Today's Agenda

1) Loss Functions

2) Activation Functions

Backprop / Optimizers

What are loss functions?

Method of evaluating how well the algorithm performs on top of the dataset. $\{f(x) = x^2 + 2\}$

high → poor

low → good

L(parameters)

$Y = mx + C$

(m, c) are the parameters

Parameters = Weights + Bias

$(y - \hat{y})^2 = ?$

LOOP

For each training sample

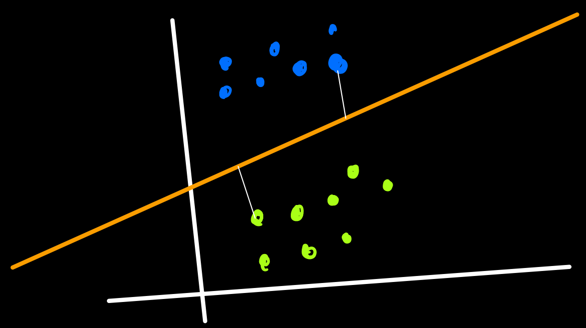
updating the line

update the parameters

Optimizers (Gradient Descent)

Condition to stop :- Least loss

Eye of the Algorithm



$L(m, c)$ = minimum

Errors

Far — More

Close — LESS

$(y-\hat{y})^2$

= not always positive

$= 0.7 + 0.1 + 0.2 - .7$

SAT | GRE | 12 | LPA

$y = 6.5$

$\hat{y} = 7.2$

$(6.5 - 7.2) = -.7$

Reduce the overall loss

Quadratic Term

$x^2 + x + 2$

1) High Penalization

$(7)^2 = 49 = \sqrt{49}$

Mean Square Error

Error / Loss / Cost

1) Error / Loss → single data point
2) Cost → for the entire dataset / batch

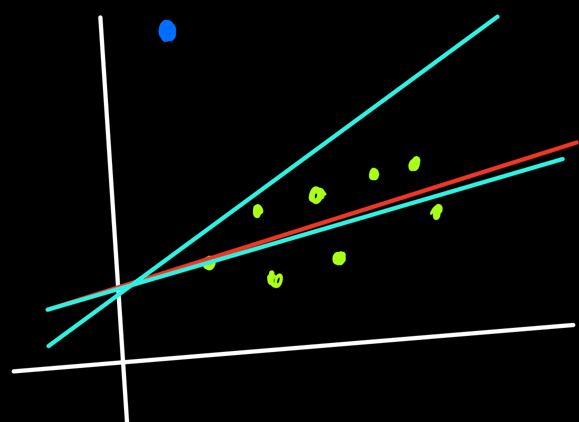
Mean Square Error

$$LF = (y - \hat{y})^2$$

$$CF = \frac{1}{n} \sum_{i=1}^{n} (y - \hat{y})^2$$

Cons

1) Cannot handle outliers



Type of Loss functions

Regression

1. MSE
2. MAE

Classification

B

1. Log loss
BCE

M

1. C.C.E
2. S.C.C.E

BCE

③ Huber Loss

③ Hinge loss

Autoencoder

① KL Divergence

GAN

① Discriminator Loss

Object Detection

Focal loss

Embeddings

Triplet loss

Object Detection

1) Co-ordinate of the object → Regression

2) Class of the object → Classification

Cost Function = Regression + Classification

Mean Square Error

10th	12th	GRE	LPA (Y)	($\hat{Y}$)
63	73	320	7.2	6.5
51	61	260	3.1	2.4

$$(\text{True} - \text{predicted})^2$$

$$= (Y - \hat{Y})^2$$

Example :- $(7.2 - 6.5)^2 = (.7)^2 = \underline{.49}$ (loss)

$$C.F = \frac{1}{2} (.49) + (3.1 - 2.4)^2$$

$$= \frac{1}{2} \quad .49 + .49$$

$$= \frac{.98}{2}$$

$$C.F = \underline{\underline{.49}}$$

Pros

1) Easy to interpret
2) Differentiable (SGD)

Cons

1) Not Robust to outliers

3) 1 local minima

Mean Absolute Error (L1 Loss)

$$L \cdot F = |y - \hat{y}|$$

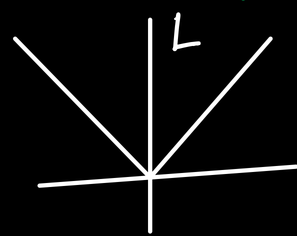
$$C \cdot F = \frac{1}{n} \sum_{i=0}^{n} |y - \hat{y}|$$

Pros

1) Easy same unit
2) Robust to outliers

Cons

1) Not differentiable
   ↳ Subgradients



Huber Loss

$$L = \begin{cases} \frac{1}{2}(y - \hat{y})^2 & \text{for } |y - \hat{y}| \le \delta \\ \delta |y - \hat{y}| - \frac{1}{2}\delta^2 & \text{otherwise} \end{cases}$$

hyperparameter

Combination of MSE and MAE

Pros

1) More Robust to Outliers

Classification

1) Binary Classification

Log loss | Binary Cross Entropy

$$\text{Loss Function} = -y \log(\hat{y}) - (1-y)\log(1-\hat{y})$$

$$\text{Cost Function} = -\frac{1}{n}\left[\sum_{i=1}^{n} y \log(\hat{y}) + (1-y)\log(1-\hat{y})\right]$$

