16 July Day 6

26 November 2022 07:00

Binary class classification
Output layer = sigmoid

Multiclass classification
Output layer = softmax.

Dead neuron ⇒ Associated with One neuron.

vanishing gradient ⇒ layer specific problem

Loss function.

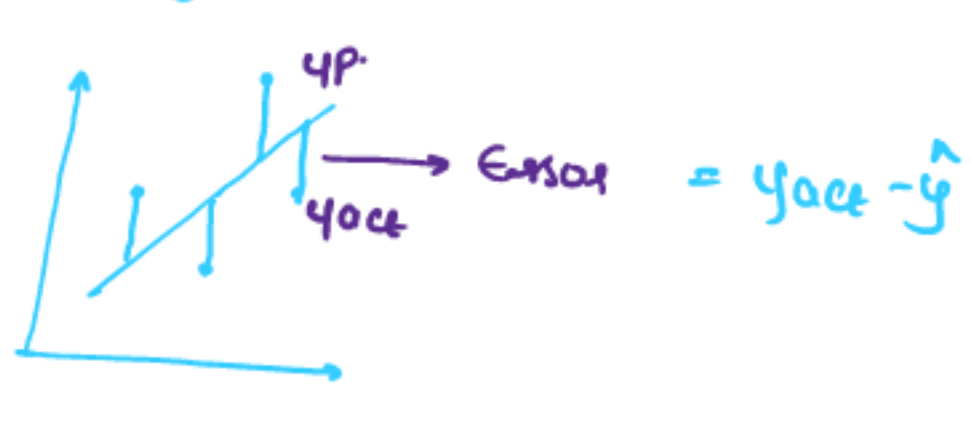
Data → NN → $\hat{y}$ → Loss ← [W,B]   ($y_{act} - \hat{y}$ ✓)

[W,B] → Data - -

$w_{new} = w_{old} - \alpha \frac{\partial L}{\partial w_{old}}$ (update)

you can't improve what you can't measure.

Loss function is the method of Evaluating how well our algorithm performing on given Dataset.

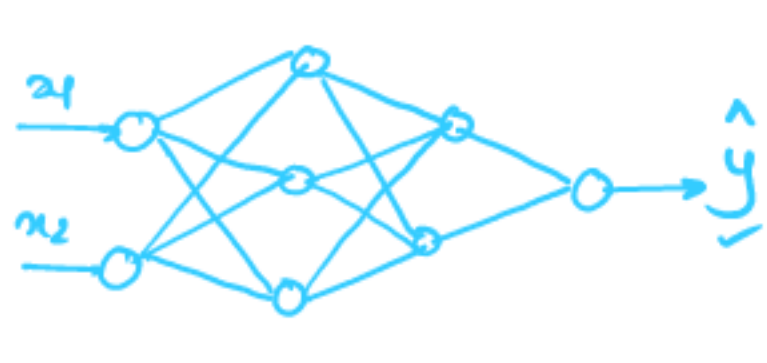


Error $= y_{act} - \hat{y}$

Loss [↓] perf [↑]
Loss [↑] perf [↓]

value of weights and bias are updated with the help of optimizer.

→ Regression
→ classification



Mean Squared Error

Loss fn. ⇒ Single Record
Cost fn ⇒ for whole batch.

$(y_{act} - \hat{y})$ ⇒ Loss fn.

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

Line	Quad
2 →	2
4 →	16
8 →	64
16 →	256

penalize. e^2

Advantages.

① If is in the form of Quadratic eq., we will get the gradient Descent with only one global minima

no local minima.
Easy to interpret
differentiable

Disadvantage

① Error unit is not same. it is in sq. form.

