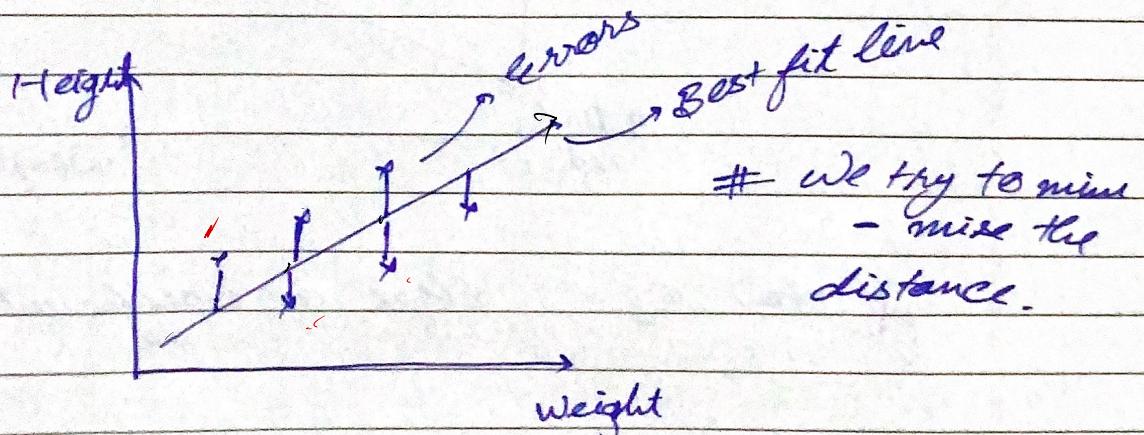
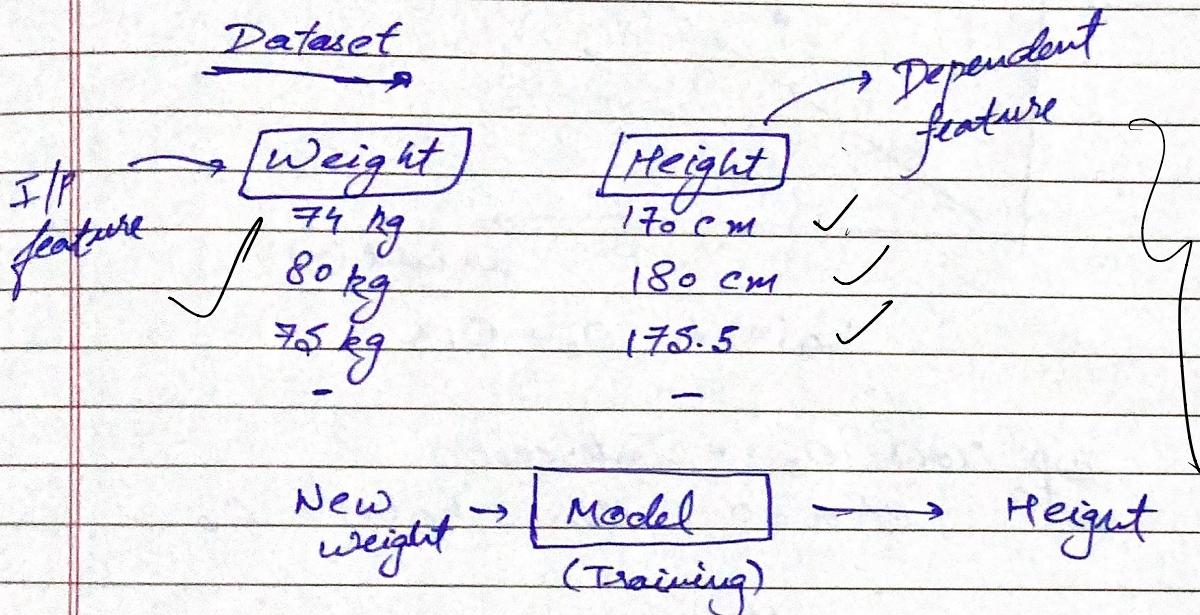
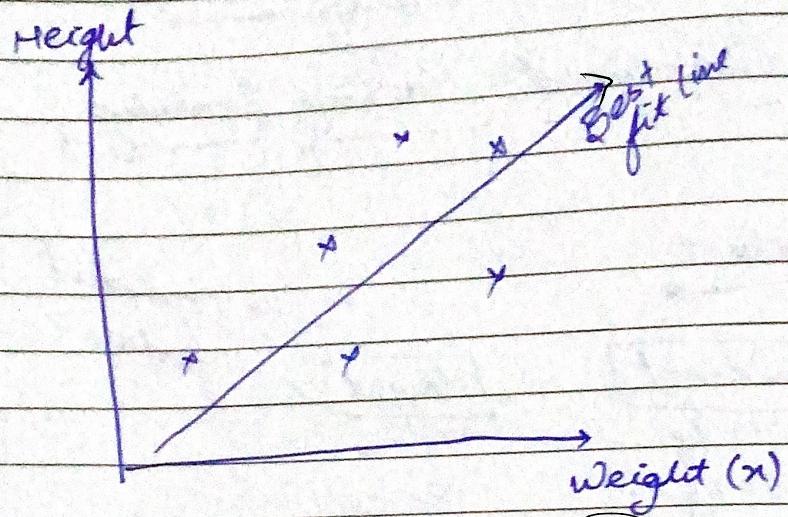


