}^{m} w_j^{(i)}}$$

} convergence



8

But we don't know label!! A.K.A don't know $z^{(i)}$

- what if we knew? Then,

log-likelihood would be

$$\ell(\phi, \mu, \Sigma) = \sum_{i=1}^{m} \log p(x^{(i)}; z^{(i)}, \mu, \Sigma)$$

& it's maximum likelihood of parameter would be

$$\phi_k = \frac{1}{m} \sum_{i=1}^{m} 1\{z^{(i)} = j\}$$

$$\mu_k = \frac{\sum_{i=1}^{m} 1\{z^{(i)} = j\} x^{(i)}}{\sum_{i=1}^{m} 1\{z^{(i)} = j\}}$$

$$\Sigma_k = \frac{\sum_{i=1}^{m} 1\{z^{(i)} = j\} (x^{(i)} - \mu_j)(x^{(i)} - \mu_j)^T}{\sum_{i=1}^{m} 1\{y^{(i)} = k\}}$$

- The function $1\{x=y\}$ is simply 1 if $x=y$, otherwise zero.

So, to solve our problem of not knowing the labels JUST ASSUME ALL & let algorithm improve it!!

7

MIXTURE OF GAUSSIAN AND EM ALGORITHM

- try to combine K-mean with generative models, so that each class in them a specific generative model.

Example (a gaussian model)
└ we can use any type of distribution for this.

- Assume 'K' class, each class choosen randomly from a multinomial distribution.

$$p(z^{(i)} = j) \propto \text{multinomial}(\phi_j)$$

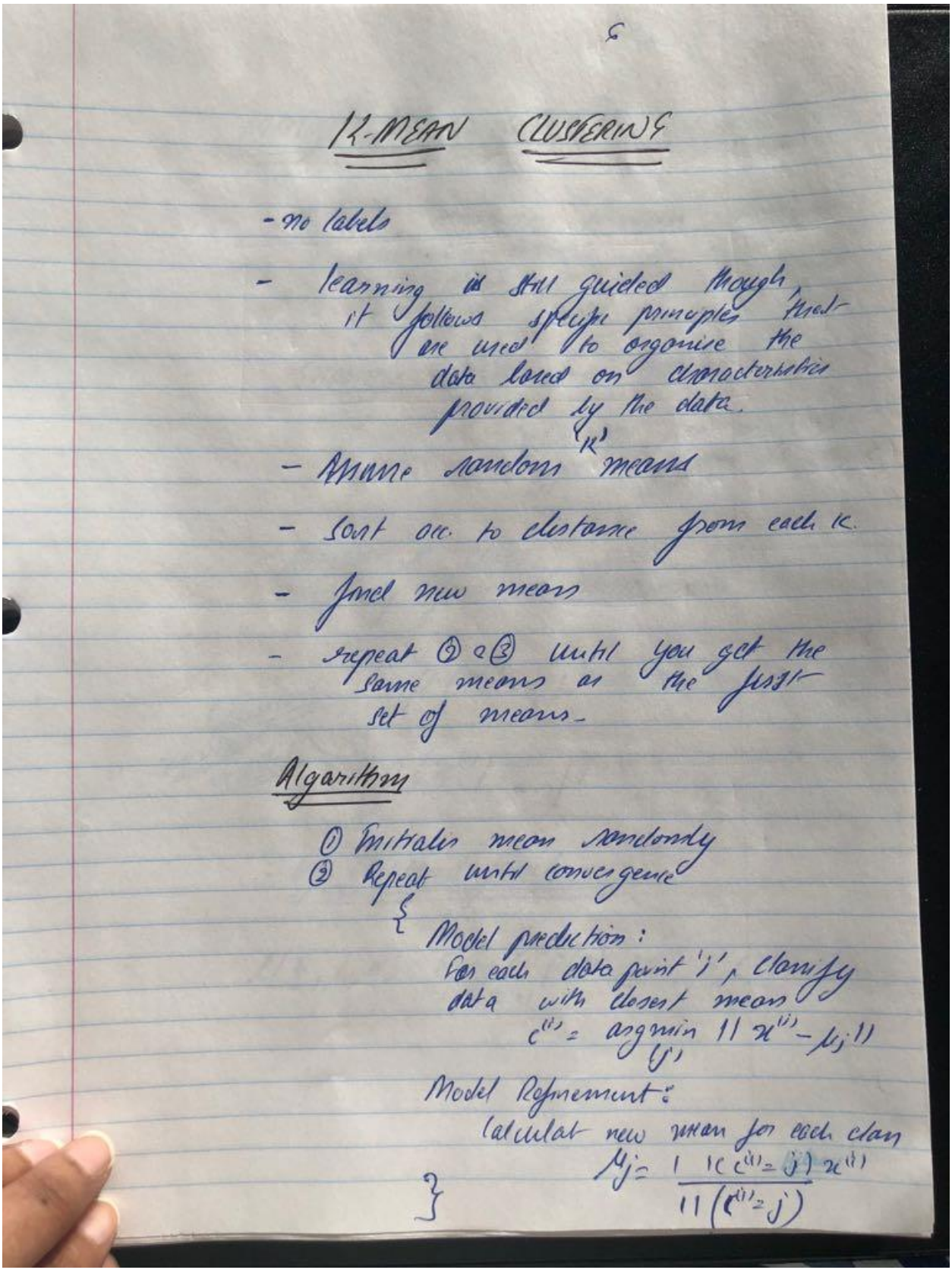
Gaussian mixture model →

$$\boxed{p(x^{(i)} | z^{(i)} = j) \propto N(\mu_j, \Sigma_j)}$$

- log-likelihood function

$$l(\phi, \mu, \Sigma) = \sum_{i=1}^{m} \log \sum_{z^{(i)}=1}^{K} p(x^{(i)} | z^{(i)}; \mu, \Sigma) p(z^{(i)}; \phi)$$

