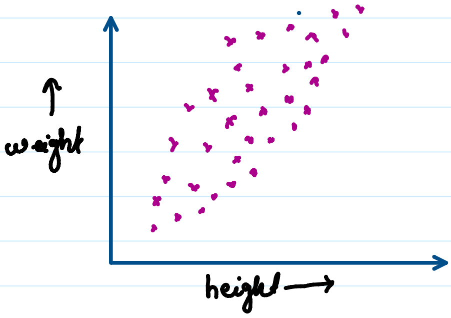
Linear Regression

12 April 2022   09:02 PM

Assumption: x and y has a linear Relation



Linear Regression

- Easy to Learn → Intuition
- Application in other Algo

$\boxed{\text{LR}}$ (Supervised) < Regression / classification

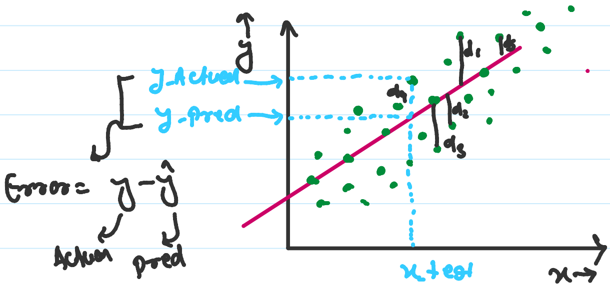
- **Linear LR** (when data is linear)
  - **Simple**: 1 - input, 1 - Output
  - **Multiple**: 1 < input, 0 - output.
- **Polynomial LR** (when data is not linear)

Ex - covid data.

Linear Regression

→ Aim is to find best fit line → Minimum Error.

Error = Actual value − Prediction value.



$$\text{Error} = d_1 + d_2 + d_3 + \dots d_n$$

$$= (y_1 - \hat{y}_1) + (y_2 - \hat{y}_2) + (y_3 - \hat{y}_3) + \dots$$

$$= \sum_{i=1} (y_i - \hat{y}_i)$$

But to manage +ve and −ve Error we have to square the Error.

$$\boxed{\therefore \text{Error}(\hat{y}) = \sum_{i=1} (y_i - \hat{y}_i)^2} \quad \text{— eq (1)}$$

(Loss function)

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

→ by putting in eq (1)

$$\boxed{\text{Loss Function } (J) = \sum_{i=1}^{n} (y_i - mx_i - c)^2}$$

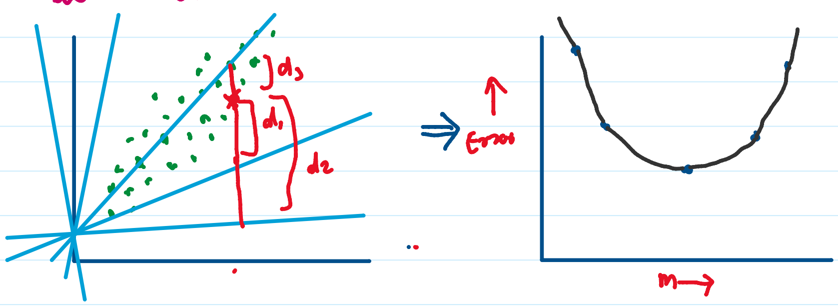
Here,

$y_i, x_i \longrightarrow$ Const, we get in training time

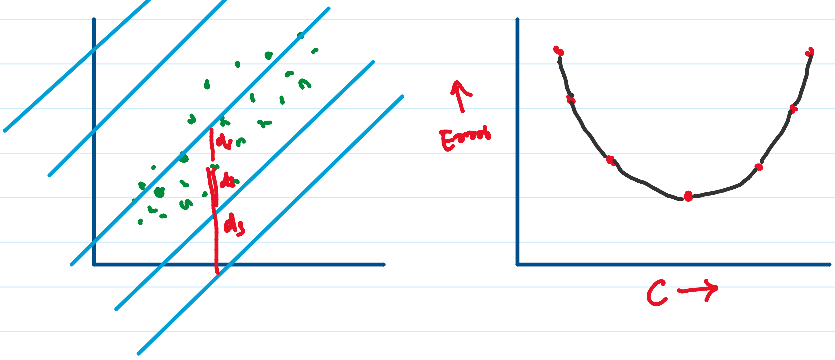
$m, c \longrightarrow$ variable, we have to calculate.