Data Science
Machine Learning
~Somyanush

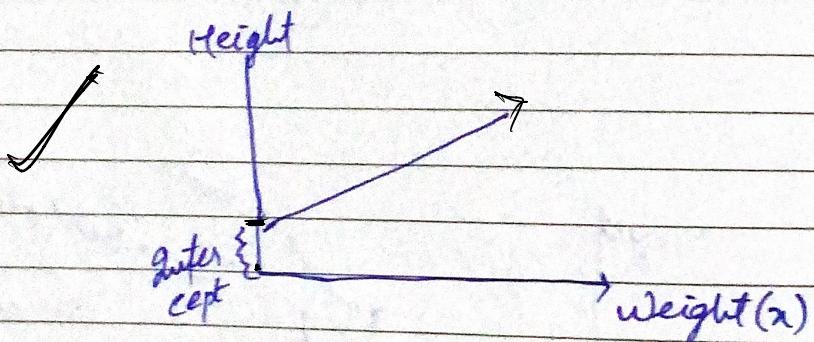
Simple Linear RegressionSupervised Machine Learning

$$y = mx + c$$

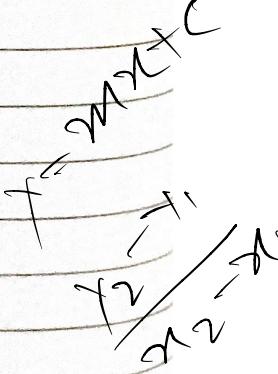
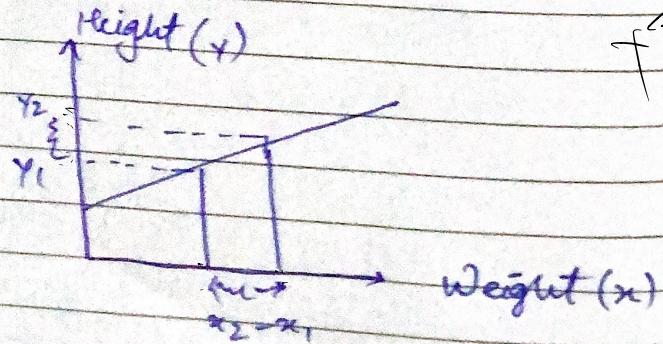


