Logistic Regression

To understand Logistic Regression, a well understanding of Linear Regression is required....

Refer Linear Regression here: https://github.com/VaradBelwalkar/helpful-tutorials/tree/main/machine_learning/Linear_Regression

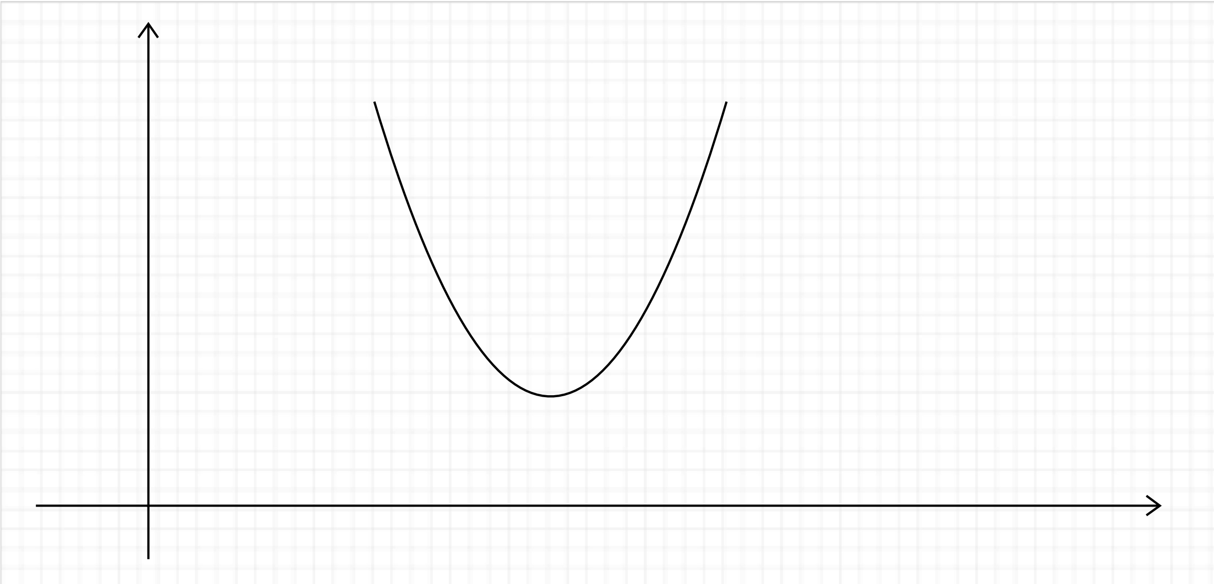
First of all why we need Logistic Regression?
We all know that linear regression finds the best fit line by reducing the cost function using the gradient descent approach and then adjusting the parameters using convergence theorem.

Here, the Convergence theorem i.e

$$z_1^0 = z_0^0 - \alpha * \frac{\delta J\left(z^0, z^1, \,\ldots\ldots\, , z^n\right)}{\delta z^0}$$

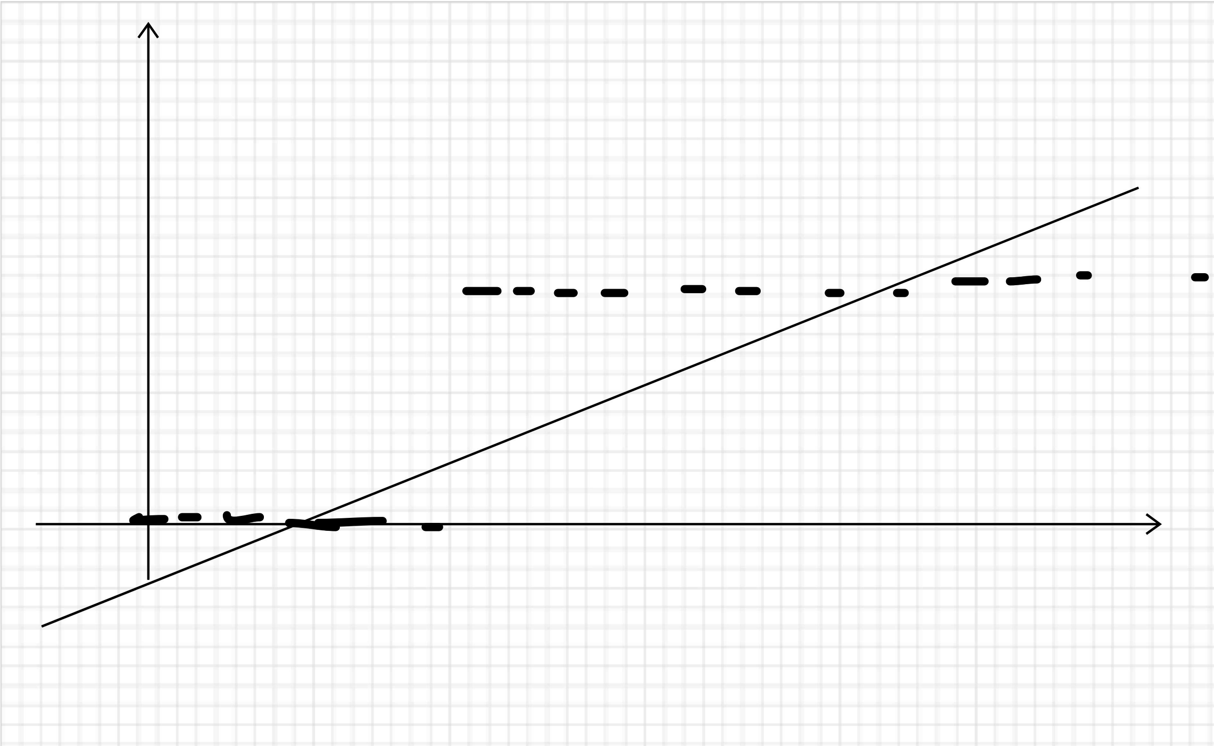
here, z^0 is a single feature,
z_1^0 is the new updated featured value,
z_0^0 is the old value,
and there is partial derivative of cost function with respect to z^0 treating other features as constant