Let's Analyse the M and C w.r.t Loss-function

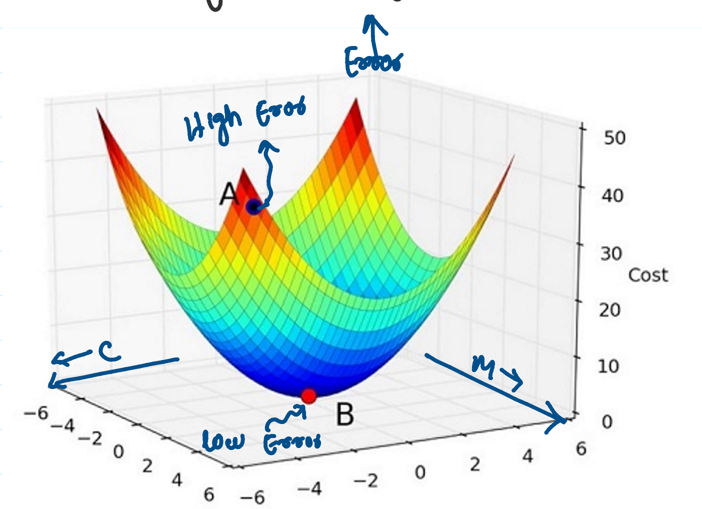
Relation bet M & loss-fun. (C = const)



Relation bet C and loss-fun (M = const)



If we Analyse M and C both together we will get the figure like below.



we can find the point of low Error by differentiating the loss Function we can find the point of low error by differentiating the loss function

Since for minima $\longrightarrow$ $f'(x) = 0$

$$f(x) = \text{loss function}(J) = \sum_{i=1}^{n} (y_i - mx_i - c)^2$$

variable

Since we have 2 variable we have to do partial derivative $\longrightarrow$ $\frac{dJ}{dc} = 0$ and $\frac{dJ}{dm} = 0$

$$\frac{dJ}{dc} = \frac{d}{dc}\left(\sum (y_i - mx_i - c)^2\right) = 0$$

$$\Rightarrow \sum \frac{d}{dc}(y_i - mx_i - c)^2 = 0$$

$$\Rightarrow \sum 2(y_i - mx_i - c) \times -1 = 0$$

$$\Rightarrow \sum (y_i - mx_i - c) = 0$$

$$\Rightarrow \sum y_i - \sum mx_i - \sum c = 0$$

$$\Rightarrow \sum y_i - \sum mx_i = \sum c$$

$$\Rightarrow \frac{\sum y_i}{n} - \frac{\sum mx_i}{n} = \frac{\sum c}{n} \quad \text{(devide by } n \text{ both side)}$$

$$\Rightarrow \bar{y} - m\bar{x} = c \quad , \quad \bar{x}, \bar{y} \rightarrow \text{mean}$$

$$\Rightarrow \boxed{c = \bar{y} - m\bar{x}} \quad \text{eq 2}$$

2) $$\frac{dJ}{dm} = \frac{d}{dm}\left(\sum (y_i - mx_i - c)^2\right) = 0$$

$$\Rightarrow \frac{d}{dm}\left(\sum (y_i - mx_i - \bar{y} + m\bar{x})^2\right) = 0$$

$$\Rightarrow \sum \frac{d}{dm}(y_i - mx_i - \bar{y} + m\bar{x})^2 = 0$$

$$\Rightarrow \sum 2(y_i - mx_i - \bar{y} + m\bar{x}) \times (-x_i + \bar{x}) = 0$$

$$\Rightarrow \sum \left(y_i - mx_i - \bar{y} + m\bar{x}\right) \times \left(x_i - \bar{x}\right) = 0$$

$$\Rightarrow \sum \left(y_i - \bar{y} - m(x_i - \bar{x})\right) \times \left(x_i - \bar{x}\right) = 0$$

$$\Rightarrow \sum \left[(y_i - \bar{y})(x_i - \bar{x}) - m(x_i - \bar{x})^2\right] = 0$$

$$\Rightarrow \sum m(x_i - \bar{x})^2 = \sum (y_i - \bar{y})(x_i - \bar{x})$$

$$\Rightarrow \boxed{m = \frac{\sum (y_i - \bar{y})(x_i - \bar{x})}{\sum (x_i - \bar{x})^2}} \quad \text{eq3}$$