$$h_0(x) = \theta_0 + \theta_1 x$$

~~If~~ New θ_0 = Intercept
~~If~~ $x=0$ then $h_0(x) = \theta_0$



~~If~~ New θ_1 = slope or coefficient

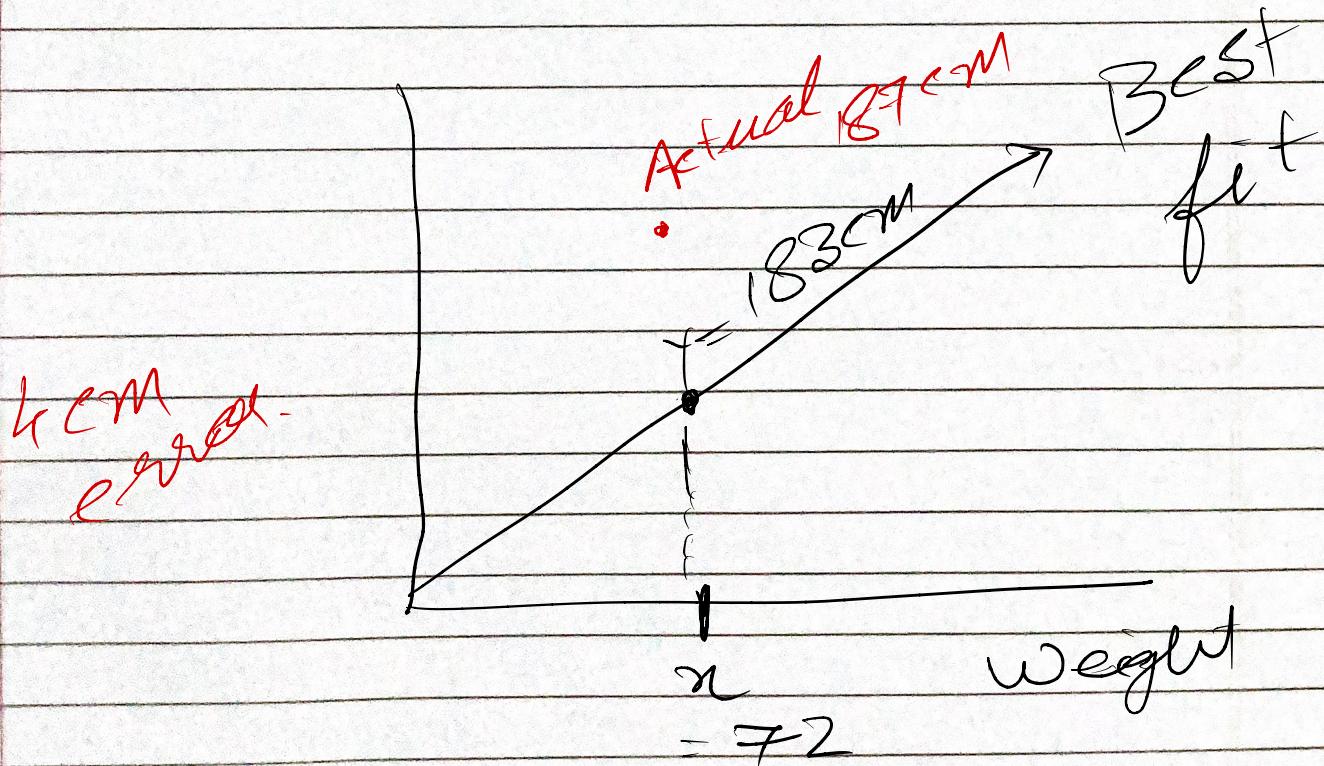


Errors = $y - \hat{y}$ [where \hat{y} = predicted value of y by model]

Now our aim is to minimise the sum of all errors

$$\min \left[\sum (y - \hat{y})^2 \right]$$

$$\hat{y} = \text{predicted}$$



Machine Learning → Notes Part 2

Cost function

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?

guess1 → 45 ?

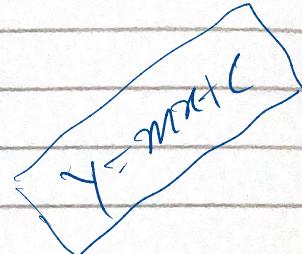
friend → Too far

Guess2 → 40 ?

friend → Still far

guess3 → 37

friend → Very close



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

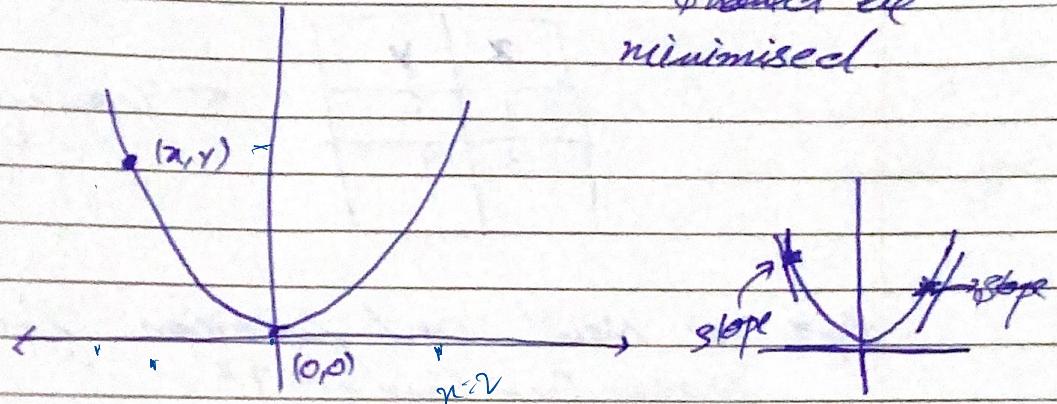
$$\text{New value} = \text{Old value} - \text{Stepsize}$$

where StepSize = Learning rate × Slope

Eg → A simple square function

$$f(x) = x^2$$

Now as per GD this function should be minimised.



New step 1) Random guess (-2, 3) ($x \& y$)

Step 2 → If $x = -2$ then $f(x) = x^2$ &

$$\text{Slope of function: } \frac{d}{dx} f(x) = 2x = 2(-2) = -4 \checkmark$$

Step 3 If $x = -1$ then $2x$ ie slope = -2 ✓

Hence we are getting close to zero ie moving in correct direction

Step 4 → 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.

Eg Cost funcⁿ for regression =

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

Step 5 → Now there are two params in b.c.

Step 6 → Again random guess let's say
 $c=0, m=1$

x	y
1	2
3	4

← Training Data

Step 7 → Now cost function for linear regn
 $= [y - (mx + c)]^2$

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

$$J(m, c) = [2 - (c + mx_1)]^2 + [4 - (c + mx_2)]^2$$

partial derivative

$$\frac{\partial J}{\partial c} = -2[2 - (c + mx_1)] - 2[4 - (c + mx_2)]$$

for $c=0 \& m=1$

$$= -2[2 - 1] - 2[4 - 3]$$

$$= -4$$

Step 8 Now because i differentiated wrt c

so my

$$\text{new } c = \text{old } c - LR \times (-4) \quad \text{slope}$$

$$= 0 - (0.001) \times (-4)$$

$$= 0.004$$

Step 9 Similar to above now we'll differentiate wrt m & new value of m will be found

Step 10 → New 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 in cost for values of c & m