Hilroy



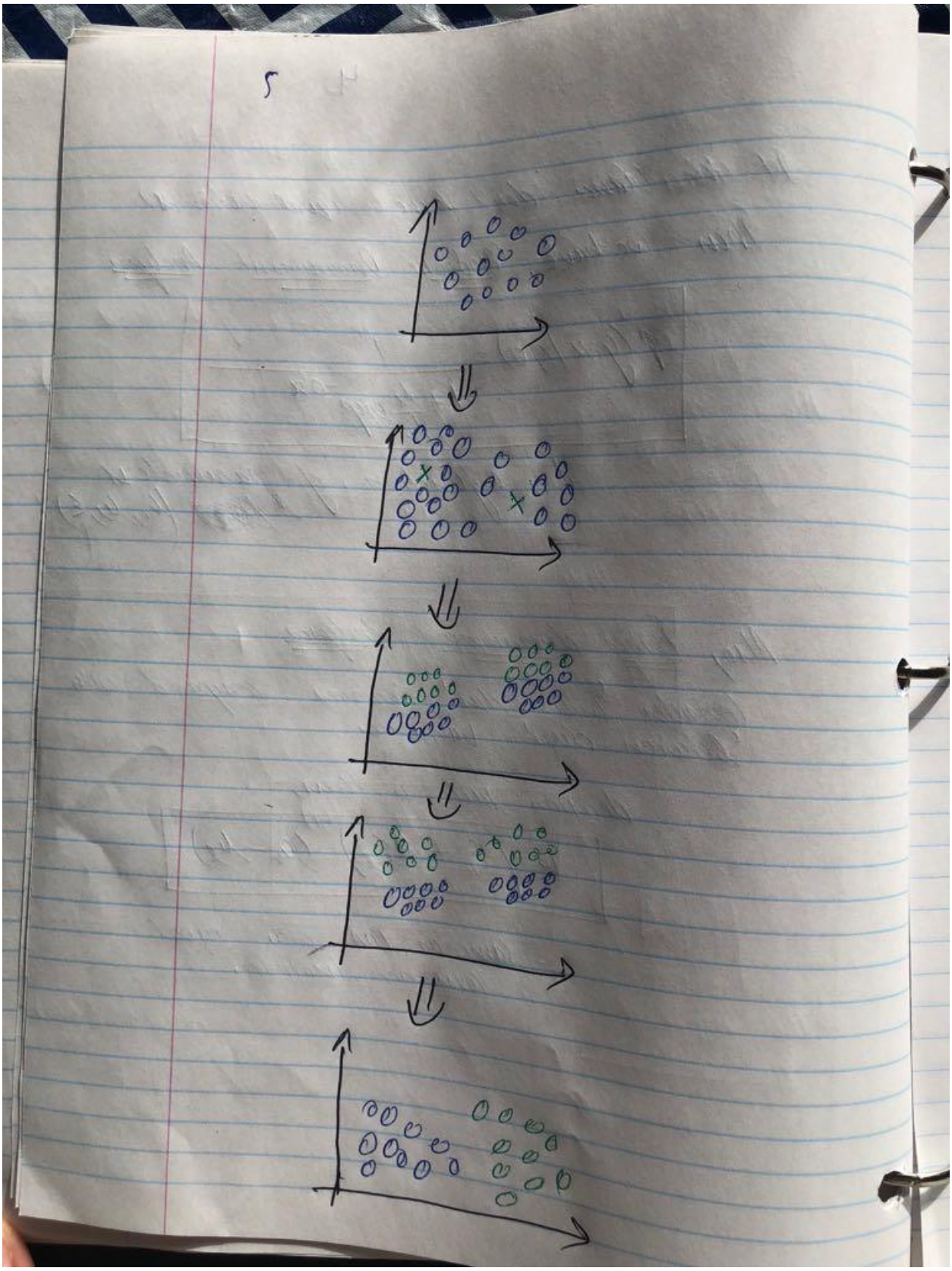
6

K-MEAN CLUSTERING

- no labels
- learning is still guided though it follows specific principles that are used to organise the data based on characteristics provided by the data.
- Name random means (K)
- Sort acc. to distance from each K.
- Find new means
- repeat ① & ② until you get the same means as the just set of means.

Algorithm

① Initialise mean randomly
② Repeat until convergence
{
Model prediction:
For each data point 'i', classify data with closest mean
$c^{(i)} = \underset{(j)}{\text{argmin}} \; \|x^{(i)} - \mu_j\|$
Model Refinement:
Calculate new mean for each class
$\mu_j = \frac{1}{\|(c^{(i)} = j)} \; 1(c^{(i)} = j)\, x^{(i)}$
}



4

If the classes don't have equal variances?

Then we have, Quadratic Discriminant Analysis

$$p(y=k|x;\theta) = \frac{1}{1+\frac{\phi_l}{\phi_k}\exp^{-\theta^T x}}$$

ϕ is a function of parameters (μ_k, μ_l, Σ) (Variance)

Also, Fisher discriminant analysis

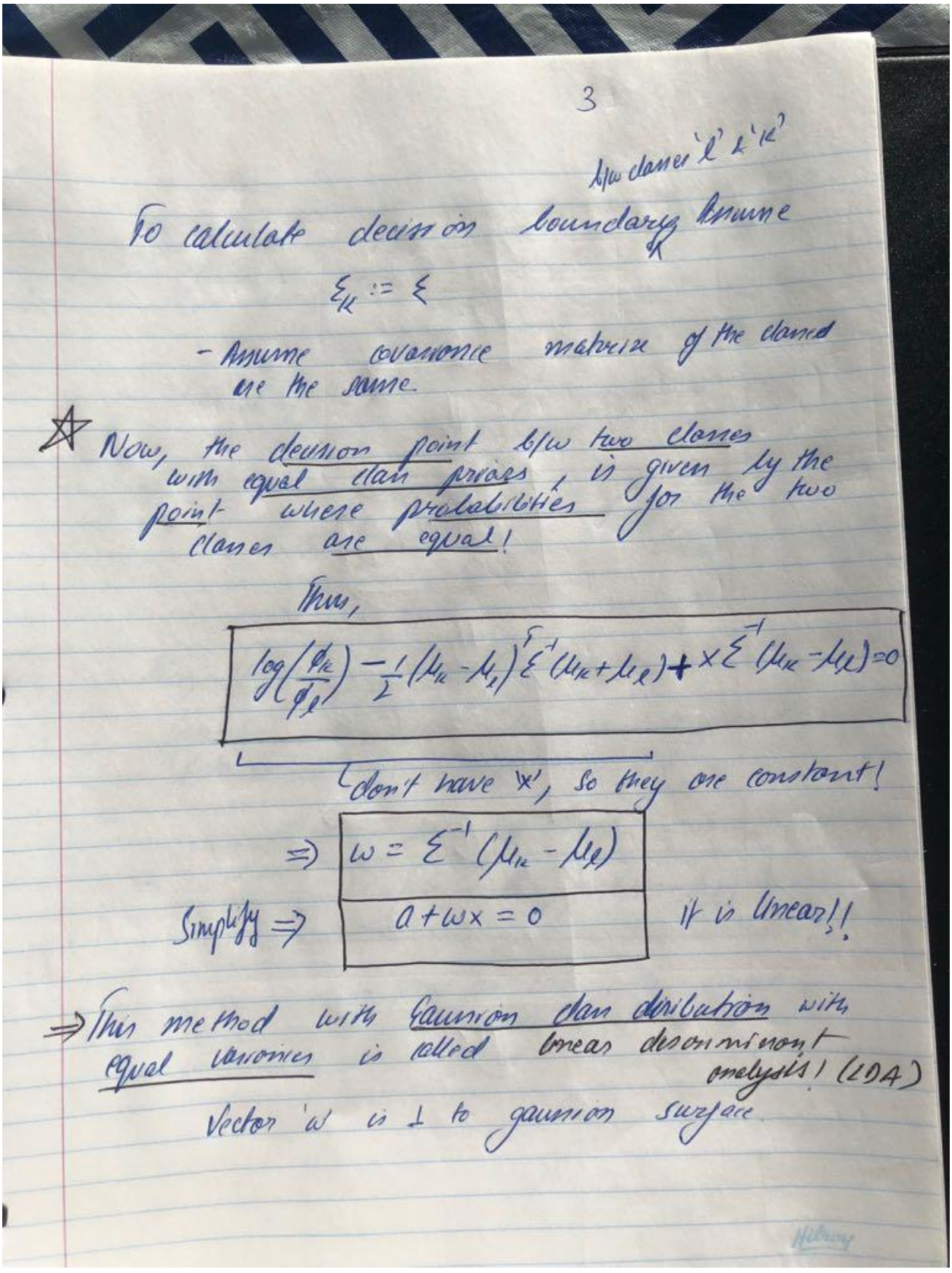
Consider 'within-class' variances compared with 'between-class' variances.

decision rule is thus,

$$w = (\Sigma_k + \Sigma_l)^{-1}(\mu_k - \mu_l)$$

- same as LDA with equal covariance matrices.

