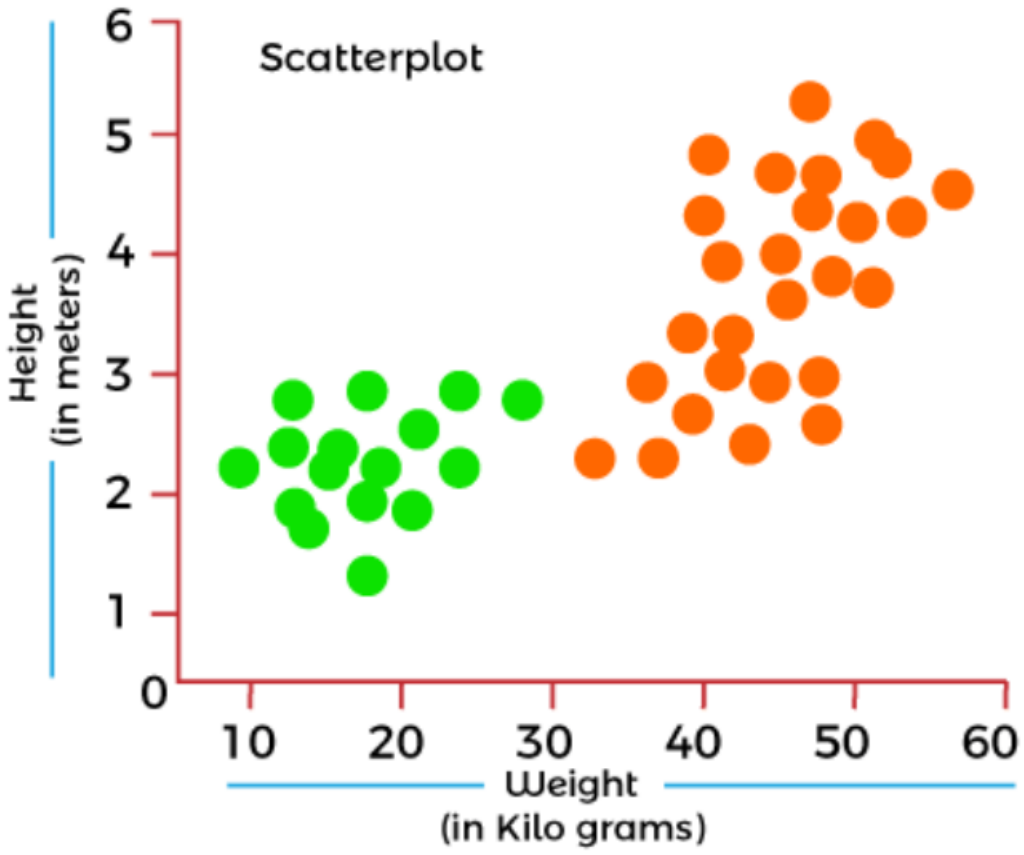
What is Cost Function?

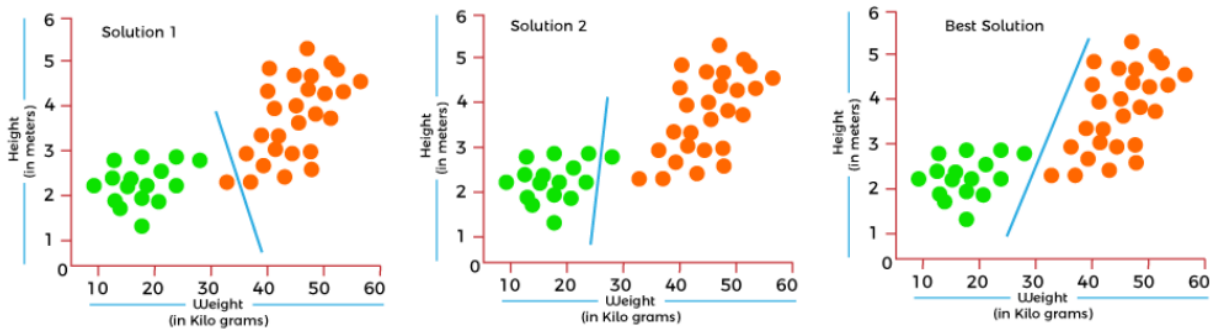
A cost function is an important parameter that determines how well a machine learning model performs for a given dataset. It calculates the difference between the expected value and predicted value and represents it as a single real number.

