Logistic Regression

Abstract

This report consists of my basic understanding of one of the popular Ml methods "Logistic Regression".

1 Logistic Regression

Logistic Regression is a ML model which adds probability into the discussion.It comes under Supervised learning, a subcategory of machine learning where the dataset has labels and it will form a model which try to predict labels of new input data.

2 When we should use this?

Logistic Regression is employed to predict how likely a binary outcome(0 or 1) can be observed given an input data.Basically it means that our dependent variable(y) has only two outcomes possible, for example, a patient has cancer or not,kohli hits a century or not,whether iam able to complete this report in time or not.So eventhough logistic regression try to predict a real number(*probability*),it is fair to say that it is used for doing binary classification(a task of classifying the input data into two groups/classes).

3 How we will do this?

Similar to linear regression first we will try to find y value from the linear combination of weights and input parameters.The difference between linear and logistic is in linear we want to predict the y value itself but in logistic we will modelled the output being 0 or 1 rather a numeric value.We will use h as linear combination of weights and input parameters since we used notation y as a final label of input data.

$$h = w_1x_1 + w_2x_2 + w_3x_3 + ... + w_mx_m + \theta \tag{3.1}$$

where $(w_1, w_2, .., w_m)$ are the weights of input vector $(x_1, x_2, ..x_m)$ with m be the no. of features(a necessary parameter to predict the output)
θ is the bias(this factor is used shift the curve up and down like an intercept of a line)

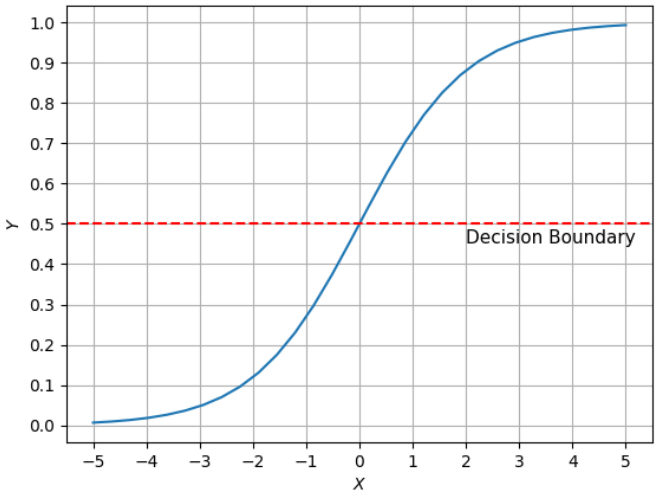

Fig. 4.1: The sigmoid function

4 The sigmoid function

Now to get the probability from this h one can use many function but the popular one is sigmoid function,i.e.,

$$\Pr(h) = \frac{1}{1+e^{-h}} \tag{4.1}$$

Now we will see the graph of sigmoid function, As we can see the output we are getting is in between 0 to 1 and for $h > 0$ the value is greater than .5 and for $h < 0$ the value is less than .5.So if we manage to get $h > 0$ whenever the input label is 1, we can correlate this sigmoid output value as the probability of getting label 1.So with the $\Pr(h) > 0.5$ we will predict label as 1 and for $\Pr(h) < 0.5$ we will predict label as 0.So our main task here is to modify the weights and labels in such a way that $h > 0$ whenever the true label is 1 and $h < 0$ whenever the true label is 0.

5 Cost Function

As you already realised we always to want minimize the error in predicting the label while modifying weights and bias.But in this case,it is not quite similar to what we did in linear regression.

Now lets put our heads straight,are main goal is to make $\Pr(h(x))$ close to 1 when the actual label of x is 1 similarly close to 0 when the actual label of x is 0.It is similar to say that we want to make maximize $\log(\Pr(h(x)))$ when the label of x (y) is 1 since

log is monotonic function but real reason that we are using it will be discussed in your higher classes just kidding!! , we will discuss that in gradient descent section.
So coming to the topic, it is similar to say that we want minimize $-\log(\Pr(h(x)))$, the thing is whenever we say cost function we want to minimize that,so the overall cost function for a input vector x looks like,

$$J(x) = \begin{cases} -\log(\Pr(h(x))) & , when y = 1 \\ -\log(1 - \Pr(h(x))) & , when y = 0 \end{cases} \tag{5.1}$$

