M T W T F S S

Page No.: YOUVA

Date: 11/4/23

LOGISTIC REGRESSION

> logistic regression is supervised learning technique it is used to predicting the categorical dependent variable using given set of independent variables.

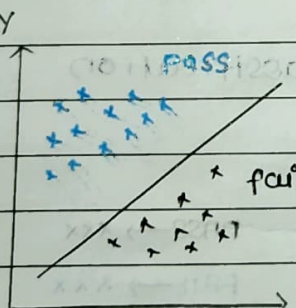
> used to solve classification problems → Binary classification, multiclass classification

> Binary classification → output = 2 categories
multiclass classification → output = greater than 2 categories.

> eg: Dataset

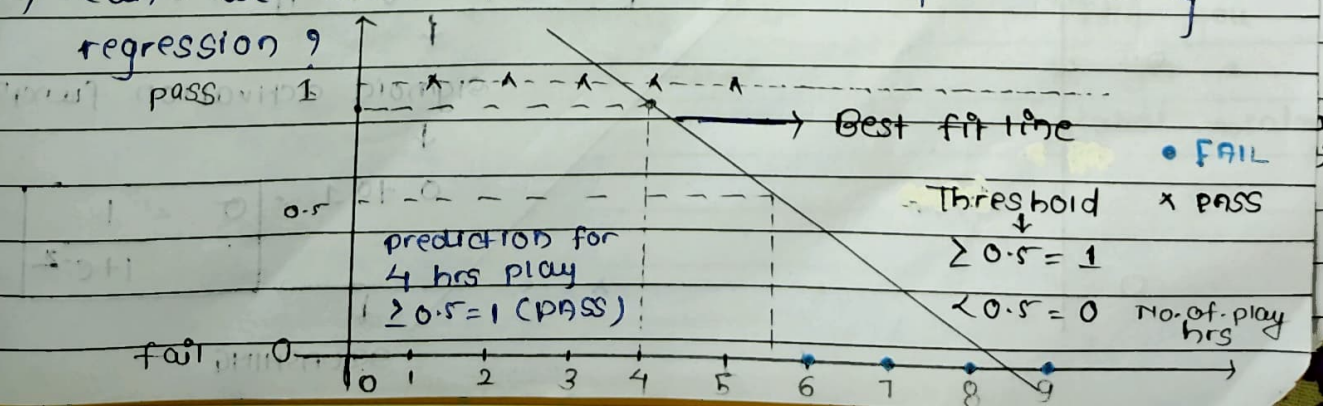
No. of play hours	pass / fail (y)	
9 hrs	fail	0
8 hrs	fail	0
7 hrs	fail	0
6 hrs	fail	0
5 hrs	pass	1
4 hrs	pass	1
3 hrs	pass	1
2 hrs	pass	1

→ Binary



> "With the help of logistic regression we are going to find best fit line in a such way that can able to categorize the pass & fail separately"

> can we solve this classification problem using regression ?



> if the student playing for 4 hours a new predicted datapoint ≥ 0.5 which means that student is pass.

> In case of outlier we will get new best fit line that can reduce our performance and result into wrong prediction.

> Why we cannot used linear regression for classification ?

Ans: 1. linear regression deals with continous values not with discrete values.

2. shift in threshold in case of outliers that can affect the performance of model result into wrong prediction.

