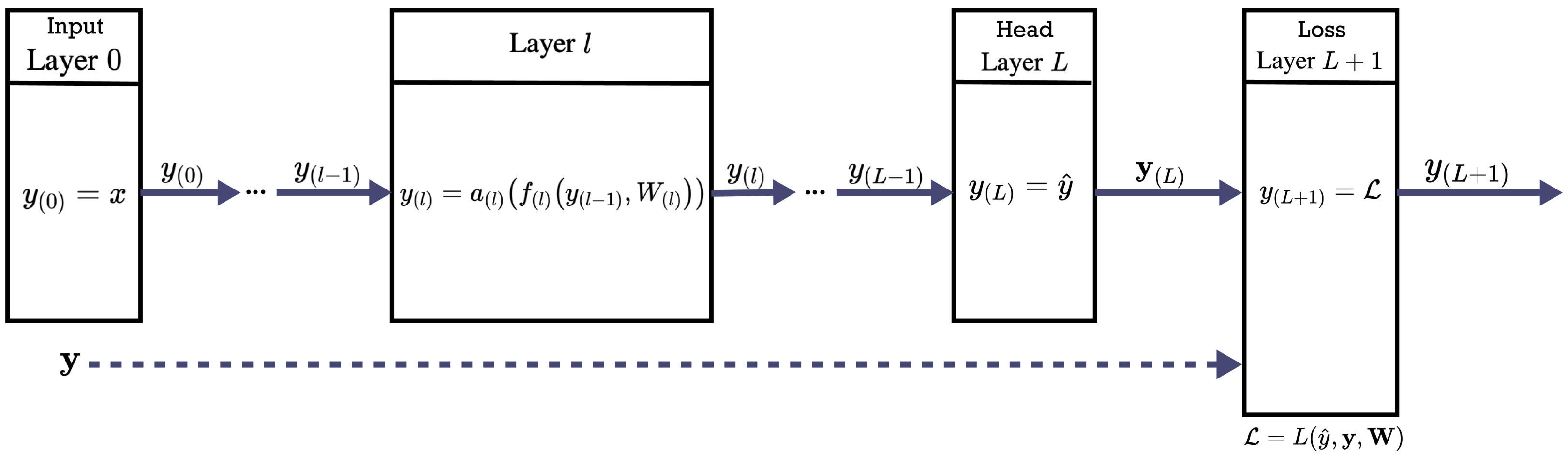
Inside a layer: Units/Neurons

Notation 1

Layer l, for $1 \leq l \leq L$:

- Produces output vector $\mathbf{y}_{(l)}$
- $\mathbf{y}_{(l)}$ is a vector of $n_{(l)}$ synthetic features
$$n_{(l)} = ||\mathbf{y}_{(l)}||$$
- Takes as input $\mathbf{y}_{(l-1)}$, the output of the preceding layer

- Layer L will typically implement Regression or Classification
- The first $(L-1)$ layers create synthetic features of increasing complexity
- We will use layer $(L+1)$ to compute a Loss


Input
Layer 0
$y_{(0)} = x$
$y_{(0)}$
$y_{(l-1)}$
Layer l
$y_{(l)} = a_{(l)}(f_{(l)}(y_{(l-1)}, W_{(l)}))$
$y_{(l)}$
$y_{(L-1)}$
Head
Layer L
$y_{(L)} = \hat{y}$
$\mathbf{y}_{(L)}$
Loss
Layer $L+1$
$y_{(L+1)} = \mathcal{L}$
$y_{(L+1)}$
$\mathbf{y}$
$\mathcal{L} = L(\hat{y}, \mathbf{y}, \mathbf{W})$

