Data Science & Machine Learning

~Soumyansh

Simple Linear Regression →

Supervised Machine Learning →

Dataset →

I/P feature → $\boxed{\text{Weight}}$ | $\boxed{\text{Height}}$ → Dependent feature

74 kg | 170 cm ✓

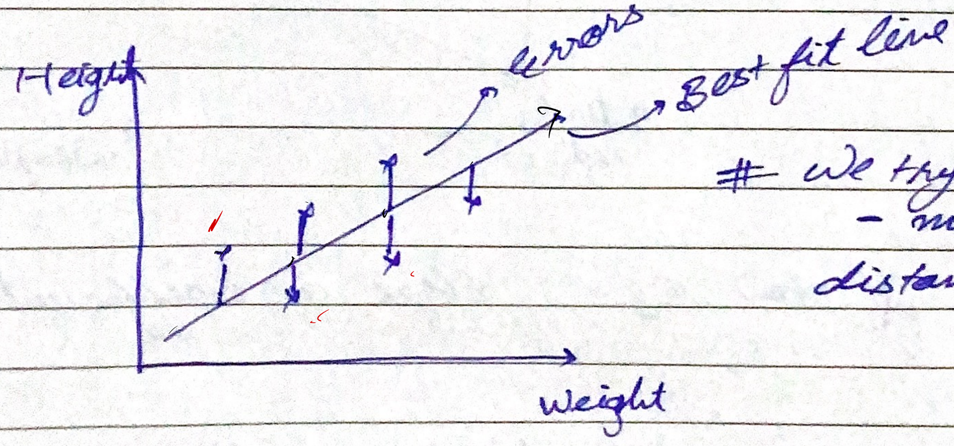
80 kg | 180 cm ✓

75 kg | 175.5 ✓

\- | -

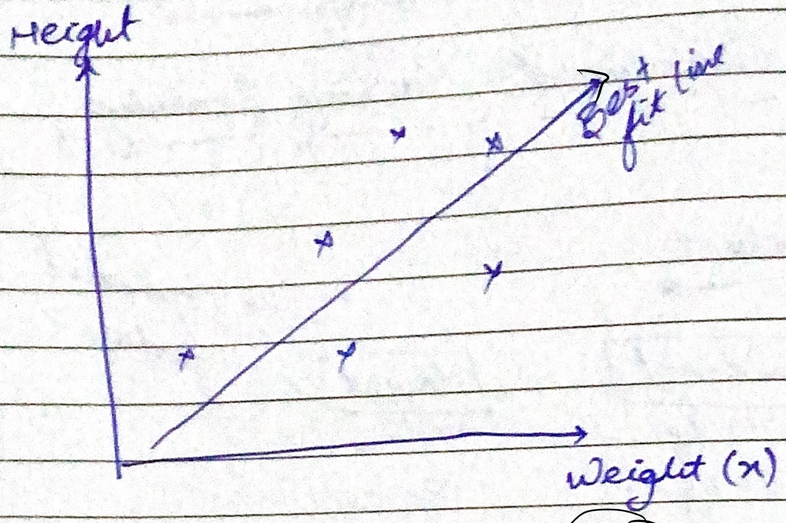
Past Data

New weight → $\boxed{\text{Model}}$ (Training) → Height



\# We try to mini-mise the distance.