Step 12 → Choosing learning Rate! Try to be moderate

Machine Learning Notes → 3

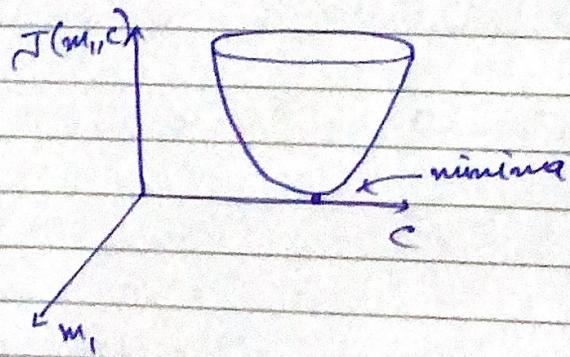
Multiple Linear Regression

House pricing dataset

No. of Rooms	Size of house	Location	Price
1	1800	x	100000 \$
2	1900	y	200000 \$
3	2000	z	1000000 \$

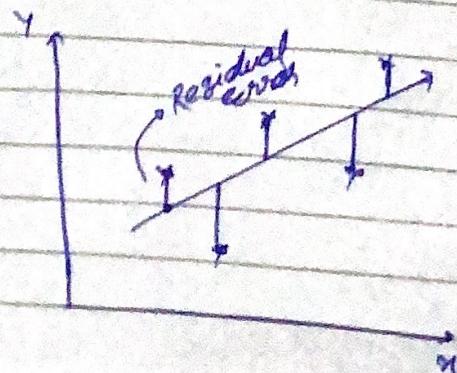
$$y = m_1 x_1 + m_2 x_2 + m_3 x_3 + \dots + m_n x_n + c$$

Example curve, multiple I/P features,



Performance Metrics used in Linear Regression

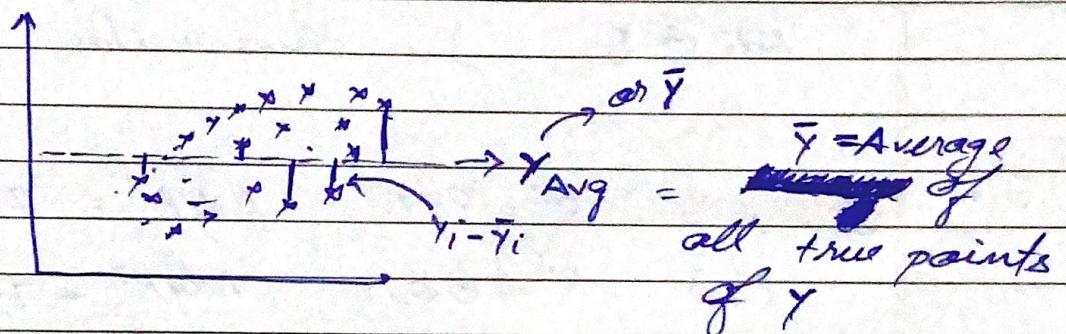
$$1) R^2 \text{ squared} = 1 - \frac{SS_{\text{Residual}}}{SS_{\text{Total}}}$$



classmate
Page

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

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



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

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

0.70 → 70% Accuracy
0.85 → 85% " & so on

2) Adjusted R squared \leftrightarrow Extra feature (Owner POB)

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

N = no. of data points

p = no. of independent features

dataset

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

$R^2 \uparrow$
 $\text{Adj } R^2 \downarrow$

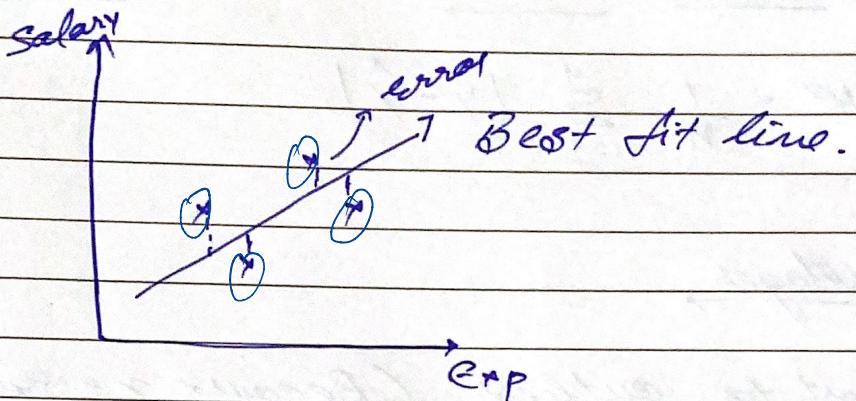
} via penalty through
denominator.

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



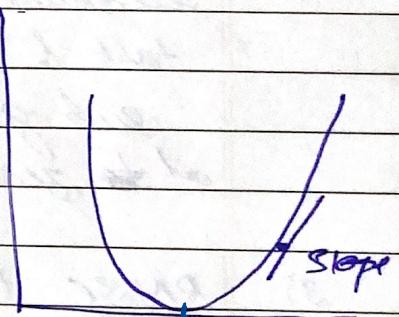
When we want to focus on error metrics.