The input $\mathbf{x}$

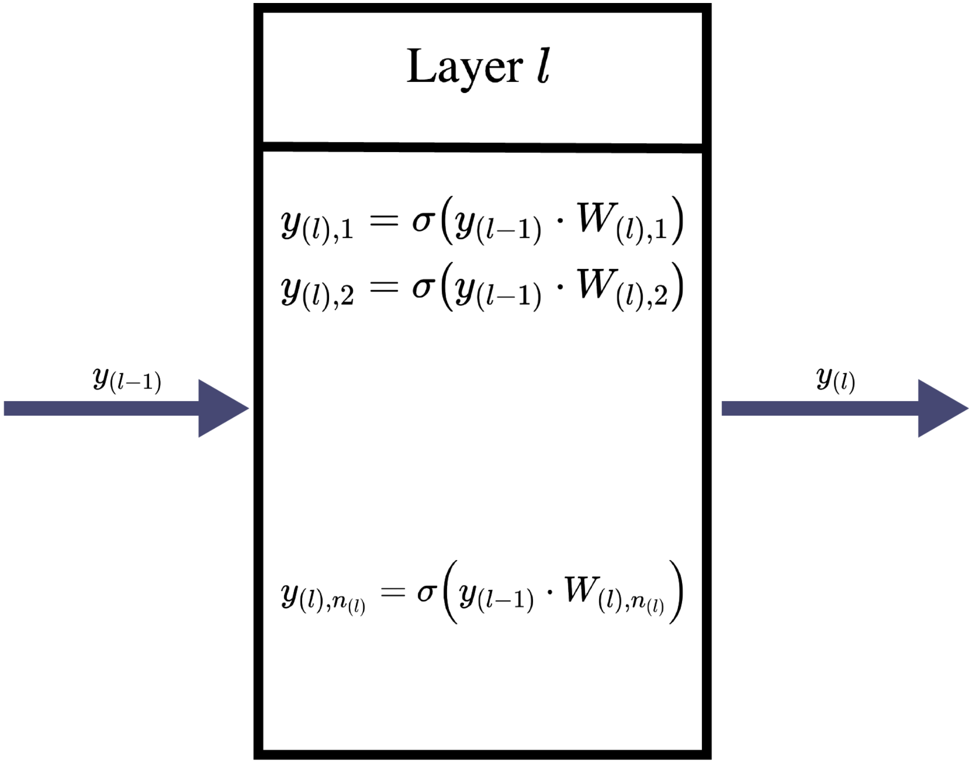
- Is called "layer 0"
- $\mathbf{y}_{(0)} = \mathbf{x}$

The output $\mathbf{y}_{(L-1)}$ of the penultimate layer $(L-1)$

- Becomes the input of a Classifier/Regression model at layer L

Let's look inside layer l (of a particular type called *Fully Connected* or *Dense*)

Layer


Layer l
$y_{(l),1} = \sigma\left(y_{(l-1)} \cdot W_{(l),1}\right)$
$y_{(l),2} = \sigma\left(y_{(l-1)} \cdot W_{(l),2}\right)$
$y_{(l),n_{(l)}} = \sigma\left(y_{(l-1)} \cdot W_{(l),n_{(l)}}\right)$
$y_{(l-1)}$
$y_{(l)}$

- Input vector of $n_{(l-1)}$ features: $\mathbf{y}_{(l-1)}$
- Produces output vector or $n_{(l)}$ features $\mathbf{y}_{(l)}$
- Feature j defined by the function

$$\mathbf{y}_{(l),j} = \sigma(\mathbf{y}_{(l-1)} \cdot \mathbf{W}_{(l),j})$$

Each feature $\mathbf{y}_{(l),j}$ is produced by a *unit* (*neuron*)

- There are $n_{(l)}$ units in layer l
- The units are *homogenous*
    - same input $\mathbf{y}_{(l-1)}$ to every unit
    - same functional form for every unit
    - units differ only in $\mathbf{W}_{(l),j}$

Units are also sometimes refered to as *Hidden Units*

- They are internal to a layer.
- From the standpoint of the Input/Output behavior of a layer, the units are "hidden"

The functional form

$$\mathbf{y}_{(l),j} = \sigma(\mathbf{y}_{(l-1)} \cdot \mathbf{W}_{(l),j})$$

is called a *Dense* or *Fully Connected* unit.

