Logistic Regression

Supervised ML approach for solving the classification problem statements

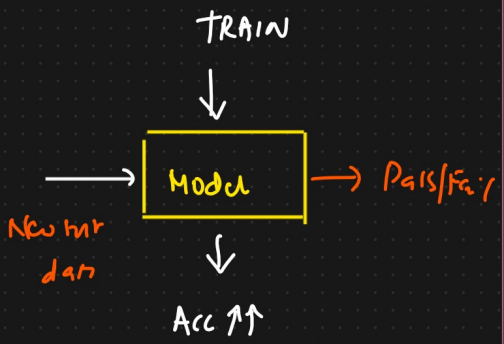
To solve classification
- Binary → Dependent → 2 categories (Dependent/Output feature have 2 classes of data points)
- Multiclass classification → More than 2 categories. (Dependent/Output feature have more than 2 classes of data points)

Example: In below dataset Pass/Fail is dependent or output feature consisting of 2 classes of data points(0 representing fail and 1 representing pass). Therefore, this is binary classification problem.

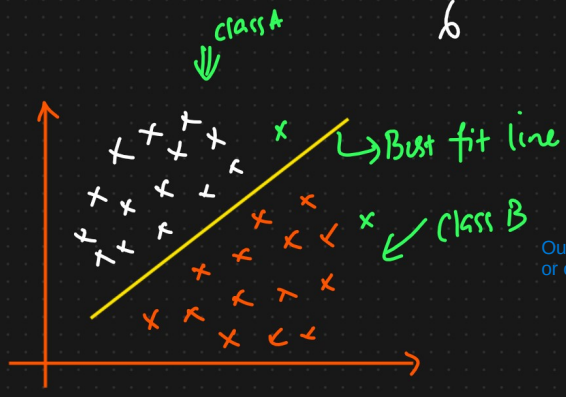
Dataset

UPSC Exam

No. of study hours	O/P Pass/Fail
0	0
1	0
2	0
3	0
2	1
4	1
5	1
6	1

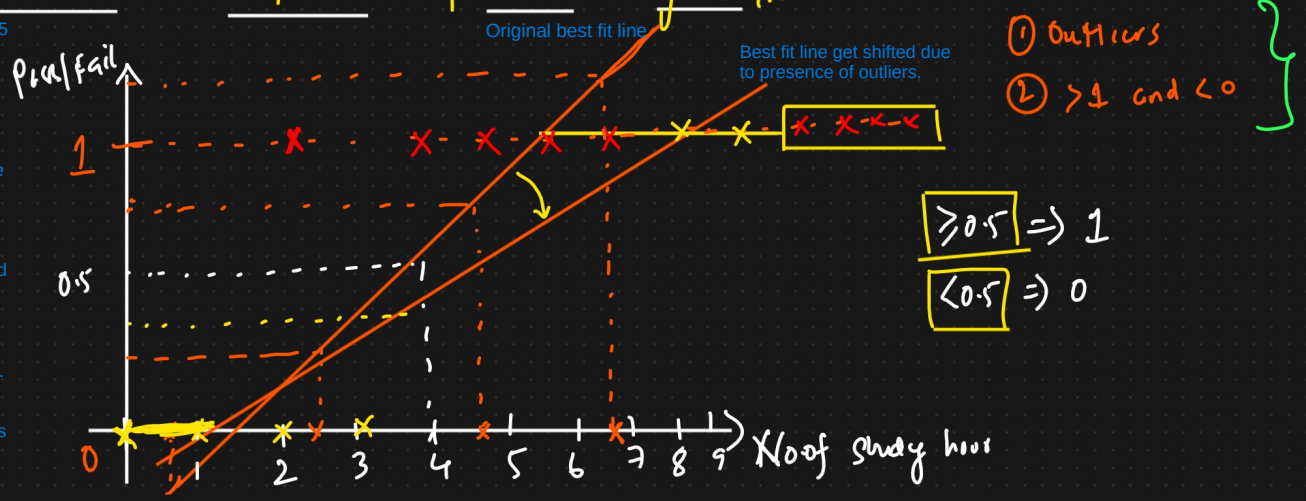


Here, we will be building our learning model using logistic regression approach. After that will train using training dataset followed by using new data points to make predictions using this model.



Our aim is to construct a best fit line that splits our datapoints in 2 or more classes based on dependent or output feature. In logistic regression we use Sigmoid function in order to construct this best fit line.

Can we solve this classification problem using Regression ?



