Natural Language Processing

Chapter 7

Artificial Neural Network

Department of Computer Science and Engineering,
Shahjalal University of Science and Technology, Sylhet

Machine Learning

Two tasks:

- Classification
- Prediction

Machine Learning

Two tasks:

- Classification
- Prediction

Machine Learning

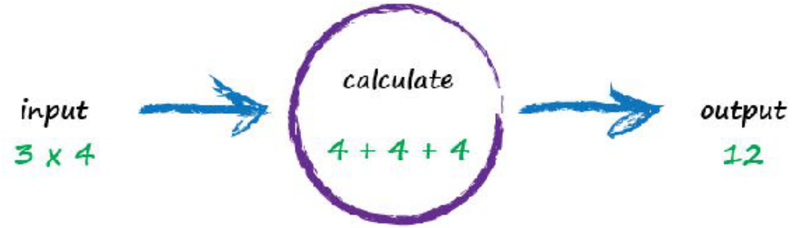
Two tasks:

- Classification
- Prediction

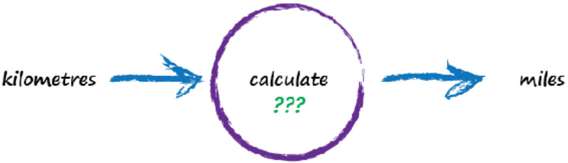
A Simple Predicting Machine

A Simple Predicting Machine

A Simple Predicting Machine

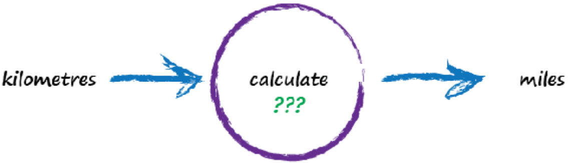


A Simple Predicting Machine



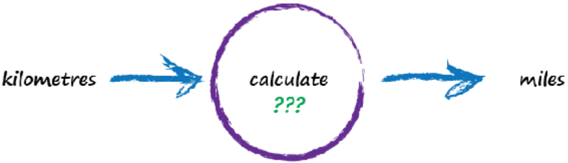
- Imagine we don't know the formula for converting between kilometres and miles.
- One clue is: the relationship between the two is linear, i.e., miles = kilometres x c
- Another clue is: real world observations.

A Simple Predicting Machine



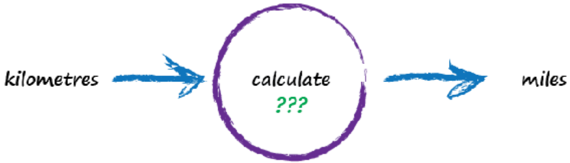
- Imagine we don't know the formula for converting between kilometres and miles.
- One clue is: the relationship between the two is linear, i.e., miles = kilometres x c
- Another clue is: real world observations.

A Simple Predicting Machine



- Imagine we don't know the formula for converting between kilometres and miles.
- One clue is: the relationship between the two is linear, i.e., miles = kilometres x c
- Another clue is: real world observations.

A Simple Predicting Machine



- Imagine we don't know the formula for converting between kilometres and miles.
- One clue is: the relationship between the two is linear, i.e., miles = kilometres x c
- Another clue is: real world observations.

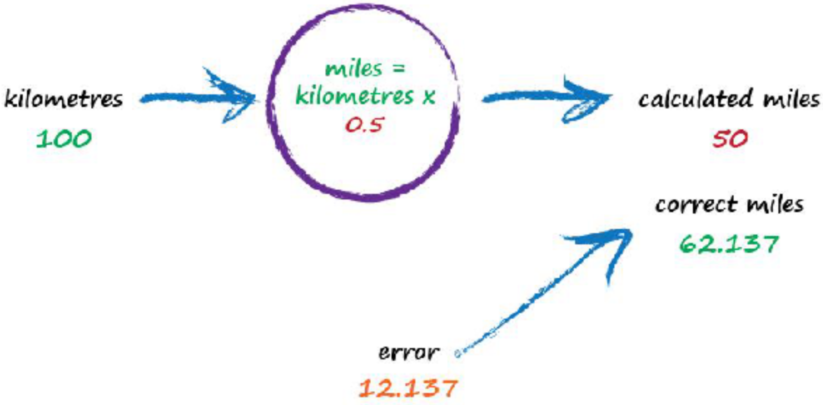
A Simple Predicting Machine

Truth Example	Kilometres	Miles
1	0	0
2	100	62.137

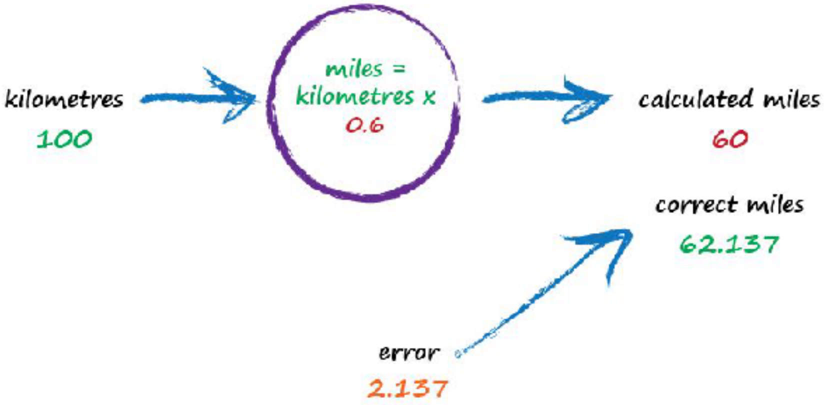
- Imagine we don't know the formula for converting between kilometres and miles.
- One clue is: the relationship between the two is linear, i.e., miles = kilometres x c
- Another clue is: real world observations.

A Simple Predicting Machine

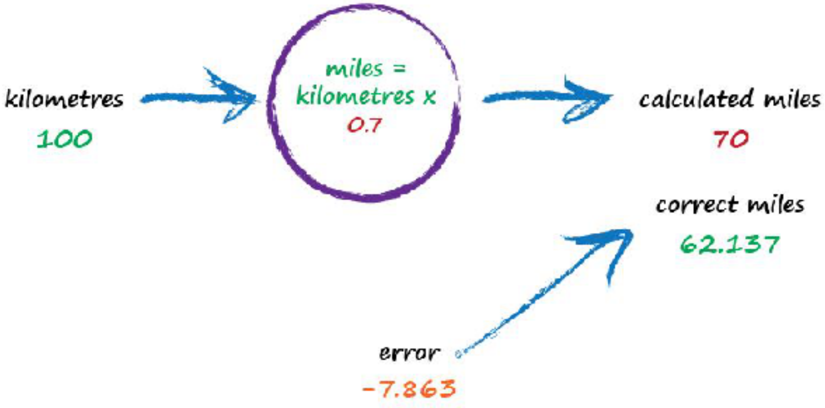
A Simple Predicting Machine



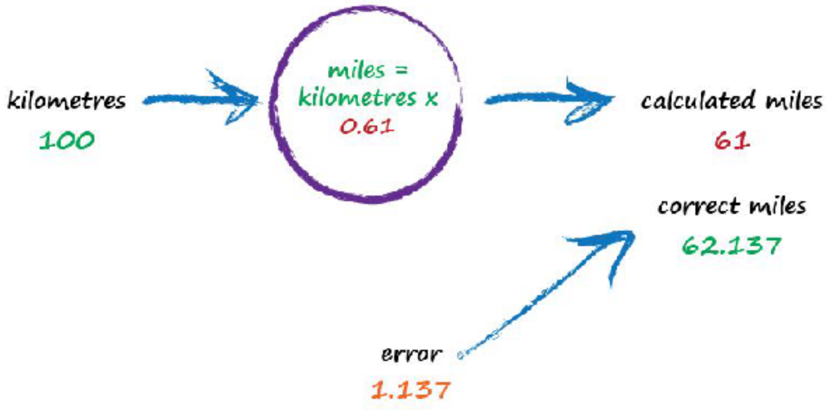
A Simple Predicting Machine



A Simple Predicting Machine



A Simple Predicting Machine



A Simple Predicting Machine

- We have trained the machine to get better and better at giving the right answer.
- This is the very core process of learning in a neural network.
- We refined that prediction by adjusting an internal parameter, informed by the error we saw when comparing with a known-true example.

A Simple Predicting Machine

- We have trained the machine to get better and better at giving the right answer.
- This is the very core process of learning in a neural network.
- We refined that prediction by adjusting an internal parameter, informed by the error we saw when comparing with a known-true example.

A Simple Predicting Machine

- We have trained the machine to get better and better at giving the right answer.
- This is the very core process of learning in a neural network.
- We refined that prediction by adjusting an internal parameter, informed by the error we saw when comparing with a known-true example.

A simple Classifier

- Classifying is Not Very Different from Predicting
- That predictor had an adjustable linear function.
- Linear functions give straight lines when you plot their output against input.
- The adjustable parameter c changed the slope of that straight line.
- A linear function inside our simple predictors can be used to classify previously unseen data.

A simple Classifier

- Classifying is Not Very Different from Predicting
- That predictor had an adjustable linear function.
- Linear functions give straight lines when you plot their output against input.
- The adjustable parameter c changed the slope of that straight line.
- A linear function inside our simple predictors can be used to classify previously unseen data.

A simple Classifier

- Classifying is Not Very Different from Predicting
- That predictor had an adjustable linear function.
- Linear functions give straight lines when you plot their output against input.
- The adjustable parameter c changed the slope of that straight line.
- A linear function inside our simple predictors can be used to classify previously unseen data.