Hilroy



3

b/w classes 'k' & 'l'

To calculate decision boundary assume

$\Sigma_k := \Sigma$

- Assume covariance matrix of the classes are the same.

☆ Now, the decision point b/w two classes with equal class priors, is given by the point where probabilities for the two classes are equal!

Thus,

$$\log\left(\frac{\phi_k}{\phi_l}\right) - \frac{1}{2}(\mu_k - \mu_l)^T \Sigma^{-1}(\mu_k + \mu_l) + x\Sigma^{-1}(\mu_k - \mu_l) = 0$$

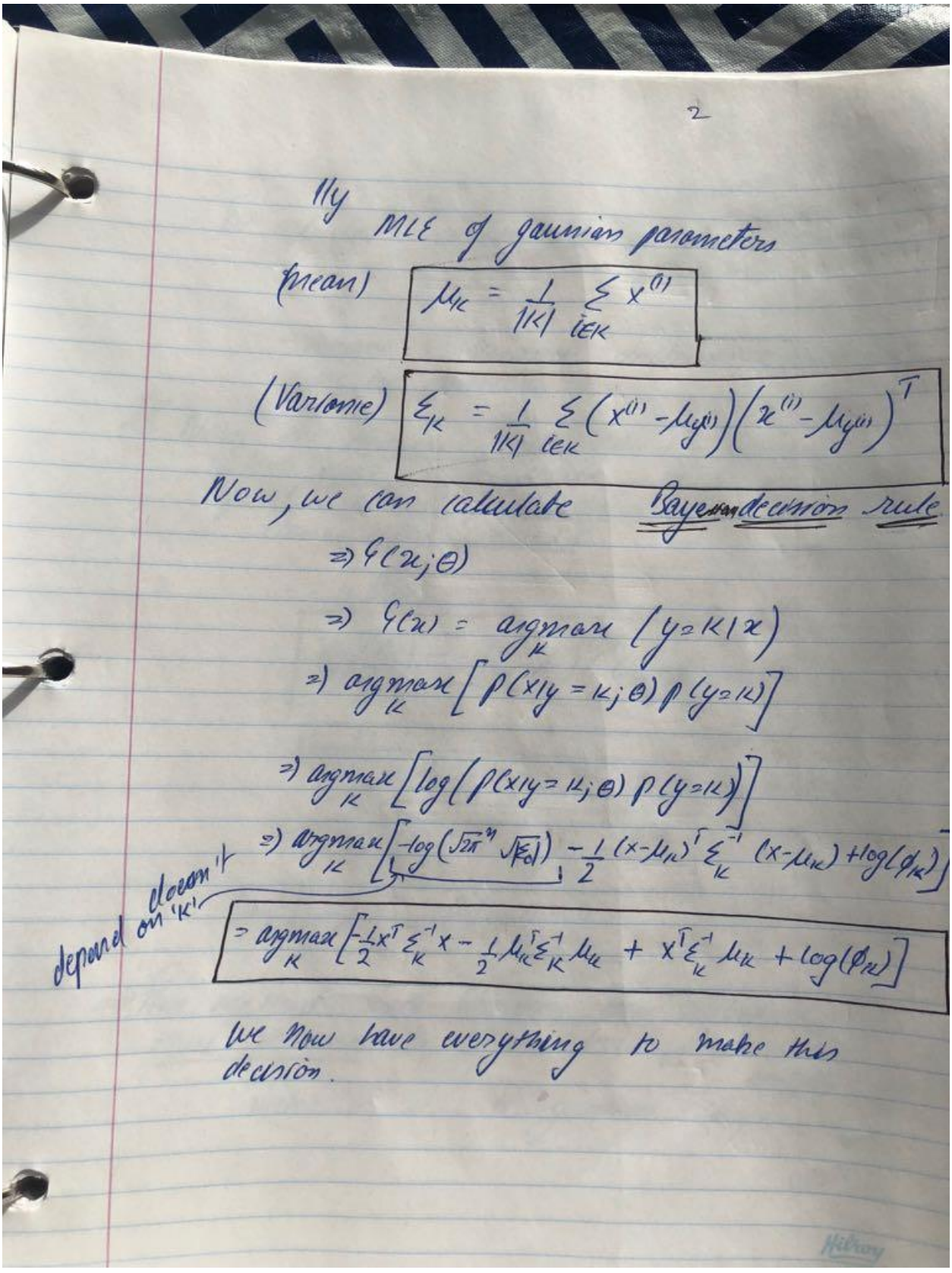
(don't have 'x', so they are constant!

$$\Rightarrow \boxed{w = \Sigma^{-1}(\mu_k - \mu_l)}$$

Simplify $\Rightarrow$ $\boxed{a + wx = 0}$ It is linear!!

$\Rightarrow$ This method with Gaussian class distribution with equal variances is called linear discriminant analysis! (LDA)

Vector 'w' is ⊥ to gaussian surface.



2

IIy MLE of gaussian parameters

(mean) $$\mu_k = \frac{1}{|k|} \sum_{i \in k} x^{(i)}$$

(Variance) $$\Sigma_k = \frac{1}{|k|} \sum_{i \in k} (x^{(i)} - \mu_{y^{(i)}})(x^{(i)} - \mu_{y^{(i)}})^T$$

Now, we can calculate Bayes decision rule

$\Rightarrow g(x;\theta)$

$\Rightarrow g(x) = \underset{k}{\text{argmax}} \, (y=k|x)$

$\Rightarrow \underset{k}{\text{argmax}} \left[p(x|y=k;\theta) \, p(y=k) \right]$

$\Rightarrow \underset{k}{\text{argmax}} \left[\log \left(p(x|y=k;\theta) \, p(y=k) \right) \right]$

$\Rightarrow \underset{k}{\text{argmax}} \left[-\log\left(\sqrt{2\pi}^n \sqrt{|\Sigma_k|}\right) - \frac{1}{2}(x-\mu_k)^T \Sigma_k^{-1} (x-\mu_k) + \log(\phi_k) \right]$

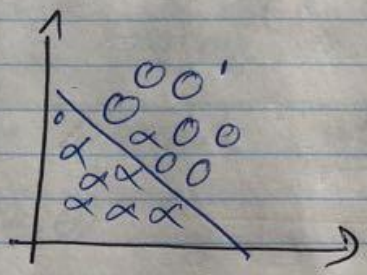
depend on 'k'

$$= \underset{k}{\text{argmax}} \left[-\frac{1}{2} x^T \Sigma_k^{-1} x - \frac{1}{2} \mu_k^T \Sigma_k^{-1} \mu_k + x^T \Sigma_k^{-1} \mu_k + \log(\phi_k) \right]$$

We now have everything to make this decision.