1. Setting the threshold at 0.5 in such a way that datapoint >= 0.5 will be classified as 1 else 0.
2. If projection of any datapoint over best fit line greater than 1 on y axis or less than 0 on x axis then we need to squash(cutt off) those projection by making line parallel to x axis so that datapoints always lie in the range[0,1]. After which based on threshold we can make the predictions.
Example:
Squash it means make the straight line parallel to x axis. b/w 8 and 9 projection on best fit line is greater than 1 but it should be [0,1] so that's why we are squashing it means cut it.

0 or 1 | >1 & <0

⇓

Best fit line $h_\theta(x) = \theta_0 + \theta_1 x_1 \Rightarrow z$

This is alphabet Z

Equation of best fit line represented as Z

⇓

Squash the best fit line ⇒ Sigmoid fn ⇒ 0 to 1 ⇒ $\frac{1}{1+e^{-z}}$

Note:
1. Squashing in such a way that the best fit line belongs [0,1]. This is done using mathematical fun^ called Sigmoid fun^.

2. Threshold is set based on problem statement and based on domain experts suggestion. For eg; Threshold for selecting whether someone is suffering from cancer or not will be different from the threshold for determining whether on tossing a coin we will be getting head/tails.

① How does logistic Regression solve classification

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

⇓

Logistic Regression

Logistic Regression

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

To summarize:
Once squashing is done using Sigmoid fun^ hypothesis for linear regression get converted into hypothesis for logistic regression. Therefore, we can say that yes, we can solve classification problems using linear regression.

It is called Logistic regression and not logistic classification because internally regression is used to obtain the best fit line followed by squashing using Sigmoid fun^.

Difference b/w linear regression and logistic regression

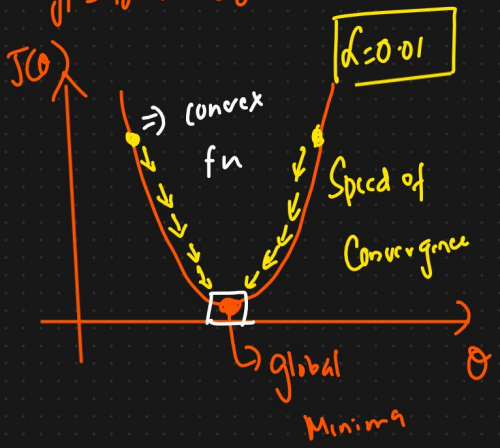
$h_\theta(x)$ ⇑

$\hat{y}_i = \theta_0 + \theta_1 x_i$

Linear Regression Cost fn

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

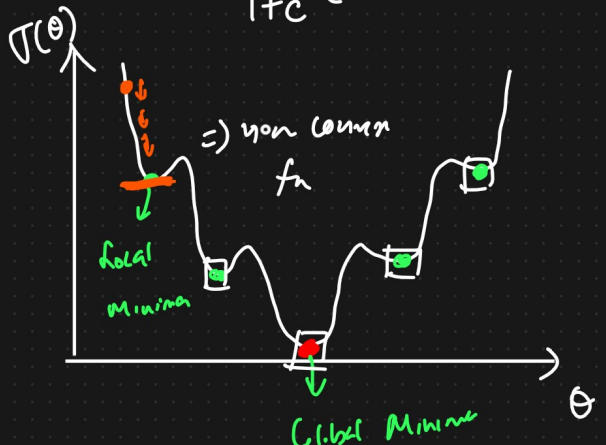
$\hat{y}_i = h_\theta(x) = \theta_0 + \theta_1 x_i$



Logistic Regression Cost fn

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

$$h_\theta(x) = \frac{1}{1+e^{-(\theta_0+\theta_1 x_i)}} \Rightarrow 0 \text{ to } 1$$



1. Cost function of Linear regression is in terms of mean squared error. Whereas cost fun^ of Logistic regression is also in terms of mean squared error where in addition to this we are also performing the squashing operation using Sigmoid function.

2. For Linear regression when cost fun^ is plotted against learning parameter(theta) then only one local minima is generated which can be also treated as global minima. Whereas in the case of logistic regression we will get multiple local minima (and global minima) as curve obtained through plot is non convex. Hence in logistic regression if we use above cost function then we will be not able to generate the optimized learning parameters as on applying gradient descent we will converge or stuck in on the local minima which will not result in producing the optimal learning parameters. In order to counter this case we use Log Loss cost function which ultimately will convert the above plot for logistic regression into convex/parabolic(since log is used) curve using which then we can generate the optimal learning parameters using gradient descent algorithm.

Log loss cost fn

$\hat{y}_i$

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

y_i: Actual value

$h_\theta(x)_i$: Predicted value obtained after applying the Sigmoid transformation

⇒ Gradient Descent Curve.