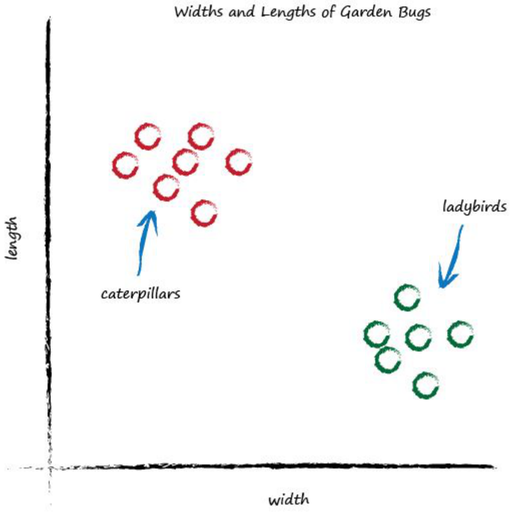
A simple Classifier

- Classifying is Not Very Different from Predicting
- That predictor had an adjustable linear function.
- Linear functions give straight lines when you plot their output against input.
- The adjustable parameter c changed the slope of that straight line.
- A linear function inside our simple predictors can be used to classify previously unseen data.

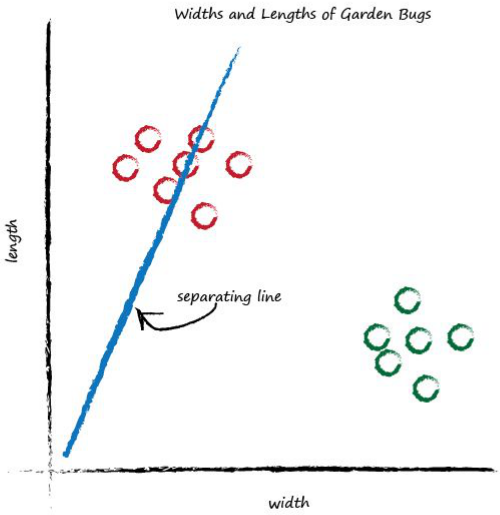
A simple Classifier

- Classifying is Not Very Different from Predicting
- That predictor had an adjustable linear function.
- Linear functions give straight lines when you plot their output against input.
- The adjustable parameter c changed the slope of that straight line.
- A linear function inside our simple predictors can be used to classify previously unseen data.

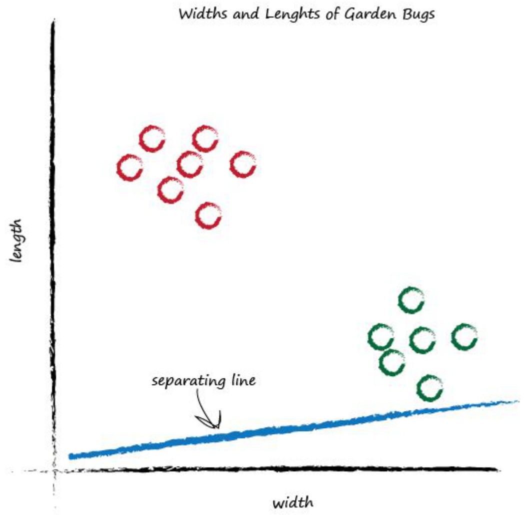
A simple Classifier



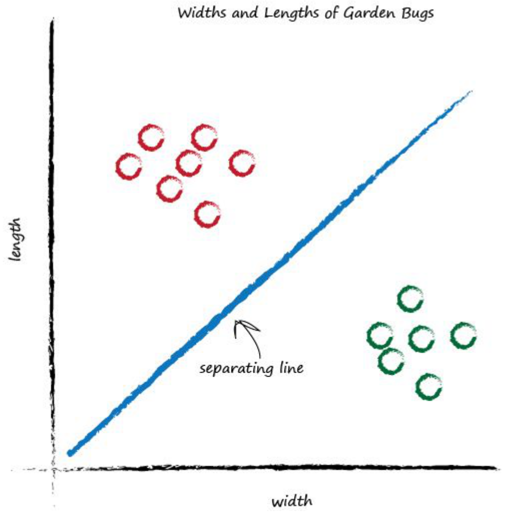
A simple Classifier



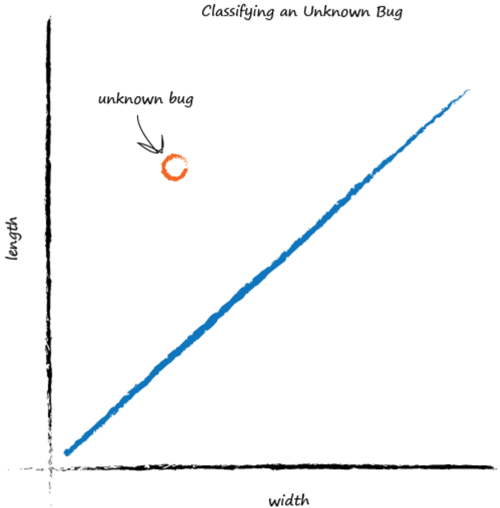
A simple Classifier



A simple Classifier



A simple Classifier



Training A Simple Classifier

- How do we get the right slope?
- How do we improve a line we know isn't a good divider between the two kinds of bugs?
- The answer to that is again at the very heart of how neural networks learn.

Training A Simple Classifier

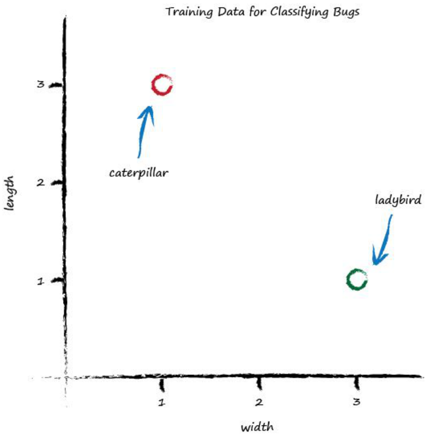
- How do we get the right slope?
- How do we improve a line we know isn't a good divider between the two kinds of bugs?
- The answer to that is again at the very heart of how neural networks learn.

Training A Simple Classifier

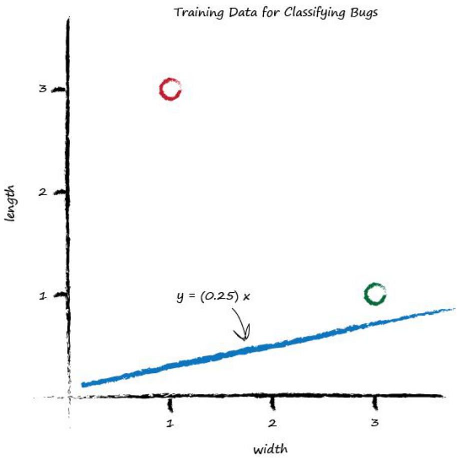
- How do we get the right slope?
- How do we improve a line we know isn't a good divider between the two kinds of bugs?
- The answer to that is again at the very heart of how neural networks learn.

Training A Simple Classifier

Example	Width	Length	Bug
1	3.0	1.0	ladybird
2	1.0	3.0	caterpillar



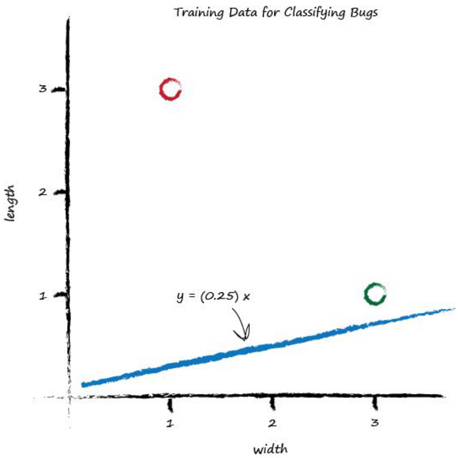
Training A Simple Classifier



$$y = Ax$$

for $A = 0.25, \; y = 0.25x$

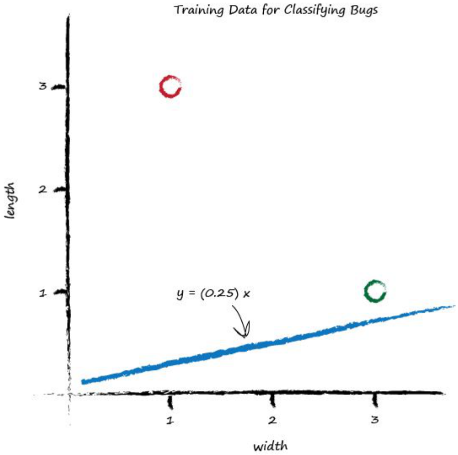
Training A Simple Classifier



$$y = Ax$$

for $A = 0.25,\ y = 0.25x$

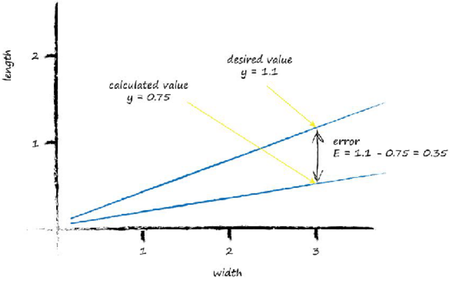
Training A Simple Classifier



$$y = Ax$$

for $A = 0.25, \; y = 0.25x$

Training A Simple Classifier



$y = Ax$

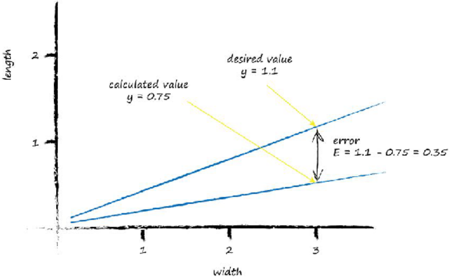
for $A = 0.25,\ y = 0.25x$

for $x = 3.0,\ y = (0.25) * (3.0) = 0.75$

error = (desired target-actual output)

