- similar for others weights too.

Forword propogation

- In backpropagation, there is two thing

i) Forword propogation

Here our data move through each layer in forword direction one-by one.

ii) Backword propogation

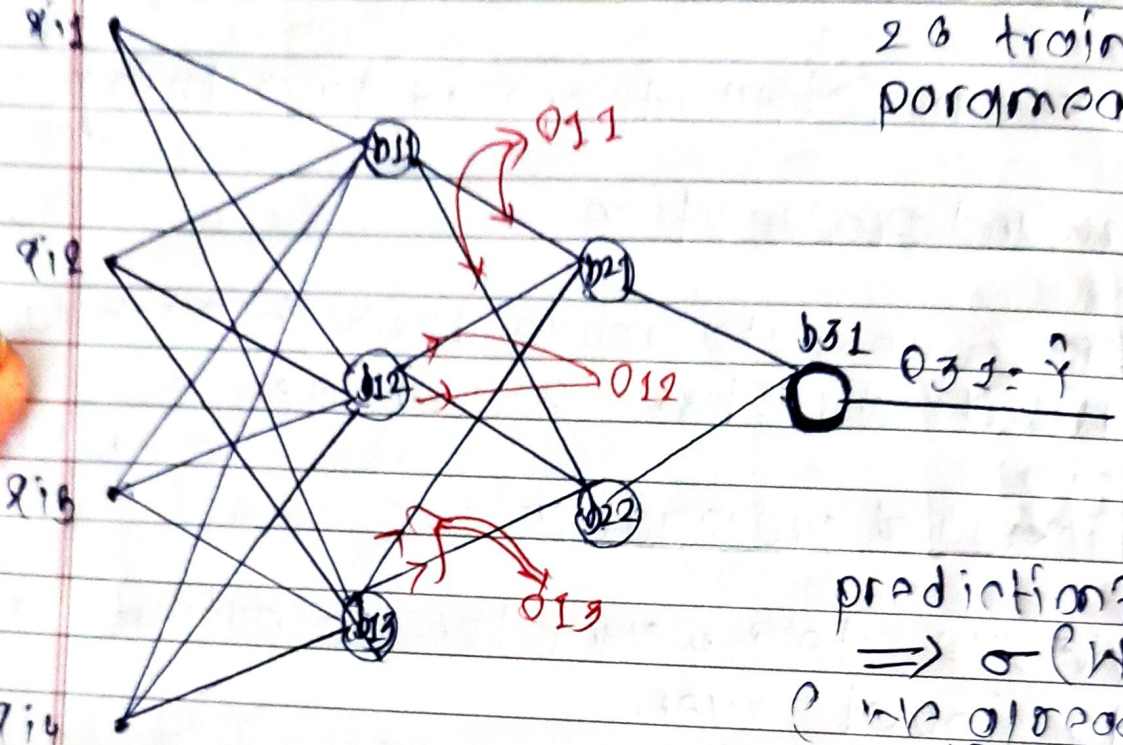
We updates weight through output layer to input layer.

- We will learn forword propogation for now, we will see if we have input and weights we will able predicts.
- We will see that how things handled by linear algebra, (At the end we have to make a dot product of matrices).
- this is our setup,

Cgpa	iq	10th Mark	12th mark	placed
7.2	72	69	81	1
8.1	92	75	76	0

→ # of trainable parameters

$12w + 6w + 2w + 2b =$
20 trainable parameter.



prediction for ~~a~~ apparently
$\Rightarrow \sigma(W^T X + b)$
(we already seen that).

* Obviously we start from random value of weight and biases.

Layer 1

$$\begin{bmatrix} w^1_{11} & w^1_{12} & w^1_{13} \\ w^1_{21} & w^1_{22} & w^1_{23} \\ w^1_{31} & w^1_{32} & w^1_{33} \\ w^1_{41} & w^1_{42} & w^1_{43} \end{bmatrix}^T \begin{bmatrix} x_{i1} \\ x_{i2} \\ x_{i3} \\ x_{i4} \end{bmatrix} + \begin{bmatrix} b_{11} \\ b_{12} \\ b_{13} \\ b_{14} \end{bmatrix}$$

$(4\times3)^T \Longrightarrow (3\times4)$ $\quad 4\times1 \quad 3\times1$

This first row comes from x_{i1} i.e. 0th column
This first column weight goes to this bias

b11, this is now the weights are designed.