Finally we got our m and c

now we can calculate y-pred. by

✓ $\boxed{\hat{y} = mx + c}$

we got from training data

y-pred x-test

Multivariant LR

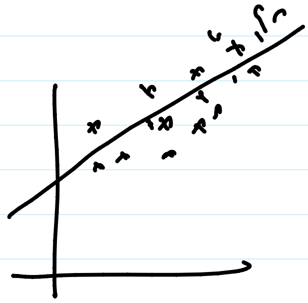
input > 1
output - 1

$\boxed{y = mx + c}$

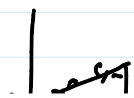
weight offset

$\boxed{y = w_1 x + w_0}$

c1	c2	c3	c4	c5

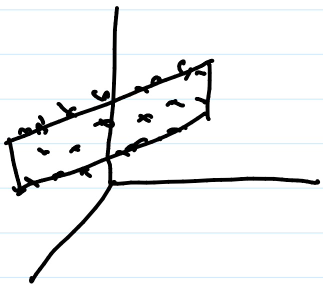


Data		Reg. line
✓ 2D	⟶	1D → line
3D	⟶	2D → plane



✓ 2-D ⟶ 1-D → line

$$\begin{cases} 3D \longrightarrow 2D \to \text{plane} \\ nD \longrightarrow (n-1)D \to \text{hyperplane} \end{cases}$$



$2D \to \underset{1D}{\underline{\text{line}}} \to \hat{y} = mx + c$

$\quad = \underline{w}x + w_0$

$3D \to \underset{2D}{\underline{\text{plane}}} \to \hat{y} = m_1x_1 + m_2x_2 + c$

$\quad = \underline{w_1}x_1 + \underline{w_2}x_2 + w_0$

$\underline{(n+1)D} \to \underset{(nD)}{\underline{\text{hyperplane}}} \to \hat{y} = (w_1x_1 + w_2x_2 + w_3x_3 \; ..... \; w_nx_n) + w_0$

$$= [w_1, w_2 \cdots w_n] \cdot \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} + w_0$$

$$\boxed{\hat{y} = w^T \cdot x + w_0}$$

$$[1, 2, 3, 4] \cdot \begin{bmatrix} 10 \\ 20 \\ 30 \\ 10 \end{bmatrix}$$

$\boxed{10 + 40 + 90 + 40}$

$$[w_1 \; w_2 \cdots w_n] \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{bmatrix}$$

$w_1x_1 + w_2x_2 + w_3x_3 \; ... \; w_nx_n$

$$w = \begin{bmatrix} w_1 \\ w_2 \\ w_3 \\ w_n \end{bmatrix} \longrightarrow w^T = [w_1 \; w_2 \; w_3 \; ...]$$

$$x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_n \end{bmatrix}$$

$1D \to \hat{y} = mx + c \qquad m \to \text{Scaler}$

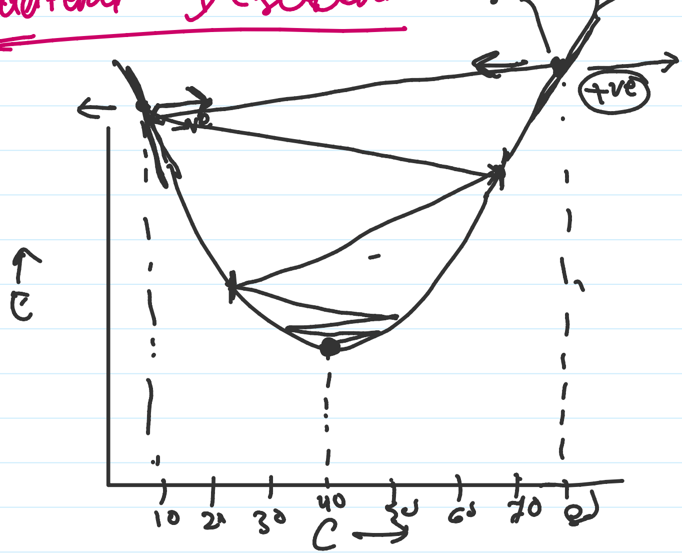
$nD \to \hat{y} = W^T x + w_0 \qquad W^T \to \text{Vector}$