It is called Fully connected since

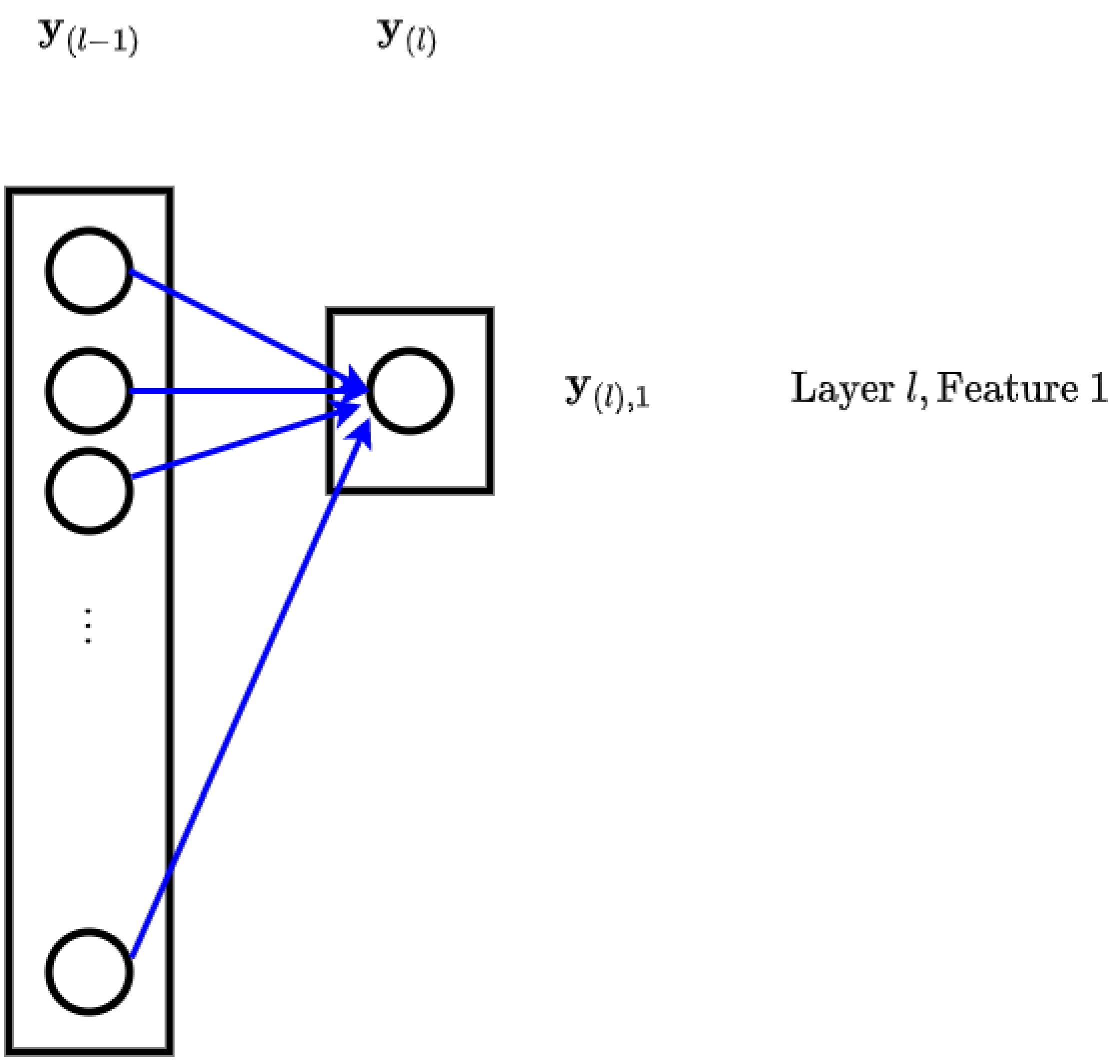
- each unit takes as input $\mathbf{y}_{(l-1)}$, **all** $n_{(l-1)}$ outputs of the preceding layer

The *Fully Connected* part can be better appreciated by looking at a diagram of the connectivity of a *single* unit producing a *single* feature.