- Linear Discriminant Analysis [Gaussian class distribution with equal variances]

We now have 'K' class with gaussian distribution over 'n' attributes.



- Each class has class prior

$$p(y=k) = \phi$$

- Each class is multivariate Gaussian distributed each with different μ_k & Σ_k

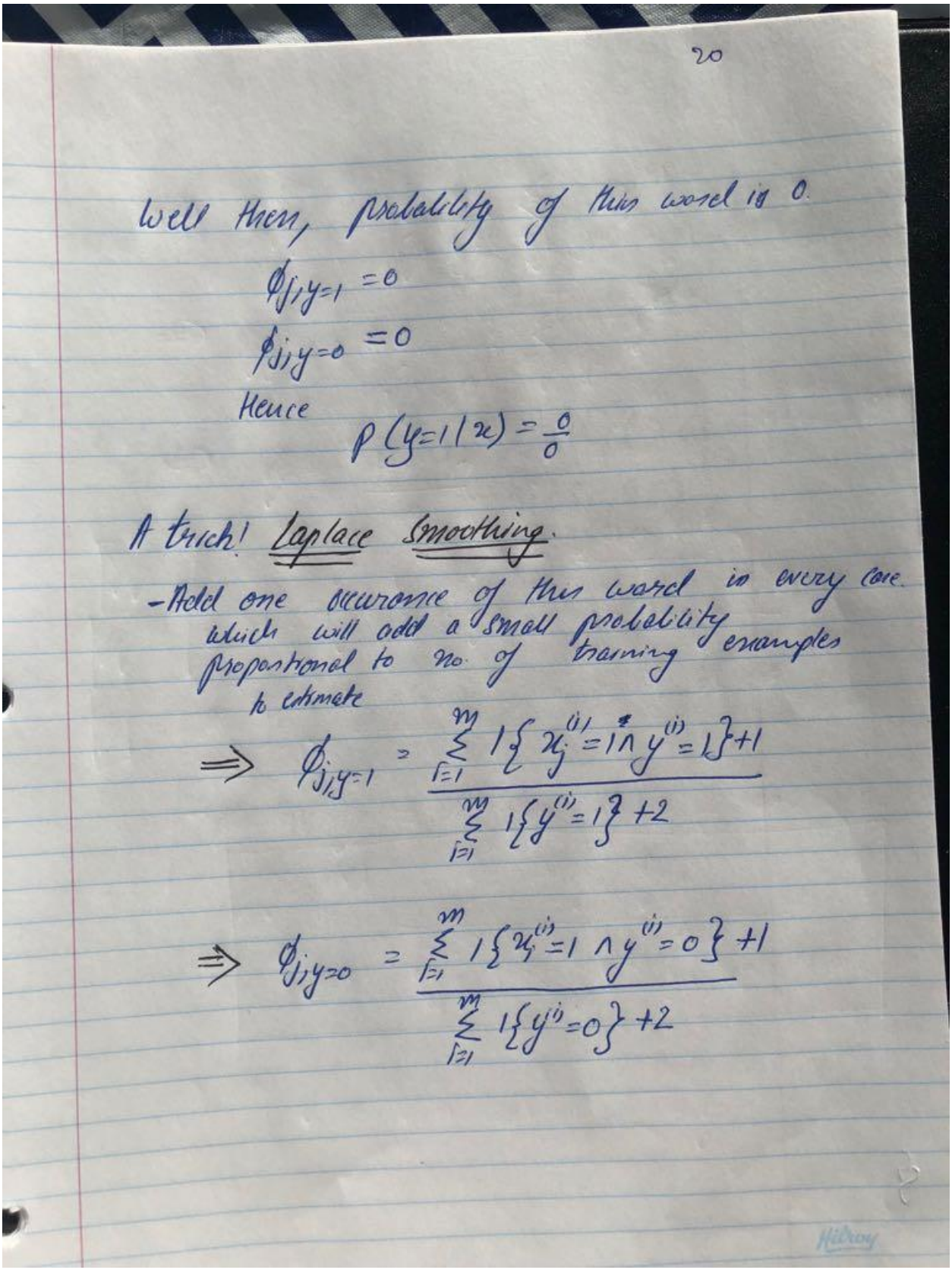
$$p(x|y=k) = \frac{1}{\sqrt{2\pi}^n \sqrt{|\Sigma_k|}} e^{-\frac{1}{2}(x-\mu_k)^T \Sigma_k^{-1} (x-\mu_k)}$$

- Use MLE to ~~predict~~ estimate the value of parameters

$$\theta = (\phi_k, \mu_k, \Sigma_k)$$

class prior $\rightarrow \phi_k = \frac{|T_k|}{m}$ $\rightarrow$ no. of examples in this set

$\rightarrow$ set of examples of class k.



20

Well then, probability of this word is 0.