1) Mean Squared Error [MSE]

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

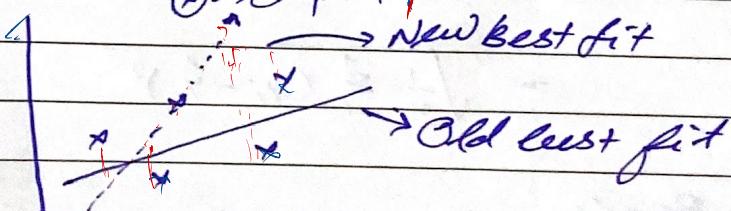
Advantages

- 1) Differentiable at all points
- 2) Converges faster.



Disadvantages

- 1) Not robust to outliers
- ∴ if this \circlearrowleft outlier
MSE will increase.



- 2) Output is not in the same unit.

e.g. let's say salary error = 5000
then MSE will be $(5000)^2$ squared

2) Mean Absolute Error (MAE)

MAE

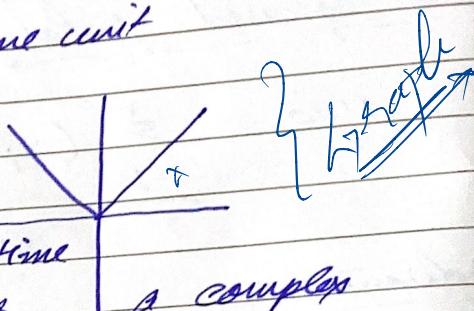
$$\text{MAE} = \frac{1}{n} \sum_{i=1}^n |y_i - \hat{y}_i|$$

Advantages →

- ① Robust to outliers (Because no squaring)
- like MSE
- ② O/p will be in the same unit

Disadvantage →

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



3) RMSE (Root mean Squared error)

RMSE

$$= \sqrt{\text{MSE}}$$

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

Benefit →

- 1) Same unit because of sqrt
- 2) Differential

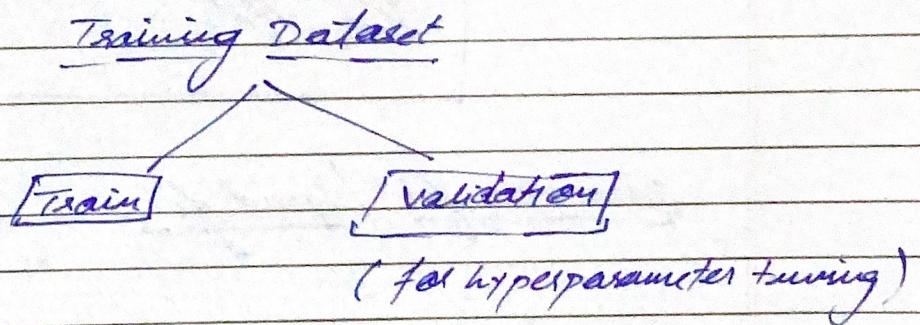
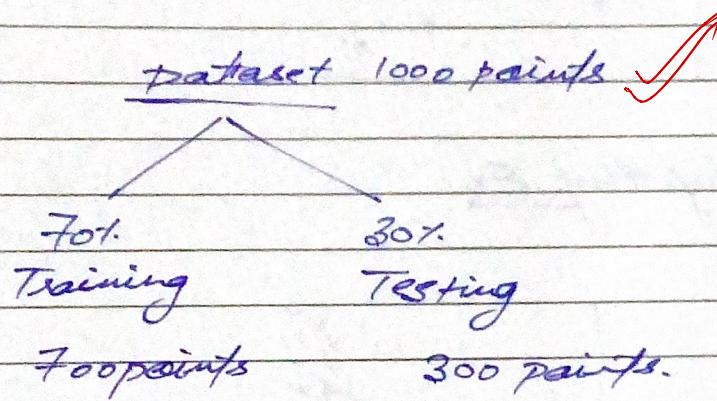
Disadvantage

- 1) Not robust to ~~extra~~ outliers

Machine Learning Notes - 5

Overfitting & Underfitting

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



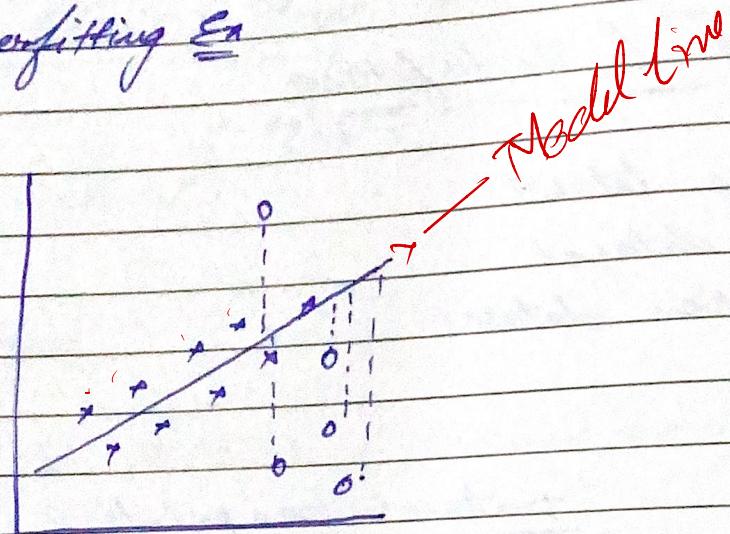
Overfitting

Very good accuracy with training data ie 90% ✓
 Bad accuracy with test data, ex → 50% ✓

