1)- Logistic Regression

used to solve classification problems :-

Classification :- → Binary Classification → o/p → 2 categories
→ Multi-class classification → o/p → 2 or more categories

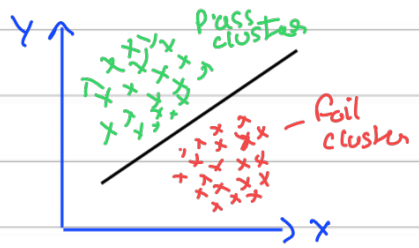
Ex: Consider a dataset with features :-

No. of play hours (X_1) (independent feature)	Pass/Fail (Y) (dependent feature)
9	fail
8	fail
7	fail
6	fail
5	pass
4	pass
3	pass

Test data → Trained Model → Prediction → Pass/fail

So, how does logistic regression work

Logistic regression creates clusters of both the classifications with boundary b/w them in order to predict the result such as

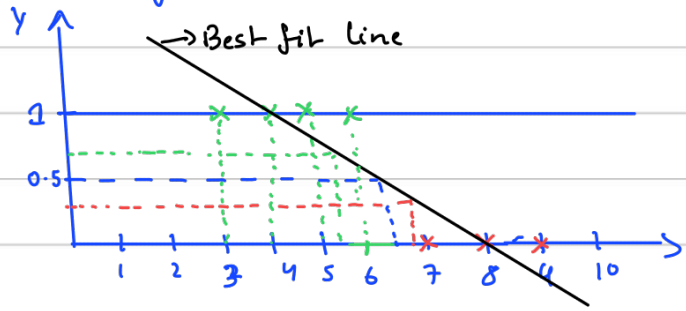


Can we solve above classification problem using Regression

No. of play hours (X_1) (independent feature)	Pass/Fail (Y) (dependent feature)
9	0 fail
8	0 fail
7	0 fail
6	0 fail
5	1 pass
4	1 pass
3	1 pass

Considering fail = 0 & pass = 1

Now plotting points



Now lets say if,

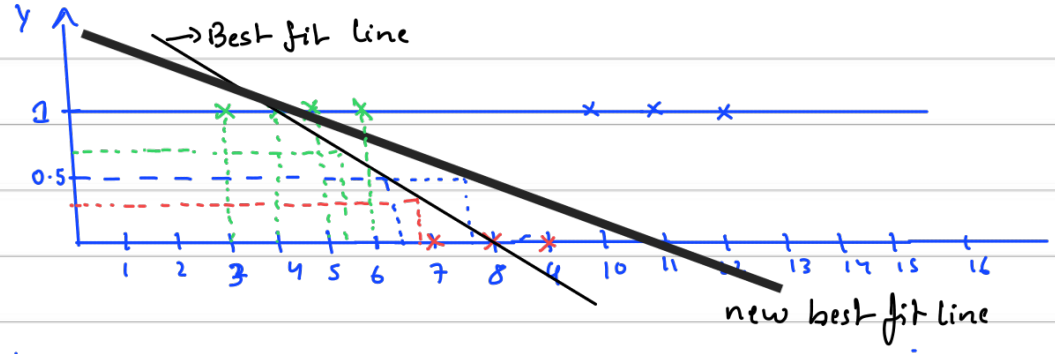
$$\left.\begin{array}{l} O/p \geq 0.5 \xrightarrow{\text{Prediction}} 1 \\ O/p < 0.5 \xrightarrow{\text{Prediction}} 0 \end{array}\right\} \text{ i.e } \underline{\text{threshold} = 0.5}$$

∴,

we are able to get the O/p so. what is the problem with using regression for classification problems?

Now,

Suppose the no. of datapoints and outliers increase so the line will ben bent towards left such that



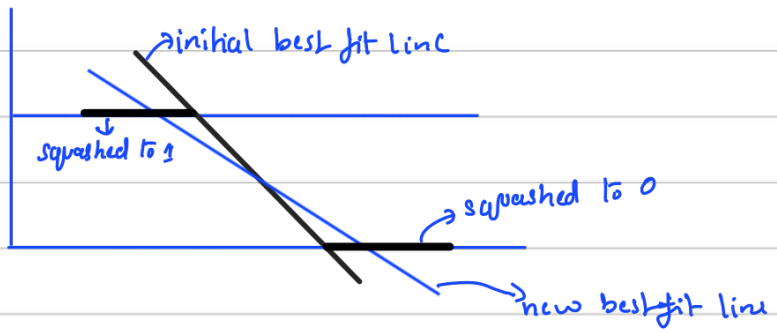
Now, as no. of datapoints are increased we can observe that for 7 also we are getting O/p > 0.5 even though it should be 0

Q) Why we cannot use Linear Regression for Classification.