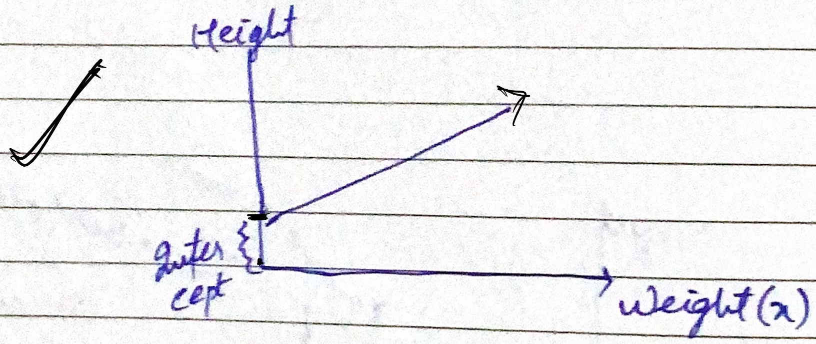
$\boxed{Y = mx + c}$



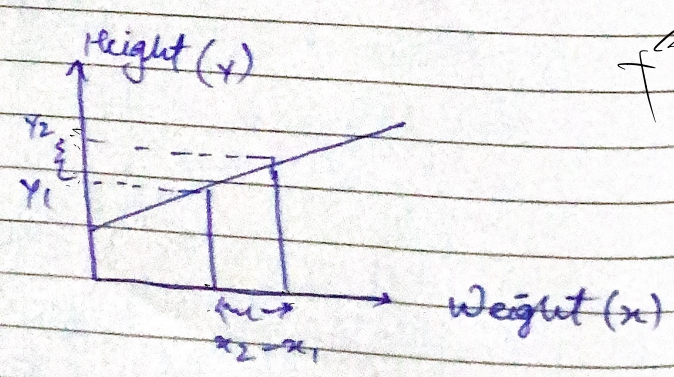
$$h_\theta(x) = \theta_0 + \theta_1 x$$

Imp Now θ_0 = Intercept

If $x = 0$ then $h_\theta(x) = \theta_0$



Imp Now θ_1 = Slope or coefficient



$Y = mx + c$

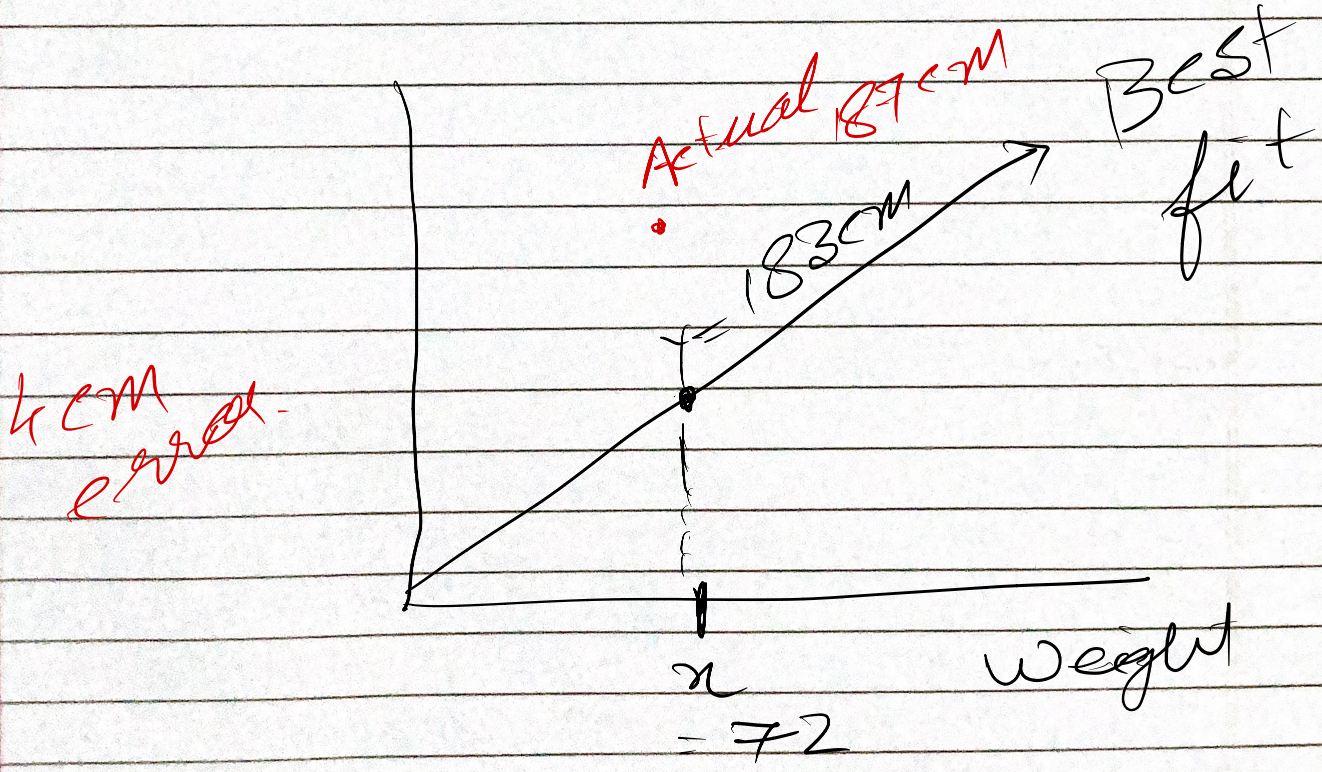
$\frac{Y_2 - Y_1}{x_2 - x_1}$

Error = $y - \hat{y}$ [where $\hat{y}$ = predicted value of y by model]

Now our aim is to minimise the sum of all errors

$$\boxed{\sum_{min} \{y - \hat{y}\}}$$

$\hat{Y}$ = Y predicted



Machine Learning → Notes Part 2

$\boxed{\text{Cost function}}$

$\boxed{\text{Gradient Descent}}$ → An algorithm to minimize a function by optimizing its parameters

Let's say there is a Science Test Max marks = 50

Friend asks you to guess his marks?
Guess 1 → 45 ?
friend → Too far
Guess 2 → 40 ?
friend → Still far
Guess 3 → 37
friend → Very close

$\boxed{Y = mx + c}$

Imp → In GD we start with random guess & then slowly move to the right answer.