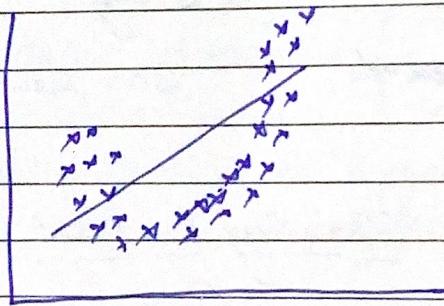
Underfitting

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

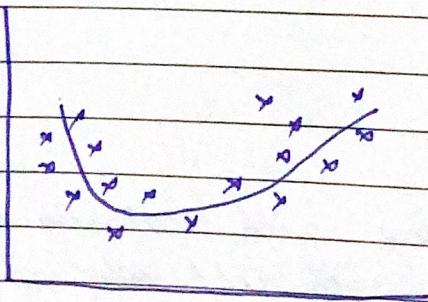
Overfitting Ex



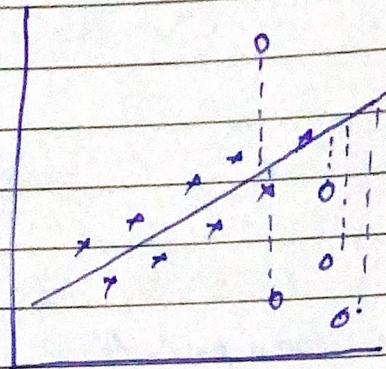
Underfitting Ex



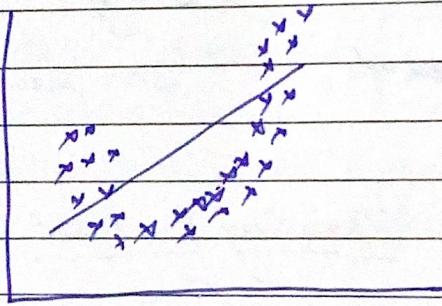
Balanced fit / Good fit



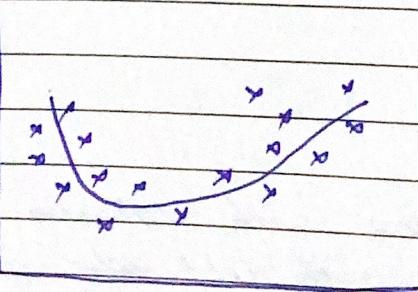
Overfitting Ex



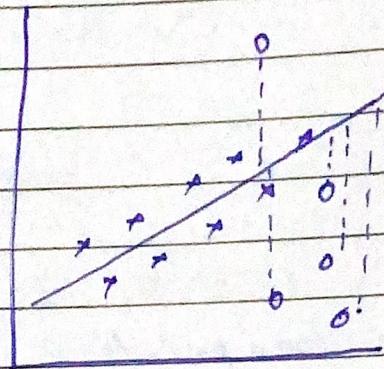
Underfitting Ex



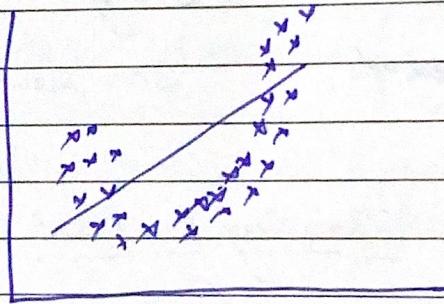
Balanced fit / Good fit



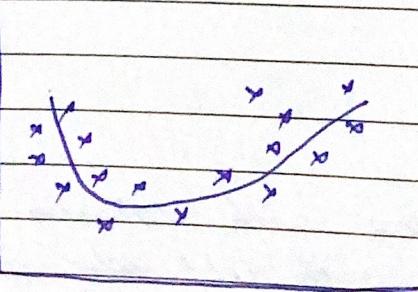
Overfitting Ex



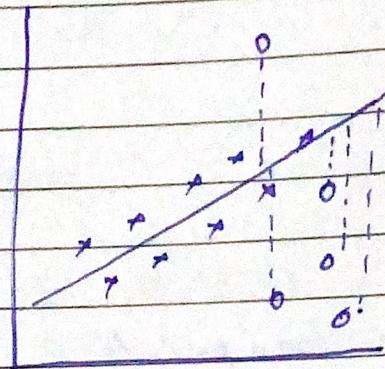
Underfitting Ex



