Multiple Linear Regression

x	y

Hypothesis's function

$$\hat{y} = mx + c$$

Error/Lost function

$$L = (y - \hat{y})^2$$

$$L(m) = (y - mx)^2 \rightarrow \text{Gradient descent}$$

↓ Minimize → Random Initialization

$m = 0.01$

$\alpha = 0.1$

↓ Derivative calculation

$$\frac{dL}{dm} = -x(y - \hat{y})$$

$\alpha = 0.1$	$w_1 = 0.01$
	$w_2 = 0.03$
	$w_3 = 0.05$
	$w_0 = 0.02$

x_1	x_2	x_3	y

Slope ⇄ weight / parameter : (w)

① Hypothesis function

$$\hat{y} = m_1x_1 + m_2x_2 + m_3x_3 + c$$

$$= w_1x_1 + w_2x_2 + w_3x_3 + w_0$$

Loss function

$$L = (y - \hat{y})^2$$

$$\Rightarrow L(w_1, w_2, w_3) = (y - w_1x_1 - w_2x_2 - w_3x_3 - w_0)^2$$

$$= -2(y - \hat{y})$$

$f(x) = x^2$

$\frac{d}{dx} f = 2x$

$f(x, y) = 3x^2 + 5y^3$

=

* Random Initialization : [θ cannot be zero]

↳ Intercept could be zero.

As Intercept or c doesn't multiply with any other feature.

$w_1 = 0.01$
$w_2 = 0.03$
$w_3 = 0.05$
$w_0 = 0.06$

$\alpha = 0.1$

Derivative calculation

$$\frac{dL}{dw_1} = -2x_1(Y-\hat{Y})$$

$$\frac{dL}{dw_2} = -2w_2(Y-\hat{Y})$$

$$\frac{dL}{dw_3} = -2x_3(Y-\hat{Y})$$

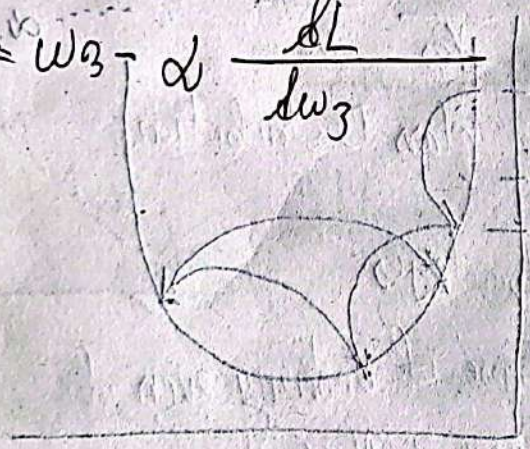
$$\frac{dL}{dw_0} = -2(Y-\hat{Y})$$

Update

$$w_1 = w_1 - \alpha \frac{dL}{dw_1}$$

$$w_2 = w_2 - \alpha \frac{dL}{dw_2}$$

$$w_3 = w_3 - \alpha \frac{dL}{dw_3}$$



$w_1 = 17.5$

$w_2 = 18.9$

$w_3 = 10.5$

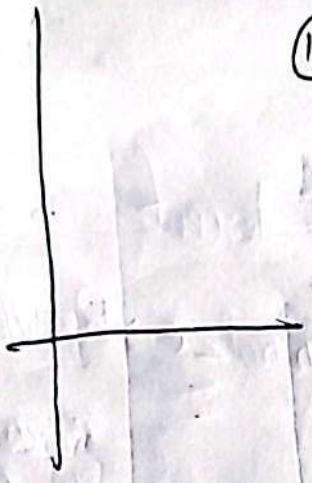
① weight has high value

② Thus value of gradient more

③ Values of update is more on steps are big

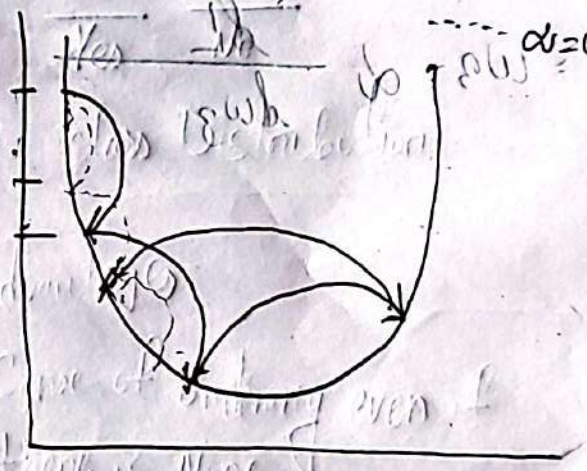
④ Thus errors are unpredictable

loop	Error
1	107
2	10 ↓
3	122 ↓
4	144
5	27



Learning Rate: [start from 100]