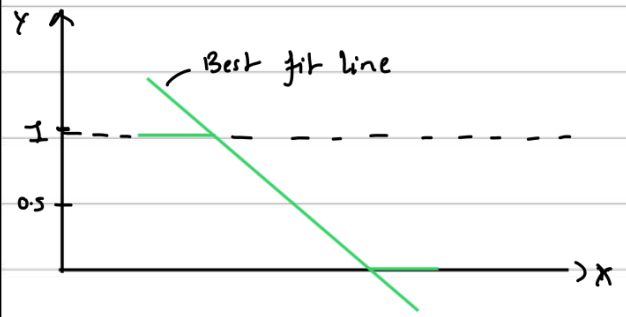
① Best fit line changes because of outliers → prediction goes wrong

② the output comes >1 and <0

Squashing Technique:- this technique is in Logistic regression, in this technique any line aside from to initial best fit line is squashed to 0 or 1 such that



1.1) How Logistic Regression Solves Classification Problem?



$$\boxed{z = h_\theta(x) = \theta_0 + \theta_1 x_1} \text{ — eq ①}$$

is used to get the best fit line but at o/p > 1 and o/p < 0 this line must be squashed ∴,

$$\boxed{\text{Sigmoid function : } \sigma = \frac{1}{1+e^{-z}}} \text{ — ②}$$

is used to squash the best fit line at 1 < o/p < 0.

∴,

New hypothesis becomes,

$$h_\theta(x) = \sigma(z) = \sigma(\theta_0 + \theta_1 x_1)$$

$$\Rightarrow \boxed{h_\theta(x) = \frac{1}{1+e^{-(\theta_0+\theta_1 x_1)}}}$$

→ New hypothesis function for Logistic Regression

1.2) Logistic Regression Cost Function :-

Linear Regress Cost function	Logistic Regression Cost function
$J(\theta_1, \theta_0) = \frac{1}{n}\sum_i^n (y_i - h_\theta(x))^2$,	$\boxed{J(\theta) = \frac{1}{n}\sum_i^n (y_i - h_\theta(x)_i)^2}$,
$h_\theta(x) = \theta_0 + \theta_1$	$h_\theta(x) = \frac{1}{1+e^{-(\theta_0+\theta_1 x_1)}}$

Q) Why this cost function is not used in Logistic Regression?

Reason → As in case of $J(\theta)$ vs θ graph of Logistic Regression there are many local minima ∴, the convergence algorithm will be stuck inside the 1st local minima

ex

Why so with Error function is used in Logistic Regression?

Solⁿ:-

★ $\boxed{\text{Log Loss function:- } J(\theta_0,\theta_1) = -y_i \log(h_\theta(x)_i) - (1-y_i)\log(1-h_\theta(x_i))}$

New cost function for Logistic Regression.

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

1.3 Convergence Algorithm:-

repeat untill convergence:- ★ $\boxed{\theta_{j_{new}} : \theta_j - \alpha \frac{\partial J(\theta_1,\theta_0)}{\partial \theta}}$

2)- Logistic Regression with Regularization

Cost function:-

$\boxed{\text{Log Loss function:- } J(\theta_0,\theta_1) = -y_i \log(h_\theta(x)_i) - (1-y_i)\log(1-h_\theta(x_i))}$ + L_2 Regularization → Reducing Overfitting

$\boxed{\text{Log Loss function:- } J(\theta_0,\theta_1) = -y_i \log(h_\theta(x)_i) - (1-y_i)\log(1-h_\theta(x_i))}$ + L_1 Regularization → feature selection

L_2 Regularization (for reducing overfitting)

$$J(\theta_0,\theta_1) = -y_i \log(h_\theta(x)_i) - (1-y_i)\log(1-h_\theta(x_i)) + \lambda \sum_1^n (\text{Slope})^2 \quad , \text{slope} = \theta_0$$

L_1 Regularization (for feature selection)

$$J(\theta_0,\theta_1) = y_i \log(h_\theta(x)_i) - (1-y_i)\log(1-h_\theta(x_i)) + \lambda \sum_1^n |\text{Slope}|$$

Elasticnet:- ($L_1 + L_2$)

$$J(\theta_0,\theta_1) = y_i \log(h_\theta(x)_i) - (1-y_i)\log(1-h_\theta(x_i)) + \lambda_1 \sum_1^n (\text{Slope})^2 + \lambda_2 \sum_1^n |\text{Slope}|$$

#note: sklearn.linear_model.LogisticRegression ($C = 1/\lambda$)

3)- Performance Matrices, Accuracy, Pression, Recall and F-Beta score.