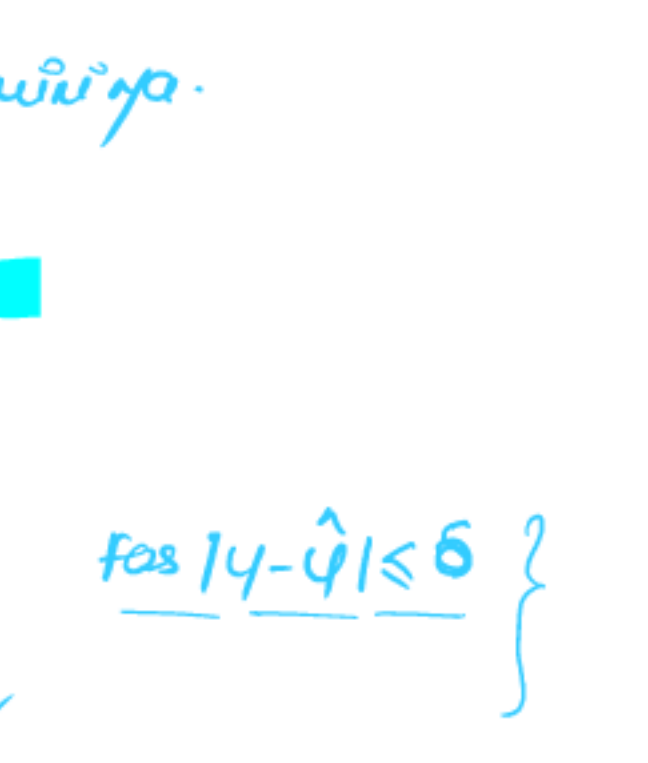
② not Robust to outlier.

true nature of our Data is not Captured and it got Deviated.

Mean Absolute Error.

$Loss = |y_{act} - \hat{y}|$ → Linear

$Cf = \frac{1}{n}\sum |y_{act} - \hat{y}|$ → Linear



Advantages.

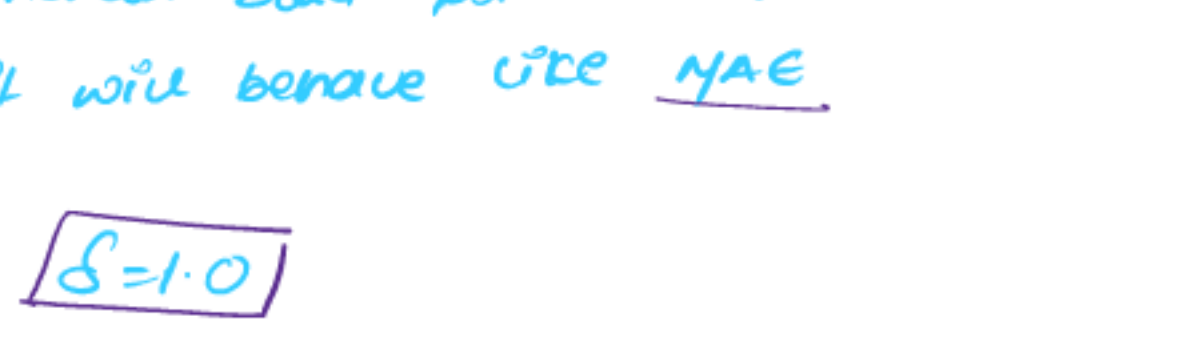
① Error unit is same.
② Easy to interpret
③ outliers are better handled than MSE, it is not penalizing the model by sq. the Error.

Disadvantage

① graph differentiable ☒
② there may have multiple local minima.

Huber Loss

$$Loss = \begin{cases} \frac{1}{2}(y_{act}-\hat{y})^{2} ✓ \\ \delta\,|y_{act}-\hat{y}| - \frac{1}{2}\delta^{2} ✓ \end{cases}$$ for $|y-\hat{y}| \le \delta$



[δ] ←

It is Less sensitive to outliers. Huber Loss is used in regression problem compared with MSE and MAE.

whenever Data point in not a outlier it will behave like MSE.

whenever Data point is an outlier it will behave like MAE

$\delta = 1.0$

Advantages.

① outliers are handled properly.
② Local minima situations are also handled here.

Disadvantage

① complex.

Classification

① Binary cross Entropy.