Why use Cost Function?

While there are different accuracy parameters, then why do we need a Cost function for the Machine learning model. So, we can understand it with an example of the classification of data. Suppose we have a dataset that contains the height and weights of cats & dogs, and we need to classify them accordingly. If we plot the records using these two features, we will get a scatter plot as below:



In the above image, the green dots are cats, and the yellow dots are dogs. Below are the three possible solutions for this classification problem.



In the above solutions, all three classifiers have high accuracy, but the third solution is the best because it correctly classifies each datapoint. The reason behind the best classification is that it is in mid between both the classes, not close or not far to any of them.

To get such results, we need a Cost function. It means for getting the optimal solution; we need a Cost function. It calculated the difference between the actual values and predicted values and measured how wrong was our model in the prediction. By minimizing the value of the cost function, we can get the optimal solution.

Cost functions or Loss function:

Why Cost Functions in Machine Learning

During the training phase, the model assumes the initial weights randomly and tries to make a prediction on training data. But how will the model get to know how much "far" it was from the prediction? It definitely needs this information so that it can adjust the weight accordingly (using gradient descent) in the next iteration on training data.

Cost function is a function that takes both predicted outputs by the model and actual outputs and calculates how much wrong the model was in its prediction.

Different types of cost functions:

Types of Cost Function

Cost functions can be of various types depending on the problem. However, mainly it is of three types, which are as follows:

1. Regression Cost Function
2. Binary Classification cost Functions
3. Multi-class Classification Cost Function.

1. Regression Cost Function

Regression models are used to make a prediction for the continuous variables such as the price of houses, weather prediction, loan predictions, etc. When a cost function is used with Regression, it is known as the "Regression Cost Function." In this, the cost function is calculated as the error based on the distance, such as:

1. Error= Actual Output-Predicted output

There are three commonly used Regression cost functions, which are as follows:

a. **Means Error**
b. **Mean Squared Error (MSE)**
c. **Mean Absolute Error (MAE)**

Now let us see what are the different types of cost functions in machine learning.