How fast / slow we'll converge to the answer is determined by learning rate

$\boxed{\text{New value} = \text{Old value} - \text{Stepsize}}$

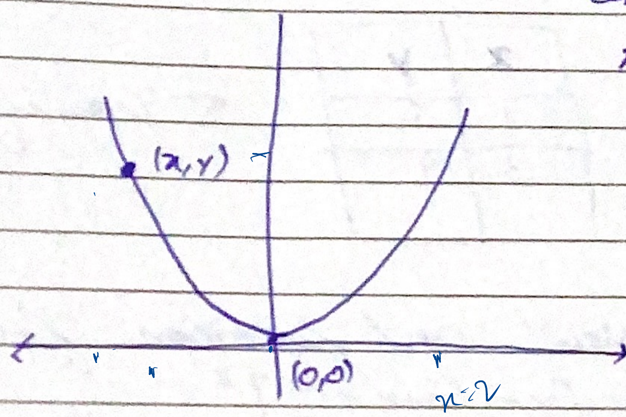
where Stepsize = Learning rate × slope

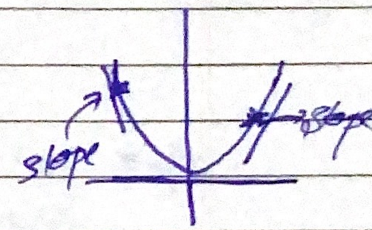
Date
Page

Ex → A simple square function

$f(x) = x^2$

Now as per GD this function should be minimised.





Now step 1) Random guess (−2, 3) (x & y)

Step 2 → Here $x = -2$ then $f(x) = x^2$ &
Slope of function $= \frac{d}{dx} f(x) = 2x = 2(-2) = -4$

Step 3 → If $x = -1$ then $2x$ ie slope $= -2$
Hence we are getting close to zero ie moving in correct direction

Step 3 → If $x = +2$ then $2x$ ie slope $= 4$ which is beyond zero so we're in wrong direction now

Step 4 → Now $\boxed{\text{new value} = \text{old value} - \text{slope} \times LR}$

& if a function has multiple parameters unlike $f(x) = x^2$, ie function having more than one parameter.