CGPA	10	Placement (Y)	($\hat{Y}$)
		1	.7
8	68		.3
7	61	0	

Case 1

Case 2

$= -y \log(\hat{y})$

$= -1 \log(.7)$

$= -1 .$

$= -(1-y) \log(1-\hat{y})$

$= -1 \cdot \log(.7)$

$= -1 .$

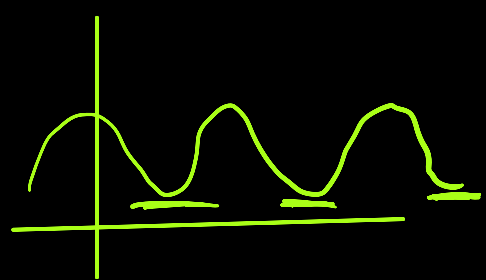
Advantages

1) Differentiable

Cons

1) M.C.C Problem

2) Multiple local minima



Multi-Class Classification

1) Categorical Cross Entropy

$$\left\{ L = -\sum_{j=1}^{K} y_j \log(\hat{y}_j) \right\}$$

K = no of classes in data

OHE

IQ	CGPA	Placed
80	8.2	Yes
60	6.1	No
70	7.1	Maybe

Yes	No	Maybe
1	0	0
0	1	0
0	0	1

$$L = -Y \log(\hat{Y}_1) - Y \log(\hat{Y}_2) - Y \log(\hat{Y}_3)$$

For Case 1 (Yes)

$$= -Y \log(\hat{Y}_1)$$

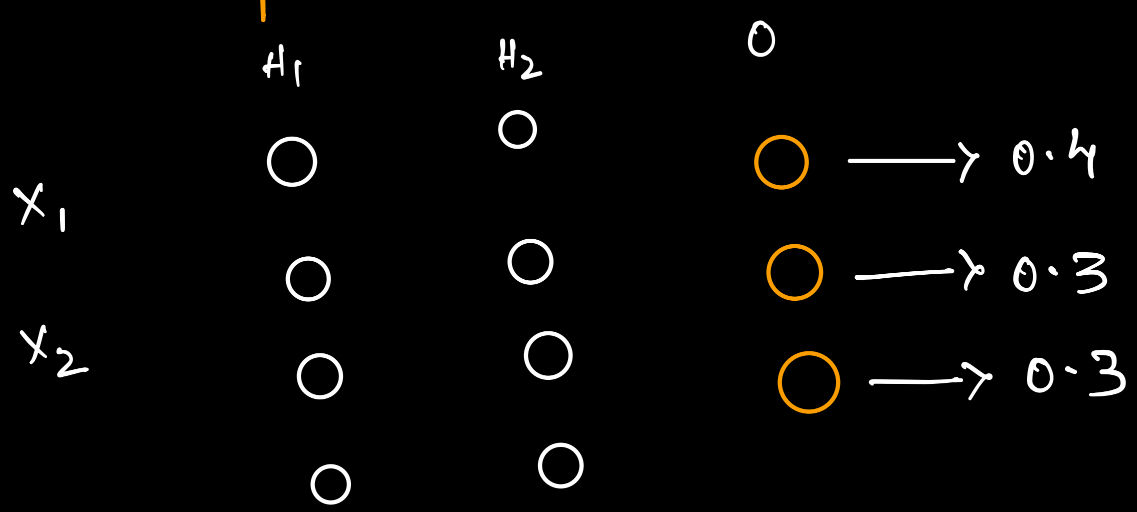
For Case 3 (Maybe)

$$= -Y \log(\tilde{Y}_3)$$

For Case 2 (No)

$$= -Y \log(\tilde{Y}_2)$$

2) Sparse Categorical Cross Entropy



Activation :- Softmax

Class

$$\text{Softmax} = \frac{e^{z_1}}{e^{z_1} + e^{z_2} + e^{z_3}}$$

Class 1

$$\frac{e^{z_2}}{e^{z_1} + e^{z_2} + e^{z_3}}$$

Case 1

$$[0.5 \quad 0.3 \quad 0.2]$$

$$1 \quad 0 \quad 0$$

$$= -y \log(\hat{y})$$

$$= -1 \log(0.5)$$

Case 2

$$[0.3 \quad 0.6 \quad 0.1]$$

$$[0 \quad 1 \quad 0]$$

$$= -1 \log(0.6)$$

12	CGPA	Placed
80	8.2	Yes
60	6.1	No
70	7.1	Maybe

Integer Encoding

1

2

3

$[0.1 \quad 0.5 \quad 0.4]$

$= -y \log(\hat{y})$

Very Fast

$$L.F = -\sum_{j=1}^{K} y_j \log(\hat{y}_j)$$

$$C.F = -\frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{K} y_{ij} \log(\hat{y}_{ij})$$

Activation Functions

Relu

$$f(x) = \max(0, x)$$

$$\text{if} \quad \begin{cases} x, & x > 0 \\ 0, & x \leq 0 \end{cases}$$

https://keras.io/api/layers/activations/#creating-custom-activations

https://www.v7labs.com/blog/neural-networks-activation-functions