3.1)- Confusion Matric

Consider the following dataset

x_1	x_2	y_i	$\hat{y}_i$
-	-	0	1
-	-	1	1
-	-	0	0
-	-	1	1
-	-	1	1
-	-	0	1
-	-	1	0

$\hat{y}_i \rightarrow$ Predicted value

$y_i \rightarrow$ actual value

$\rightarrow$ wrong predictions

$\Downarrow$

Predicted value \ actual value	1	0
1	1+1+1	1+1
0	1	1

$\Downarrow$

★ Confusion matrix

	3	2
	1	1

$\rightarrow$ correct predictions

3.2)- Accuracy:

to calculate accuracy use the formulae below :-

confusion matrix :-

	1	0
1	TP	FP
0	FN	TN

TP – True +ve & TN = True -ve

, FP → False +ve & FN = False -ve

★ accuracy

$$\therefore \boxed{\text{accuracy} = \frac{TP + TN}{TP + TN + FP + FN}} \Rightarrow \text{accuracy} = \frac{3+1}{3+2+1+1} = \frac{4}{7} = 57.1\%$$

3.3)- Precision: ★[used when: FP is to be reduced (FP↓↓)]

★ $$\boxed{\text{Precision} = \frac{TP}{TP + FP}}$$

$\Rightarrow$ idicates How many are correctly pridicted.

Predicted \ Actual	1	0
1	TP	FP
0	FN	FP

3.4)- Recall: [Used when: FN is to be removed (FN ↓)]

$$\boxed{\text{Recall} = \frac{TP}{TP + FN}}$$ => indicates out of all predicted values how many are correctly predicted with actual values.

used it to reduce FN

Q) How to decide when False Negitive and when is False +ve important?

Ex 1: usecase 1: Spam Classification

Text $\xrightarrow{\text{training}}$ Model $\xrightarrow{\text{Prediction}}$ Span/ Not Spam

Various Scenarios:-

① Mail → Spam, Pred^n → Spam } accurate Scenario

② Mail → Not Spam, Pred^n → Not spam } accurate Scenario

③ Mail → Spam, Pred^n → Not Spam } FN

④ Mail → Not Spam, Pred^n → Spam } FP

in the above case the false +ve (FP) is really important because in case ④ if its an imp. mail and is missed due to being in a spam folder, it will create a big problem

∴,

In this scenario we use "Precision" to calculate the accuracy of the model

Ex 2: usecase 2: To predict weather the person has diabetes or not?

dataset → Model → Prediction < 1 → Diabetes, 0 → No Diabetes

Predict \ actual	1	0
1	TP	FP
0	FN	TN

In this case False Negative is very important and can be very dangerous for the patient

∴ in this case we use "Recall"

3.5) F-beta Score:

$$\boxed{\text{F-Beta score} = (1+\beta^2)\,\frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}}$$

① $\beta = 1$: If FP & FN both important

$$\Rightarrow \boxed{F_1 \text{ Score} = 2\,\frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}}$$

② $\beta = 0.5$: If FP is more important then FN

$$\Rightarrow \boxed{F_{(0.5)} \text{ Score} = 1.25\,\frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}}$$