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

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

Here y represents the actual value

Final Aim : Minimize cost fn $J(\theta_0,\theta_1)$ by changing θ_0, θ_1

Similar to what we did in Linear regression

Convergence

Repeat until convergence

{

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

}

Old learning parameter

This will give the tangent or slope which will give ideal in which direction we need to move:
1. down the hill from left side of global maxima (-ve slope value)
2. down the hill from right side of global maxima (+ve slope value)

New learning parameter

This is the learning parameter that represents the step size by which by should move down the hill based on the tangent or slope defined.

Logistics Regression With Regularization Parameters

This part is similar to what we discussed in Linear regression.

Cost fn

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

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

[Reduce Overfitting]

$$J(\theta_0,\theta_1) = -y\log(h_\theta(x)) - (1-y)\log(1-h_\theta(x)) + \alpha_2 \text{ Regularization}$$

$$J(\theta_0,\theta_1) = -y\log(h_\theta(x)) - (1-y)\log(1-h_\theta(x)) + \alpha_1 \text{ Reg} \quad \text{[Feature Selection]}.$$

$$J(\theta_0,\theta_1) = -y\log(h_\theta(x)) - (1-y)\log(1-h_\theta(x)) + \alpha_2 \text{ Reg} + \alpha_1 \text{Reg}.$$

L2 Regularization $\Rightarrow$ Reduce Overfitting $\Rightarrow$ Ridge Regression.

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

L1 Regularization $\Rightarrow$ feature selection $\Rightarrow$ Lasso

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

Elastic Net

$$J(\theta_0,\theta_1) = -y\log(h_\theta(x)) - (1-y)\log(1-h_\theta(x)) + \lambda_1 \sum_{i=1}^{n} (\text{slope})^2 + \lambda_2 \sum_{i=1}^{n} |\text{slope}|.$$

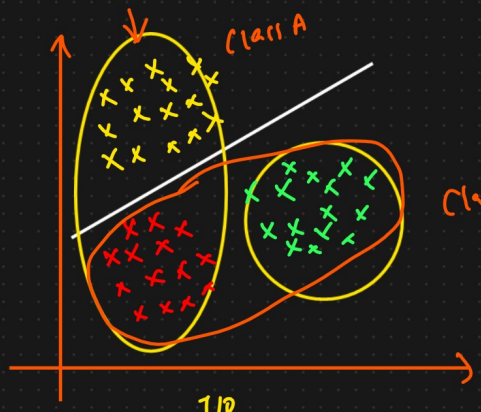
$C \propto \frac{1}{\lambda}$

C

Multiclass Logistic Regression

Multiple logistic reg ex doing 3 best fits internally.

In logistic regression is output feature:
-->is indexes (0/1) then multinomial classification.
-->is probability then OVR(One Versus Rest) classification.



Multinomial => Index of o/p

OVR {One Versus Rest} => Probabilities of all categories.

label =>

	M1	M2	M3
A	1	0	0
B	0	1	0
C	0	0	1
C	0	0	1
B	0	1	0

O/P

OVR

M1 M2 M3 → [0.20, 0.30, 0.50] → C

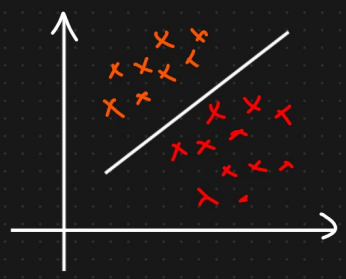
One Versus rest(OVR):
M1, M2 and M3 represents the probability of a data point getting classified as A, B and C respectively. Here, since M3 is highest so the probability of any datapoint to get classified as C is maximum.

For multinomial classification(A,B and C) this is how we can represent them in the form of 0/1 index.

A → 900
B → 100
} Imbalance dataset
⇒ 1000

{B : 10}

Through multiclass attribute we are defining multinomial or OVR. Read once from official documentation of sklearn for logistic reg.

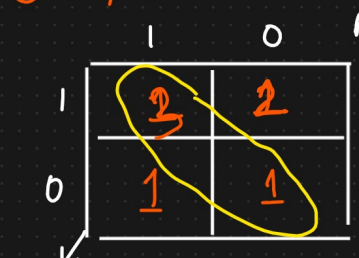


Performance metrics are the metrices that are used for determining the performance of the ML based learning model

Performance Metrics, Accuracy, Precision, Recall, F-Beta.

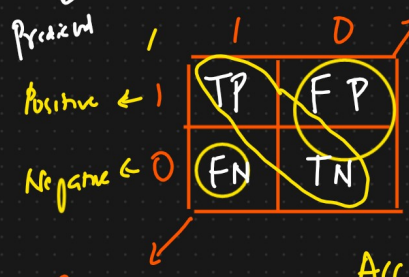
Imp NOTE:
Performance metric like accuracy, Precision, Recall and F Beta score is only applicable to Classification problem statement and not in case of regression problem statement.