We also know that for the equation of line, when trying to fit towards the number of points, of course analytically we can understand that the cost function for this will be convex one, i.e something like this:



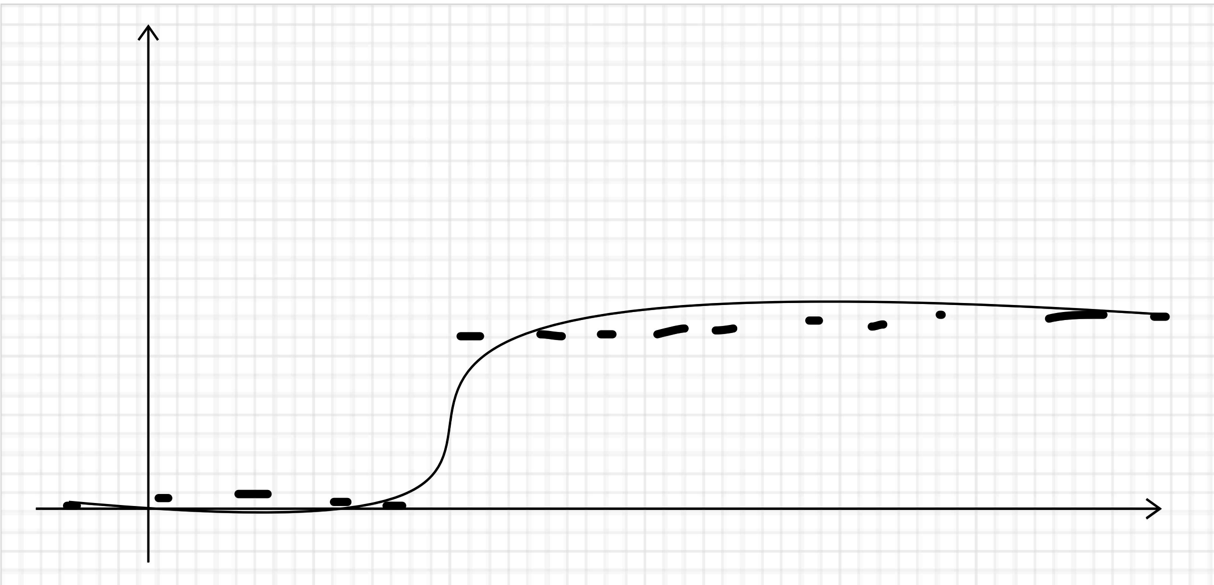
As we can see that there is only one minima and is itself Global minima, so working with the simple MSE cost function has no issue as the gradient descent curve will be convex for sure.

Sure we can use Linear Regression with the binary data points, but as you might have seen many fit lines trying to fit such points, fail to classify and also fails to deal with outliers
The failed approach with Linear Regression is something like this



Here, first of all, sorry for my bad point notations :)
But you got the point, and if some outlier comes, then this line is entirely going to get oriented somewhere else.