94 1N 14 9N
✓ uncertain ✓

10Y 0N 0Y 10N
certain

5Y 5N 5Y 5N

Entropy [↑] ⇒ [1] ✓

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

$y = y_{act}$
$\hat{y} = y_{pred}$

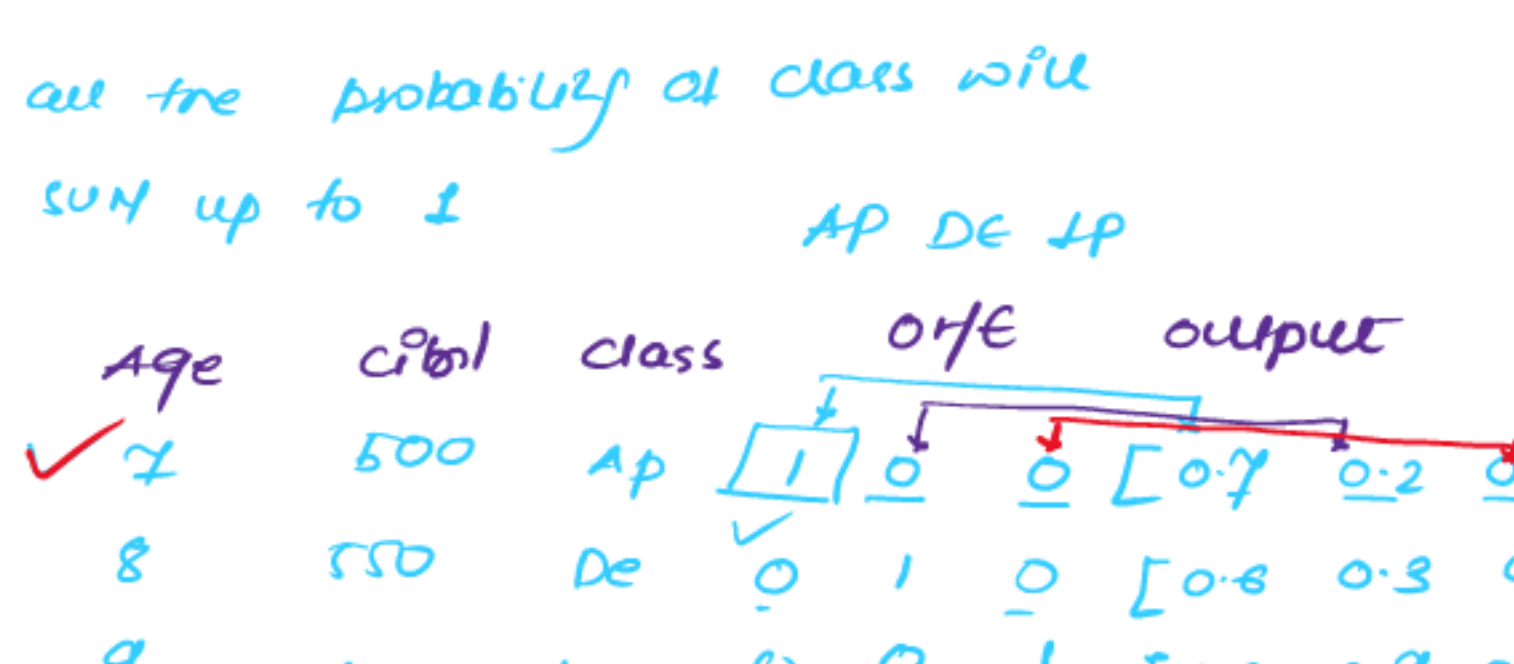
$$CF = -\frac{1}{n}\left[\sum_{i=1}^{n} y(\log(\hat{y}) + (1-y)\log(1-\hat{y})\right]$$

[class 0] ⇒ $-\log(1-\hat{y})$ ✓
[class 1] ⇒ $-\log(\hat{y})$

Categorical Cross Entropy.

used for multiclass classification problem.