Ex Cost func^n for regression =

$$J(m, c) = \sum_{i=1}^{n} (y_i - (mx_i + c))^2$$

Date
Page

Step5 → Now there are two params m & c

Step6 → Again random guess let's say
{ c=0 , m=1 }

x	y
1	2
3	4

← Training Data

Step7 → Now cost function for linear regression
$= [y-(mx+c)]^2$

Plugging the values for training data (1,2) & (3,4)

$J(m,c) = [2-(c+m\times 1)]^2 + [4-(c+3m)]^2$

partial derivative
$\frac{dJ}{dc} = -2[2-(c+m)] - 2[4-(c+3m)]$
for $c=0$ & $m=1$

↓
differentiation wrt c
$= -2[2-1] - 2[4-3]$
$= -4$

Step8 Now because i differentiated wrt c
so my
new c = old c − LR × (−4) ↗ slope
$= 0 - (0.001) \times (-4)$
$= 0.004$

Step9 Similar to above now we'll differentiate wrt m & new value of m will be found

Step 10 → Now new values will be taken for consideration & entire process will get repeated

Step 11 → When algo will stop?
Ans → When there won't be much improvement on cost for values of c & m

Imp → Choosing learning Rate: Try to be moderate

Machine Learning Notes → 3

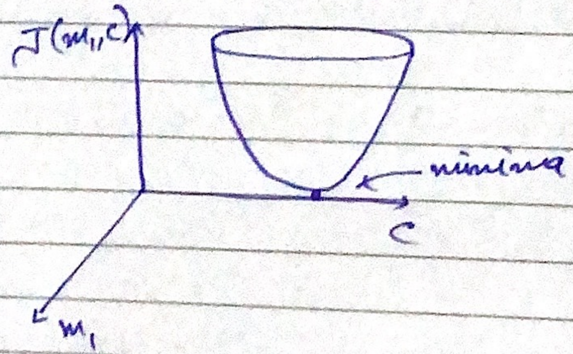
Multiple Linear Regression →

House pricing Dataset →

No. of Rooms	Size of house	Location	Price
1	1800	X	100000 $
2	1900	Y	200000 $
3	2000	Z	1000000 $

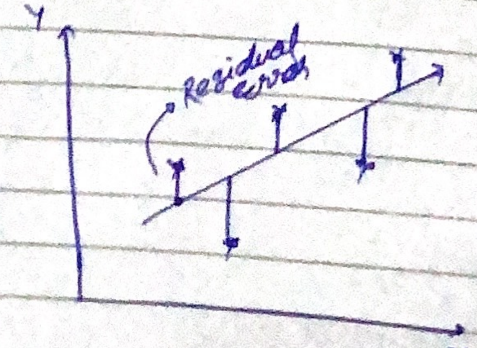
$$Y = m_1x_1 + m_2x_2 + m_3x_3 - - - - + m_nx_n + c$$

Example curve. Multiple I/P features →



Performance Metrics used in Linear Regression →

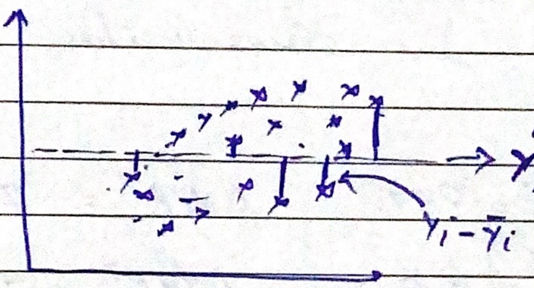
1) R Squared $= 1 - \frac{SS_{Residual}}{SS_{Total}}$



$\nearrow$ Y true value $\quad \nearrow$ Y predicted

SS Residual (Sum of Square Residual) $= \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$

SS Total (Sum of square Total) $= \sum_{i=1}^{n} (y_i - \bar{y}_i)^2$



$\rightarrow Y_{Avg}$ $\nearrow$ or $\bar{Y}$

$Y_{Avg} = \bar{Y}$ = Average of all true points of Y

$$\therefore \boxed{R\ squared = 1 - \frac{\sum_{i=1}^{n} (y_i - \hat{y}_i)^2}{\sum_{i=1}^{n} (y_i - \bar{y}_i)^2}}$$