$$\phi_{j|y=1} = 0$$

$$\phi_{j|y=0} = 0$$

Hence

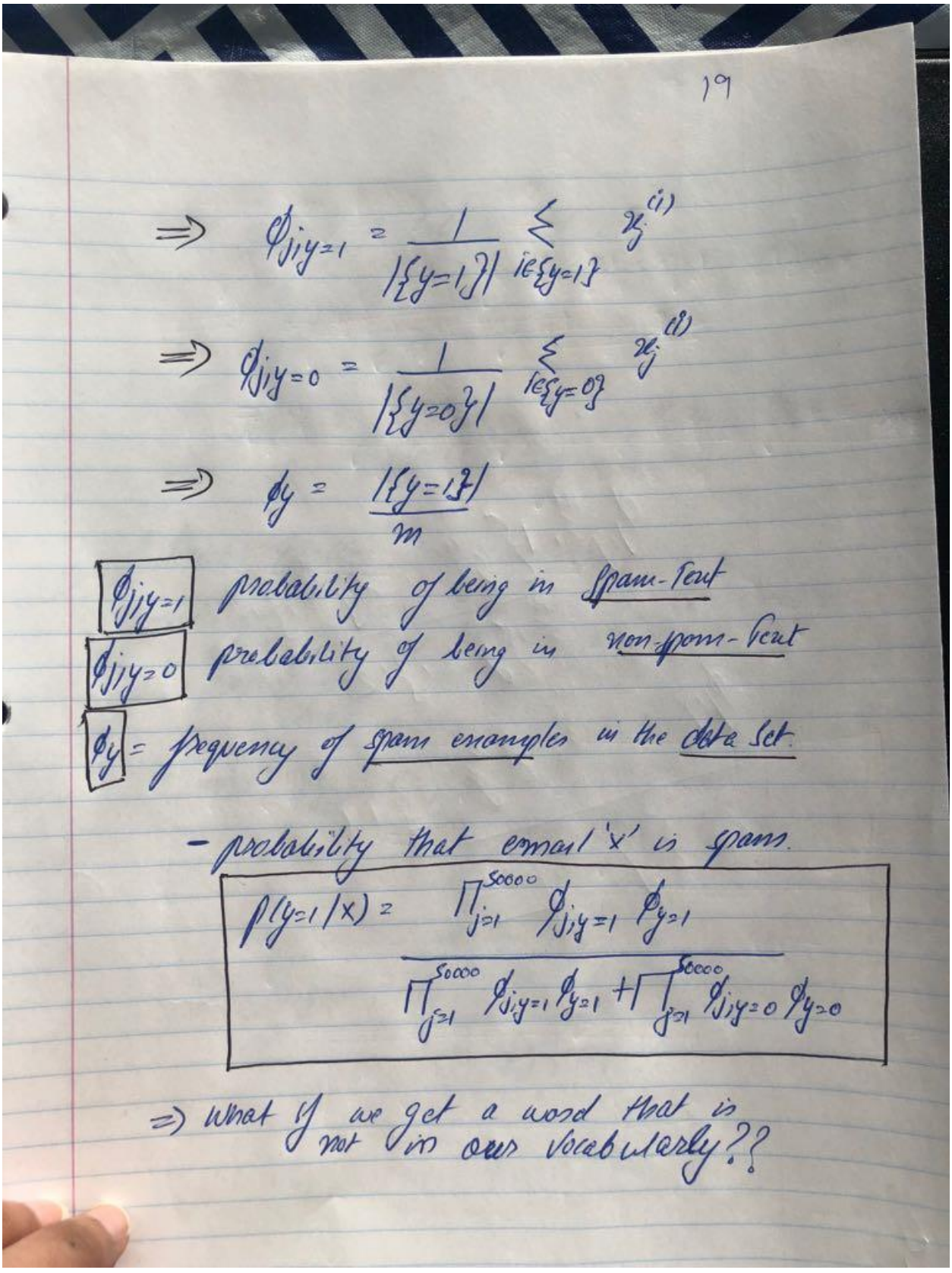
$$p(y=1|x) = \frac{0}{0}$$

A trick! Laplace Smoothing.

- Add one occurance of this word in every case which will add a small probability proportional to no. of training examples to estimate

$$\Rightarrow \phi_{j|y=1} = \frac{\sum_{i=1}^{m} 1\{x_j^{(i)} = 1 \wedge y^{(i)} = 1\} + 1}{\sum_{i=1}^{m} 1\{y^{(i)} = 1\} + 2}$$

$$\Rightarrow \phi_{j|y=0} = \frac{\sum_{i=1}^{m} 1\{x_j^{(i)} = 1 \wedge y^{(i)} = 0\} + 1}{\sum_{i=1}^{m} 1\{y^{(i)} = 0\} + 2}$$



19

$$\Rightarrow \phi_{j|y=1} = \frac{1}{|\{y=1\}|} \sum_{i \in \{y=1\}} x_j^{(i)}$$

$$\Rightarrow \phi_{j|y=0} = \frac{1}{|\{y=0\}|} \sum_{i \in \{y=0\}} x_j^{(i)}$$

$$\Rightarrow \phi_y = \frac{|\{y=1\}|}{m}$$

$\phi_{j|y=1}$ probability of being in Spam-Text

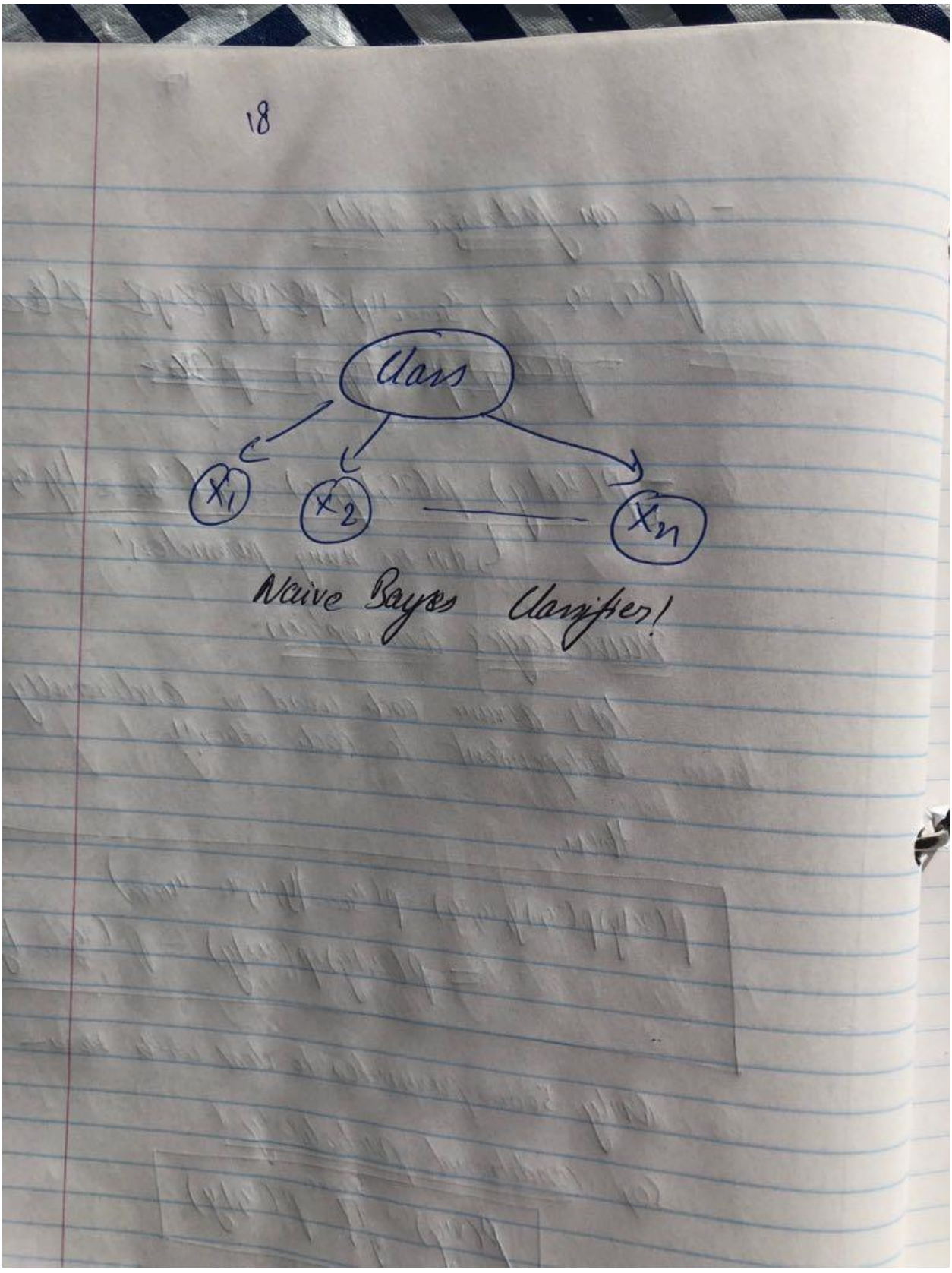
$\phi_{j|y=0}$ probability of being in non-spam-text