[1000]



[=1]

(LP) ←

3 Approved Decline pipeline

$$Loss = -\sum_{i=1}^{K} y_{act}\log(\hat{y})$$

$-y_1\log(\hat{y}_1) - y_2\log(\hat{y}_2) - y_3\log(\hat{y}_3)$ ✓

K = no. of classes.

all the probability of class will sum up to 1

Age	cibil	class	OHE (AP)	OHE (DE)	OHE (LP)	output
✓ 7	500	AP	[1]	0	0	[0.7 0.2 0.5] = ①
8	550	DE	0	1	0	[0.6 0.3 0.1]
9	700	LP	0	0	1	[0.2 0.7 0.1]

y_{act} value $\hat{y}$ value

Loss $-y_1\log(\hat{y}_1) - y_2\log(\hat{y}_2) - y_3\log(\hat{y}_3)$

→ 0.7

✓ $-1\times\log(0.3) - 0\times\log(0.2) - 0\times\log(0.5)$

$-1\times\log(0.3) - 0 - 0$

⇒ $[-1\times\log(0.3)]$ ✓ = (+ve) Loss | - Result

⇒ [Result] ✓

① produce one hot Array containing to probable match for each category.

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

Sparse Categorical Cross Entropy.

It is frustrating when you are dealing with large classification problems. ie 1000 classes.

And when we do one hot Encoding of 1000 classes → 1000 vectors necessary significant amount of memory.

Age	cibil	class	CCE	SPCE
7	500	AP	100	1 [x x x]
8	550	DE	010	2 [x x x]
9	700	LP	001	3 [x x x]

Computation fast

Drop out Ratio.

Dropout ratio is the technique that drops the neuron from the neural net as ignore them during training.

Some neurons are involved in training and Rest are turned off.

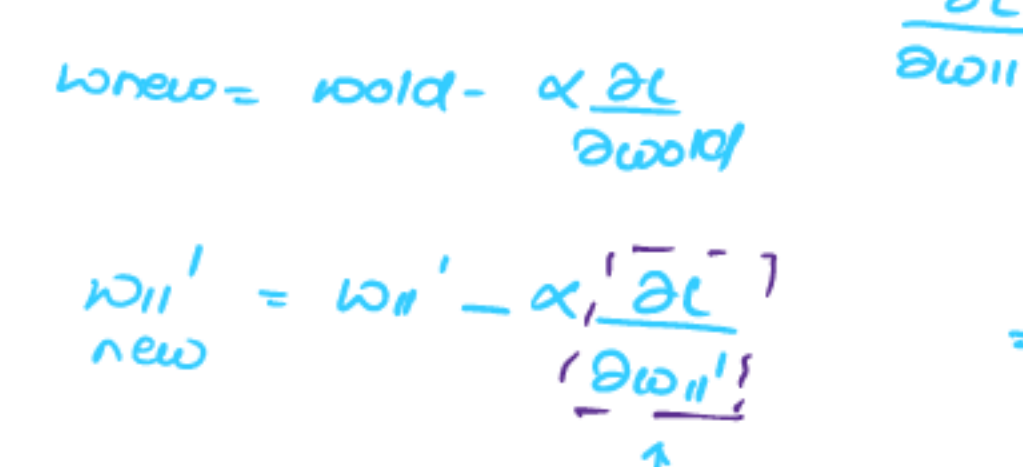
"to prevent overfitting"

these units are not consider During particular forward or backward propagation

$P = 0.5$

① Dropout is an approach in neural network which helps in Reducing independent Learning among Neuron.

Exploding gradient



$w_{new} = w_{old} - \alpha\frac{\partial L}{\partial w_{old}}$

$$\frac{\partial L}{\partial w_{11}'} = \frac{\partial L}{\partial O_{31}} \times \frac{\partial O_{31}}{\partial O_{21}} \times \frac{\partial O_{21}}{\partial O_{11}} \times \frac{\partial O_{11}}{\partial w_{11}'}$$

$w_{11\,new}' = w_{11}' - \alpha\left[\frac{\partial L}{\partial w_{11}'}\right]$

$= 0.2 \times 10 \times 0.6 \times 1$

$=$ [Big]

$= w_{11}' - \alpha\left[\frac{\partial L}{\partial w_{11}}\right]$

Solution.

① proper weight initialization. ✓
② non saturating Act fn. ✓

Model is experiencing high Learning rate. Exponential growth in model parameters.