Loop 1	E1	E2	E3
1	50	75	08
2	8	50	07
3	48	25	06
4	52	6	05
5	12	58	04



If error sudden changes positive or negative means big α

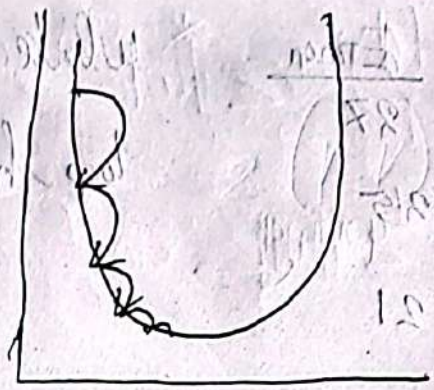
α sudden changes = big α

α change small = small α.

Learning Rate Scheduling

Loop	α
1	0.1
2	0.05
3	0.033

$\alpha = \frac{\alpha_0}{t}$

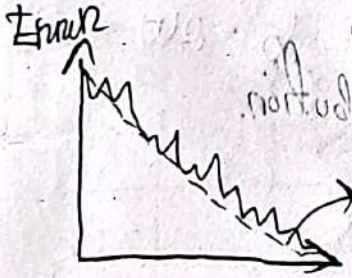


Gradient decent type:

① Stochastic Gradient Descent :

→ If one data is used in every loop. Thus the overall slope is not achieved.

~~When we take the whole~~



② Batch gradient decent: when the whole batch is taken in a loop and after several loops ~~several~~ avg is taken

In real life mini-batch is used as the memory is limited

③ Mini batch gradient decent: Smoother then stochastic rougher then Batch gradient

Loop	Error
1	27
2	35
3	21
4	25
5	26

while sudden increase of error the loop should not be stoped.

* Logistic Regression

x_1	x_2	x_3	y
			Yes
			No
			Yes
			No
			Yes
			No

categorical variable

class

3	2
Yes	No

class Distribution

categorical to Numerical

① Label Encoding : Yes → 1

Dis advantages:

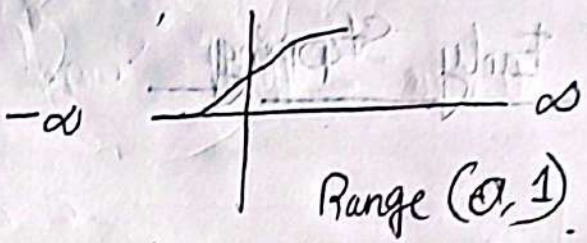
① Sense of ordering even if

Step 1: hypothesis function : $\hat{y} = w_1 x_1 + w_2 x_2 + w_3 x_3 + w_0$

~~Step 2: Loss function i, $L = (y - \hat{y})^2$~~

$\hat{y} = \text{sigmoid}(w_1 x_1 + w_2 x_2 + w_3 x_3 + w_0)$

Sigmoid



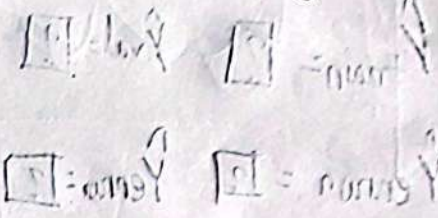
Range (0, 1)

* Threshold = 0.5

$\hat{y} = 0.7 \rightarrow 1$ > Threshold = 1.

$\hat{y} = 0.49 \rightarrow 0$. ≤ Threshold = 0.