$E = 1.1 - 0.75 = 0.35$

Training A Simple Classifier



$y = Ax$

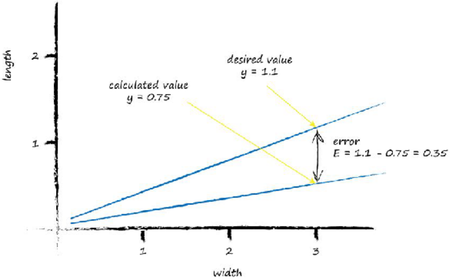
for $A = 0.25, \; y = 0.25x$

for $x = 3.0, \; y = (0.25) * (3.0) = 0.75$

error $=$ (desired target-actual output)

$E = 1.1 - 0.75 = 0.35$

Training A Simple Classifier



$y = Ax$

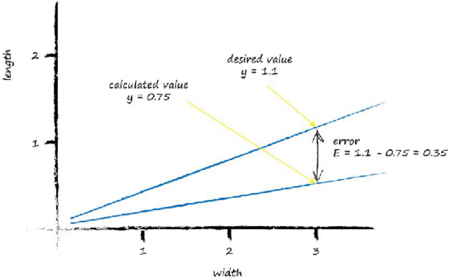
for $A = 0.25,\ y = 0.25x$

for $x = 3.0,\ y = (0.25) * (3.0) = 0.75$

error $=$ (desired target-actual output)

$E = 1.1 - 0.75 = 0.35$

Training A Simple Classifier



$y = Ax$

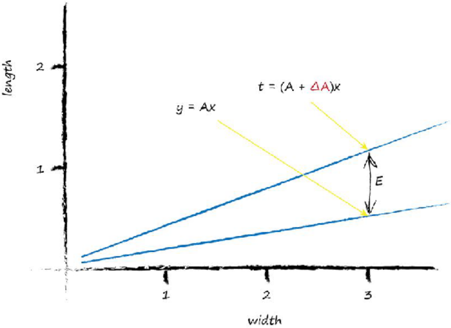
for $A = 0.25,\ y = 0.25x$

for $x = 3.0,\ y = (0.25) * (3.0) = 0.75$

error $=$ (desired target-actual output)

$E = 1.1 - 0.75 = 0.35$

Training A Simple Classifier



$$y = Ax$$

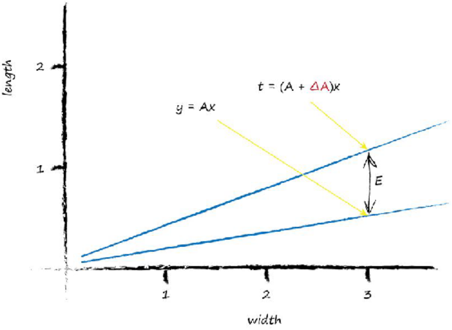
$$t = (A + \Delta A)x$$

$$E = t - y = Ax + (\Delta A)x - Ax$$

$$E = (\Delta A)x$$

$$\Delta A = E/x$$

Training A Simple Classifier



$$y = Ax$$

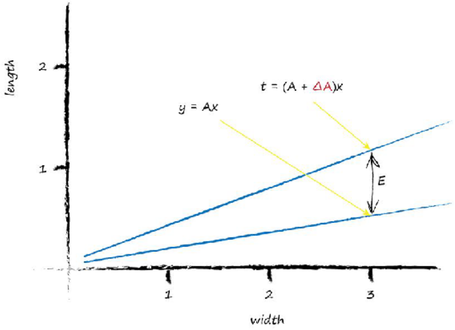
$$t = (A + \Delta A)x$$

$$E = t - y = Ax + (\Delta A)x - Ax$$

$$E = (\Delta A)x$$

$$\Delta A = E/x$$

Training A Simple Classifier



$$y = Ax$$

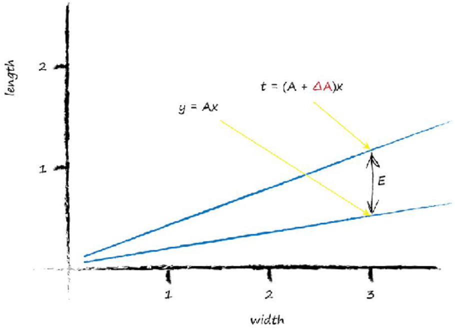
$$t = (A + \Delta A)x$$

$$E = t - y = Ax + (\Delta A)x - Ax$$

$$E = (\Delta A)x$$

$$\Delta A = E/x$$

Training A Simple Classifier



$$y = Ax$$

$$t = (A + \Delta A)x$$

$$E = t - y = Ax + (\Delta A)x - Ax$$

$$E = (\Delta A)x$$

$$\Delta A = E/x$$

Training A Simple Classifier

$$\Delta A = E/x = 0.35/3.0 = 0.1167$$

$$A = (A + \Delta A) = 0.25 + 0.1167 = 0.3667$$

$$y = 0.3667 * 3.0 = 1.1$$

Training A Simple Classifier

$$\Delta A = E/x = 0.35/3.0 = 0.1167$$

$$A = (A + \Delta A) = 0.25 + 0.1167 = 0.3667$$

$$y = 0.3667 * 3.0 = 1.1$$

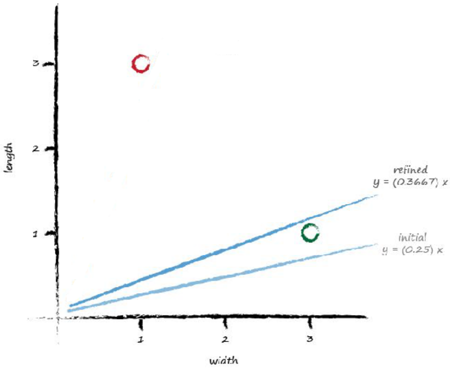
Training A Simple Classifier

$\Delta A = E/x = 0.35/3.0 = 0.1167$

$A = (A + \Delta A) = 0.25 + 0.1167 = 0.3667$

$y = 0.3667 * 3.0 = 1.1$

Training A Simple Classifier



$$\Delta A = E/x = 0.35/3.0 = 0.1167$$

$$A = (A + \Delta A) = 0.25 + 0.1167 = 0.3667$$

$$y = 0.3667 * 3.0 = 1.1$$

Training A Simple Classifier

Another Training Data: x=1.0 and y=3.0

Now, Slope, $A = 0.3667$

For $x = 1.0$,
$y = 0.3667 * 1.0 = 0.3667$

$E = t - y = 2.9 - 0.3667 = 2.5333$

$\Delta A = E/x = 2.5333/1.0 = 2.5333$

$A = (A + \Delta A) = 0.3667 + 2.5333 = 2.9$

$y = 2.9 * 1.0 = 2.9$

Training A Simple Classifier

Another Training Data: x=1.0 and y=3.0

Now, Slope, $A = 0.3667$

For $x = 1.0$,
$y = 0.3667 * 1.0 = 0.3667$

$E = t - y = 2.9 - 0.3667 = 2.5333$

$\Delta A = E/x = 2.5333/1.0 = 2.5333$

$A = (A + \Delta A) = 0.3667 + 2.5333 = 2.9$

$y = 2.9 * 1.0 = 2.9$

Training A Simple Classifier

Another Training Data: x=1.0 and y=3.0

Now, Slope, $A = 0.3667$

For $x = 1.0$,
$y = 0.3667 * 1.0 = 0.3667$

$E = t - y = 2.9 - 0.3667 = 2.5333$

$\Delta A = E/x = 2.5333/1.0 = 2.5333$

$A = (A + \Delta A) = 0.3667 + 2.5333 = 2.9$

$y = 2.9 * 1.0 = 2.9$

Training A Simple Classifier

Another Training Data: x=1.0 and y=3.0

Now, Slope, $A = 0.3667$

For $x = 1.0$,
$y = 0.3667 * 1.0 = 0.3667$

$E = t - y = 2.9 - 0.3667 = 2.5333$

$\Delta A = E/x = 2.5333/1.0 = 2.5333$

$A = (A + \Delta A) = 0.3667 + 2.5333 = 2.9$

$y = 2.9 * 1.0 = 2.9$

Training A Simple Classifier

Another Training Data: x=1.0 and y=3.0

Now, Slope, $A = 0.3667$

For $x = 1.0$,
$y = 0.3667 * 1.0 = 0.3667$

$E = t - y = 2.9 - 0.3667 = 2.5333$

$\Delta A = E/x = 2.5333/1.0 = 2.5333$

$A = (A + \Delta A) = 0.3667 + 2.5333 = 2.9$

$y = 2.9 * 1.0 = 2.9$

Training A Simple Classifier

Another Training Data: x=1.0 and y=3.0

Now, Slope, $A = 0.3667$

For $x = 1.0$,
$y = 0.3667 * 1.0 = 0.3667$

$E = t - y = 2.9 - 0.3667 = 2.5333$

$\Delta A = E/x = 2.5333/1.0 = 2.5333$

$A = (A + \Delta A) = 0.3667 + 2.5333 = 2.9$

$y = 2.9 * 1.0 = 2.9$

Training A Simple Classifier

Another Training Data: x=1.0 and y=3.0

Now, Slope, $A = 0.3667$

For $x = 1.0$,
$y = 0.3667 * 1.0 = 0.3667$

$E = t - y = 2.9 - 0.3667 = 2.5333$

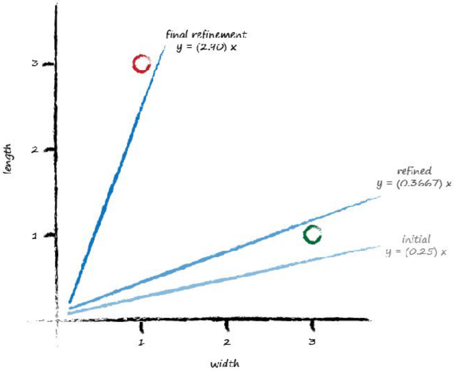
$\Delta A = E/x = 2.5333/1.0 = 2.5333$

$A = (A + \Delta A) = 0.3667 + 2.5333 = 2.9$

$y = 2.9 * 1.0 = 2.9$

Training A Simple Classifier

Another Training Data: x=1.0 and y=3.0



Now, Slope, $A = 0.3667$

For $x = 1.0$,
$y = 0.3667 * 1.0 = 0.3667$

$E = t - y = 2.9 - 0.3667 = 2.5333$

$\Delta A = E/x = 2.5333/1.0 = 2.5333$

$A = (A + \Delta A) = 0.3667 + 2.5333 = 2.9$

$y = 2.9 * 1.0 = 2.9$

Learning Rate

- How do we fix this?
- This is an important idea in machine learning.
- We moderate the updates.
- Add a moderation into the update formula:

$$\Delta A = L(E/x)$$

- This moderating factor is called a learning rate.