③ $\beta = 2$: If FN is more important FP

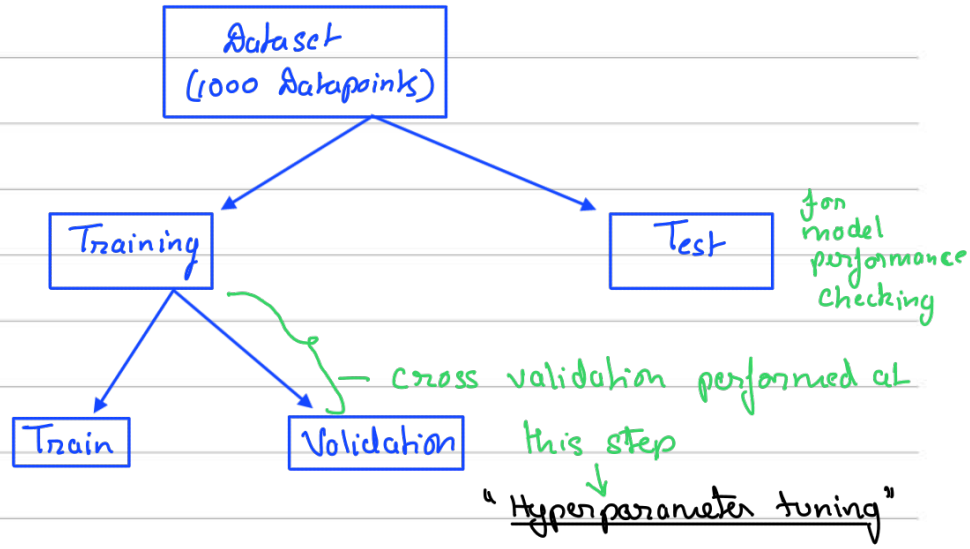
$$\Rightarrow \boxed{F_2 \text{ score} = 5\,\frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}}$$

4)- Cross Validation and its types:

Consider a dataset with 1000 datapoints:

Dividing dataset into test & train dataset:

dividing training data into Train & Validation:



Q) Why is cross validation Performed?

Sol^n -> as during Train-Test split based on diff.-diff. split (i.e random state) the accuracy of the model also differs. Therefore while cross validation we take diff. diff. Train-Validation split and find the avg. accuracy and choose the accuracy.

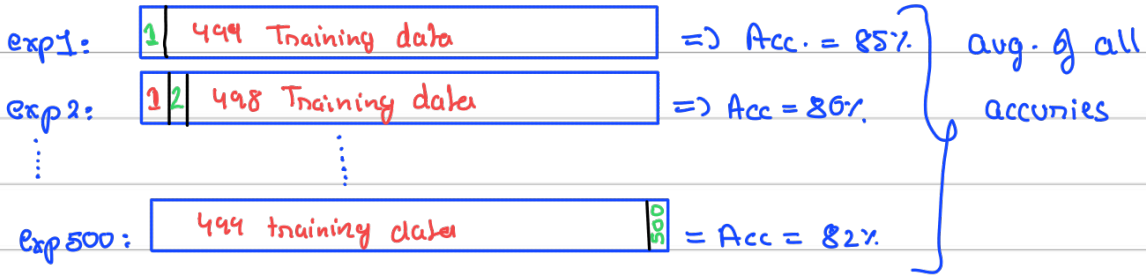
Ex:- exp 1: Train_1 Validation_1 → Accuracy 1
exp 2: Train_2 Validation_2 → Accuracy 2
⋮
exp n: Exp_n Validation_n Accuracy n

→ final accuracy = mean accuracy

4.1)- Types of Cross Validation Techniques:

4.1.1)- Leave One Out Cross Validation (LOOCV):

Consider a dataset: Dataset (500 datapoints)



Disadvantage:-

1. Time complexity is very huge for training big data
2. Model Overfit to training data => Train accuracy ↑↑
   New data Acc: ↓↓

4.1.2)- Leave P Out Cross Validation (LPOCV):

$\boxed{\text{Leave P out CV (LPOCV)}}$: p = hyperparameter, $p = 10, 20 \ldots$ etc.

4.1.3)- Satisfied K Fold Cross Validation :