Cost functions for Regression problems

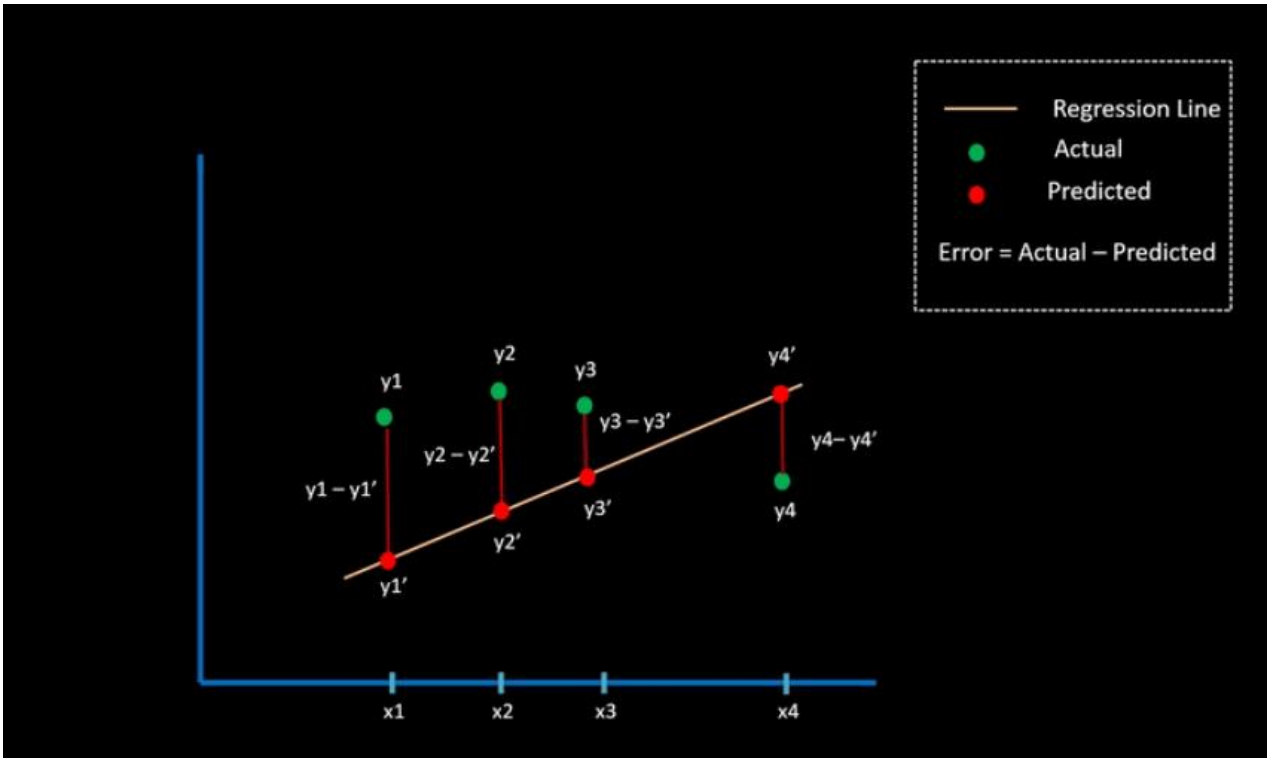
In regression, the model predicts an output value for each training data during the training phase. The cost functions for regression are calculated on distance-based error. Let us first understand this concept first.

Initial Concept – Distance Based Error

Let us say that for a given set of input data, the actual output was y and our regression model predicts y' then the error in prediction is calculated simply as

$$Error = y-y'$$

This also known as distance-based error and it forms the basis of cost functions that are used in regression models. Below animation gives a more clear geometrical interpretation of distance-based error.



Mean Error (ME)

In this cost function, the error for each training data is calculated and then the mean value of all these errors is derived. Calculating mean of the errors is the simplest and most intuitive way possible. But there is a catch here.

Y (Actual)	Y' (Predicted)	Error = Y-Y'
10.2	9.4	0.8
3.5	1.7	1.8
7.1	6.9	0.2
14.5	15.4	-0.9
17.2	18.4	-1.2
9.5	11.3	-1.8
2.7	2.5	0.2
11.5	11.1	0.4
5.9	6.7	-0.8
15.3	15.2	0.1
Sum		-1.2
Mean Error		(-1.2/10) = -0.12

The errors can be both negative or positive and during summation, they will tend to cancel each other out. In fact, it is theoretically possible that the errors are such that positive and negatives cancel each other to give zero error mean error for the model.

So Mean Error is not a recommended cost function for regression. But it does lay the foundation for our next cost functions.

Mean Squared Error (MSE)

This improves the drawback we saw in Mean Error above. Here a square of the difference between the actual and predicted value is calculated to avoid any possibility of negative error.

So in this cost function, MSE is calculated as mean of squared errors for N training data.

MSE = (Sum of Squared Errors)/N

The below example should help you to understand MSE much better. MSE is also known as L2 loss.

Y (Actual)	Y' (Predicted)	Error = $(Y-Y')^2$
10.2	9.4	0.64
3.5	1.7	3.24
7.1	6.9	0.04
14.5	15.4	0.81
17.2	18.4	1.44
9.5	11.3	3.24
2.7	2.5	0.04
11.5	11.1	0.16
5.9	6.7	0.64
15.3	15.2	0.01
Sum		10.26
Mean Square Error		(10.26/10) = 1.02

c) Mean Absolute Error (MAE)

Mean Absolute error also overcome the issue of the Mean error cost function by taking the absolute difference between the actual value and predicted value.