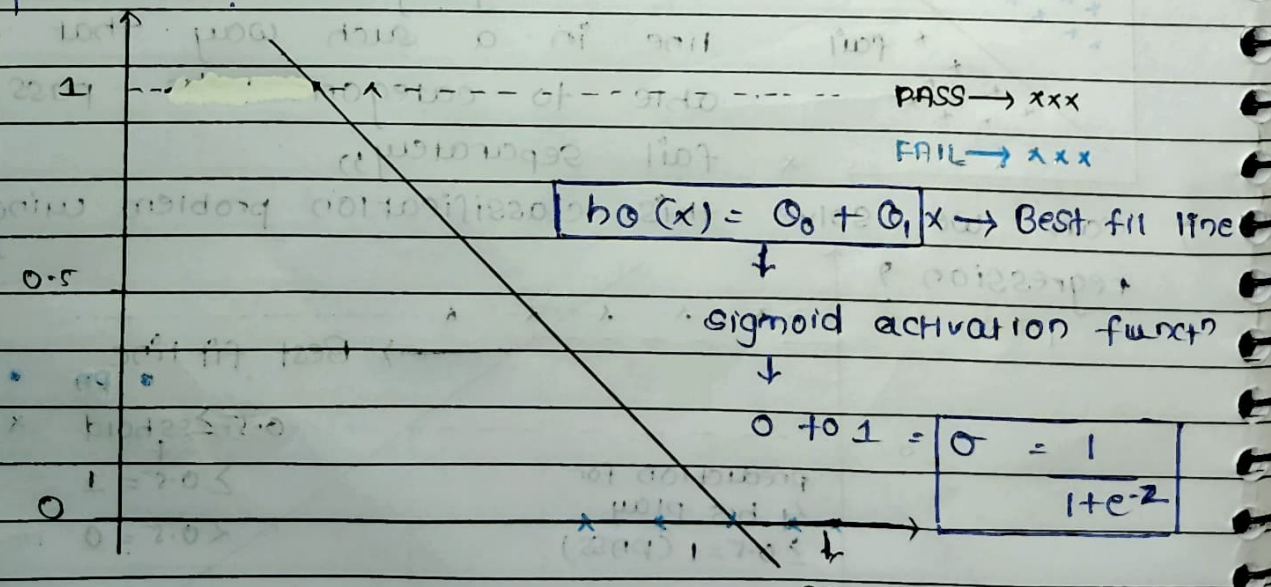
3. The outcomes comes to >1 & <0.

- to solve this problem we will used logistic regression.

↓

0 to 1	$\Longrightarrow$ squashing technique.

> How logistic regression solve classification problems ?



$\therefore \quad h_\theta(x) = \underbrace{\sigma}_{\downarrow}\left(\underbrace{\theta_0 + \theta_1 x_1}_{\downarrow z}\right)$

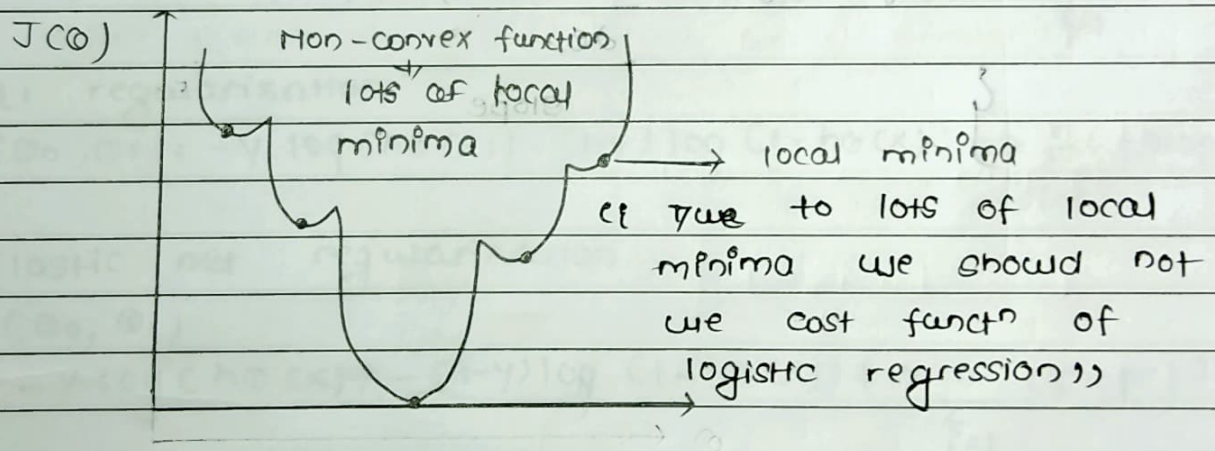
: equating the best fit line.

$$\therefore \quad h_\theta(x) = \frac{1}{1 + e^{-(\theta_0 + \theta_1 x_1)}} \Rightarrow \text{logistic regression}$$

> cost function for logistic regression

$$J(\theta_0, \theta_1) = \frac{1}{n} \sum_{i=1}^{n} (y_i - h_\theta(x)_i)^2$$

$\therefore h_\theta(x) = \frac{1}{1 + e^{-(\theta_0 + \theta_1 x_1)}}$ } value 0 to 1



" due to lots of local minima we should not use cost funct^n of logistic regression "

> log loss function

$$J(\theta_0, \theta_1) = -y_i \log(h_\theta(x)_i) - (1-y) \log(1 - h_\theta(x))$$