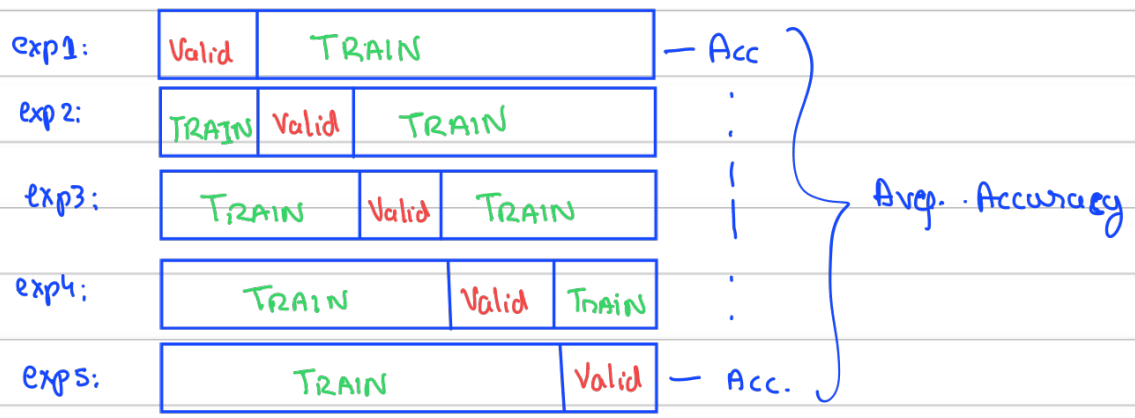
Let say, $k = 5$ and $n = 500$

k = no. of experiments

n = no. of datapoints in a dataset

$\therefore$,

validation data size $= \frac{n}{k} = \frac{500}{5} = 100$



4.1.3)- Stratified K Fold Cross Validation : (for imbalanced dataset)

In Stratified K fold :- Validation data will have equal no. of categories (1s or 0s)

Ex: $n = 500$ $\quad 1 \to 350$

$0 \to 150$

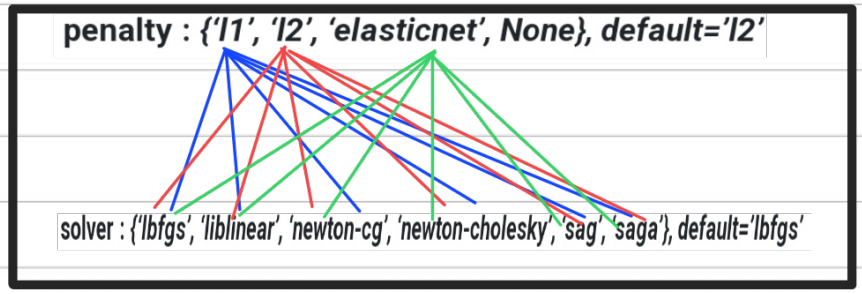
$k = 50$ $\therefore$ Stratified K fold technique will make sure that there will be 25 · 1s and 25 0s in validation data at all times.

5)- HYPERPARAMETER TUNING With Cross Validation:

→ used for finding best parameter while training the models.

Types : ① Grid Search CV

② Randomized Search CV



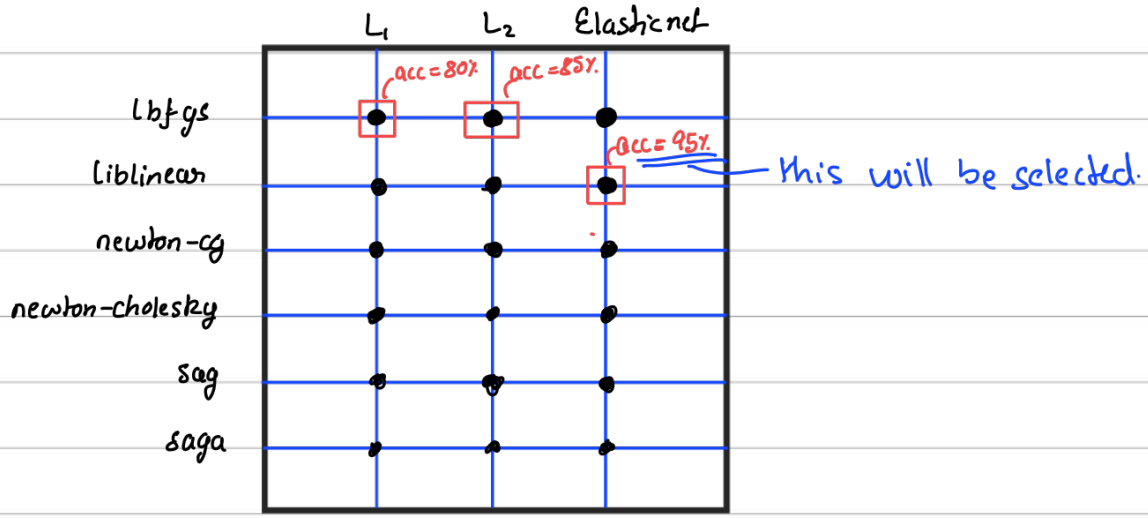
which combination of above parameters suits the best is decided using above two techniques.