$\qquad\qquad\qquad\qquad\qquad c, w_0 \to \text{Const.}$

we have to find

Gradient Descent

loss funct = L



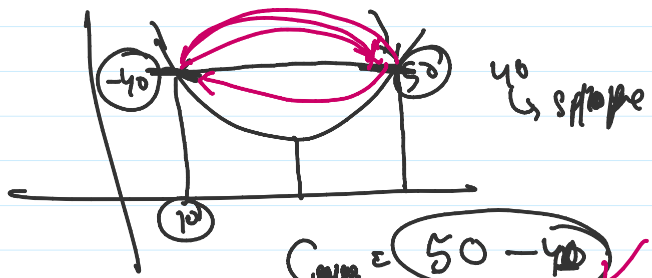
$$C_{new} = C_{old} - \text{slope}$$

slope → $\dfrac{dL}{dC}$

$$C_{new} = 80 - 70$$
$$= 10$$

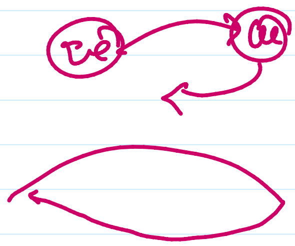
$$C_{new} = 10 - (-60)$$
$$= 70$$

$$C_{new} = 70 - 50$$
$$80 - (-40)$$



$$C_{new} = 50 - 40$$

$$C_{new} = 10 - (-40)$$
$$= 50$$



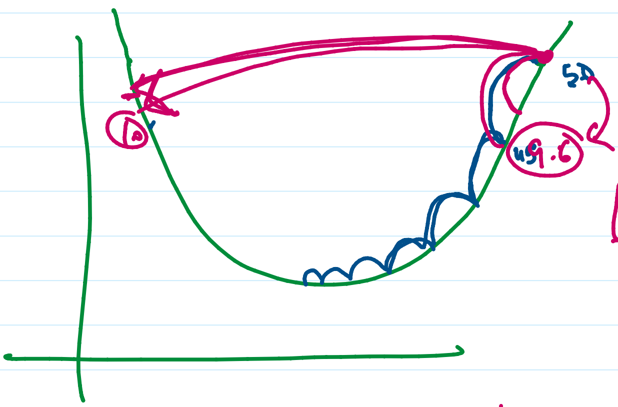
$$C_{new} = C_{old} - \lambda\ \text{slope}$$