Threshold is a value used to determine a decision boundary



Step 2:

$L = -y \log \hat{y} - (1-y) \log (1-\hat{y})$ ← Binary log loss function

Suppose, $y = 0$

$\hat{y}$	0	0.25	0.50	0.90
$-\log(1-\hat{y})$	0	-0.124	0.30	1

L

$y = 1$

$\hat{y}$	1	0.75	0.5	0.25
$-\log \hat{y}$	0	0.12	0.30	0.60

As loss function works to find the distance between actual and prediction, Thus it works the same here.

~~* Sig~~

* Sigmoid function * Binary log-loss function
* Threshold. * Encoding.

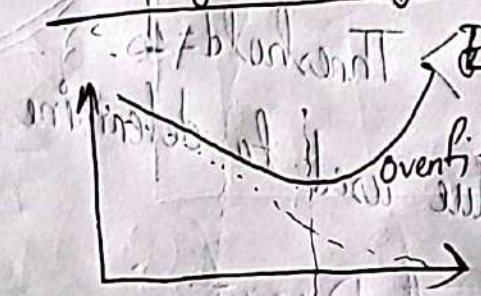
* Early Stopping

Epoch/loop

$\hat{Y}_{train}$ = ? $\hat{Y}_{val}$ = ?

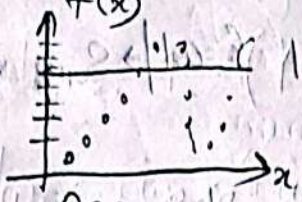
$\hat{Y}_{error}$ = ? $\hat{Y}_{error}$ = ?

Early stopping



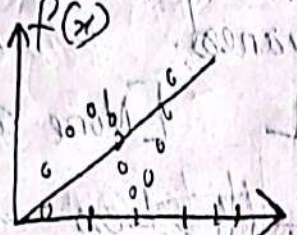
Polynomial Regression:

① $f(x) = 5$



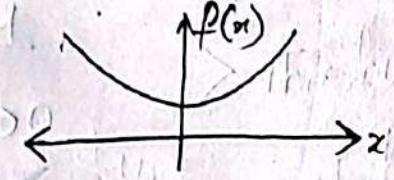
0 degree polynomial

② $f(x) = 2x$



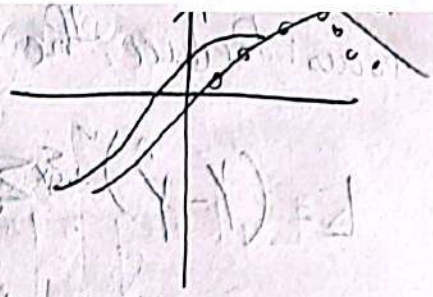
one-degree polynomial

③ $f(x) = x^2$



two-degree polynomial.

④ $f(x)^3 =$



The more power = more curve.

Degree of an Equation = Highest power of that variable.

x_1	x_2	x_3	y

→ 2 degree Polynomial

x_1	x_1^2	x_2	x_2^2	x_3	x_3^2	Target y

$0^\circ = x^0$

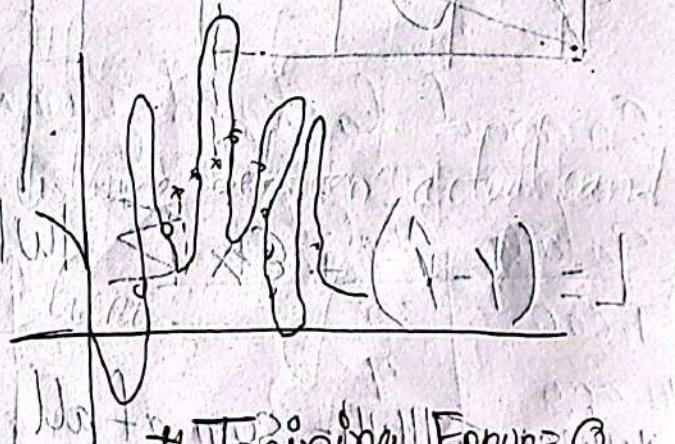
1 deg $= x$

2 deg $= x + x^2$

3 deg $= x + x^2 + x^3$

hypothesis $= \hat{Y} = w_1 x_1 + w_1 x_1^2 + w_2 x_2 + w_2 x_2^2 + w_3 x_3 + w_3 x_3^2 + w_0$

$f(x) = x^{20} + \ldots$



Training Error = 0.