The most suitable plot i think should be like:



So, the only function coming in my mind which has similar curve is a SIGMOID Function,

$$f(x) = \frac{1}{1 - e^{-x}}$$

and, this is now, our hypothesis which will predict values, but wait, this is the standard function which will always output same thing!
So, what needs to get changed?
Of Course, tell me what can be changed? of course value in the numerator can be changed, in denominator can be changed, and most importantly, the power of e can be changed!

So why touch numerator and part of denominator, simply change the power!

fine, so our new expected cost function looks like this:

$$f(x) = \frac{1}{1 - e^{-(a + bx)}}$$

Here, the parameters a and b will take care of tuning the hypothesis so that it can correctly fit the given points!

Wait, what is that power? isn't that the same linear regression hypothesis?
Yes, we simply feed the general hypothesis of linear regression to the power of e

Now, after we have got the hypothesis for the Logistic Regression, its time to find cost function in order to fine tune the parameters using again gradient descent.
But can we use the same MSE cost function?
No, the cost function we are going to use here is:

$$\begin{array}{ll} g(x) = -log(f(x)) & for\ y = 1 \\ g(x) = -log(1 - f(x)) & for\ y = 0 \end{array}$$

$here,\ the\ terms\ "\ for\ y\ =\ "\ refer\ to\ the\ value\ calculated\ at\ the\ instance\ of\ the\ original\ value$

$$Combining\ these\ together$$

$$g(x) \ = \ -y * log(f(x)) \ - \ (1-y) * log(1-f(x))$$

Putting value of $f(x)$

$$g(x) \ = \ -y * log\left(\frac{1}{1- \ e^{-(a \ + \ bx)}}\right) \ - \ (1-y) * log\left(1-\frac{1}{1- \ e^{-(a \ + \ bx)}}\right)$$

This function is for calculating loss for single feature set

So, finding average,

$$g(x) \ = \ - \ \frac{1}{N}\sum_{i=1}^{N}\left\{y * log\left(\frac{1}{1- \ e^{-(a \ + \ bx)}}\right) \ + \ (1-y) * log\left(1-\frac{1}{1- \ e^{-(a \ + \ bx)}}\right)\right\}$$

This is our final cost function!
which is also called as cross entropy loss.

Differentiating w.r.t a,

$$\frac{d\,g(x)}{da} \ = \ -\frac{\left(\mathrm{e}^{bx}y-\mathrm{e}^{bx}\right)\mathrm{e}^{a}-y}{\mathrm{e}^{a+bx}-1}$$

Differentiating w.r.t b,

$$\frac{d\,g(x)}{db} \ = \ -\frac{x\cdot\left(\left(\mathrm{e}^{a}y-\mathrm{e}^{a}\right)\mathrm{e}^{xb}-y\right)}{\mathrm{e}^{xb+a}-1}$$

Well, implementing this in the code is just another day's task, we can follow much better approach!

The differential of our hypothesis can be reduced to,

$$\frac{d\,\sigma(x)}{dx} = \frac{d}{dx}\left\{\frac{1}{1 - e^{-(a+bx)}}\right\}$$

$$To,$$

$$\frac{d\,\sigma(x)}{dx} = \sigma(x) * (1 - \sigma(x))$$

$$This\ will\ be\ helpful\ much\ more.$$

$$So,\ using\ this,\ finding\ partial\ derivative\ w.r.t\ a\ and\ b\ becomes\ simpler\ than\ ever,$$

$$where\ as\ compared\ to\ above,\ we\ get\ the$$

$$\frac{d\,g(x)}{da} = -\frac{1}{N}\sum(y_i - y_{pred})$$

$$\frac{d\,g(x)}{db} = -\frac{1}{N}\sum(y_i - y_{pred}) * x$$

$$Where,$$

$$y_{pred} = \frac{1}{1 - e^{-(a+bx)}}$$

(You can refer in-depth guide of how exactly partial derivative is simplified here: https://medium.com/analytics-vidhya/derivative-of-log-loss-function-for-logistic-regression-9b832f025c2d)

Here, you will notice one thing for sure, the partial derivatives of 'a' and 'b' is in-fact, equal to the same calculated with linear regression!

This is the beauty of this cost function!

So, once you initialize the parameters in the code, and start the fitting, you simply need to calculate the y_{pred} by putting the feature set values into the hypothesis, and after getting predictions,
simply using them to calculate latest differentials and then updating the parameters using the convergence theorem! that's it!

So, you in real, don't need understand the derivation of cost function, but it is indeed good thing to know :)

You can find teh **CODE** here : https://github.com/VaradBelwalkar/helpful-tutorials/tree/main/machine_learning/Logistic_Regression

If you found the content useful, consider following me on my github: https://github.com/VaradBelwalkar