$1 - \frac{\text{Small Number}}{\text{Big Number}} \Rightarrow 1 - \text{Small Number} \approx 1$

0.70 $\rightarrow$ 70% Accuracy

0.85 $\rightarrow$ 85% " & so on

2) Adjusted R squared $\rightarrow$ Extra feature (Owner DOB)

$$1 - \frac{(1-R^2)(N-1)}{N-P-1}$$

N = no. of data points

P = no. of independent features

Dataset

No. of Rooms	Size	Location	Owner DOB	Price
—	—	—	—	—

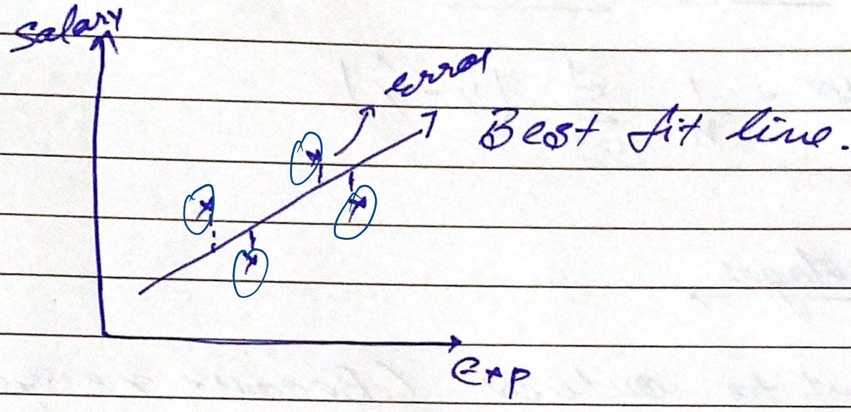
$$\left.\begin{matrix} R^2 \uparrow \\ \text{Adj } R^2 \downarrow \end{matrix}\right\}$$ via penalty through denominator.

Ex $p=2$ $R^2 = 80\%$ $\text{Adj}\,R^2 = 76\%$

$p=3$ $R^2 = 85\%$ $\text{Adj}\,R^2 - 79\%$

(Penalty of non correlated feature)

Machine Learning Notes → 4



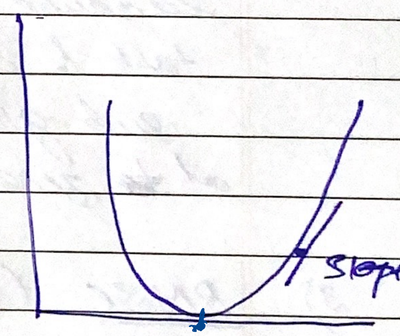
→ When we want to focus on Error metrics.

1) Mean Squared Error $\boxed{MSE}$

$$MSE = \frac{\sum_{i=1}^{n} (y_i - \hat{y}_i)^2}{n} \rightarrow \text{Cost function}$$

Advantages →

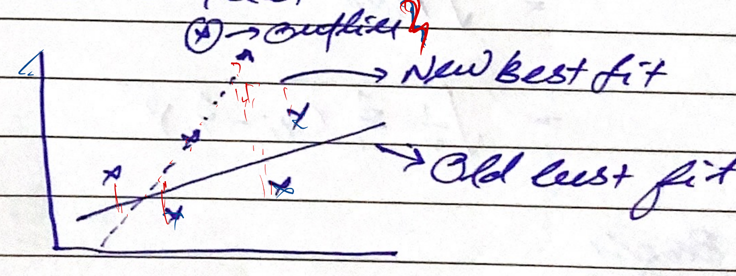
1) Differentiable at all points
2) Converges faster.



Disadvantages →

1) Not robust to outliers

∵ of this MSE will increase.



2) Output is not in the same unit.