$y_i \rightarrow$ real or actual data

$h_\theta(x)_i \rightarrow$ predicted value $= \hat{y}$

" help you to get a global minima "

if $y_i = 1$

$$J(\theta_0, \theta_1) = \begin{cases} -\log(h_\theta(x)i) & \text{if } y = 1 \\ -\log(1 - h_\theta(x)) & \text{if } y = 0 \end{cases}$$

> final aim: minimize cost function $J(\theta_0, \theta_1)$ by changing θ_0, θ_1

> convergence algorithm:

Repeat untill convergence

{

$$\theta_j = \theta_j + \alpha \frac{\partial}{\partial \theta} J(\theta_0, \theta_1)$$

(learning rate ↑, slope ↓)

}

LOGISTIC REGRESSION WITH REGULARIZATION

M T W T F S S | Page No.: | Date: | YOUVA

> Regularization ⟶ reducing overfitting
> Applying regularization techniques

$$J(\theta_0, \theta_1) = -y \log(h_\theta(x)) - (1-y) \log(1-h_\theta(x)) + L_2$$
$$J(\theta_0, \theta_1) = -y \log(h_\theta(x)) - (1-y) \log(1-h_\theta(x)) + L_1$$

∴ L2 regularization ⟶ reducing overfitting
∴ L1 regularization ⟶ feature selection.

$$J(\theta_0, \theta_1) = -y \log(h_\theta(x)) - (1-y) \log(1-h_\theta(x)) + L_1 + L_2$$

∴ L2 reg + L1 reg ⇒ Elastic net

> L2 regularization

$$J(\theta_0, \theta_1) = -y \log(h_\theta(x)) - (1-y) \log(1-h_\theta(x)) + \lambda \sum_{i=1}^{n} (\text{slope})^2$$

> L1 regularization

$$J(\theta_0, \theta_1) = -y \log(h_\theta(x)) - (1-y) \log(1-h_\theta(x)) + \lambda \sum_{i=1}^{n} |\text{slope}|$$

> Elastic net regularization

$$J(\theta_0, \theta_1) = -y \log(h_\theta(x)) - (1-y) \log(1-h_\theta(x)) + \lambda \sum_{i=1}^{n} (\text{slope})^2 + \lambda \sum_{i=1}^{n} |\text{slope}|$$

PERFORMANCE METRICS, ACCURACY, PRECISION RECALL AND F-BETA.

M T W T F S S | Page No.: | Date: 2/4/23 | YOUVA

1.) Confusion matrix

>

	1	0
1	1	1
predicted 0	1	1
($\hat{Y}$)		

Actual (Y)

eg: Dataset

x_1	x_2	Y	$\hat{Y}$
-	-	0	1
-	-	1	1
-	-	0	0
-	-	1	1
-	-	1	1
-	-	0	1

predicted

> Notation

1 - P , 0 - N

1 0 → Actual

1	TP	FP
0	FN	TN

↓ predicted

$$\frac{TP + TN}{TP + FP + FN + TN} \Rightarrow \text{Model Accuracy}$$

$$\therefore \text{predicted} = \frac{3+1}{3+2+1+1} = \frac{4}{7} = 57.1\%$$

TP, TN → Correct match , FP, FN → wrong match

> eg:

Dataset → Imbalance dataset

1000 datapoint → 900 → 1, 100 → 0 } Imbalance

↓

confusion matrix

Dumb model → 1 ⇒ "we get 90% Acuracy"

	1	0
1	900	100
0	0	0

← Actual

↓ predicted

= Acaray = 90%

"if the accuracy is 90%. it is not sufficient model is not good. To overcome this problem we can use recall & precision"

2.) precision

> Measure the accuracy of positive prediction.

> precision = $\frac{TP}{TP+FP}$ } out of all the actual values how many are correctly predicted.

3.) Recall

> Number of correct post prediction made out of all post prediction that could have been made.

> Recall = $\frac{TP}{TP+FN}$ } out of all the predicted value how many are correctly predicted with actual value.