A Fully Connected/Dense Layer producing a *single* feature at layer l computes

$$\mathbf{y}_{(l),1} = a_{(l)}\big(\mathbf{y}_{(l-1)} \cdot \mathbf{W}_{(l),1}\big)$$

Fully connected unit, single feature

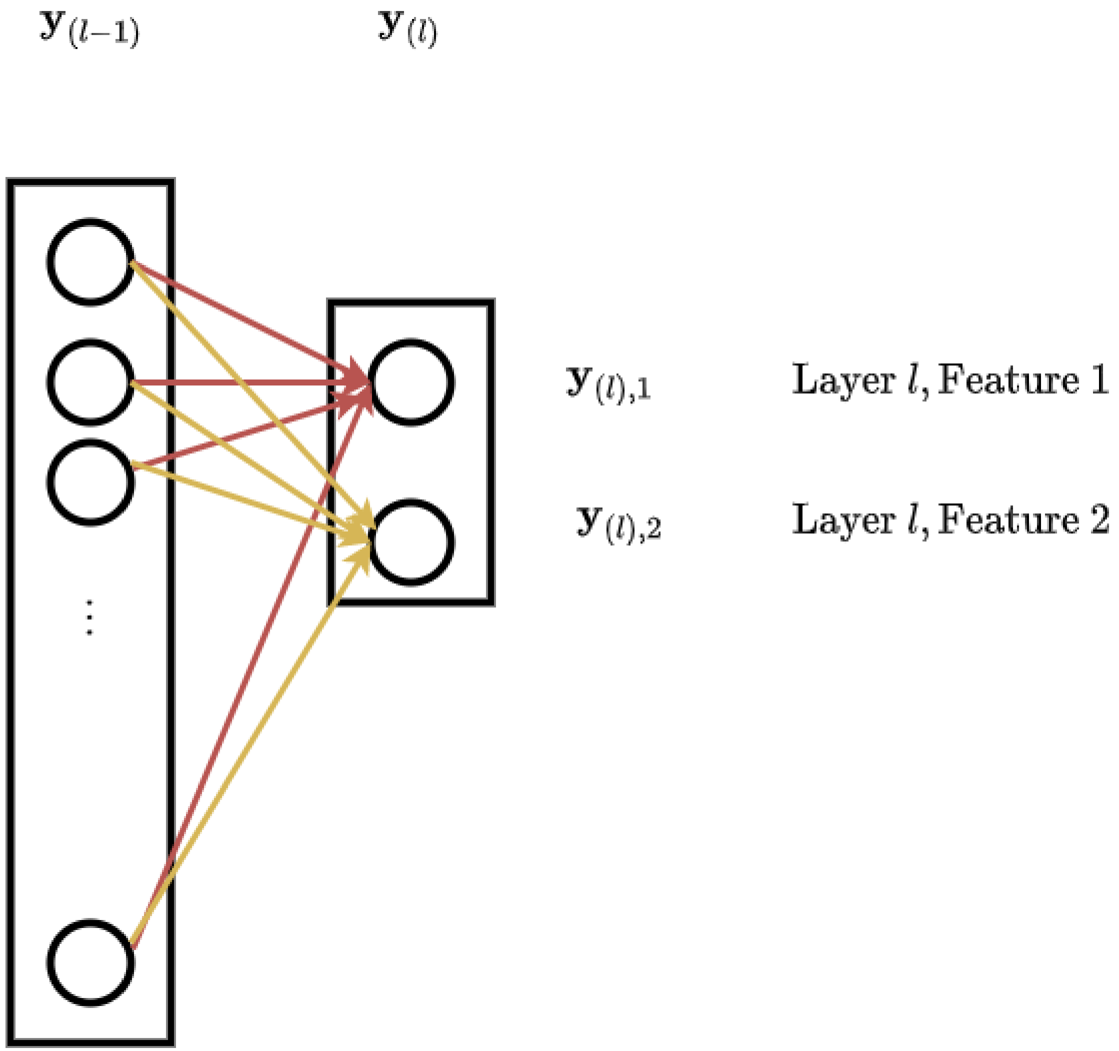


The edges into the single unit of layer l correspond to $\mathbf{W}_{(l),1}$.

A Fully Connected/Dense Layer with multiple units producing *multiple* feature at layer l computes

$$\mathbf{y}_{(l),j} = a_{(l)}(\mathbf{y}_{(l-1)} \cdot \mathbf{W}_{(l),j})$$

Fully connected, two features