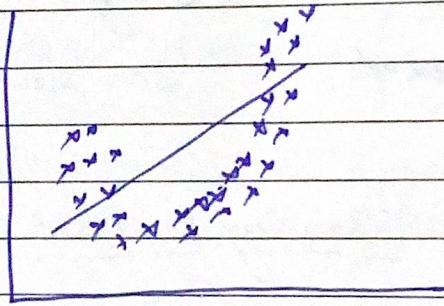
Balanced fit / Good fit



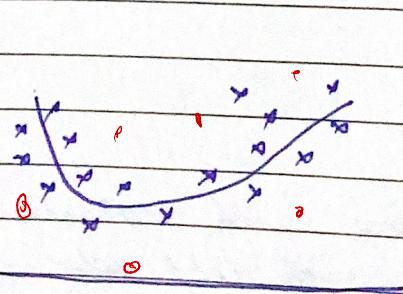
Overfitting Ex



Underfitting Ex



Balanced fit / Good fit



Machine Learning Note-6

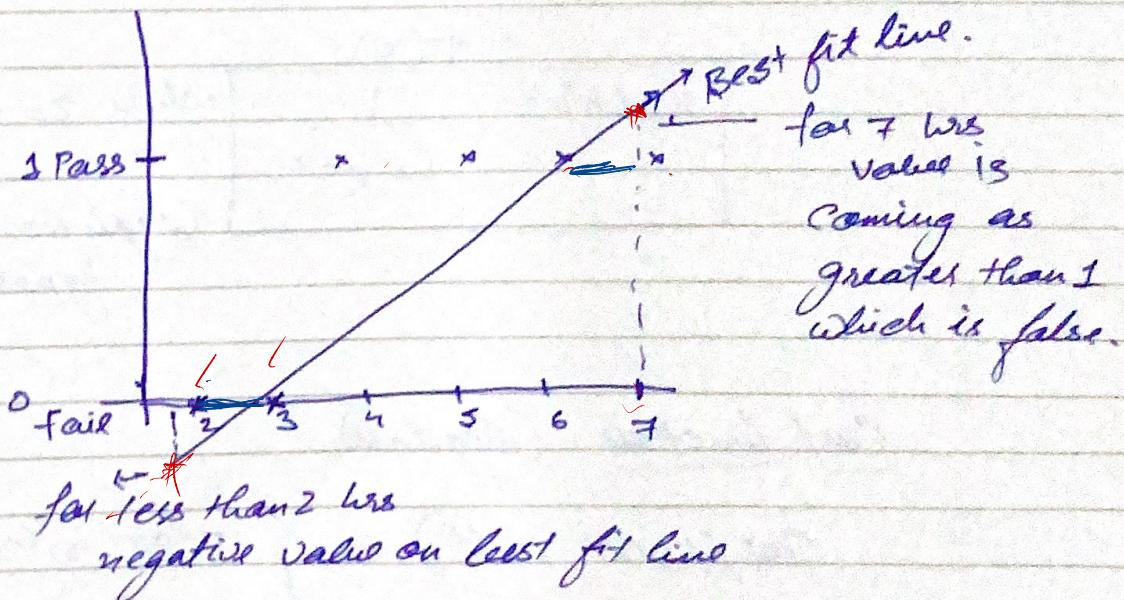
Logistic Regression, (Binary classification)

Dataset O/P (Binary Categories)

(Input variable)	Study hours.	O/P (Dependent variable)
2		Fail
3		Fail
4		Pass
5		Pass
6		Pass
7		Pass

} 2 categories.

✓ Why classification problem can't be solved using Simple Linear Regression?

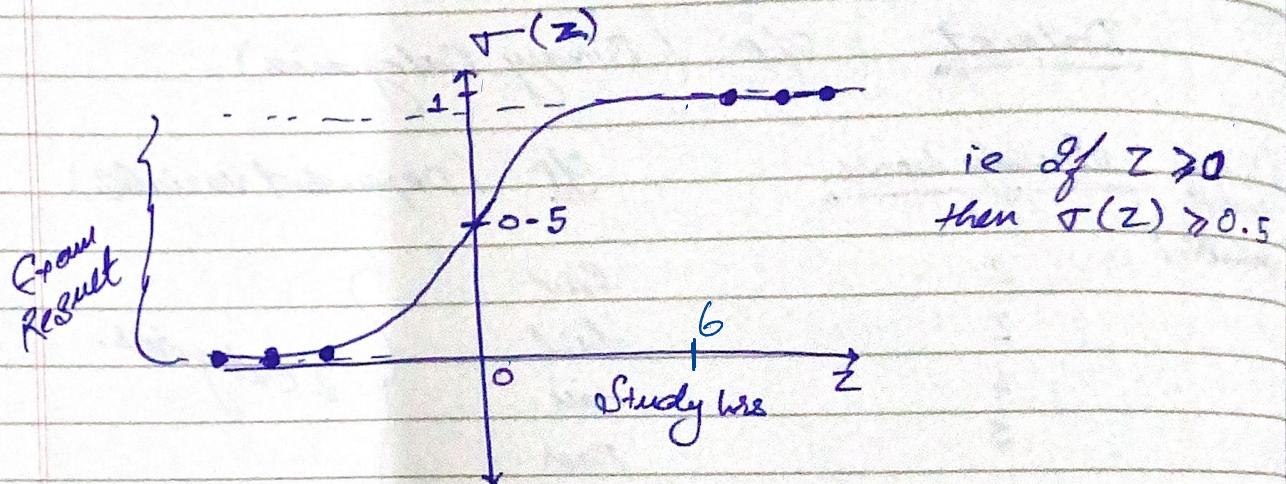


Logistic Regression → (Email / Spam / Non-Spam)