The formula for calculating Mean Absolute Error is given below:

$$MAE = \frac{\sum_{i=0}^{n} |y - y'|}{n}$$

This means the Absolute error cost function is also known as **L1 Loss**. It is not affected by noise or outliers, hence giving better results if the dataset has noise or outlier.

Cost functions for Classification problems

Cost functions used in classification problems are different than what we saw in the regression problem above. **There is a reason why we don't use regression cost functions for classification problem and we will see it later.** But before that let us see the classification cost functions.

Initial Concept – Cross Entropy Intuition

The name cross entropy might give you a bummer at first, but let me give you a very intuitive understanding. Cross Entropy can be considered as a way to measure the distance between two probability distributions. (This is an intuitive understanding, however, as it does not really measure the distance between the probability distribution in mathematics term)

So how does cross entropy help in the cost function for classification? Let us understand this with a small example.

Let us consider that we have a classification problem of 3 classes as follows.

Class(Orange,Apple,Tomato)

The machine learning model will actually give a probability distribution of these 3 classes as output for a given input data. The class having the highest probability is considered as a winner class for prediction.

Output = [P(Orange),P(Apple),P(Tomato)]

The actual probability distribution for each class is shown below.

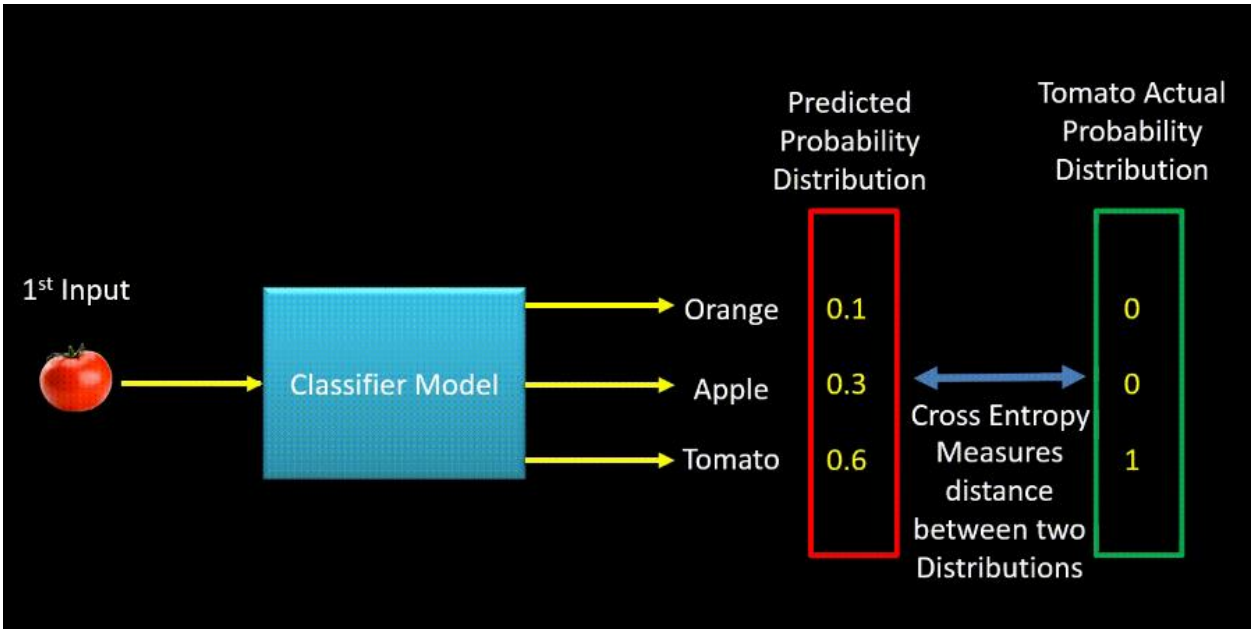
Orange = [1,0,0]