The same thing can be written as,

$$J(x) = -y \log(\Pr(h(\mathbf{w}, \theta))) - (1 - y) \log(\Pr(h(\mathbf{w}, \theta))) \tag{5.2}$$

Here I changed $h(x)$ to $h(\mathbf{w}, \theta)$ because they are the actual parameters and this notation useful for later purpose.
Now for a dataset $(\mathbf{x_i}, y_i)_{i=1}^{N}$, the cost function looks like,

$$J = \frac{1}{N}\left(\sum_{i=1}^{N} -y_i \left(\log(\Pr(h_i(\mathbf{w}, \theta))) - (1 - y_i) \log(\Pr(h_i(\mathbf{w}, \theta)))\right)\right) \tag{5.3}$$

Now we will try to minimize the cost function gradient descent method.

6 Gradient Descent

Before going to the method, first we will compare it with linear regression. The cost function of it looks like,

$$J = \sum_{i=1}^{N} \frac{(Y_i - y_i)^2}{2} \tag{6.1}$$

$$= \sum_{i=1}^{N} \frac{\left(Y_i - \left(\mathbf{w}^T X + \theta\right)\right)}{2} \tag{6.2}$$

Now if we plotted the graph w.r.t weights and J it looks like Fig 6.2,
As you can see since it is convex graph using gradient descent method is effective, we will move down to minima of cost function in each step. Now in the case of logistic, if we don't use log of $\Pr(h(\mathbf{w}, \theta))$,the plot between J and weights looks like,6.3
As you can see since their are many local minima, the gradient descent method may fail to reach the global minima it entirely depends on the starting weight.That the reason why we replace it with log term. As we can see here, for $y = 1$ if probability tends to 1 the cost tend to 0 and if probability tends to 0 the cost tends to infinity.Similar resemblance can be seen for $y = 0$. This makes easier for gradient descent method to

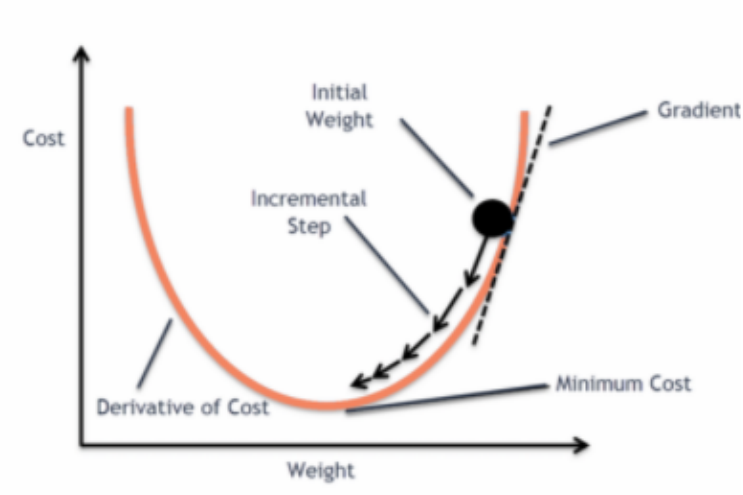

Fig. 6.2: convex curve

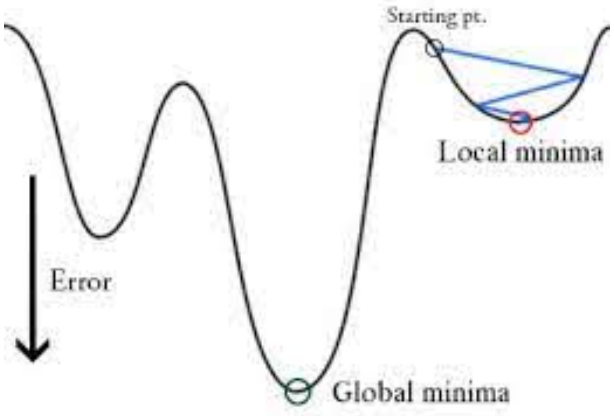

Fig. 6.3: Non convex

reach the minimum.So in each iteration we will modify the weights and bias with α as learning rate as,

