Loss and Cost Function

CSE451:Neural Network & Fuzzy Logic

MD Tamim Hossain
Lecturer
Department of Computer Science and Engineering
Premier University

“You're going to go through tough times - that's life. But I say, 'Nothing happens to you, it happens for you.' See the positive in negative events..”

-**Joel Osteen**

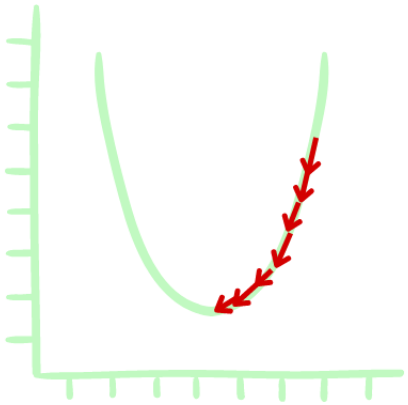

DREAM BIG IN
2024
HAPPY NEW YEAR

Loss Function vs Cost Function

- We usually consider both terms synonyms and can use them interchangeably.
- Loss function is associated with every training example
- Cost function is the average of the loss function values over all the data samples.
- In Machine learning, we optimize our cost rather than our loss function.

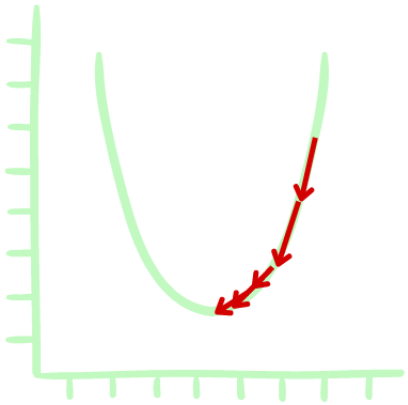
Quick Revision

Too low



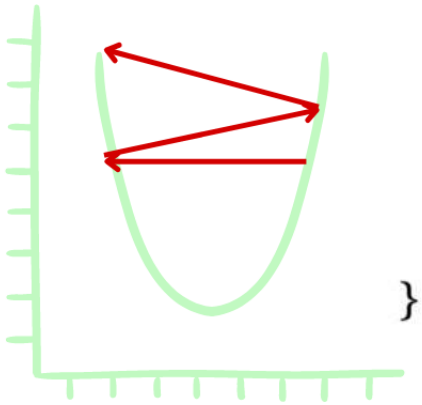
Smaller learning rate requires many updates, hence learning would be very slow

Just Right



Optimal learning rate smoothly reaches to the minima

Too Hight



Larger learning rate may lead to drastic updates and GD may diverge from the minima.

Repeat until convergence {

$$\theta_j := \theta_j - \alpha * \frac{\partial J(\theta)}{\partial \theta}$$

(for j=1 and j=0)

}

Loss Functions In Regression Problems

- **Square Error**
  - Also Called **L2 loss**

$$Loss_i = (Y_i - f(x_i))^2$$

- **Mean Squared Error (MSE)**

$$MSE = \frac{1}{n}\Sigma_i^n(Y_i - f(X_i))^2$$

Limitations of MSE

The difference between actual and predicted will be higher, and squaring that difference will make them even more prominent.

- So MSE is less robust to the outlier's presence, and it is advised not to use MSE when there are too many outliers in the dataset.

Absolute Error

- Also Called **L1 loss**

$$Loss_i = |Y_i - f(x_i)|$$

- **Mean Absolute Error(MAE)**

$$MAE = \frac{1}{n}\Sigma_i^n |Y_i - f(X_i)|$$

Limitations of MAE

It is a non-differentiable function, so we need more complicated techniques to find the gradients and update the parameters. When the cost values approach minimum, the gradient becomes undefined, making MAE unstable.

- It is also widely used in industries, especially when the training data is more prone to outliers

Loss Functions for Classification

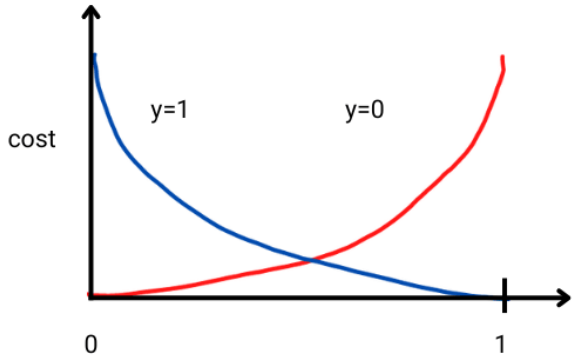
- **Binary Cross Entropy/log loss**
  - Used in binary classification problems like two classes

$$\text{Log Loss} = -\frac{1}{N}\sum_{i=1}^{N} y_i \log \hat{y}_i + (1-y_i)\log(1-\hat{y}_i)$$