The edges into each unit of layer l correspond to

- $\mathbf{W}_{(l),1}, \mathbf{W}_{(l),2} \cdots$
- Separate colors for each units/row of $\mathbf{W}$

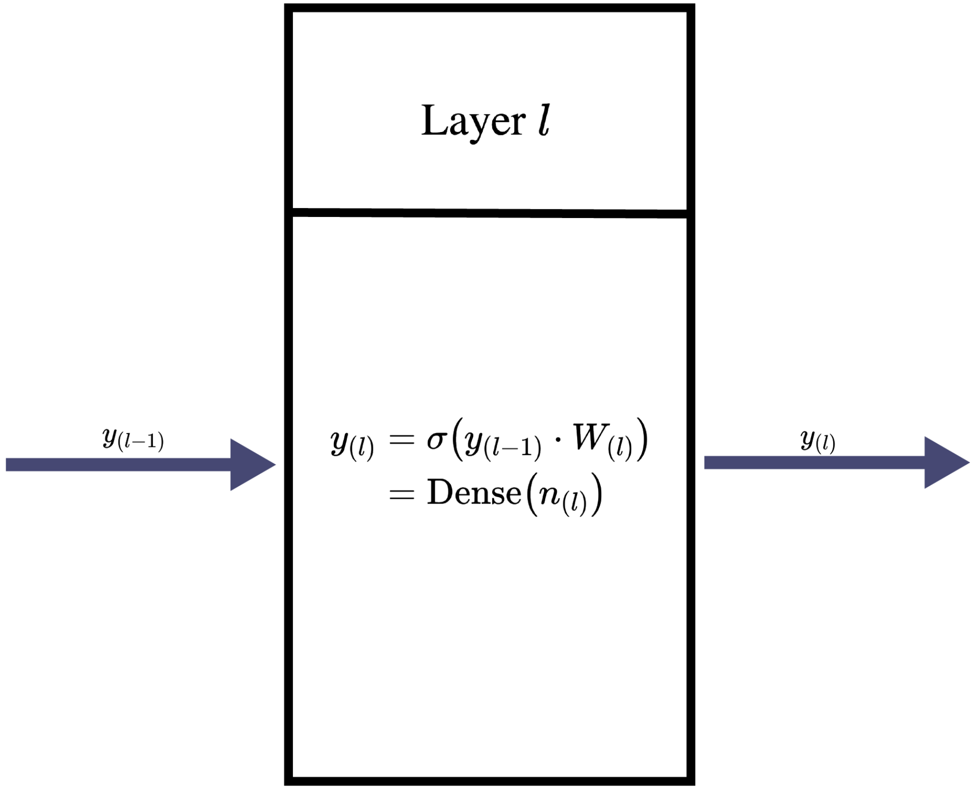
Each unit $\mathbf{y}_{(l),j}$ in layer l creates a new feature using pattern$\mathbf{W}_{(l),j}$

The functional form is of

- A dot product $\mathbf{y}_{(l-1)} \cdot \mathbf{W}_{(l),j}$
    - Which can be thought of matching input $\mathbf{y}_{(l-1)}$ against pattern $\mathbf{W}_{(l),j}$
- Fed into σ, the *sigmoid* function we have previously encountered in Logistic Regression.