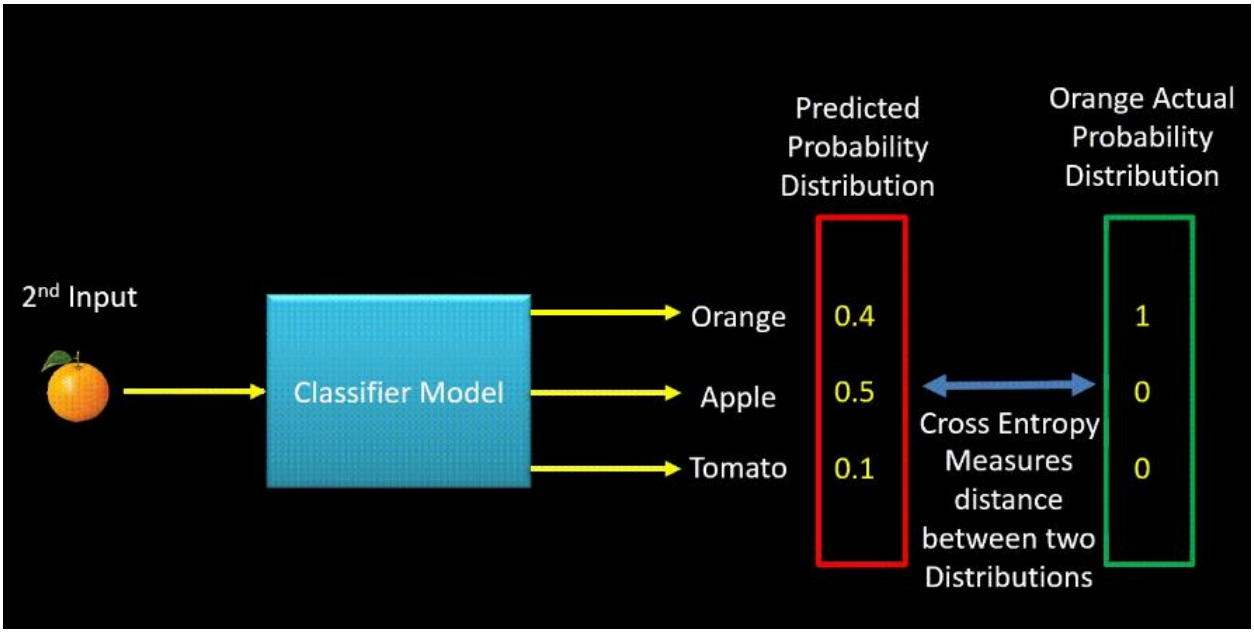
Apple = [0,1,0]

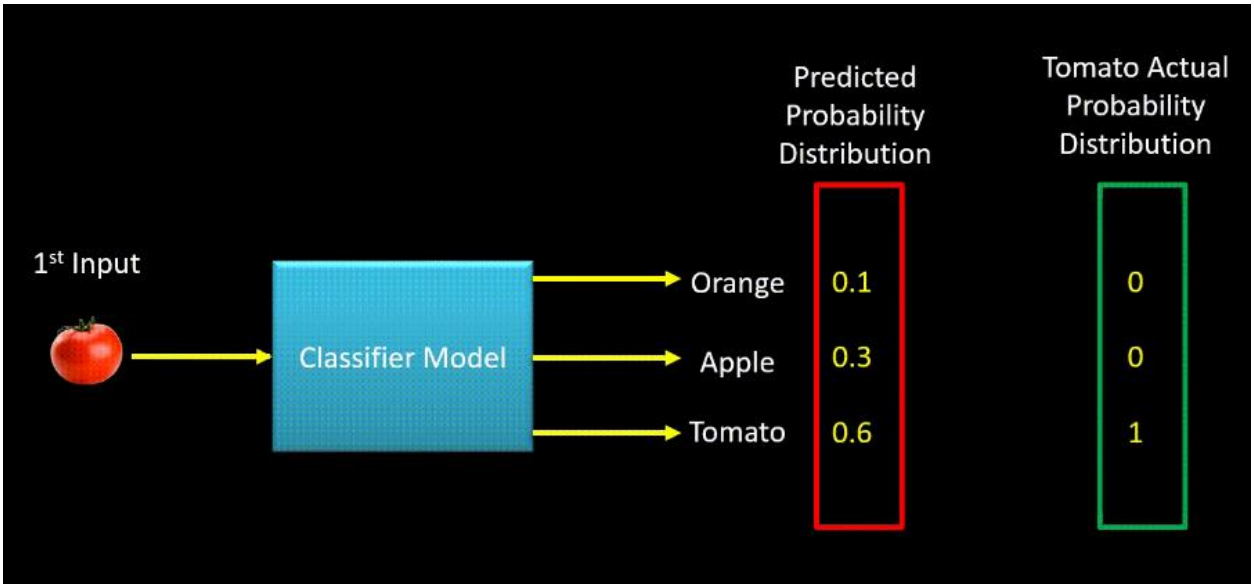
Tomato = [0,0,1]