Sigmoid Function

$$\frac{1}{1+e^{-z}}$$

Always b/w 0 & 1



Logistic Regression = Sigmoid function on Simple Linear regression

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

\downarrow

$= T(z)$

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

where $z = \theta_0 + \theta_1 x_1$

(Logistic Regression hypothesis)

Cost function (LogLoss)

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

$$\text{Ex} \quad \theta_0 = 1.5, \theta_1 = 0.6 \quad \text{at } x=5 \text{ hrs}$$

$$h_0(x) = \frac{1}{1 + e^{-z}}$$

$$= \frac{1}{1 + e^{-(\theta_0 + \theta_1 x_1)}}$$

$$= \frac{1}{1 + e^{-(1.5 + 3)}} = \frac{1}{1 + e^{-4.5}} = .818$$

i.e. near to 1 hence pass

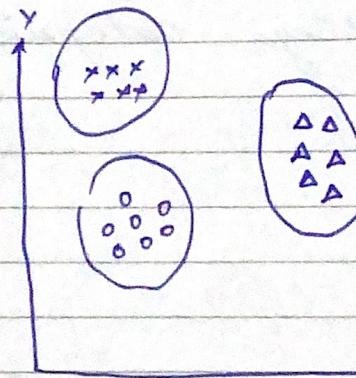
O/P \rightarrow O/P is always categorical in Logistic Regression

Machine Learning Notes - 7

Logistic Regression → (Multiclass - Classification Problem)

OVR Approach → One v/s Rest

f_1	f_2	f_3	O/P
-	-	-	O_1
-	-	-	O_2
-	-	-	O_3
-	-	-	O_2
-	-	-	O_3
-	-	-	O_1



Sub model $M_1 \rightarrow$ Cat A / Cat B + Cat C
 " " $M_2 \rightarrow$ Cat B / Cat A + Cat C
 " " $M_3 \rightarrow$ Cat C / Cat A + Cat B

Taken as 1 category

One hot encoding for Sub Models

f_1	f_2	f_3	O/P	O_1	O_2	O_3
-	-	-	O_1	1	0	0
-	-	-	O_2	0	1	0
-	-	-	O_3	0	0	1
-	-	-	O_1	1	0	0
-	-	-	O_3	0	0	1
-	-	-	O_2	0	1	0

New Sub model trainings

Model $M_1 \rightarrow$ I/P features $\rightarrow f_1, f_2, f_3$ O/P $\rightarrow O_1$

Model $M_2 \rightarrow$ I/P features $\rightarrow f_1, f_2, f_3$ O/P $\rightarrow O_2$

Model $M_3 \rightarrow$ I/P features $\rightarrow f_1, f_2, f_3$ O/P $\rightarrow O_3$

Now new test Data comes

Step 1 \rightarrow Passed thru M_1 model \rightarrow probability $\rightarrow 0.2$

Step 2 \rightarrow " " $M_2 \rightarrow P_2 \rightarrow 0.3$

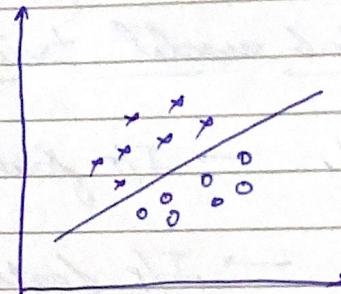
Step 3 \rightarrow " " $M_3 \rightarrow P_3 \rightarrow 0.5$

Now because probability 0.5 is highest
for Model M_3 hence O/P will be
category 3 or category C

Machine Learning Notes - 8

Performance Metrics in Classification Problems

- Topics
- 1) Confusion Matrix
 - 2) Accuracy
 - 3) Precision
 - 4) Recall
 - 5) F-Beta Score

Dataset

x_1	x_2	Actual O/P	Predicted
-	-	0 ✓	1 ✓
-	-	1	1 ✓
✓	-	0	0 ✓
✓	-	1	1 ✓
-	-	1	1 ✓
-	-	0	1 ✓
-	-	1	0

Confusion Matrix

		1	0 ✓	Actual Values.
		✓	✓	
Predicted Values.	1	1+1 +1	1+1	
	0	1	1	

3	2
1	1

Actual

		1	0
Predict	1	TP	FP
	0	FN	TN

TP = True Positive

FP = False Positive

TN = True Negative

FN = False Negative.

Ans New Accuracy = $\frac{TP + TN}{TP + FP + TN + FN}$ ✓ = $\frac{4}{7}$

Precision = $\frac{TP}{TP + FP}$

Ans → Out of all the actual values, how many are correctly predicted.

Recall = $\frac{TP}{TP + FN}$

Ans → Out of all the predicted values, how many are correctly predicted

F1 Score = $2 * \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}$