Learning Rate

- How do we fix this?
- This is an important idea in machine learning.
- We moderate the updates.
- Add a moderation into the update formula:

  $$\Delta A = L(E/x)$$

- This moderating factor is called a learning rate.

Learning Rate

- How do we fix this?
- This is an important idea in machine learning.
- We moderate the updates.
- Add a moderation into the update formula:

$$\Delta A = L(E/x)$$

- This moderating factor is called a learning rate.

Learning Rate

- How do we fix this?
- This is an important idea in machine learning.
- We moderate the updates.
- Add a moderation into the update formula:

$$\Delta A = L(E/x)$$

- This moderating factor is called a learning rate.

Learning Rate

- How do we fix this?
- This is an important idea in machine learning.
- We moderate the updates.
- Add a moderation into the update formula:

$$\Delta A = L(E/x)$$

- This moderating factor is called a learning rate.

Training A Simple Classifier

Let us pick Learning rate, L = 0.5

1st Training Data: x=3.0 and y=1.0

Initial $A = 0.25$

For $x = 3.0$, $y = 0.25 * 3.0 = 0.75$

$E = t - y = 1.1 - 0.75 = 0.35$

$\Delta A = L(E/x) = 0.5 * 0.35/3.0 = 0.0583$

$A = (A + \Delta A) = 0.25 + 0.0583 = 0.3083$

2nd Training Data: x=1.0 and y=3.0

For $x = 1.0$, $y = 0.3083 * 1.0 = 0.3083$

$E = t - y = 2.9 - 0.3083 = 2.5917$

$\Delta A = L(E/x) = 0.5 * 2.5917/1.0 = 1.2958$

$A = (A + \Delta A) = 0.3083 + 1.2958 = 1.6042$

$y = Ax = 1.6042 * x$

Training A Simple Classifier

Let us pick Learning rate, L = 0.5
1st Training Data: x=3.0 and y=1.0
Initial $A = 0.25$

For $x = 3.0$, $y = 0.25 * 3.0 = 0.75$

$E = t - y = 1.1 - 0.75 = 0.35$

$\Delta A = L(E/x) = 0.5 * 0.35/3.0 = 0.0583$

$A = (A + \Delta A) = 0.25 + 0.0583 = 0.3083$

2nd Training Data: x=1.0 and y=3.0
For $x = 1.0$, $y = 0.3083 * 1.0 = 0.3083$

$E = t - y = 2.9 - 0.3083 = 2.5917$

$\Delta A = L(E/x) = 0.5 * 2.5917/1.0 = 1.2958$

$A = (A + \Delta A) = 0.3083 + 1.2958 = 1.6042$

$y = Ax = 1.6042 * x$

Training A Simple Classifier

Let us pick Learning rate, L = 0.5

1st Training Data: x=3.0 and y=1.0

Initial $A = 0.25$

For $x = 3.0$, $y = 0.25 * 3.0 = 0.75$

$E = t - y = 1.1 - 0.75 = 0.35$

$\Delta A = L(E/x) = 0.5 * 0.35/3.0 = 0.0583$

$A = (A + \Delta A) = 0.25 + 0.0583 = 0.3083$

2nd Training Data: x=1.0 and y=3.0

For $x = 1.0$, $y = 0.3083 * 1.0 = 0.3083$

$E = t - y = 2.9 - 0.3083 = 2.5917$

$\Delta A = L(E/x) = 0.5 * 2.5917/1.0 = 1.2958$

$A = (A + \Delta A) = 0.3083 + 1.2958 = 1.6042$

$y = Ax = 1.6042 * x$

Training A Simple Classifier

Let us pick Learning rate, L = 0.5
1st Training Data: x=3.0 and y=1.0
Initial $A = 0.25$

For $x = 3.0$, $y = 0.25 * 3.0 = 0.75$

$E = t - y = 1.1 - 0.75 = 0.35$

$\Delta A = L(E/x) = 0.5 * 0.35/3.0 = 0.0583$

$A = (A + \Delta A) = 0.25 + 0.0583 = 0.3083$

2nd Training Data: x=1.0 and y=3.0
For $x = 1.0$, $y = 0.3083 * 1.0 = 0.3083$

$E = t - y = 2.9 - 0.3083 = 2.5917$

$\Delta A = L(E/x) = 0.5 * 2.5917/1.0 = 1.2958$

$A = (A + \Delta A) = 0.3083 + 1.2958 = 1.6042$

$y = Ax = 1.6042 * x$

Training A Simple Classifier

Let us pick Learning rate, L = 0.5
1st Training Data: x=3.0 and y=1.0
Initial $A = 0.25$

For $x = 3.0$, $y = 0.25 * 3.0 = 0.75$

$E = t - y = 1.1 - 0.75 = 0.35$

$\Delta A = L(E/x) = 0.5 * 0.35/3.0 = 0.0583$

$A = (A + \Delta A) = 0.25 + 0.0583 = 0.3083$

2nd Training Data: x=1.0 and y=3.0
For $x = 1.0$, $y = 0.3083 * 1.0 = 0.3083$

$E = t - y = 2.9 - 0.3083 = 2.5917$

$\Delta A = L(E/x) = 0.5 * 2.5917/1.0 = 1.2958$

$A = (A + \Delta A) = 0.3083 + 1.2958 = 1.6042$

$y = Ax = 1.6042 * x$

Training A Simple Classifier

Let us pick Learning rate, L = 0.5

1st Training Data: x=3.0 and y=1.0

Initial $A = 0.25$

For $x = 3.0$, $y = 0.25 * 3.0 = 0.75$

$E = t - y = 1.1 - 0.75 = 0.35$

$\Delta A = L(E/x) = 0.5 * 0.35/3.0 = 0.0583$

$A = (A + \Delta A) = 0.25 + 0.0583 = 0.3083$

2nd Training Data: x=1.0 and y=3.0

For $x = 1.0$, $y = 0.3083 * 1.0 = 0.3083$

$E = t - y = 2.9 - 0.3083 = 2.5917$

$\Delta A = L(E/x) = 0.5 * 2.5917/1.0 = 1.2958$

$A = (A + \Delta A) = 0.3083 + 1.2958 = 1.6042$

$y = Ax = 1.6042 * x$

Training A Simple Classifier

Let us pick Learning rate, L = 0.5

1st Training Data: x=3.0 and y=1.0

Initial $A = 0.25$

For $x = 3.0$, $y = 0.25 * 3.0 = 0.75$

$E = t - y = 1.1 - 0.75 = 0.35$

$\Delta A = L(E/x) = 0.5 * 0.35/3.0 = 0.0583$

$A = (A + \Delta A) = 0.25 + 0.0583 = 0.3083$

2nd Training Data: x=1.0 and y=3.0

For $x = 1.0$, $y = 0.3083 * 1.0 = 0.3083$

$E = t - y = 2.9 - 0.3083 = 2.5917$

$\Delta A = L(E/x) = 0.5 * 2.5917/1.0 = 1.2958$

$A = (A + \Delta A) = 0.3083 + 1.2958 = 1.6042$

$y = Ax = 1.6042 * x$

Training A Simple Classifier

Let us pick Learning rate, L = 0.5
1st Training Data: x=3.0 and y=1.0
Initial $A = 0.25$

For $x = 3.0$, $y = 0.25 * 3.0 = 0.75$

$E = t - y = 1.1 - 0.75 = 0.35$

$\Delta A = L(E/x) = 0.5 * 0.35/3.0 = 0.0583$

$A = (A + \Delta A) = 0.25 + 0.0583 = 0.3083$

2nd Training Data: x=1.0 and y=3.0
For $x = 1.0$, $y = 0.3083 * 1.0 = 0.3083$

$E = t - y = 2.9 - 0.3083 = 2.5917$

$\Delta A = L(E/x) = 0.5 * 2.5917/1.0 = 1.2958$

$A = (A + \Delta A) = 0.3083 + 1.2958 = 1.6042$

$y = Ax = 1.6042 * x$

Training A Simple Classifier

Let us pick Learning rate, L = 0.5

1st Training Data: x=3.0 and y=1.0

Initial $A = 0.25$

For $x = 3.0$, $y = 0.25 * 3.0 = 0.75$

$E = t - y = 1.1 - 0.75 = 0.35$

$\Delta A = L(E/x) = 0.5 * 0.35/3.0 = 0.0583$

$A = (A + \Delta A) = 0.25 + 0.0583 = 0.3083$

2nd Training Data: x=1.0 and y=3.0

For $x = 1.0$, $y = 0.3083 * 1.0 = 0.3083$

$E = t - y = 2.9 - 0.3083 = 2.5917$

$\Delta A = L(E/x) = 0.5 * 2.5917/1.0 = 1.2958$

$A = (A + \Delta A) = 0.3083 + 1.2958 = 1.6042$

$y = Ax = 1.6042 * x$

Training A Simple Classifier

Let us pick Learning rate, L = 0.5

1st Training Data: x=3.0 and y=1.0

Initial $A = 0.25$

For $x = 3.0$, $y = 0.25 * 3.0 = 0.75$

$E = t - y = 1.1 - 0.75 = 0.35$

$\Delta A = L(E/x) = 0.5 * 0.35/3.0 = 0.0583$

$A = (A + \Delta A) = 0.25 + 0.0583 = 0.3083$

2nd Training Data: x=1.0 and y=3.0

For $x = 1.0$, $y = 0.3083 * 1.0 = 0.3083$

$E = t - y = 2.9 - 0.3083 = 2.5917$

$\Delta A = L(E/x) = 0.5 * 2.5917/1.0 = 1.2958$

$A = (A + \Delta A) = 0.3083 + 1.2958 = 1.6042$

$y = Ax = 1.6042 * x$

Training A Simple Classifier

Let us pick Learning rate, L = 0.5

1st Training Data: x=3.0 and y=1.0

Initial $A = 0.25$

For $x = 3.0$, $y = 0.25 * 3.0 = 0.75$

$E = t - y = 1.1 - 0.75 = 0.35$

$\Delta A = L(E/x) = 0.5 * 0.35/3.0 = 0.0583$

$A = (A + \Delta A) = 0.25 + 0.0583 = 0.3083$

2nd Training Data: x=1.0 and y=3.0

For $x = 1.0$, $y = 0.3083 * 1.0 = 0.3083$

$E = t - y = 2.9 - 0.3083 = 2.5917$

$\Delta A = L(E/x) = 0.5 * 2.5917/1.0 = 1.2958$

$A = (A + \Delta A) = 0.3083 + 1.2958 = 1.6042$

$y = Ax = 1.6042 * x$

Training A Simple Classifier

Let us pick Learning rate, L = 0.5
1st Training Data: x=3.0 and y=1.0
Initial $A = 0.25$