Ex let's say salary error = 5000

then MSE will be $(5000)^2$ squared

$\boxed{MAE}$

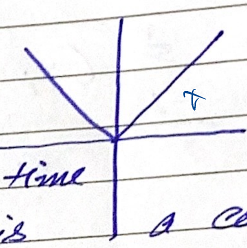
2) Mean Absolute Error (MAE)

$$MAE = \frac{1}{n} \sum_{i=1}^{n} |y - \hat{y}|$$

Advantages →

① Robust to outliers (Because no squaring like MSE

② O/p will be in the same unit



Graph

Disadvantage →

1) Convergence takes more time because optimisation is a complex task & differentiables are happening at sub gradients & no differentiation at ~~to~~ zero

3) RMSE (Root mean squared error) $\boxed{RMSE}$

$$= \sqrt{MSE}$$

$$= \sqrt{\frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2}$$

Benefit →

1) Same unit because of Sqrt

2) Differential

Disadvantage

1) Not robust to ~~~~ outliers

Machine Learning Notes - 5

Overfitting & Underfitting

1) Training dataset
2) Testing dataset
3) Validation dataset

Dataset 1000 points

70% Training — 700 points

30% Testing — 300 points.

Training Dataset

[Train] [Validation]

(for hyperparameter tuning)

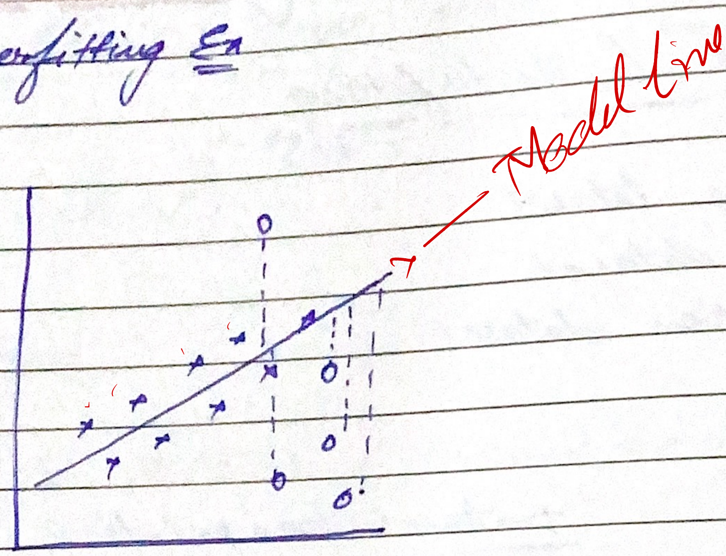
Overfitting

Very good accuracy with training data ie 90%.
Bad accuracy with test data, Ex → 50%.

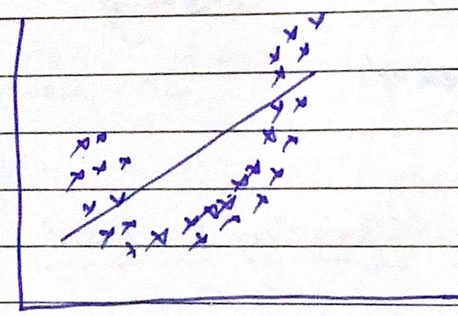
Underfitting.

Training score is low (50%)
Testing score is low as well (50%)