$$\mathbf{w_k} = \mathbf{w_k} - \alpha \frac{\partial J}{\partial w_k} \tag{6.3}$$

$$\theta_k = \theta_k - \alpha \frac{\partial J}{\partial \theta_k} \tag{6.4}$$

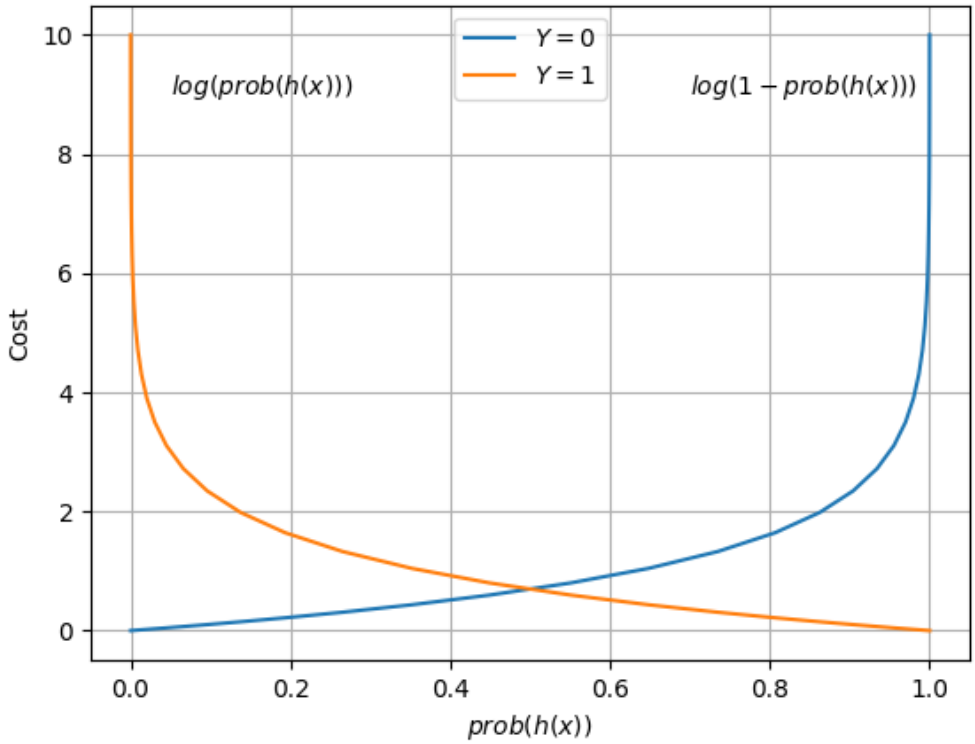

Fig. 6.4: $prob(h(x))$ vs cost

where w_k and θ_k are weight vector and bias after k^{th} iteration. Now we will understand (6.3), let say $\mathbf{w} = \begin{pmatrix} w_1 \\ w_2 \\ w_3 \\ . \\ . \\ w_m \end{pmatrix}$, now we want to modify each of these weights in a iteration, to do that first we will simplify the cost function,

$$J = \frac{1}{N}\left(\sum_{i=1}^{N} -y_i \log\left(\Pr\left(h_i\left(\mathbf{w}, \theta\right)\right)\right) - \left(1 - y_i\right) \log\left(\Pr\left(h_i\left(\mathbf{w}, \theta\right)\right)\right)\right) \tag{6.5}$$

Using (4.1),

$$J = \frac{1}{N}\left(\sum_{i=1}^{N} -y_i \log\left(\frac{1}{1+e^{-h_i}}\right) - (1-y_i)\log\left(1 - \frac{1}{1+e^{-h_i}}\right)\right) \tag{6.6}$$

$$= \frac{1}{N}\left(\sum_{i=1}^{N} y_i \log\left(1+e^{-h_i}\right) + (1-y_i)\log\left(1+e^{h_i}\right)\right) \tag{6.7}$$

Now if we do the paritial derivative w.r.t w_j where $j \in \{1, 2, ...m\}$,

$$\frac{\partial J}{\partial w_j} = \frac{1}{N}\left(\sum_{i=1}^{N} y_i \frac{1}{1+e^{-h_i}} - e^{-h_i}\frac{\partial h_i}{\partial w_j} + (1-y_i)\frac{1}{1+e^{h_i}} e^{h_i}\frac{\partial h_i}{\partial w_j}\right) \tag{6.8}$$