For $x = 3.0$, $y = 0.25 * 3.0 = 0.75$

$E = t - y = 1.1 - 0.75 = 0.35$

$\Delta A = L(E/x) = 0.5 * 0.35/3.0 = 0.0583$

$A = (A + \Delta A) = 0.25 + 0.0583 = 0.3083$

2nd Training Data: x=1.0 and y=3.0
For $x = 1.0$, $y = 0.3083 * 1.0 = 0.3083$

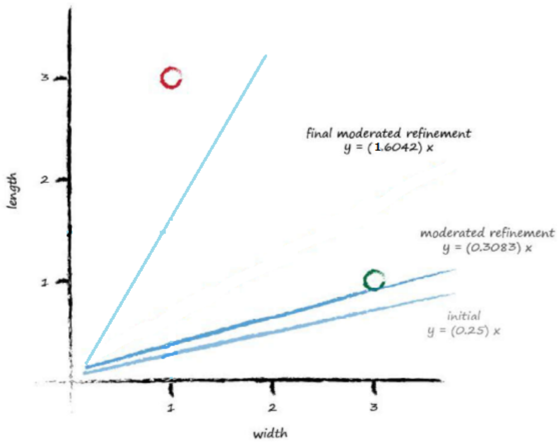
$E = t - y = 2.9 - 0.3083 = 2.5917$

$\Delta A = L(E/x) = 0.5 * 2.5917/1.0 = 1.2958$

$A = (A + \Delta A) = 0.3083 + 1.2958 = 1.6042$

$y = Ax = 1.6042 * x$

Training A Simple Classifier



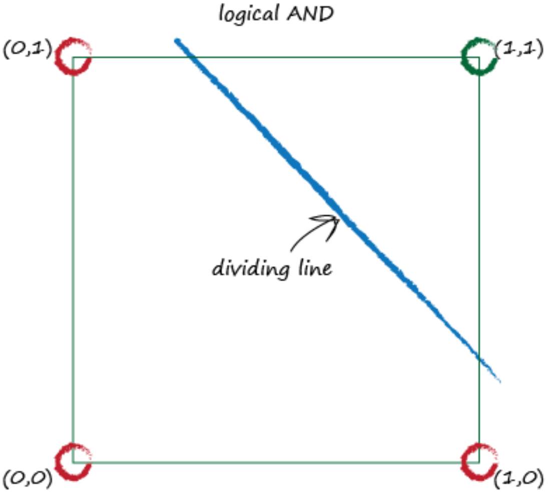
Training A Simple Classifier

- Understand the relationship between the output error of a linear classifier and the adjustable slope parameter.
- Moderate the updates with a learning rate so no single training example totally dominates the learning.
- Training examples from the real world can be noisy or contain errors. Moderating updates in this way helpfully limits the impact of these false examples.

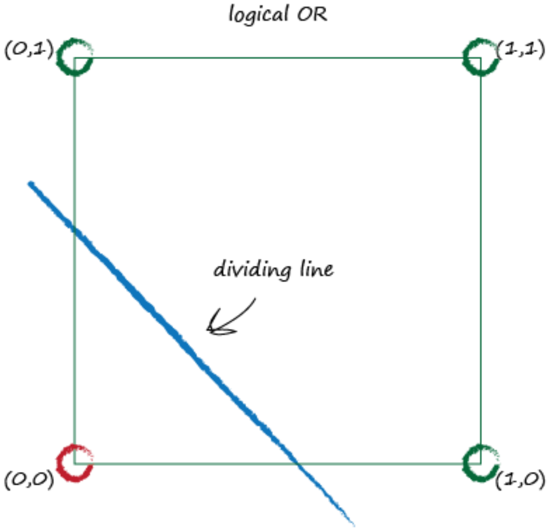
Sometimes One Classifier Is Not Enough

Input A	Input B	Logical AND	Logical OR
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	1