During the training phase, for example, if the training data is Orange, the predicted probability distribution should tend towards the actual probability distribution of Orange. If predicted probability distribution is not closer to the actual one, the model has to adjust its weight.

This is where cross entropy becomes a tool to calculate how much far is the predicted probability distribution from the actual one.







Now with this understanding of cross entropy, let us now see the classification cost functions.

Categorical Cross Entropy Cost Function

This cost function is used in the classification problems where there are multiple classes and input data belongs to only one class.

Before defining the cost function let us first understand how cross entropy is calculated.

Let us assume the model gives the probability distribution as below for M classes for a particular input data D.

$$P(D) = [y1' , y2' , y3' \dots yM']$$

And the actual or target probability distribution of the data D is

$$A(D) = [y1 , y2 , y3 \dots yM]$$

Then cross entropy for that particular data D is calculated as

$$CrossEntropy(A,P) = – (y1*log(y1') + y2*log(y2') + y3*log(y3') + \dots + yM*log(yM'))$$

Let us quickly understand this with the help of above example where .

$$P(Orange) = [0.6, 0.3, 0.1]$$

$$A(Orange) = [1, 0, 0]$$

$$Cross\_Entropy(A,P) = – (1*Log(0.6) + 0*Log(0.3)+0*Log(0.1)) = 0.51$$

The above formula just measures the cross entropy for a single observation or input data.

The error in classification for the complete model is given by *categorical cross entropy* which is nothing but the mean of cross entropy for all N training data.

$$Categorical\_Cross\_Entropy = (Sum\ of\ Cross\ Entropy\ for\ N\ data)/N$$

The below example will help you understand Categorical Cross Entropy better.

Data	Actual Probability Distribution	Predicted Probability Distribution	Cross Entropy
Orange	[1,0,0]	[0.6,0.3,0.1]	- (1*Log(0.6) + 0*Log(0.3)+0*Log(0.1)) = 0.51
Orange	[1,0,0]	[0.9,0.1,0]	- (1*Log(0.9) + 0*Log(0.1)+0*Log(0)) = 0.1
Apple	[0,1,0]	[0.2,0.7,0.1]	- (0*Log(0.2) + 1*Log(0.7)+0*Log(0.1)) = 0.35
Tomato	[0,0,1]	[0.3,0.2,0.5]	- (0*Log(0.3) + 0*Log(0.2)+1*Log(0.5)) = 0.69
Apple	[0,1,0]	[0.6,0.1,0.3]	- (0*Log(0.6) + 1*Log(0.1)+0*Log(0.3)) = 2.3
Orange	[1,0,0]	[0.5,0.2,0.3]	- (1*Log(0.5) + 0*Log(0.2)+0*Log(0.3)) = 0.69
Tomato	[0,0,1]	[0.1,0.1,0.8]	- (0*Log(0.1) + 0*Log(0.1)+1*Log(0.8)) = 0.22
		Sum	0.51+0.1+0.35+0.69+2.3+0.69+0.22 = 4.76
		Cross Entropy Cost Function	4.67/7 = 0.68