NOTE:	> Whenever FP (False post) is important we will use precision. (FP) ↓ reducing.
	> Whenever FN (False neg-) is important we will use recall. (FN) ↓ reducing.

usecase 1: Spam classification (precision) → FP is imp

TEXT → Model → Spam / Not Spam

	1	0 → Actual
1	TP	FP
0	FN	TN

↓ predicted

TP (1)

Mail (Spam) → Model → Spam } Accurate ⇒

Mail (Not Spam) → Model → Not Spam ⇒

TN (0)

Mail (Not Spam) → Model → Spam } wrong prediction ↓ (Blunder)

↓

FP → (imp)

Mail (Spam) → model → Not Spam } wrong predictio

↓

FN

Usecase 2: FN is important (Recall)

> To predict person has diabetic or not

	Diabetis	No diabetis
Diabetis	TP	FP
No diabetic	FN	TN

TP ⇐ Actual → diabetis → model → diabetis } → correct

TN ⇐ Actual → No diabetis → model → no diabetis } → correct

FP ⇐ Actual → No diabetis → model → diabetis } → wrong

FN ⇐ Actual → diabetic → model → No diabetic } → wrong
↓
(Blunder)

4.) F-Beta score

∴ F-Beta score $= (1+\beta^2) \frac{\text{precision} * \text{Recall}}{\text{precision} + \text{Recall}}$

i.) if FP and FN are both important

$\beta = 1$

$$F1 \text{ score} = \left\{ (1+1) \frac{P * R}{P + R} \right\} \Rightarrow \text{Harmonic Mean}$$

ii.) if FP is more important than FN

$\beta = 0.5$

$$\left\{ F0.5 \text{ score} = (1+0.25) \frac{P \times R}{P+R} \right\}$$

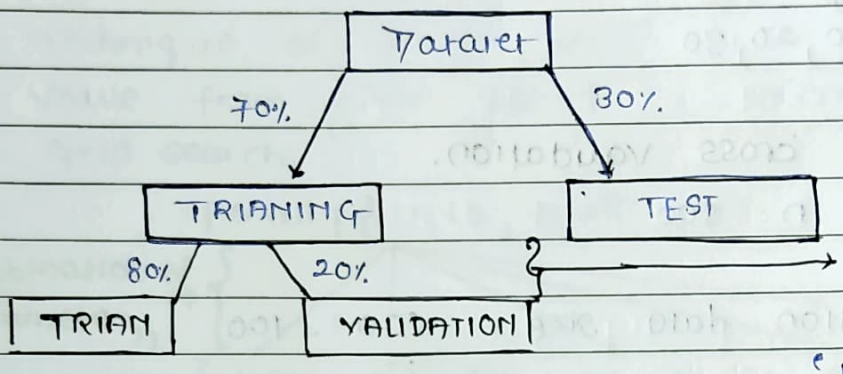
iii) if FN is more important than FP

$\beta = 2$; $\left\{ F2 = (1+4) \frac{P \times R}{P+R} \right\}$

CROSS VALIDATION AND ITS TYPES

M T W T F S S
Page No.: YOUVA
Date:

> eg: consider dataset with 1000 datapoints



random state = 100
= 42

→ 85% accuracy } "getting different Accuray because we changing the random state value"
75%.
65%.

TRAINING ⇒ CV=5 Exp1 Train : validation Acc 1
Exp2 Train validatipn Acc 2
⋮
Exp Train Validation Acc 3

"mean of accuracy"

TYPE OF CROSS VALIDATION:

1.) leave one out cross validation (Loocv)

Training data ⟶ 500 datapoints

validation ↓

Exp1 = [1 | 499 training data] ⇒ ACC1 = 85%.

validation ↑

Exp2 = [1 | 2 | 498 Traing data] ⇒ ACC2 = 80%.

} Avg of all accuray

Disadvantage:

① Time complexity is huge
② Model tends to overfit. ⇒ Train: Acc ↑
New data ↓

2.) leave p out cross validation

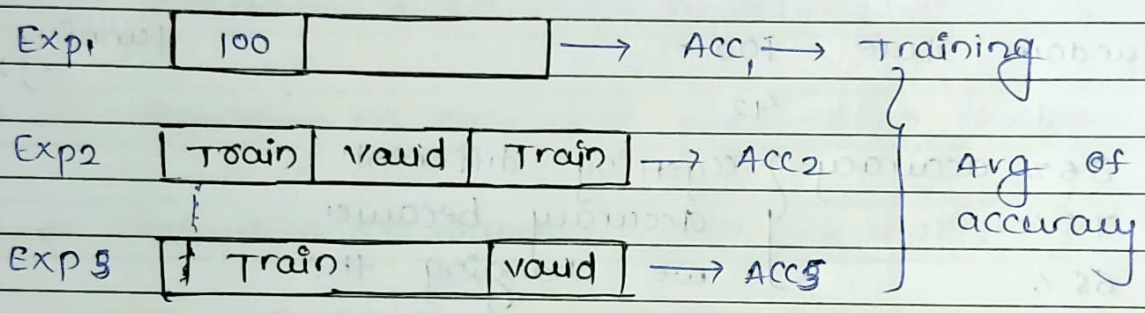
p ⇒ hyperparameter

eg: p=10, 20, 30

3.) K-fold cross validation.