Test Error = ∞.

Overfitting	Underfitting
values of weight ↑	values of weight ↓
↓	model complexity ↓
model complexity ↑	Decrease overfitting
↓	
overfitting ↑	

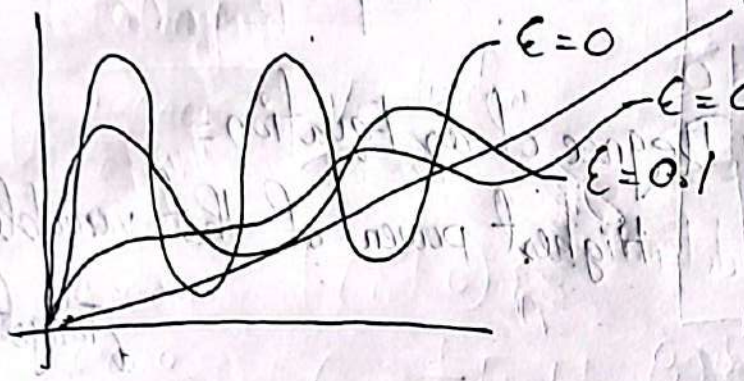
Focus: Reduce the weight.

$\epsilon = 0 \rightarrow$ No penalty $\rightarrow$ complex

$\epsilon = 0-1$ $\epsilon = 0.1 \rightarrow$ Some $n \rightarrow$ Reduce (n)

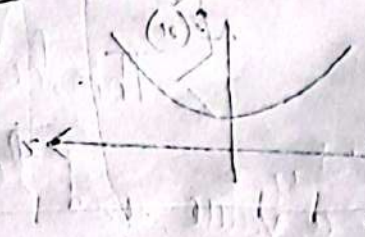
$$L = (Y - \hat{Y})^2 + \epsilon \underbrace{\sum \|w\|^2}_{l_2\text{-penalty}}$$

where, $\sum \|w\|^2$ = Sum of weight squared.

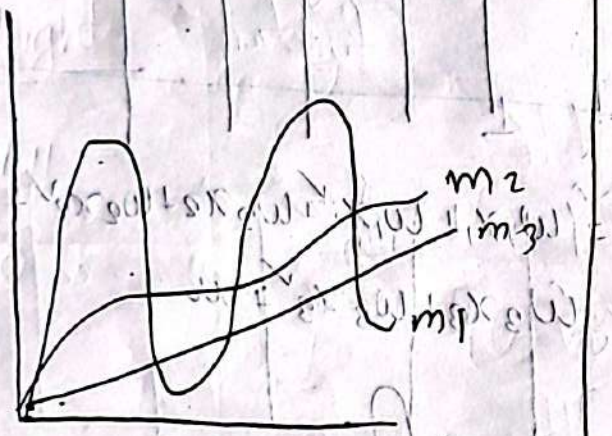


[More complex less ϵ

Less n More ϵ



Q. Based on model arrange based on ϵ.



m_1 has no penalty

$m_1 < m_2 < m_3$

$$L = (Y - \hat{Y}) + \epsilon \times \sum \|w\|$$

where $\sum \|w\|$ = sum of all weights

* l_2-penalty $\rightarrow$ Ridge Regression

* l_1-penalty $\rightarrow$ Lasso Regression

$L = (Y-\hat{Y})^2 + \lambda \sum \|w\|^2 + (1-\alpha) \sum \|w\|$ Here, $0 < r < 1$

l_2 l_1

Using l_2 l_1 together is called Elastic Net.

Regularization
- * l_2 / Ridge Regression
- * l_1 / Lasso Regression
- * Elastic Net.

→ Label Encoding
→ One hot

Feature Eg / Explanatory / Data preprocessing.
Data analysis
↳ log transform
→ Data Scaling [Min Max Normalizer]
→ standardizer.

[Data scaling must for gradient decent]

Feature Eng / Feature Composition: Creating new feature from Existing one.