- We have to multiply this matrix with inputs i.e. $\begin{bmatrix} x_{i1} \\ x_{i2} \\ x_{i3} \\ x_{i4} \end{bmatrix}$

- After multiplication,

$$= \begin{bmatrix} w^1_{11} x_{i1} + w^1_{21} x_{i2} + w^1_{31} x_{i3} + w^1_{41} x_{i4} \\ w^1_{12} x_{i1} + w^1_{22} x_{i2} + w^1_{32} x_{i3} + w^1_{42} x_{i4} \\ w^1_{13} x_{i1} + w^1_{23} x_{i2} + w^1_{33} x_{i3} + w^1_{43} x_{i4} \end{bmatrix} + \begin{bmatrix} b_{11} \\ b_{12} \\ b_{13} \end{bmatrix}$$

$$= \begin{bmatrix} - + - + - + - + b_{11} \\ - + - + - + - + b_{12} \\ - + - + - + - + b_{13} \end{bmatrix}$$

- After taking sigmoide we have this matrix of output.

$$\begin{bmatrix} O_{11} \\ O_{12} \\ O_{13} \end{bmatrix} \Longrightarrow \text{Assume this as, } a^{[1]}$$

Layer 2

$$= \begin{bmatrix} W^2_{11} & W^2_{12} \\ W^2_{21} & W^2_{22} \\ W^2_{31} & W^2_{32} \end{bmatrix}^T \begin{bmatrix} O_{11} \\ O_{12} \\ O_{13} \end{bmatrix} + \begin{bmatrix} b_{21} \\ b_{22} \end{bmatrix}$$

$(3\times2)^T \Rightarrow 2\times3 \qquad 3\times1 \qquad 2\times1$

After calculating we get this, Also after applying sigmoide.

$$= \begin{bmatrix} O_{21} \\ O_{22} \end{bmatrix} \Longrightarrow \text{Assume this as, } a^{[2]}$$

Layer 3

$$\begin{bmatrix} W^3_{11} \\ W^3_{21} \end{bmatrix} \begin{bmatrix} O_{21} \\ O_{22} \end{bmatrix} + \begin{bmatrix} b_{31} \end{bmatrix}$$

$(2\times1)^T \Rightarrow (1\times2) \qquad 2\times1 \qquad 1\times1$

After calculating and applying sigmoide we get

thing.

= 031 = $\hat{y}_i$ ⟹ assume this as a[3]

- And input layer output as a[0].
- This steps we do let me explaine one by one,

$$a^{[1]} = \sigma(a^{[0]} W^{[1]} + b^{[1]}) \Rightarrow \text{layer1 output}$$

$$a^{[2]} = \sigma(a^{[1]} W^{[2]} + b^{[2]}) \Rightarrow \text{layer2 output.}$$

$$\hat{y}_i = a^{[3]} = \sigma(a^{[2]} W^{[3]} + b^{[3]}) \Rightarrow \text{layer3 output.}$$

- We can write above each layer output in one expression,

$$\sigma(\sigma(\underbrace{(\sigma(a^{[0]} W^{[1]} + b^{[1]}))}_{a^{[1]}} W^{[2]} + b^{[2]}) W^{[3]} + b^{[3]})$$

↓

Whole thing give's a[3]

classmate
Date
Page

Customer churn prediction

- Scale all values first since neural net work converge fast with scaled val

```
import tensorflow
from tensorflow import keras
from tensorflow.keras import Sequential
from tensorflow.keras import Dense
```

Layers are added one after another in sequence (No branching, No loop, just straight layers)

- Dense parameter

1) 'units' => number of neurons.
2) 'activation' => activation function
3) 'input_dim' => only used for first layer
4) 'use_bias' = 'true' => Each neurons get bias without bias no shifting of line, each line passes to origine.
5) 'kernel_initializer' => "Different value present (How the intial weights are created).

6) "bias_intializer" = 'zeros' ⇒ Default (How bias ~~are~~ values intialed.
7) We can gives name to layers 'name'

Important note

- Binary classifier unit ⇒ 1, activation function = sigmoid/side
- Multiclass classifier unit = 3, softmax
- Linear unit = 1 linear

- So,

```
model = Sequential()
model.add(Dense(11, activation='sigmoid', input_dim=11))
model.add(Dense(11, activation='sigmoid'))
model.add(Dense(1, activation='sigmoid'))
```

- model.summary() # Gives total no. of trainable parameter.

Model compilation stage.

- Here we tell which optimizer we using or which loss function we were using.

model.compile (loss='binary_cross_entropy (log-loss)', optimizer='Adam'

Compile parameters

- loss = mathamatical function model tries to minimize.

binary_crossentropy ⟹ Binary classification

categorical_crossentropy ⟹ one_hot
sparse_categorical_crossentropy ⟹ integer labels

Multiclass classification

Regression mse/mae

- Optimizer (controls how models updates weight).

Adam ⟹ Default.
'sgd' ⟹ simple gradient discent.
'rmsprop' ⟹ Good for RNN.
'adagrad' ⟹ Sparse data.

- Matrics

What performance we want to see during training.

'accuracy', 'mae', 'mse'.

classmate
Date
Page

layer attributes

```
model.layers[0].get_weights()

# we get weight and bias value for
# layer '0';
```

Prediction

```
y_pred = model.predict(X_test)
```

Handwritten digit classification by using ANN (MNIST) (Info is about multiclass classification by using ANN)

- Resolution of this images are low 28×28 = 784 pixels.