Sometimes One Classifier Is Not Enough



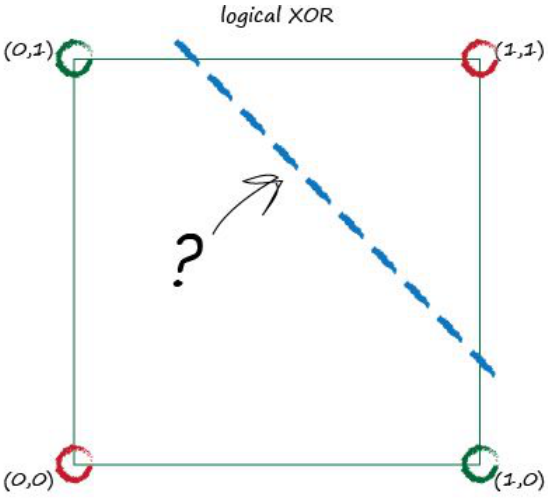
Sometimes One Classifier Is Not Enough



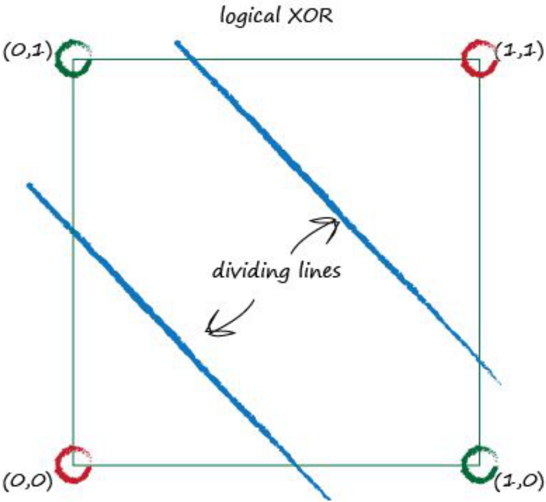
Sometimes One Classifier Is Not Enough

Input A	Input B	Logical XOR
0	0	0
0	1	1
1	0	1
1	1	0

Sometimes One Classifier Is Not Enough

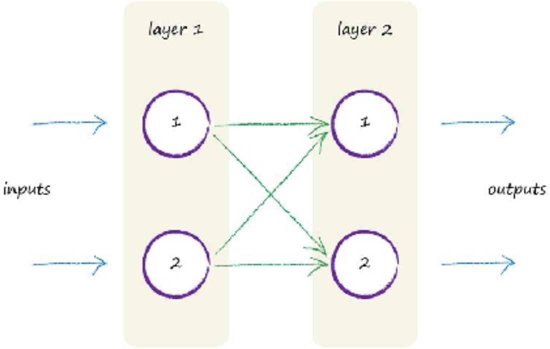


Sometimes One Classifier Is Not Enough



Break

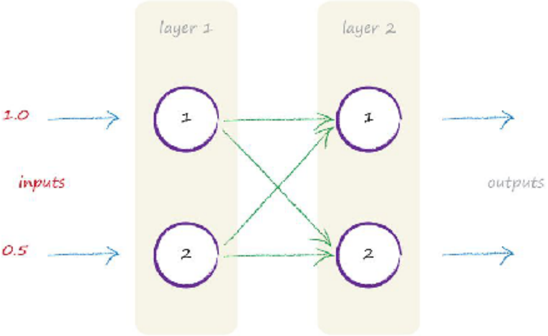
Signals Through A Neural Network



$\text{input}_1 = 1.0$

$\text{input}_2 = 1.0$

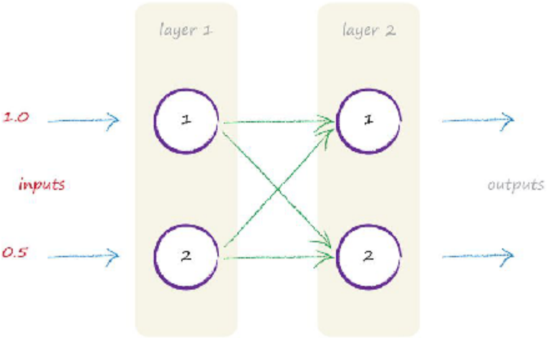
Signals Through A Neural Network



$\text{input}_1 = 1.0$

$\text{input}_2 = 1.0$

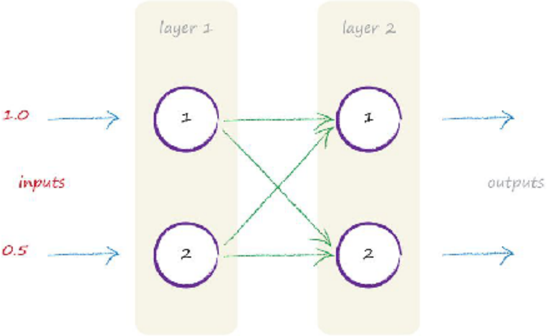
Signals Through A Neural Network



$\text{input}_1 = 1.0$

$\text{input}_2 = 1.0$

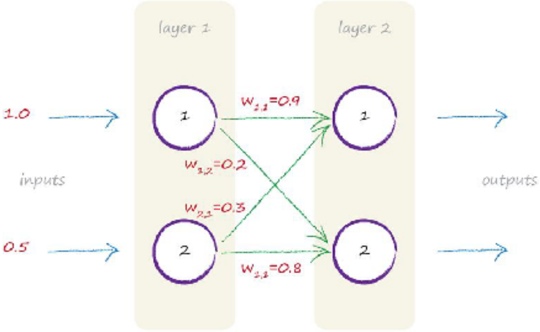
Signals Through A Neural Network



$\text{input}_1 = 1.0$

$\text{input}_2 = 1.0$

Signals Through A Neural Network



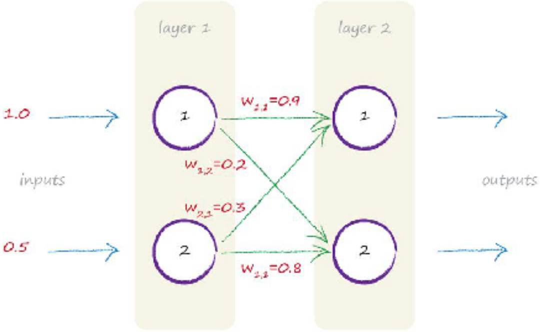
$w_{1,1} = 0.9$

$w_{1,2} = 0.2$

$w_{2,1} = 0.3$

$w_{2,2} = 0.8$

Signals Through A Neural Network



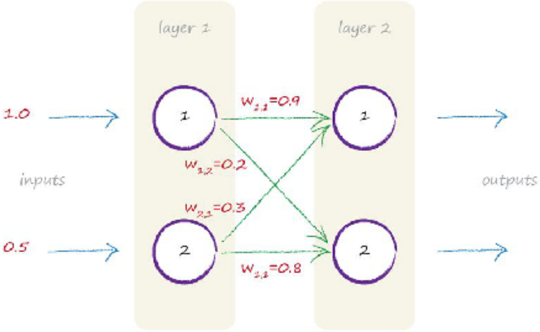
$$w_{1,1} = 0.9$$

$$w_{1,2} = 0.2$$

$$w_{2,1} = 0.3$$

$$w_{2,2} = 0.8$$

Signals Through A Neural Network



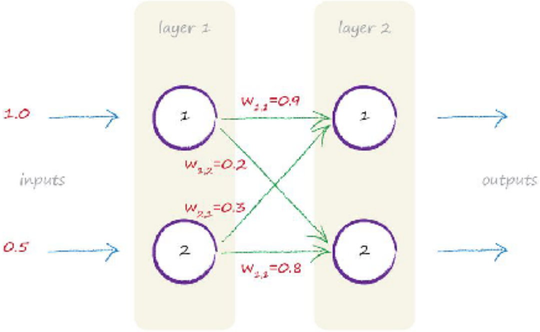
$w_{1,1} = 0.9$

$w_{1,2} = 0.2$

$w_{2,1} = 0.3$

$w_{2,2} = 0.8$

Signals Through A Neural Network



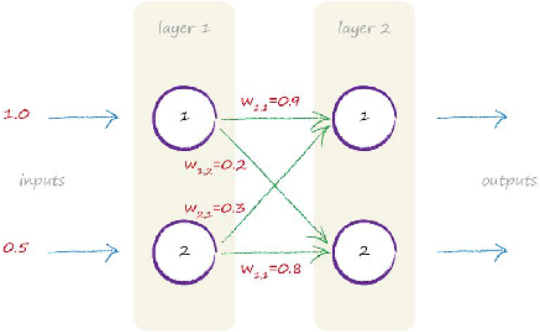
$$w_{1,1} = 0.9$$

$$w_{1,2} = 0.2$$

$$w_{2,1} = 0.3$$

$$w_{2,2} = 0.8$$

Signals Through A Neural Network



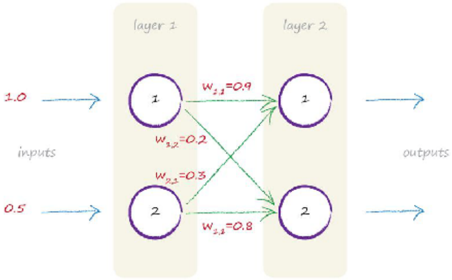
$w_{1,1} = 0.9$

$w_{1,2} = 0.2$

$w_{2,1} = 0.3$

$w_{2,2} = 0.8$

Signals Through A Neural Network



$$x_1 = (input_1 * w_{1,1} + input_2 * w_{2,1})$$

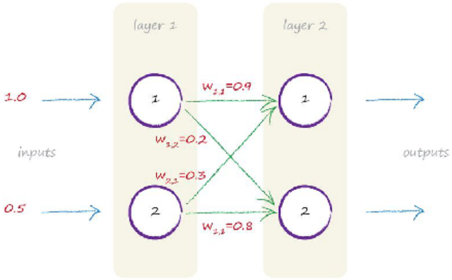
$$x_1 = (1.0 * 0.9) + (0.5 * 0.3)$$

$$x_1 = 0.9 + 0.15 = 1.05$$

$$y_1 = 1/(1 + e^{-x}) = 1/(1 + e^{-1.05})$$

$$y_1 = 1/(1 + 0.3499) = 1/1.3499 = 0.7408$$

Signals Through A Neural Network



$$x_1 = (input_1 * w_{1,1} + input_2 * w_{2,1})$$

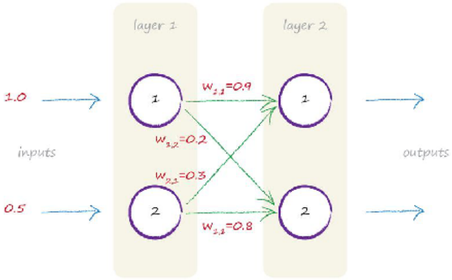
$$x_1 = (1.0 * 0.9) + (0.5 * 0.3)$$

$$x_1 = 0.9 + 0.15 = 1.05$$

$$y_1 = 1/(1 + e^{-x}) = 1/(1 + e^{-1.05})$$

$$y_1 = 1/(1 + 0.3499) = 1/1.3499 = 0.7408$$

Signals Through A Neural Network



$$x_1 = (input_1 * w_{1,1} + input_2 * w_{2,1})$$

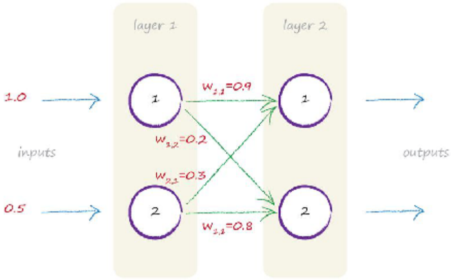
$$x_1 = (1.0 * 0.9) + (0.5 * 0.3)$$

$$x_1 = 0.9 + 0.15 = 1.05$$

$$y_1 = 1/(1 + e^{-x}) = 1/(1 + e^{-1.05})$$

$$y_1 = 1/(1 + 0.3499) = 1/1.3499 = 0.7408$$

Signals Through A Neural Network



$$x_1 = (input_1 * w_{1,1} + input_2 * w_{2,1})$$

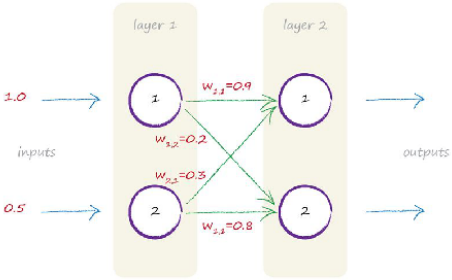
$$x_1 = (1.0 * 0.9) + (0.5 * 0.3)$$

$$x_1 = 0.9 + 0.15 = 1.05$$

$$y_1 = 1/(1 + e^{-x}) = 1/(1 + e^{-1.05})$$

$$y_1 = 1/(1 + 0.3499) = 1/1.3499 = 0.7408$$

Signals Through A Neural Network



$$x_1 = (input_1 * w_{1,1} + input_2 * w_{2,1})$$

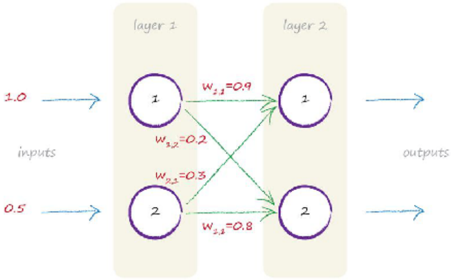
$$x_1 = (1.0 * 0.9) + (0.5 * 0.3)$$

$$x_1 = 0.9 + 0.15 = 1.05$$

$$y_1 = 1/(1 + e^{-x}) = 1/(1 + e^{-1.05})$$

$$y_1 = 1/(1 + 0.3499) = 1/1.3499 = 0.7408$$

Signals Through A Neural Network



$x_1 = (input_1 * w_{1,1} + input_2 * w_{2,1})$

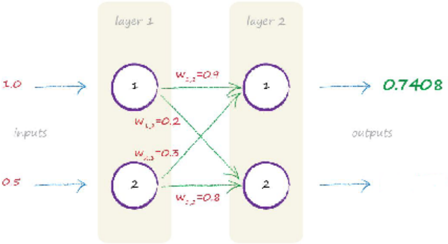
$x_1 = (1.0 * 0.9) + (0.5 * 0.3)$

$x_1 = 0.9 + 0.15 = 1.05$

$y_1 = 1/(1 + e^{-x}) = 1/(1 + e^{-1.05})$