Now we know that,

$$h_i = \mathbf{w^T x_i} + \theta \tag{6.9}$$

$$= w_1 x_{1i} + w_2 x_{2i} + .. + w_j x_{ji} .. w_m x_{mi} + \theta \tag{6.10}$$

$$\frac{\partial h_i}{\partial w_j} = x_{ji} \tag{6.11}$$

Here the notation x_{ji} can be understood as the j^{th} feature of i^{th} input vector. Using (6.11),we can modify (6.8) as,

$$\frac{\partial J}{\partial w_j} = \frac{1}{N}\left(\sum_{i=1}^{N} y_i \frac{-1}{e^{h_i}+1} x_j i + (1-y_i)\frac{e^{h_i}}{1+e^{h_i}} x_j i\right) \tag{6.12}$$

$$= \frac{1}{N}\left(\sum_{i=1}^{N} x_{ji}\left(y_i(-1) + \frac{1}{1+e^{-h_i}}\right)\right) \tag{6.13}$$

$$= \frac{1}{N}\left(\sum_{i=1}^{N} x_{ji}\left(-y_i + \Pr(h_i)\right)\right) \tag{6.14}$$

$$= \frac{1}{N}\left(\sum_{i=1}^{N} x_{ji}\left(\Pr(h_i) - y_i\right)\right) \tag{6.15}$$

So with α as learning rate we will modify each weight in k^{th} iteration as shown below,

$$w_{1k} = w_{1k} - \alpha \frac{1}{N} \left(\sum_{i=1}^{N} x_{1i} \left(\Pr\left(h_i\right) - y_i \right) \right) \tag{6.16}$$

$$w_{2k} = w_{2k} - \alpha \frac{1}{N} \left(\sum_{i=1}^{N} x_{2i} \left(\Pr\left(h_i\right) - y_i \right) \right) \tag{6.17}$$

$$\ldots$$

$$\ldots$$

$$w_{mk} = w_{mk} - \alpha \frac{1}{N} \left(\sum_{i=1}^{N} x_{mi} \left(\Pr\left(h_i\right) - y_i \right) \right) \tag{6.18}$$

Similarly solving (6.4) gives,

$$\theta_k = \theta_k - \alpha \frac{1}{N} \sum_{i=1}^{N} \left(\Pr\left(h_i\right) - y_i \right) \tag{6.19}$$

In this way we will modify weights and bias over the iterations in gradient descent method.

7 Regularization

Similar to linear regression, logistic regression also requires regularisation since it also assumes linear relationship(with a non-linear transform of output). The goal of regularization is to minimize the generalization error without effecting the training error.Often, the model tend to learn too perfectly which leads to problem of overfitting.To avoid that we add square of norm of β to the cost function.

$$\beta = \begin{pmatrix} \mathbf{w} \\ \theta \end{pmatrix} \tag{7.1}$$

This shrinks the weights/coefficients of the model without disturbing the training error of the model.This is necessary because in our data there may lie some o utliers which may helps the model more flexible (outliers are data points which don't follow normal trend) but at the risk of overfitting.
The new cost function looks like,

$$J = \frac{1}{N} \left(\sum_{i=1}^{N} -y_i \left(\log\left(\Pr\left(h_i\left(\mathbf{w}, \theta\right) \right) \right) - \left(1 - y_i\right) \log\left(\Pr\left(h_i\left(\mathbf{w}, \theta\right) \right) \right) \right) \right) + \lambda \left\| \beta \right\|^2 \tag{7.2}$$

where, λ is a tuning factor which decides how much we want penalize the flexibility of the model.

8 Summary

In this report we discussed about,

1) What is logistic regression?
2) When we should use logistic regression?
3) The cost function of logistic regression
4) The gradient descent method for logistic regression.
5) Regularization of the model

9 Questions

1) How logistic regression is different from linear regression?.
2) Why do we need bias term in logistic regression?
3) Can you think of a function which can do same job as sigmoid function?
4) How the value of learning rate affects the training of the model?
5) How regularization is justified?