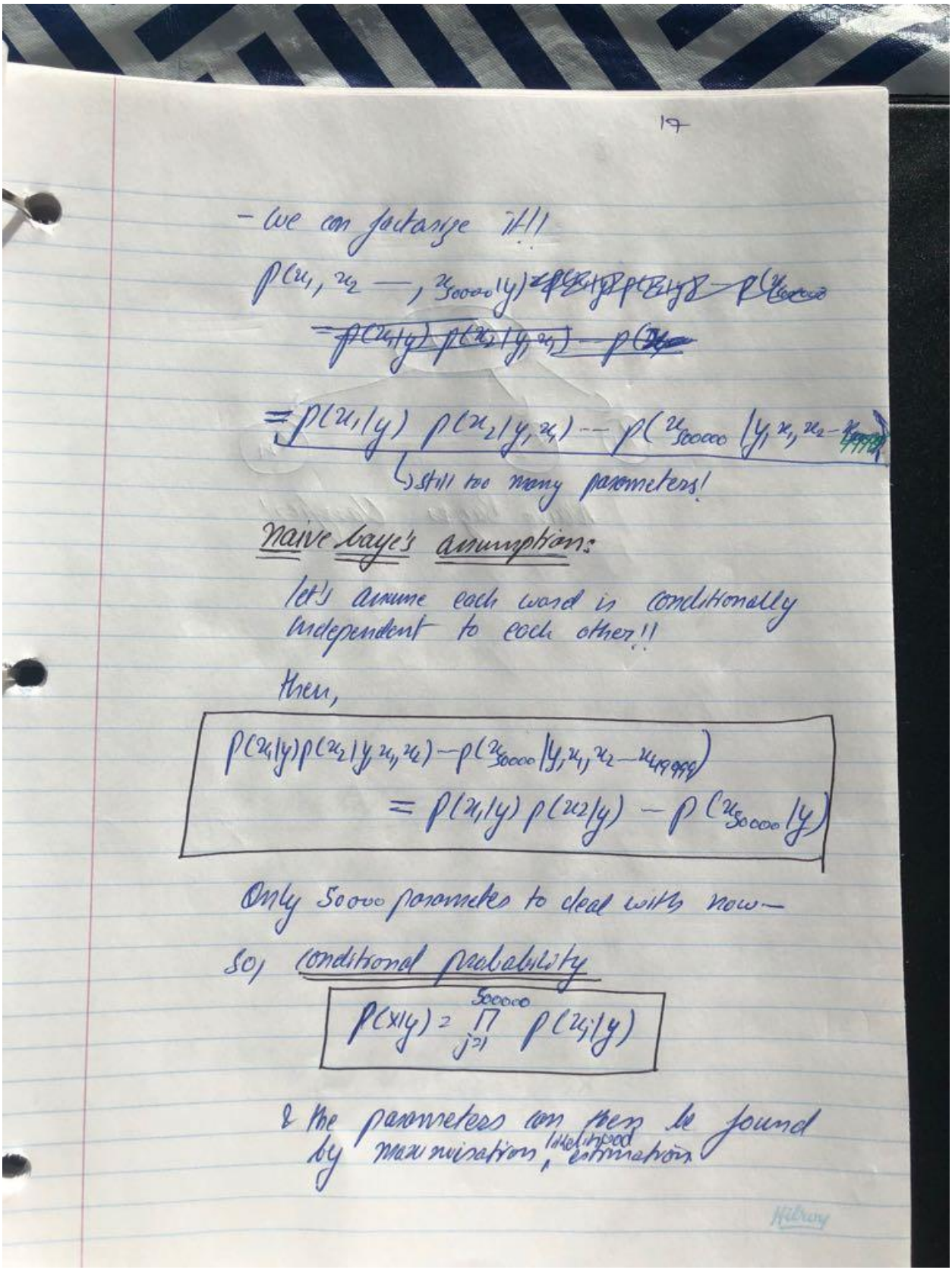
ϕ_y = frequency of spam examples in the data set

- probability that email 'x' is spam.

$$p(y=1|x) = \frac{\prod_{j=1}^{50000} \phi_{j|y=1} \phi_{y=1}}{\prod_{j=1}^{50000} \phi_{j|y=1} \phi_{y=1} + \prod_{j=1}^{50000} \phi_{j|y=0} \phi_{y=0}}$$

⇒ What if we get a word that is not in our vocabulary??


18
Class
X_1
X_2
X_n
Naive Bayes Classifier!



17

- we can factorize it!!

$p(x_1, x_2 \dots, x_{50000} | y)$ ~~$= p(x_1|y)\, p(x_2|y, x_1) \dots p(x_{50000}|y, x_1, x_2 \dots x_{49999})$~~

$= p(x_1|y)\, p(x_2|y, x_1) \dots p(x_{50000}|y, x_1, x_2 \dots x_{49999})$

↳ still too many parameters!

Naive Bayes assumption:

let's assume each word is conditionally independent to each other!!

Then,

$$p(x_1|y)\, p(x_2|y, x_1, x_2) \dots p(x_{50000}|y, x_1, x_2 \dots x_{49999}) = p(x_1|y)\, p(x_2|y) \dots p(x_{50000}|y)$$

Only 50000 parameters to deal with now—

So, conditional probability

$$p(x|y) = \prod_{j=1}^{50000} p(x_j|y)$$

& the parameters can then be found by maximization, likelihood estimation

16

NAIVE BAYES

Example Classify Spam & not Spam
($y=1$) ($y=0$)

Store all the words in a vocabulary vector.

$$x = \begin{bmatrix} a \\ ab \\ c \\ d \\ e \\ \vdots \end{bmatrix}$$

then, text is represented by 1 if it has that word.

$$x = \begin{bmatrix} 1 \\ 0 \\ 1 \\ 1 \\ \vdots \end{bmatrix}$$

If this vocabulary has the count of each word then it is called a "bag of words".

So, our vocabulary has 50,000 words now.

i.e. we want to model

$$p(x|y) = p(x_1, x_2 \text{ —— }, x_{50000} | y)$$

- this has $2^{50,000} - 1$ parameters!!

~~x_0~~ $x_0 = 1$ [normalization condition]

15

Example

- joined distribution of the five variables

$$P(B, E, A, J, M) = P(B|E, A, J, M)\, P(E|A, J, M)\, P(A|J, M)\, P(J|M)\, P(M)$$

- However, we have a specific factorization

$$P(B, E, A, J, M) = P(B)\, P(E)\, P(A|B, E)\, P(J|A)\, P(M|A)$$

which is much easier to handle!!