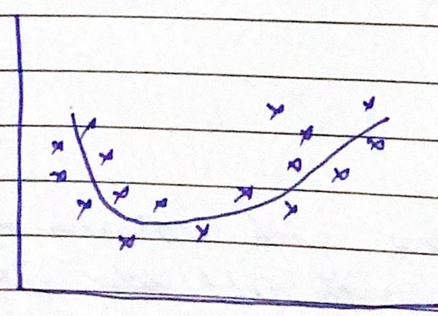
Overfitting Ex



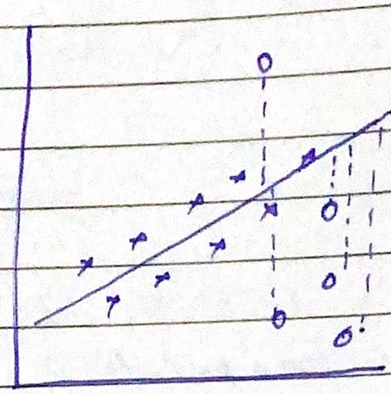
Underfitting Ex



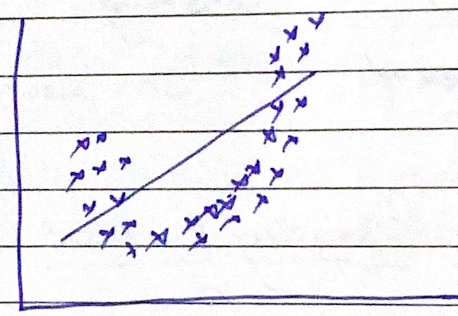
Balanced fit / Good fit



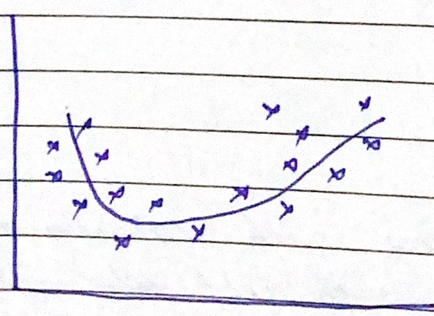
Overfitting Ex



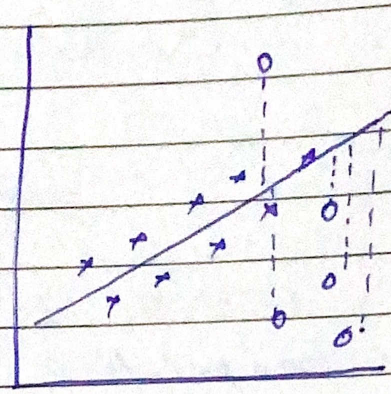
Underfitting Ex



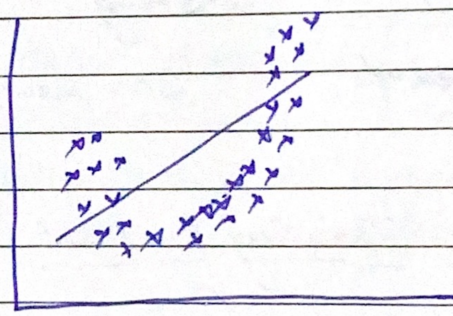
Balanced fit / Good fit



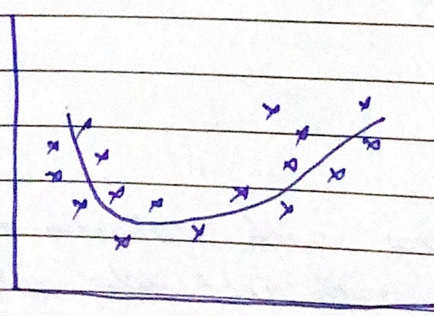
Overfitting Ex



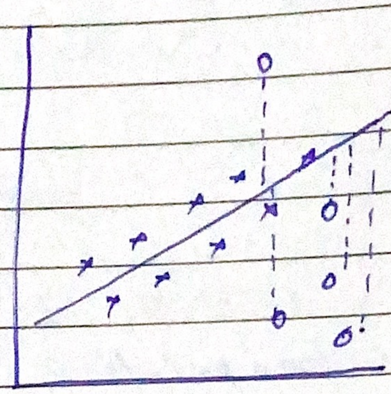
Underfitting Ex



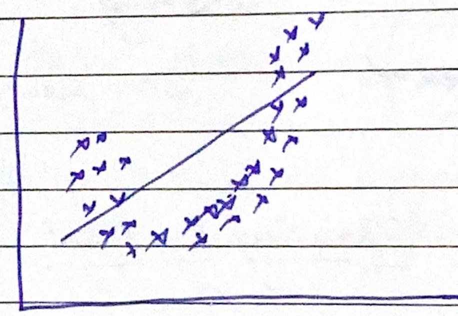
Balanced fit / Good fit



Overfitting Ex



Underfitting Ex



Balanced fit / Good fit