Because the units are homogeneous, we can depict it as

Layer



Layer l
$y_{(l)} = \sigma\left(y_{(l-1)} \cdot W_{(l)}\right)$
$= \mathrm{Dense}\left(n_{(l)}\right)$
$y_{(l-1)}$
$y_{(l)}$

where

- $\mathbf{y}_{(l)}$ is a vector of length $n_{(l)}$
- $\mathbf{W}_{(l)}$ is a matrix
    - $n_{(l)}$ rows
    - $\mathbf{W}^{(j)}_{(l)} = \mathbf{W}_{(l),j}$

Written with the shorthand `Dense(` n_l `)`

We will introduce other types of layers.

- Most will be homogeneous
- Not all will be fully Connected
- The dot product will play a similar role

The sigmoid function σ may be the *most significant part* of the functional form

- The dot product is a *linear* operation
- The outputs of sigmoid are *non-linear* in its inputs

So the sigmoid induces a non-linear transformation of the features $\mathbf{y}_{(l-1)}$

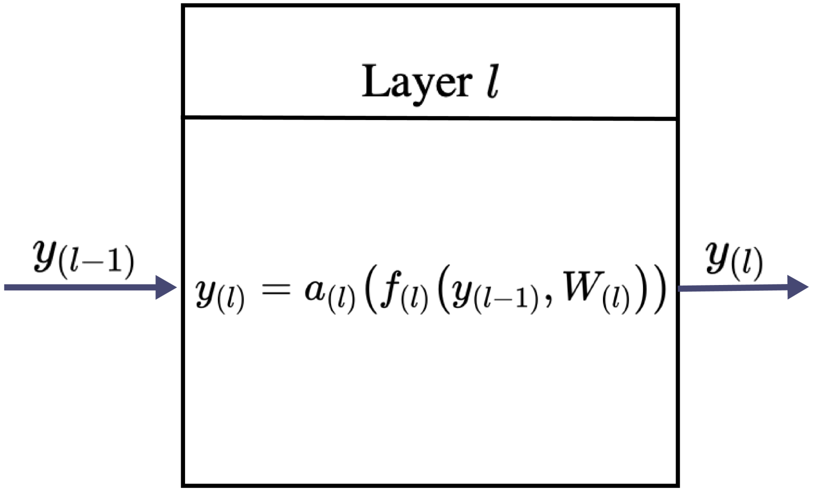
The outer function which applies a non-linear transformation to linear inputs

- Is called an *activation function*
- Sigmoid is one of several activation functions we will study

- The operation of a layer does not always need to be a dot production
- The activation function of a layer need not always be the sigmoid