↳ learning rate → 0.01

↳ 0.1

↳ 0.001

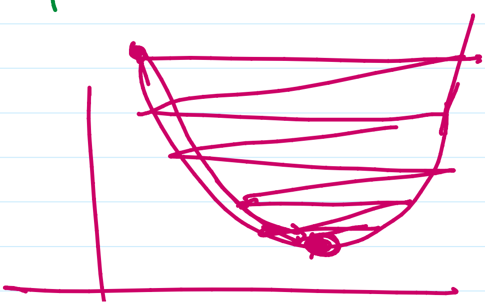
↳ 0.0001

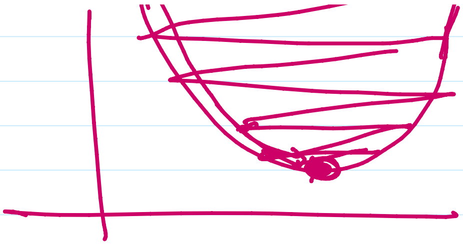


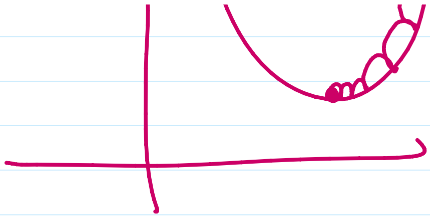
$50 - 40 \times 0.01$

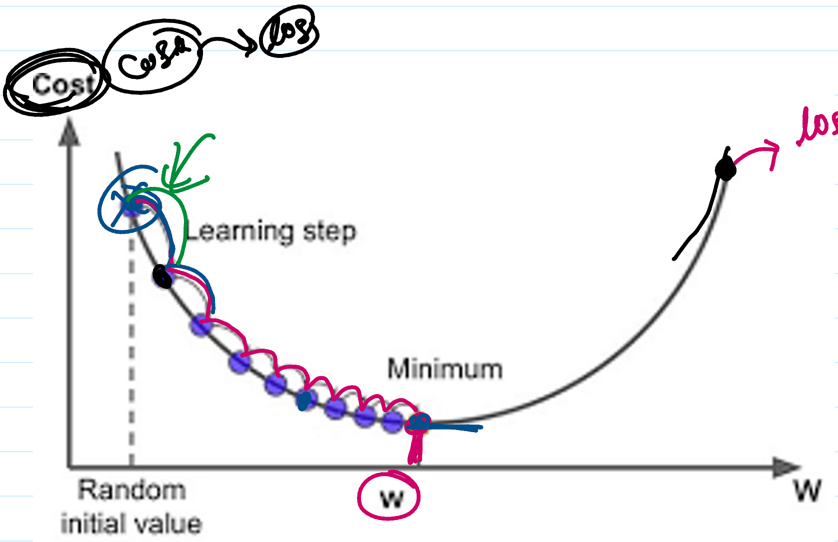
$\boxed{49.6}$

4×0.4









$$w_{new} = w_{old} - \lambda \times \text{slope}$$

slope $\rightarrow \dfrac{dL}{dw}$

$$\boxed{y = w^T x + w_0}$$

$$\text{loss} = \sum (y - \hat{y})^2$$

$$\text{loss} = \sum (y - w^T x - w_0)^2$$

$$\text{Cost} = \frac{1}{n} \sum (y - \hat{y})^2$$

$$= \frac{1}{n} \sum (y - w^T x - c)^2$$

Assumption

$$w_{new} = w_{old} - \text{slope} \times \lambda$$

$$\frac{d\,\text{cost}}{dw} = \frac{1}{n} \sum 2 (y - w^T x - c) \times (-x)$$

$$\boxed{\frac{d\,\text{cost}}{dw} = \frac{1}{n} \sum -2x (y - w^T x - c)}$$

Math

$$\frac{d\,cost}{d\omega}$$

→ ω_{new} → Ans

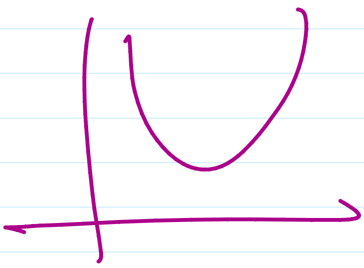
$$\frac{d\,cost}{dC} = \frac{1}{n}\sum 2\,(y-\omega^T x - c)\,(-1)$$

→

$$\boxed{\frac{d\,cost}{dC} = \frac{1}{n}\sum -2\,(y-\omega^T x - c)}$$

Gradient des

$$\boxed{\omega_{new} = \omega_{old} - \lambda \times slope}$$

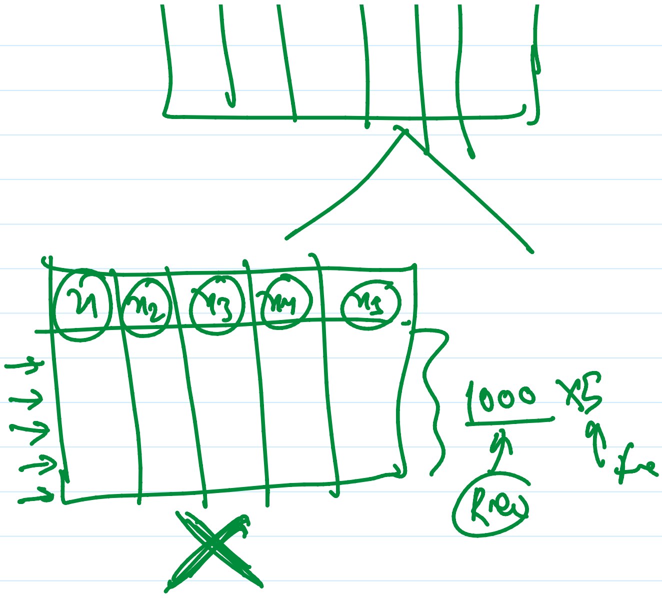


$$\lambda = 0.01$$

$$x = \begin{bmatrix} x_2 \\ x_3 \end{bmatrix} \begin{matrix} \to col2 \\ \to col3 \end{matrix}$$

c1	2	3	4	5	y

[5]



$$\cancel{X} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}$$

$$\frac{d\,\text{cost}}{dc} = \frac{1}{n} \sum -2 \left(y - w^T x - c \right.$$