Question 1: Define Binary Cross Entropy as a cost function

Ans :

Binary Classification is a problem where we have to segregate our observations in any of the two labels on the basis of the features. Suppose you have some images and have to put each in a stack one for Dogs and the other for Cats. Here you are solving a binary classification problem.

The loss function tells how good your model is in predictions. If the model predictions are closer to the actual values the Loss will be minimum and if the predictions are totally away from the original values the loss value will be the maximum.

Binary cross entropy compares each of the predicted probabilities to the actual class output which can be either 0 or 1. It then calculates the score that penalizes the probabilities based on the distance from the expected value. That means how close or far from the actual value.

Binary Cross Entropy is the negative average of the log of corrected predicted probabilities.

Let us take an example:

Object	Actual	Predicted Probability	Corrected Probability	Log
OBJ_1	1	0.94	0.94	-0.0269
OBJ_2	1	0.90	0.90	-0.0458
OBJ_3	1	0.78	0.78	-0.1079
OBJ_4	0	0.56	0.44	-0.3565

Here, Predicted Probability indicates the probability of belonging to class 1 for an object. For OBJ_1, OBJ_2 and OBJ_3 predicted probability and corrected probability is the same. But for OBJ_4 as we can see the actual output is 0 and the predicted probability is 0.56 for class 1 . so the corrected probability will be (1-0.56) = 0.44.
We calculated the log value for each of the corrected probabilities. The reason behind using the log value is, the log value offers less penalty for small differences between predicted probability and corrected probability. when the difference is large the penalty will be higher.

Here we have calculated log values for all the corrected probabilities. Since all the corrected probabilities lie between 0 and 1, all the log values are negative.

In order to compensate for this negative value, we will use a negative average of the values -

$$-\frac{1}{N}\sum_{i=1}^{N}(\log p_i)$$

The value of the negative average of corrected probabilities we calculate comes to be 0.134 which is our Log loss or Binary cross-entropy for this particular example.

Further, instead of calculating corrected probabilities, we can calculate the Log loss using the formula given below.

$$Log\ loss = \frac{1}{N}\sum_{i=1}^{N} -(y_i * log(p_i) + (1 - y_i) * log(1 - y_i))$$

Here, pi is the probability of class 1, and (1-pi) is the probability of class 0. When the observation belongs to class 1 the first part of the formula becomes active and the second part vanishes and vice versa in the case observation's actual class are 0. This is how we calculate the Binary cross-entropy.
If you are dealing with a multi-class classification problem you can calculate the Log loss in the same way. Just use the formula given below.

$$logloss = -\frac{1}{N}\sum_{i}^{N}\sum_{i}^{M} y_{ij} log(p_{ij})$$

- N is the number of samples
- M is the number of classes

Question 2 : Difference between adam optimizer & gradient descent

Ans:

Gradient Descent is an optimization algorithm used for minimizing the cost function in various machine learning algorithms. **In Gradient Descent algorithm the learning rate α is fixed .**
Learning rate : It is a parameter that provides the model a scale of how much model weights should be updated.
Cost function : A cost function is used to calculate the cost that is the difference between the predicted value and the actual value.
Gradient descent works best for most purposes. However, it has some downsides too. It is expensive to calculate the gradients if the size of the data is huge. Gradient descent works well for convex functions but it doesn't know how far to travel along the gradient for nonconvex functions.

Gradient Descent algorithms are of 3 types :

- Batch Gradient Descent
- Mini Batch Garident Descent
- Stochastic Gradient Descent

Adam optimizer tried to improve several things from normal gradient Descent algorithm which includes -

- instead of taking the whole dataset for each iteration, we randomly select the batches of data. That means we only take few samples from the dataset.
- Learning rate α is not fixed
- It tends to focus on faster computation time

The adam optimizer has several benefits, due to which it is used widely. It is adapted as a benchmark for deep learning papers and recommended as a default optimization algorithm. Moreover, the algorithm is straightforward to implement, has faster running time, low memory requirements, and requires less tuning than any other optimization algorithm.

So basically adam optimizer is a modified version of stochastic gradient descent algorithm where computation time is faster, learning rate is not fixed , doesn't take all data samples , works better on non convex function .