- what if we want to deduce about parent from child nodes?

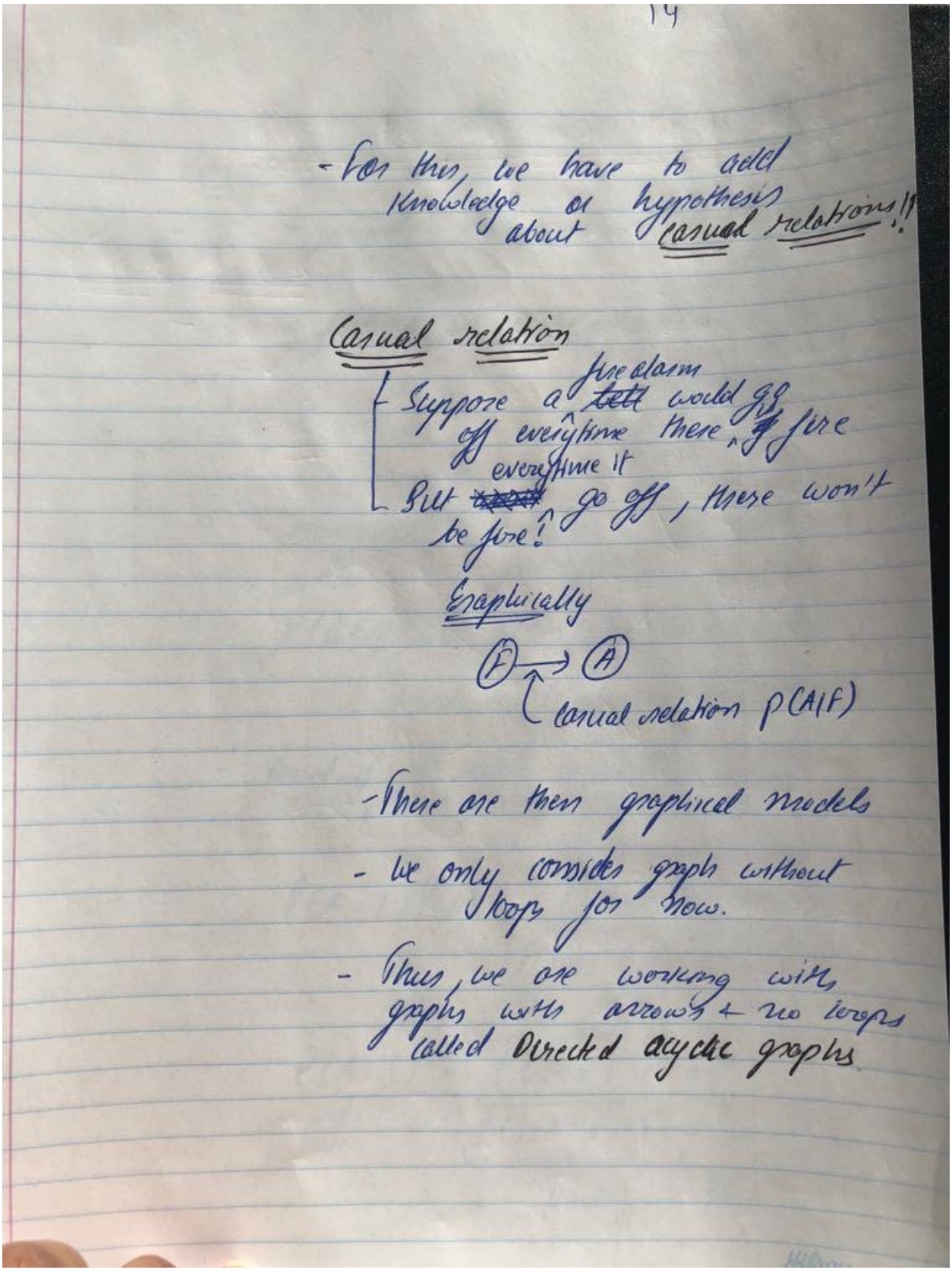
$$P(A = t|J = t) = \frac{P(J = t|A = t)\, P(A = t)}{P(J|A)\, P(A) + P(J|A)\, P(A)}$$

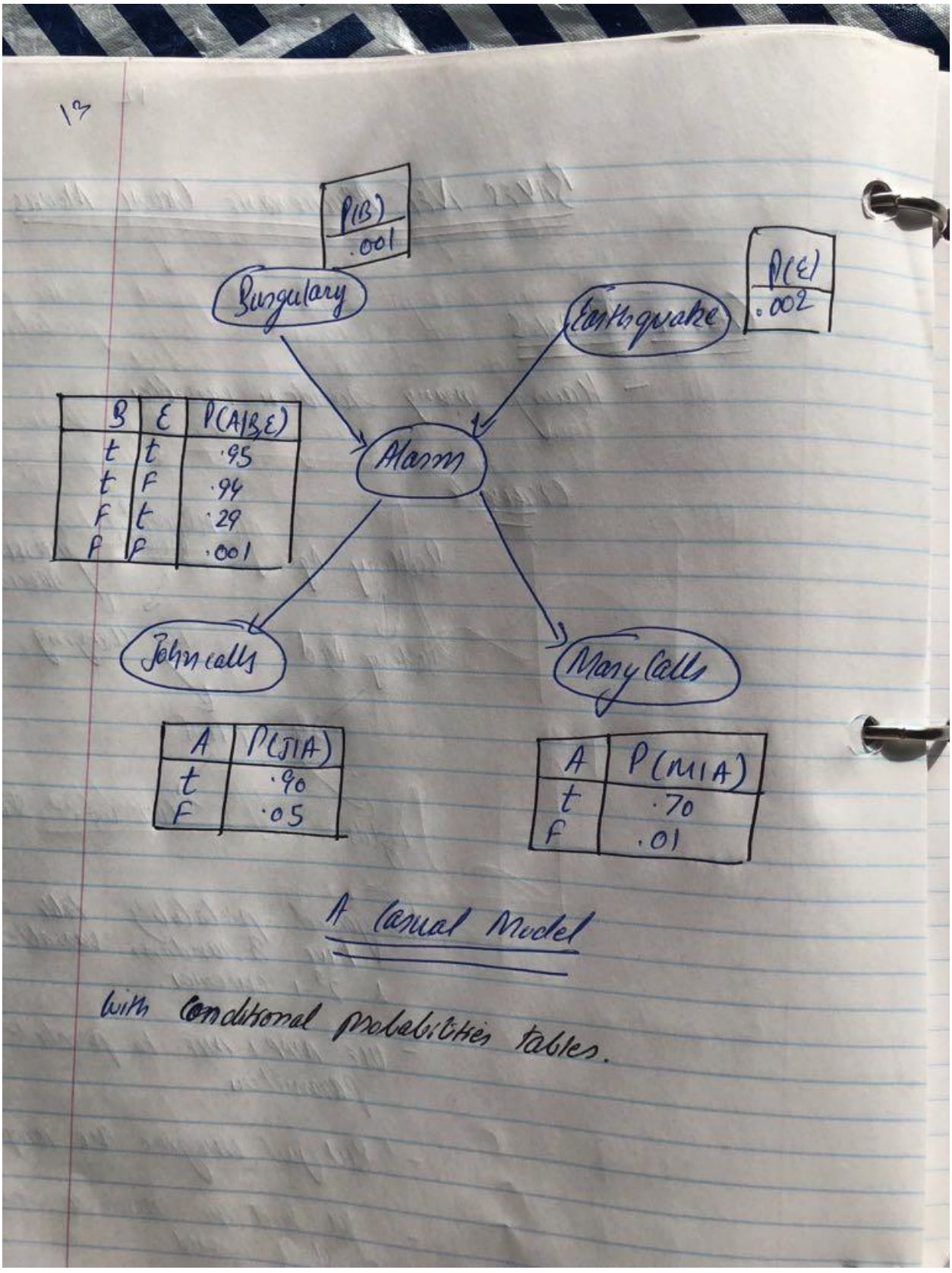
($P(J|A)$ with $J = t$, $A = t$; $P(A)$ with $A = t$; $P(J|A)$ with $J = t$, $A = f$; $P(A)$ with $A = f$)