Binary Cross Entropy Cost Function

Binary cross entropy is a special case of categorical cross entropy when there is only one output which just assumes a binary value of 0 or 1 to denote negative and positive class respectively

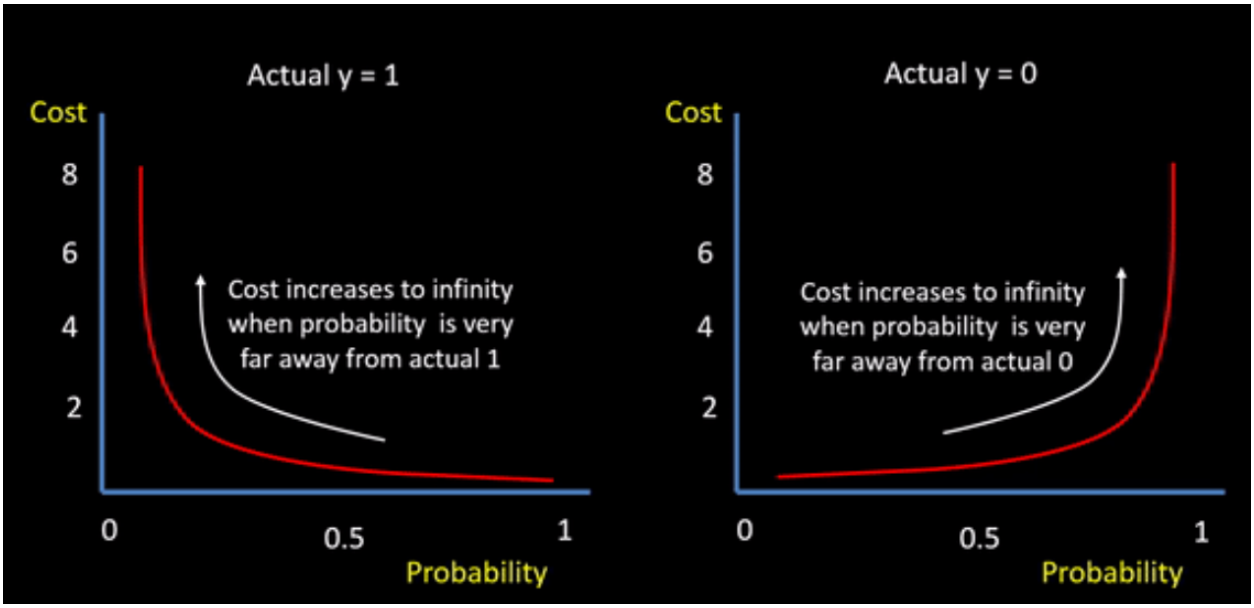
Let us assume that actual output is denoted by a single variable y, then cross entropy for a particular data D is can be simplified as follows –

$$cross\_entropy(D) = – y*log(y') \quad when\ y = 1$$

$$cross\_entropy(D) = – (1-y)*log(1-y') \quad when\ y = 0$$

The error in binary classification for the complete model is given by binary cross entropy which is nothing but the mean of cross entropy for all N training data.

$$Binary\_Cross\_Entropy = (Sum\ of\ Cross\_Entropy\ for\ N\ data)/N$$



Cost functions in machine learning, also known as loss functions, calculates the deviation of predicted output from actual output during the training phase.

- ✓ Cost functions are an important part of the optimization algorithm used in the training phase of models like logistic regression, neural network, support vector machine.
- ✓ There are many cost functions to choose from and the choice depends on type of data and type of problem (regression or classification).
- ✓ Mean squared error (MSE) and Mean Absolute Error (MAE) are popular cost functions used in regression problems.
- ✓ For classification problems, the models which give probability output mostly use categorical cross entropy and binary cross entropy cost functions.