$y_1 = 1/(1 + 0.3499) = 1/1.3499 = 0.7408$

Signals Through A Neural Network



$$x_1 = (input_1 * w_{1,1} + input_2 * w_{2,1})$$

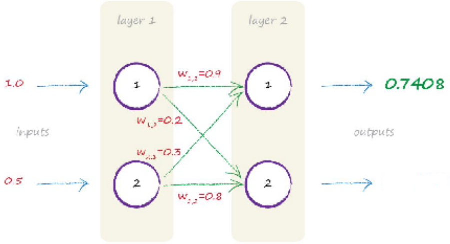
$$x_1 = (1.0 * 0.9) + (0.5 * 0.3)$$

$$x_1 = 0.9 + 0.15 = 1.05$$

$$y_1 = 1/(1 + e^{-x}) = 1/(1 + e^{-1.05})$$

$$y_1 = 1/(1 + 0.3499) = 1/1.3499 = 0.7408$$

Signals Through A Neural Network



$$x_2 = (input_1 * w_{1,2} + input_2 * w_{2,2})$$

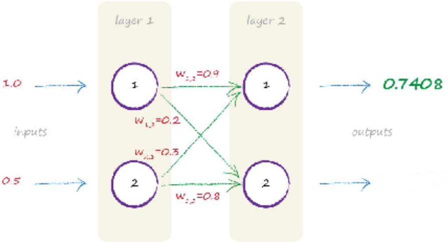
$$x_2 = (1.0 * 0.2) + (0.5 * 0.8)$$

$$x_2 = 0.2 + 0.4 = 0.6$$

$$y_2 = 1/(1 + e^{-x}) = 1/(1 + e^{-0.6})$$

$$y_2 = 1/(1 + 0.5488) = 1/1.5488 = 0.6457$$

Signals Through A Neural Network



$$x_2 = (input_1 * w_{1,2} + input_2 * w_{2,2})$$

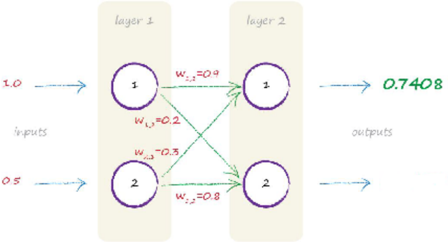
$$x_2 = (1.0 * 0.2) + (0.5 * 0.8)$$

$$x_2 = 0.2 + 0.4 = 0.6$$

$$y_2 = 1/(1 + e^{-x}) = 1/(1 + e^{-0.6})$$

$$y_2 = 1/(1 + 0.5488) = 1/1.5488 = 0.6457$$

Signals Through A Neural Network



$$x_2 = (input_1 * w_{1,2} + input_2 * w_{2,2})$$

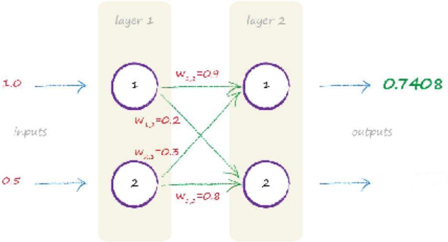
$$x_2 = (1.0 * 0.2) + (0.5 * 0.8)$$

$$x_2 = 0.2 + 0.4 = 0.6$$

$$y_2 = 1/(1 + e^{-x}) = 1/(1 + e^{-0.6})$$

$$y_2 = 1/(1 + 0.5488) = 1/1.5488 = 0.6457$$

Signals Through A Neural Network



$$x_2 = (input_1 * w_{1,2} + input_2 * w_{2,2})$$

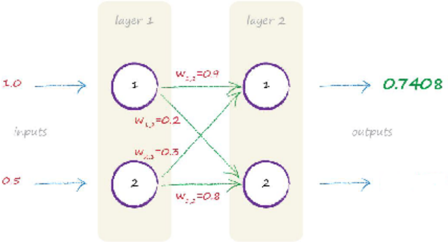
$$x_2 = (1.0 * 0.2) + (0.5 * 0.8)$$

$$x_2 = 0.2 + 0.4 = 0.6$$

$$y_2 = 1/(1 + e^{-x}) = 1/(1 + e^{-0.6})$$

$$y_2 = 1/(1 + 0.5488) = 1/1.5488 = 0.6457$$

Signals Through A Neural Network



$$x_2 = (input_1 * w_{1,2} + input_2 * w_{2,2})$$

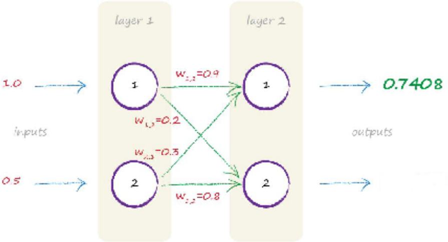
$$x_2 = (1.0 * 0.2) + (0.5 * 0.8)$$

$$x_2 = 0.2 + 0.4 = 0.6$$

$$y_2 = 1/(1 + e^{-x}) = 1/(1 + e^{-0.6})$$

$$y_2 = 1/(1 + 0.5488) = 1/1.5488 = 0.6457$$

Signals Through A Neural Network



$$x_2 = (input_1 * w_{1,2} + input_2 * w_{2,2})$$

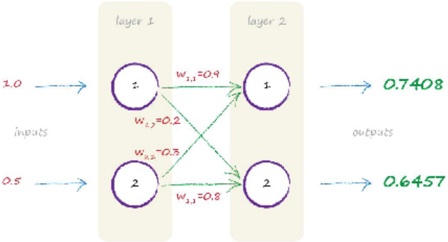
$$x_2 = (1.0 * 0.2) + (0.5 * 0.8)$$

$$x_2 = 0.2 + 0.4 = 0.6$$

$$y_2 = 1/(1 + e^{-x}) = 1/(1 + e^{-0.6})$$

$$y_2 = 1/(1 + 0.5488) = 1/1.5488 = 0.6457$$

Signals Through A Neural Network



$$x_2 = (input_1 * w_{1,2} + input_2 * w_{2,2})$$

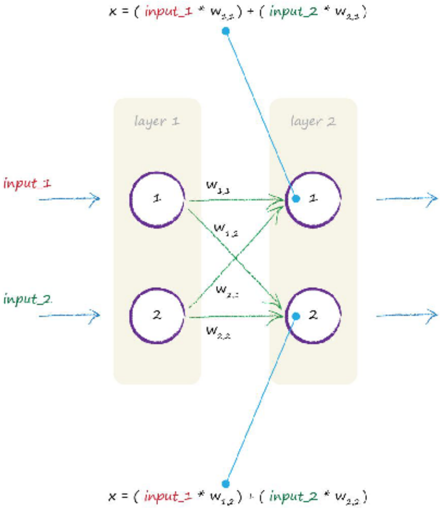
$$x_2 = (1.0 * 0.2) + (0.5 * 0.8)$$

$$x_2 = 0.2 + 0.4 = 0.6$$

$$y_2 = 1/(1 + e^{-x}) = 1/(1 + e^{-0.6})$$

$$y_2 = 1/(1 + 0.5488) = 1/1.5488 = 0.6457$$

Matrix Multiplication is Useful



$$\begin{pmatrix} w_{1,1} & w_{2,1} \\ w_{1,2} & w_{2,2} \end{pmatrix} \begin{pmatrix} input_1 \\ input_2 \end{pmatrix}$$

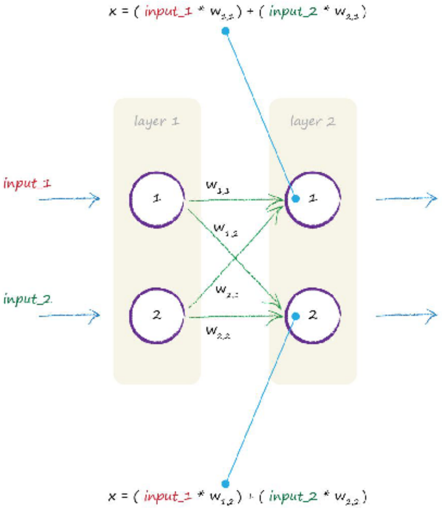
$$=$$

$$\begin{pmatrix} (input_1 * w_{1,1}) + (input_2 * w_{2,1}) \\ (input_1 * w_{1,2}) + (input_2 * w_{2,2}) \end{pmatrix}$$

$$\mathbf{X} = \mathbf{W}.\mathbf{I}$$

$$\mathbf{O} = \text{sigmoid}(\mathbf{X})$$

Matrix Multiplication is Useful



$$\begin{pmatrix} w_{1,1} & w_{2,1} \\ w_{1,2} & w_{2,2} \end{pmatrix} \begin{pmatrix} input_1 \\ input_2 \end{pmatrix}$$

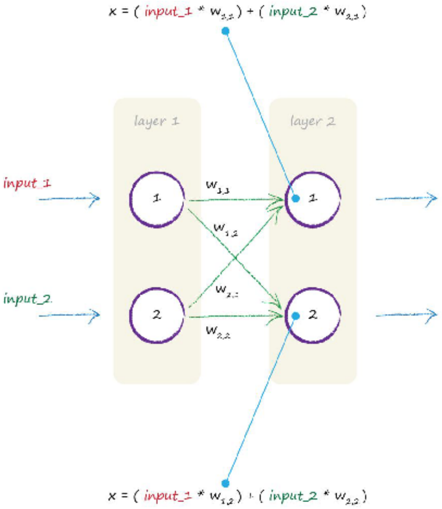
$$=$$

$$\begin{pmatrix} (input_1 * w_{1,1}) + (input_2 * w_{2,1}) \\ (input_1 * w_{1,2}) + (input_2 * w_{2,2}) \end{pmatrix}$$

$$\mathbf{X} = \mathbf{W}.\mathbf{I}$$

$$\mathbf{O} = \text{sigmoid}(\mathbf{X})$$

Matrix Multiplication is Useful



$$\begin{pmatrix} w_{1,1} & w_{2,1} \\ w_{1,2} & w_{2,2} \end{pmatrix} \begin{pmatrix} input_1 \\ input_2 \end{pmatrix}$$