OR

$$P(A|J) = \frac{\overbrace{P(J|A)\, P(A)}^{\text{true}}}{\underbrace{P(J|A)\, P(A)}_{\text{true}} + \underbrace{P(J|A)\, P(A)}_{\text{false}}}$$

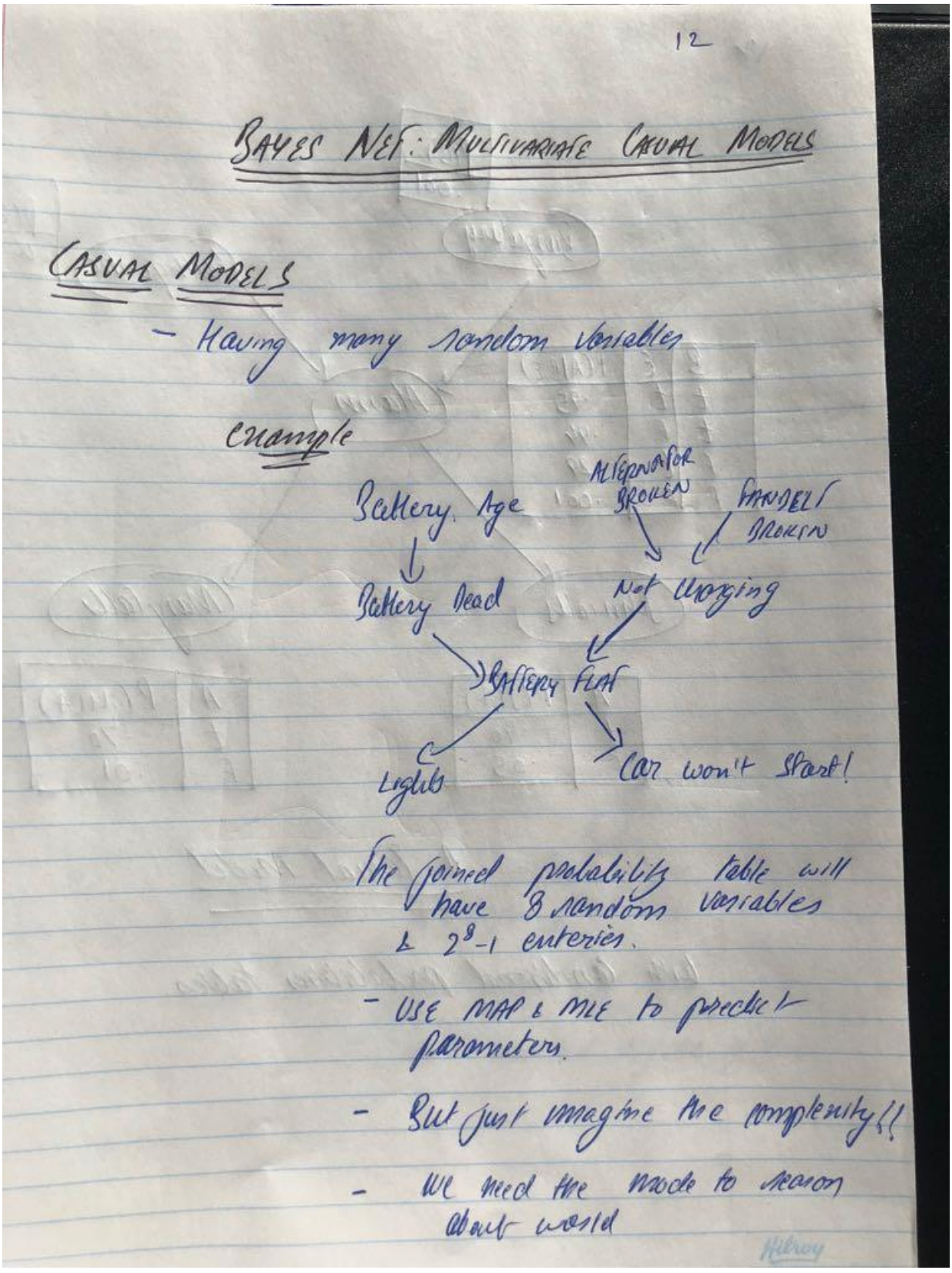
$P(A|J)$: J true


14
- For this, we have to add knowledge or hypothesis about casual relations!!
Casual relation
Suppose a fire alarm would go off everytime there is fire
everytime it
But go off, there won't be fire?
Graphically
F
A
casual relation p(A|F)
- There are then graphical models
- We only consider graph without loops for now.
- Thus, we are working with graphs with arrows + no loops called Directed acyclic graphs



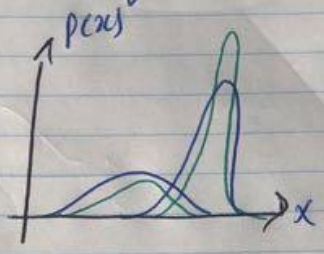
A casual Model

with conditional probabilities tables.

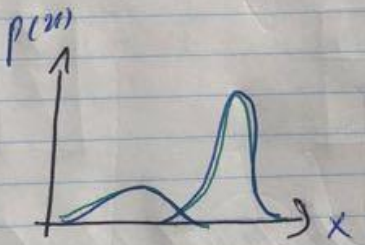

12
BAYES NET: MULTIVARIATE CASUAL MODELS
CASUAL MODELS
- Having many random variables
Example
Battery Age
Battery Dead
Alternator Broken
Fan belt Broken
Not charging
Battery Flat
Lights
Car won't start!
The joined probability table will have 8 random variables & 2^8-1 enteries.
- USE MAP & MLE to predict parameters.
- But just imagine the complexity!!
- We need the model to reason about world

11

- We choose parameters randomly!
- We can assign each data point to the Gau which produces the larger probability, within the current model
- Thus, we are using this hypothesis as ~~recognition model~~ recognition model!



After 3 updates



After 9 updates