K=5 n = 500

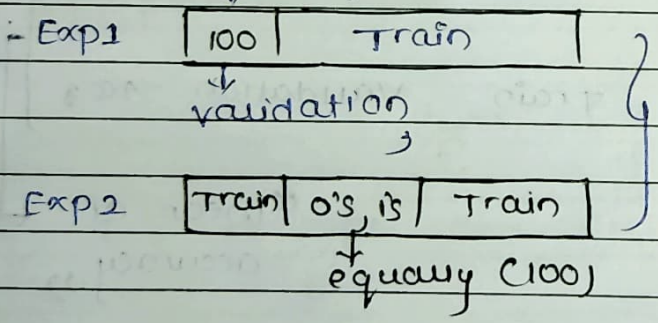
validation data size = 500 / 5 = 100



4.) Stratified K fold CV → { imbalanced dataset }

0's, 1's equally K=5 n=500

{ 1 category = 350 }
{ 0 category = 150 }



HYPERMETER TUNNING WITH CROSS VALIDATION

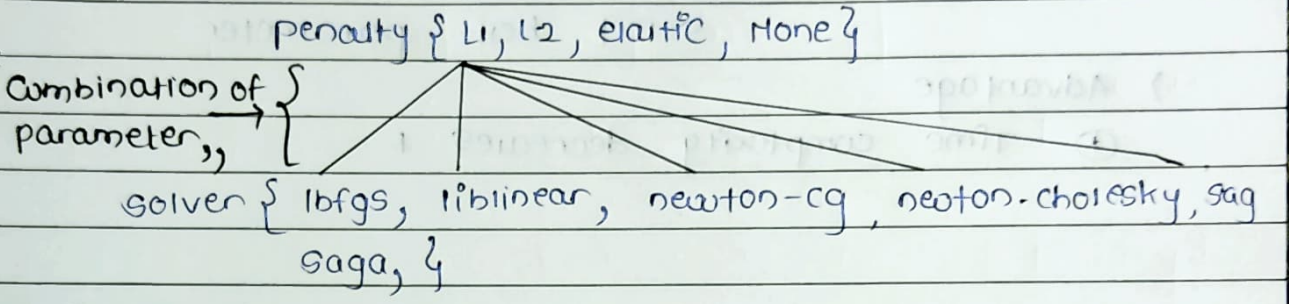
M T W T F S S | Page No.: | Date: | YOUVA

> Finding the best parameter while training the models.

1.) Grid search cv [Grid search + CV]

> Technique of finding optimal or best parameter value from given set of parameter.

> Grid search



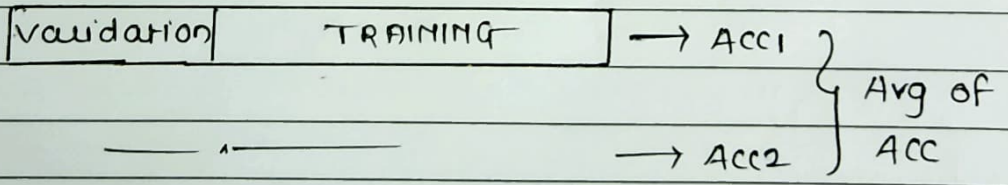
" Grid search cv making sure that go ahead and checking every combination try to find out best combination with high or best accuracy,,

> lets assume that L1 & lbfg is best parameters

logistic regression

↓

cross validation [k fold cv]



> Disadvantage

① Time complexity increases with huge dataset. for training the model.

2.) RandomizedSearchCV:

> Randomly select the best set of parameters.

> eg: n_iter = 10 , CV = 5

10 different combination * CV=5 ⇒ 50

⇓

select best parameter

> Advantage

① Time complexity decreases ↓