* Feature Engineering / Exploratory Data Analysis / Data Preprocessing
↳ Data visualization
↳ Histogram
↳ log Transform

→ Exploring & preprocessing

Numeric data:

Salary
1,000
50,000
8,000
10,000
100,000

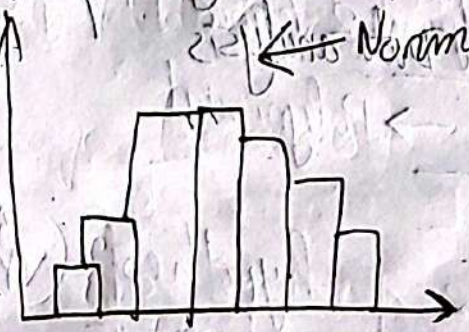
Visualization →

Binning

Height

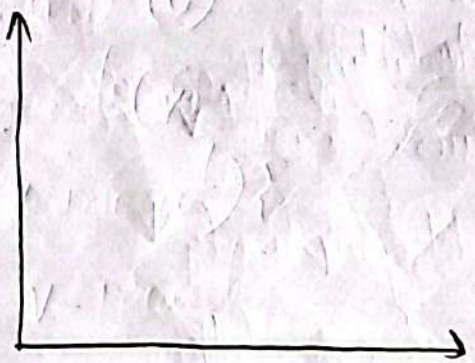
4'11

4'9



In case of skewed column use log, log(salary)

Salary	Log(Sal)
1,000	3
10,000	4
2,000	3.30
1,500	3.17
2,500	3.39
12,000	4.07



$$D_{EU} = \sqrt{(4.11-4.0)^2 + (1000-50{,}000)^2}$$

Different Numeric Feature Have Different Scale/Range.

$$= \sqrt{0.04 + 49{,}000^2}$$

$$=$$

Min Max Normalizer

$$= \frac{x - x_{min}}{x_{max} - x_{min}}$$

Range $[0, 1]$

	x_1	x_2
	5	1000
	10	2000
	15	3000
	20	4000

→

x_1	x_2
0	0
0.03	0.33
0.66	0.66
1	1

Standardizen $\left[x_{new} = \frac{x - x_{mean}}{x_{standard\ dev.}} \right]$

Here centen is "0".

Numinaton brings close to zero.

Denominaton → spread data nicely

* Disjoint Set / Union Find Algorithm
* Kruskal's Algorithm ↓ For identifying disjoint sets.

int parent[6] = {0, 1, 2, 3, 4, 5}

0	1	2	3	4	5

Disjoint set;

A = {2, 3, 5}

B = {1, 4, 6}

A ∩ B = ∅

A and B are disjoint sets.

Union Find

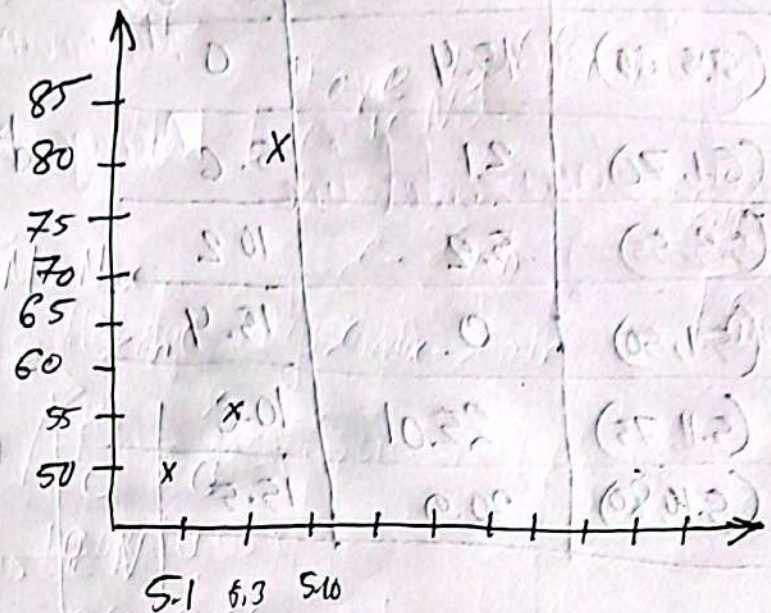
```
int parent (int u) {
    while (parent[u] != u) {
        u = parent[u];
    }
} return u;

int connected (int u, int v) {
    return parent(u) == parent(v);
}

int connect (int u, int v) {
    int u-parent = parent(u);
    int v-parent = parent(v);
    if (u-parent != v-parent) {
        parent[u] = v-parent;
    }
}
```

Clustering

Height	weight
5.5	65
6.1	70
5.3	55
5.1	50
5.11	70
5.10	79
	80



K-means clustering [Possible number of groups].

step 1 : Select the value of "k"

$k = 2$

step 2 : Initialize k centroids ← mean or center point of the clusters.
[start Randomly].