More generically we write a layer as

Layers



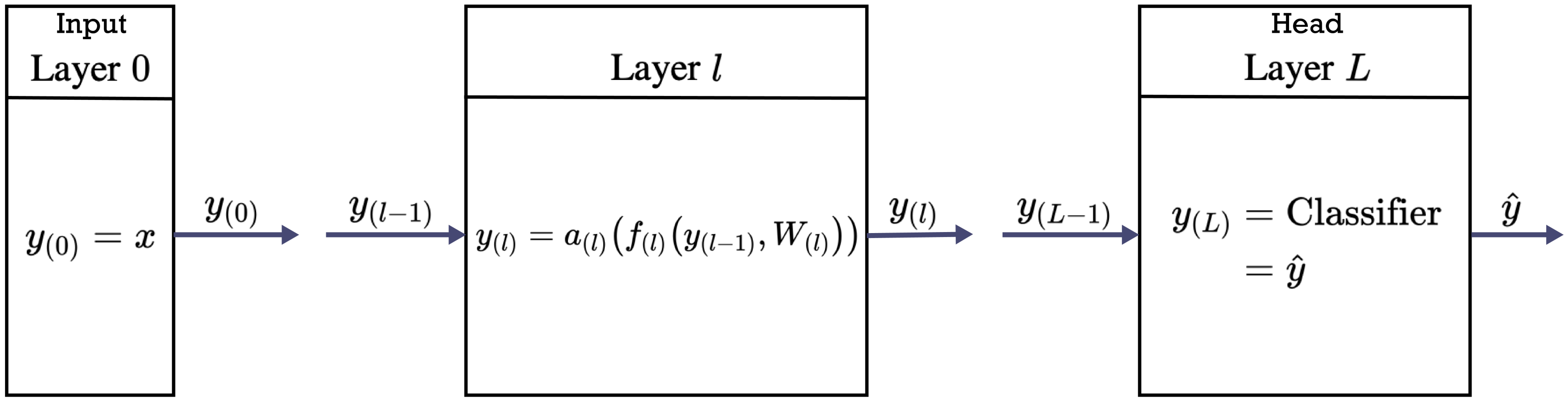
$$\mathbf{y}_{(l)} = \boldsymbol{a}_{(l)} \left(\boldsymbol{f}_{(l)} (\mathbf{y}_{(l-1)}, \mathbf{W}_{(l)}) \right)$$

where

- $\boldsymbol{f}_{(l)}$ is a function of $\mathbf{y}_{(l)-1}$ and $\mathbf{W}_{(l)}$
- $\boldsymbol{a}_{(l)}$ is an activation function

So our multi-layer Neural Network (using Dense layers) looks like

Layers



Input
Layer 0
$y_{(0)} = x$
$y_{(0)}$
$y_{(l-1)}$
Layer l
$y_{(l)} = a_{(l)}\big(f_{(l)}\big(y_{(l-1)}, W_{(l)}\big)\big)$
$y_{(l)}$
$y_{(L-1)}$
Head
Layer L
$y_{(L)} = \text{Classifier}$
$= \hat{y}$
$\hat{y}$

In slightly more mathematical terms: Layer l is computing a function $\mathbf{y}_{(l)} = \boldsymbol{F}_{(l)}$

$$\boldsymbol{F}_{(l)}(\mathbf{y}_{(l-1)}; \mathbf{W}_{(l)}) = \mathbf{y}_{(l)}$$

$$\boldsymbol{F}_{(l)} : \boldsymbol{\mathcal{R}}^{||\mathbf{y}_{(l-1)}||} \mapsto \boldsymbol{\mathcal{R}}^{||\mathbf{y}_{(l)}||}$$

If we expand $\boldsymbol{F}_{(l)}$, we see that it is the $\boldsymbol{l}$-fold composition of functions $\boldsymbol{F}_{(1)}, \ldots, \boldsymbol{F}_{(l)}$

$$\begin{aligned}
\mathbf{y}_{(l)} &= \boldsymbol{F}_{(l)}(\mathbf{y}_{(l-1)}; \mathbf{W}_{(l)}) \\
&= \boldsymbol{F}_{(l)}(\ \boldsymbol{F}_{(l-1)}(\mathbf{y}_{(l-2)};\ \mathbf{W}_{(l-1)});\ \mathbf{W}_{(l)}\ ) \\
&= \boldsymbol{F}_{(l)}(\ \boldsymbol{F}_{(l-1)}(\ \boldsymbol{F}_{(l-2)}(\mathbf{y}_{(l-3)};\ \mathbf{W}_{(l-2)});\ \mathbf{W}_{(l-1)}\ ); \mathbf{W}_{(l)}\ ) \\
&= \vdots
\end{aligned}$$

So the layer-wise architecture is nothing more than a way of computing a nested (composed) function.

In [4]:

```python
print("Done")
```

```
Done
```