Binary Cross Entropy/log loss

$$cost(y_p, y) = -(y \,.\, log(y_p) + (1 - y) \,.\, log(1 - y_p)$$

$$cost(y_p, y) = -log(y_p),\ if\ y = 1$$
$$= -log(1 - y_p),\ if\ y = 0$$



Categorical Cross Entropy

- Categorical Cross entropy is used for Multiclass classification and softmax regression.

$$\text{Loss} = -\sum_{j=1}^{K} y_j \log(\hat{y}_j)$$

where k is number of classes in the data

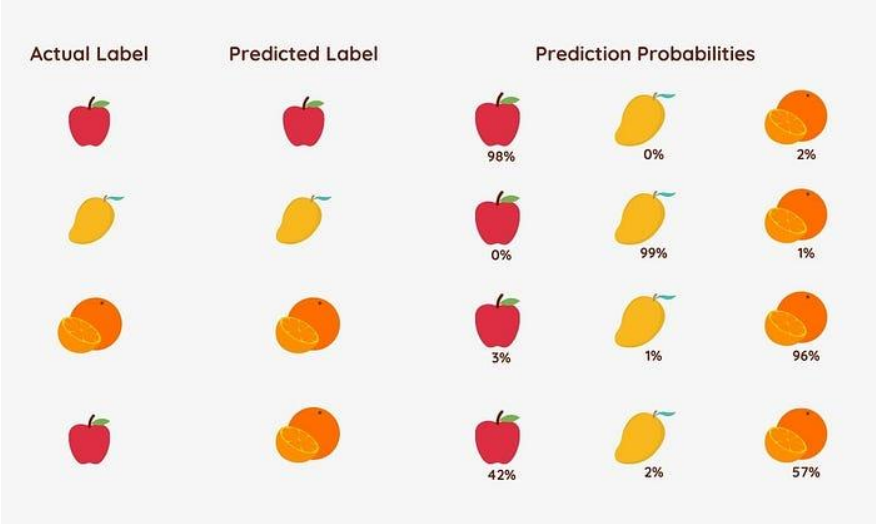
Categorical Cross Entropy

$$\text{Cost} = \frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{k}[y_{ij}\log(\hat{y}_{ij})]$$

- In multi-class classification at the last neuron use the softmax activation function

$$\sigma(\vec{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$$

Categorical Cross Entropy



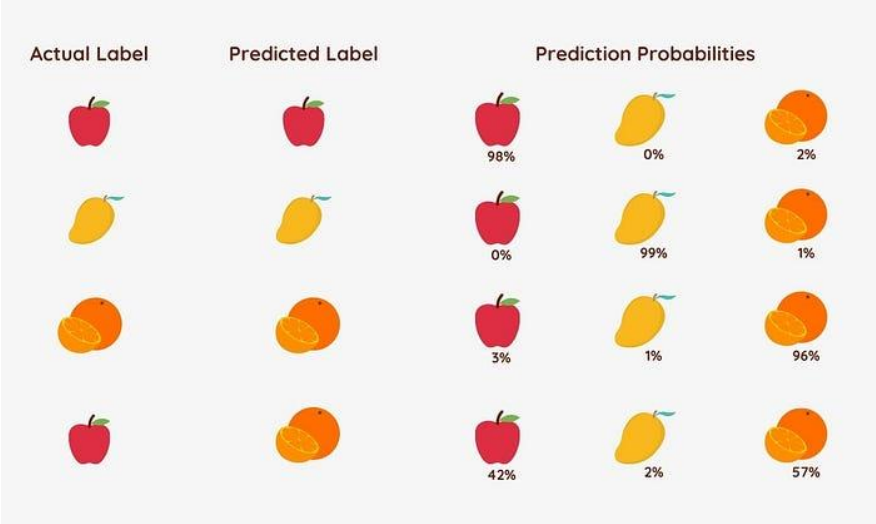
Actual Label	Predicted Label	Prediction Probabilities
[1, 0, 0]	[1, 0, 0]	[0.98 , 0 , 0.02]
[0, 1, 0]	[0, 1, 0]	[0 , 0.99 , 0.01]
[0, 0, 1]	[0, 0, 1]	[0.03 , 0.01 , 0.96]
[1, 0, 0]	[1, 0, 0]	[0.42 , 0.02 , 0.57]

one-hot encoding

Sparse Categorical Cross entropy

- The categorical cross-entropy becomes very memory in-efficient when we have a large number of classes, say 1000
- This means we have a large array of all zeros and a single 1
- In such cases, we use the **sparse categorical crossentropy** loss function
- Works on label-encoded data instead of one-hot encoded data
- Makes computation **very fast**

Sparse Categorical Cross Entropy



Actual Label	Predicted Label	Prediction Probabilities
1	1	[0.98 , 0 , 0.02]
2	2	[0 , 0.99 , 0.01]
3	3	[0.03 , 0.01 , 0.96]
1	3	[0.42 , 0.02 , 0.57]

Label-encoded