$C_1 = (5.1, 50)$

$C_2 = (5.5, 65)$

step 3 : Calculate Distance from each data to each centroid.

		c_1 (5.1, 50)	c_2 (5.9, 65)
1	(5.5, 65)	15.4	0
2	(6.1, 70)	21	5.6
3	(5.3, 55)	5.2	10.2
4	(5.1, 50)	0	15.4
5	(5.11, 75)	25.01	10.6
6	(5.10, 80)	30.9	15.5

x_1, y_1

x_2, y_2

$d = |x_2 - x_1| + |y_2 - y_1|$

Step 4: Assign each Data to its closest centroid

1	c_2
2	c_2
3	c_1
4	c_1
5	c_2
6	c_2

c_1 updated x position =

Average of previous positions.

Step 5: Updated centroid

$$c_1 = \left(\frac{5.1 + 5.3}{2}, \frac{50, 55}{2}\right)$$

$$= (5.2, 52.5)$$

$$C_2 = \frac{5.5+6.1+5.10+3.11}{4}, \frac{65+70+75+8}{4}$$

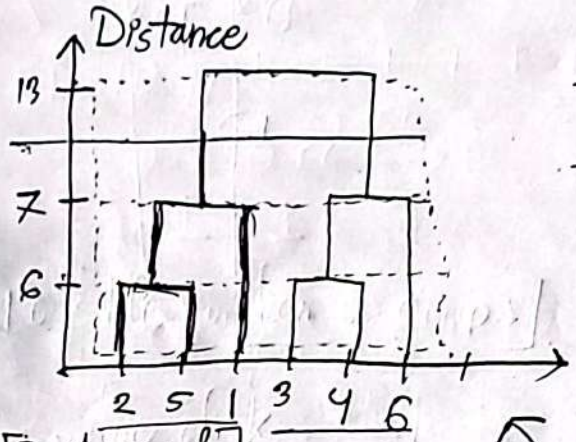
$$= (5.5, 72.5)$$

	1,2,3,4,5,6
1,2,3,4,5,6	

* Hierarchical / Agglomerative / clustering.

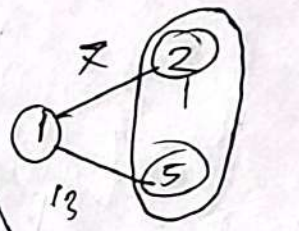
→ Initially, each data is an individual cluster.

	x_1	x_2
1	5	2
2	10	4
3	25	8
4	30	9
5	15	5
6	35	11



[Dendrogram]

	1,2,5	3,4,6
1,2,5	0	
3,4,6	13	0



1. Minimum
2. Maximum
3. Average.

Distance Matrix

	1	2	3	4	5	6
1	0	7				
2	7	0				
3	26	10	0			
4	32	25	6	0		
5	13	6	13	10	0	26
6	30	32	13	7	26	0

x_1, y_1

x_2, y_2

$$\varphi = |x_2 - x_1| + |y_2 - y_1|$$

=

	1	(2,5)	3	4	6
1	0				
(2,5)	7	0			
3	26	13	0		
4	32	10	6	0	
6	30	26	13	7	0

	(1,2,5)	3,4	6
(1,2,5)	0		
(3,4)	13	0	
6	26	7	0

	(2,5)	(3,4)	
1	3,4	6	
(2,5)	0		
(3,4)	7	0	
6	26	13	0
	30	26	7	0

Dendogram

↳ Rectangular area

↳ but no horizontal line.

Minimum ⇒ Single Linkage

Maximum → Complete Linkage

Average → Average Linkage

Minimum