5.1)- Grid Search Cv: [Grid Search + Cross Validation]

Grid Search: use to choose best possible parameters

So grid search goes through all the combinations and

chooses the one with the best accuracy we are able to get.
Such as:



Now after selecting the best possible combination then the model is trained with K fold CV method

ex:- Logistic Regression (Elasticnet, Liblinear)

⇓

Cross Validation (K fold CV, k = n)

<u>Disadvantage</u>: Time complexity increases with huge dataset for training the model.

5.2)- Randomized Search CV:

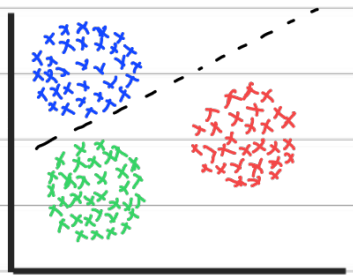
So in Randomized CV instead of interating to all possible combinations we define provide no. of iterations it should perform and the the combinations are selecting randomly.

eg:

n_itr = 10 & CV = 5 => 50 iterations max

<u>Advantage</u>:- Time Complexity increases.

6)- Logistic Regression for Multiclass Classification:



can't categorize the folling clusters with one perticular best-fit line.

Using following techniques Logistic Regression is able to classify multiclass classifications

1- One Versus Rest (OVR)
2- Multinomial

1)- One Versus rest (OVR):

Consider dataset with 3 classification o/p:-

f_1	f_2	f_3	o/p
-	-	-	1
-	-	-	0
-	-	-	2
-	-	-	2
-	-	-	0
-	-	-	1

So, what OVR does is that it creates one hot encoding for the o/p classes 1st

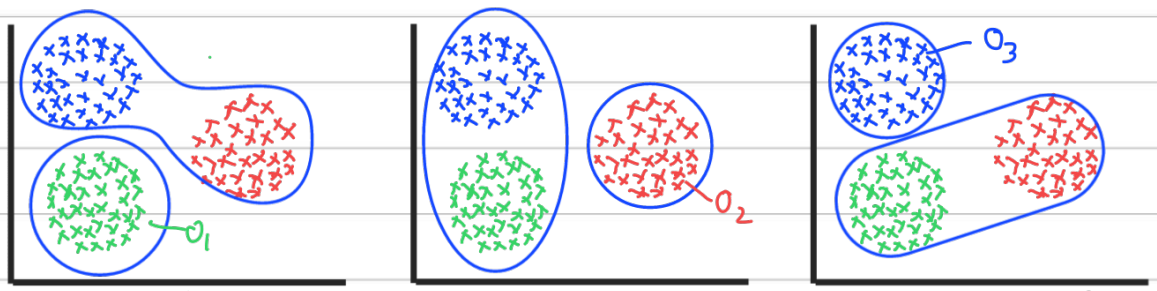
=>

f_1	f_2	f_3	$O_1^{(0)}$	$O_2^{(1)}$	$O_3^{(2)}$
-	-	-	0	1	0
-	-	-	1	0	0
-	-	-	0	0	1
-	-	-	0	0	1
-	-	-	1	0	0
-	-	-	0	1	0

Now,

After creating one-hot encoding for the o/p column the 2 classification are created by combining 2 classes. and doing such 3 models are trained (cuz 3 classes are present, suppose if there are 5 o/p classes then 5 models will be trained).

ex:



model 1 (M_1) model 2 (M_2) model 3 (M_3)

all the above 3 models are trained by consider o/p as binary o/p.

① M_1 Model → $I/p = f_1\ f_2\ f_3$ & $o/p = O_1$

② M_2 Model → $i/p = f_1\ f_2\ f_3$ & $o/p = O_2$

③ M_3 Model → $i/p = f_1\ f_2\ f_3$ & $o/p = O_3$

f_1	f_2	f_3	O_1	O_2	O_3
-	-	-	0	1	0
-	-	-	1	0	0
-	-	-	0	0	1
-	-	-	0	0	1
-	-	-	1	0	0
-	-	-	0	1	0

So, new test data → $[M_1\ \ M_2\ \ M_3]$

$[0.25\ \ 0.25\ \ 0.5]$

↳ highest probability is in M_3 model so, the ans will be O_3 i.e new data