(1) Confusion Matrix



In confusion matrix left diagonal elements represents true values.

x_1	x_2	y	ŷ	
		0	1	→ Wrong
		1	1	→ Correct
		0	0	
		1	1	
		1	1	
		0	1	
		1	0	



FP & FN Are Errors

For Eg; TN then T represents actual value and N represents the predicted value.

$$\text{Acc} = \frac{TP+TN}{FP+TP+FN+TN} = \frac{4}{7} = 57.1\%$$

Accuracy

3+2+1+1

DATASET : Imbalance Dataset

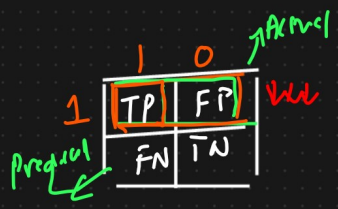
1000 datapoints → 900 1's, 100 0's → Imbalanced → Acc = 90%

In this case we have around 1000 datapoints from which around 900 are classified as 1 and only 100 are classified as 0. Due to this data imbalance trained model will most likely behave in a biased manner that is even if classification belongs to 0 category model may falsely predict it as 1

If we calculate the accuracy in this case it will come around 90%. Although accuracy is high but in this case where there is huge data imbalance accuracy cannot be used as an only parameter to determine the performance of the ML model.

That's why we are also using other learning parameters such as Precision and Recall to judge the performance of model instead of just relying on accuracy score.

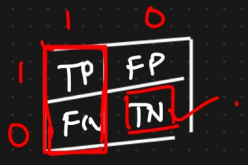
→ Dumb Model → $\hat{y} \to 1$



When false positive is important we use Precision where we want to reduce false positive.

③ Precision : $\frac{TP}{TP+FP}$ } Out of all the actual values how many are correctly predicted

↑↑ FP is Important ↓↓ FP



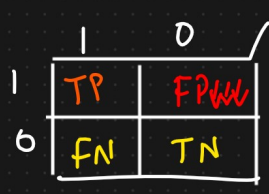
When false negative is important we use Recall where we want to reduce false negative.

④ Recall : $\frac{TP}{TP+FN}$ } Out of all the predicted values how many are correctly predicted with actual values.

↓ FN ↓↓↓ ⇒ Reduce FN

Usecase 1 : Spam classification.

Note: Relate this example with the email box(gmail/outlook)



1= Spam
0=Not spam

Text ⇒ Model ⇒ Spam / Not Spam.

FN↓↓↓

Wrong Scenario { Text → Spam, Model → Not a Spam } ↑↑ FN

Accurate. { Text ⇒ Spam, Model ⇒ Spam } ⇒ good Scenario.

If, text is spam and model is predicting it as not spam then it is acceptable. That is FN is not that important.

Blunder. { Text → Not a Spam, Model ⇒ Spam } ↑↑ FP

{ Text ⇒ Not a Spam, Model ⇒ Not a Spam } ⇒ Accurate

But if text is not spam and model is predicting it as spam then this case will be blunder and unacceptable. That is FP is important as compared to earlier case and we should try reducing it.

Usecase : FN is Important

To predict whether a person has diabetes or Not

Actual → Diabetes
Model → Diabetes
} Correct :) TP

Actual :) No Diabetes
Model :) Diabetes
} FP Wrong Prediction.

Actual → No Diabetes
Model → No Diabetes
} Correct :) TN

Actual :) Diabetes
Model :) No Diabetes
} FN Blunder

I was joking

Gulab Jamoon

Assignment : Tomorrow the Stock Market will crash or Not

Reduce FP or FN

1. Protection of people = Reduce FN
2. Protection of companies = Reduce FP

✓ F-Beta Score

1. Precision and Recall are valuable metrics that are widely used across the industry. Still, they have a massive drawback. They produce two values that must be first analyzed separately and then together to better understand an algorithm's performance. So, to overcome this disadvantage, Data Scientists came up with a way to combine Precision and Recall into an aggregate quality metric.

2. To define the term, in Machine Learning, the F-beta score (or F-measure) is a Classification metric featuring a harmonic mean of Precision and Recall. To evaluate a Classification model using the F-beta score, you need to have:
   -->The ground truth classes;
   -->And the model's predictions.

https://hasty.ai/docs/mp-wiki/metrics/f-beta-score#:~:text=In%20the%20F%2Dbeta%20score,the%20measured%20value%2C%20the%20better.

Follow above link to learn more about f beta score.

I need to prepare a ipnb notebook discussing performance metrices(precision, recall, Fbeta score)