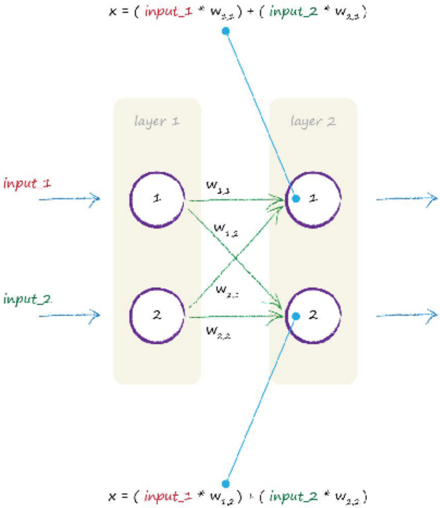
$$=$$

$$\begin{pmatrix} (input_1 * w_{1,1}) + (input_2 * w_{2,1}) \\ (input_1 * w_{1,2}) + (input_2 * w_{2,2}) \end{pmatrix}$$

$$\mathbf{X} = \mathbf{W}.\mathbf{I}$$

$$\mathbf{O} = \text{sigmoid}(\mathbf{X})$$

Matrix Multiplication is Useful



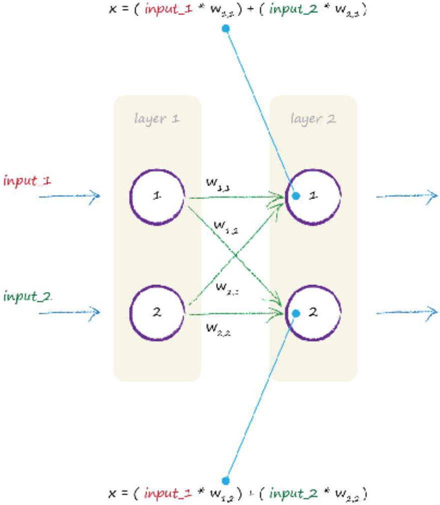
$$\begin{pmatrix} w_{1,1} & w_{2,1} \\ w_{1,2} & w_{2,2} \end{pmatrix} \begin{pmatrix} input_1 \\ input_2 \end{pmatrix}$$

$$= \begin{pmatrix} (input_1 * w_{1,1}) + (input_2 * w_{2,1}) \\ (input_1 * w_{1,2}) + (input_2 * w_{2,2}) \end{pmatrix}$$

$$\mathbf{X} = \mathbf{W}.\mathbf{I}$$

$$\mathbf{O} = \text{sigmoid}(\mathbf{X})$$

Matrix Multiplication is Useful



$$\begin{pmatrix} w_{1,1} & w_{2,1} \\ w_{1,2} & w_{2,2} \end{pmatrix} \begin{pmatrix} input_1 \\ input_2 \end{pmatrix}$$

$$=$$

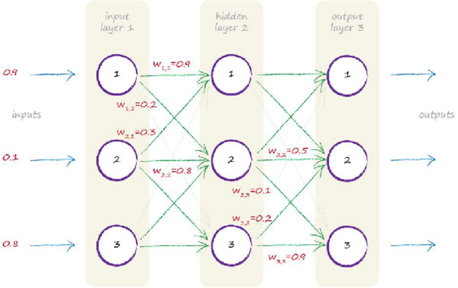
$$\begin{pmatrix} (input_1 * w_{1,1}) + (input_2 * w_{2,1}) \\ (input_1 * w_{1,2}) + (input_2 * w_{2,2}) \end{pmatrix}$$

$$\mathbf{X} = \mathbf{W}.\mathbf{I}$$

$$\mathbf{O} = \text{sigmoid}(\mathbf{X})$$

A Three Layer Example

$$\mathbf{I} = \begin{pmatrix} 0.9 \\ 0.1 \\ 0.8 \end{pmatrix}$$

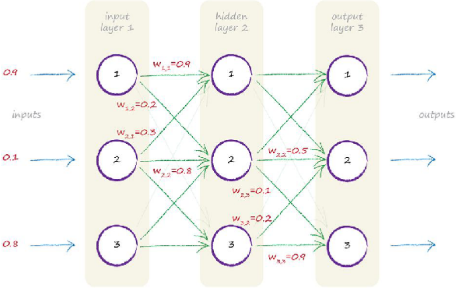


$$\mathbf{W}_{input\_hidden} = \begin{pmatrix} 0.9 & 0.3 & 0.4 \\ 0.2 & 0.8 & 0.2 \\ 0.1 & 0.5 & 0.6 \end{pmatrix}$$

$$\mathbf{W}_{hidden\_output} = \begin{pmatrix} 0.3 & 0.7 & 0.5 \\ 0.6 & 0.5 & 0.2 \\ 0.8 & 0.1 & 0.9 \end{pmatrix}$$

A Three Layer Example

$$\mathbf{I} = \begin{pmatrix} 0.9 \\ 0.1 \\ 0.8 \end{pmatrix}$$

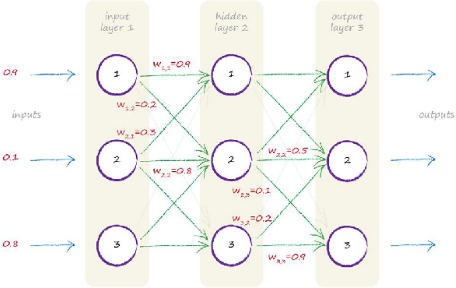


$$\mathbf{W}_{input\_hidden} = \begin{pmatrix} 0.9 & 0.3 & 0.4 \\ 0.2 & 0.8 & 0.2 \\ 0.1 & 0.5 & 0.6 \end{pmatrix}$$

$$\mathbf{W}_{hidden\_output} = \begin{pmatrix} 0.3 & 0.7 & 0.5 \\ 0.6 & 0.5 & 0.2 \\ 0.8 & 0.1 & 0.9 \end{pmatrix}$$

A Three Layer Example

$$\mathbf{I} = \begin{pmatrix} 0.9 \\ 0.1 \\ 0.8 \end{pmatrix}$$

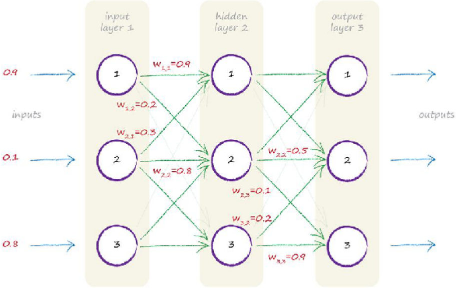


$$\mathbf{W}_{input\_hidden} = \begin{pmatrix} 0.9 & 0.3 & 0.4 \\ 0.2 & 0.8 & 0.2 \\ 0.1 & 0.5 & 0.6 \end{pmatrix}$$

$$\mathbf{W}_{hidden\_output} = \begin{pmatrix} 0.3 & 0.7 & 0.5 \\ 0.6 & 0.5 & 0.2 \\ 0.8 & 0.1 & 0.9 \end{pmatrix}$$

A Three Layer Example

$$\mathbf{I} = \begin{pmatrix} 0.9 \\ 0.1 \\ 0.8 \end{pmatrix}$$

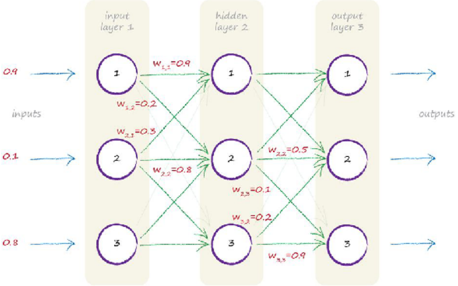


$$\mathbf{W}_{input\_hidden} = \begin{pmatrix} 0.9 & 0.3 & 0.4 \\ 0.2 & 0.8 & 0.2 \\ 0.1 & 0.5 & 0.6 \end{pmatrix}$$

$$\mathbf{W}_{hidden\_output} = \begin{pmatrix} 0.3 & 0.7 & 0.5 \\ 0.6 & 0.5 & 0.2 \\ 0.8 & 0.1 & 0.9 \end{pmatrix}$$

A Three Layer Example

$$\mathbf{X_{hidden}} = \mathbf{W_{input\_hidden}} . \mathbf{I}$$

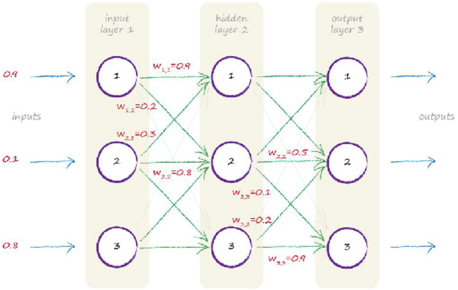


$$\mathbf{X_{hidden}} = \begin{pmatrix} 0.9 & 0.3 & 0.4 \\ 0.2 & 0.8 & 0.2 \\ 0.1 & 0.5 & 0.6 \end{pmatrix} . \begin{pmatrix} 0.9 \\ 0.1 \\ 0.8 \end{pmatrix}$$

$$\mathbf{X_{hidden}} = \begin{pmatrix} 1.16 \\ 0.42 \\ 0.62 \end{pmatrix}$$

A Three Layer Example

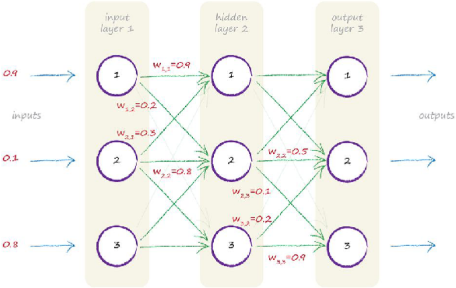
$$\mathbf{X_{hidden}} = \mathbf{W_{input\_hidden}} . \mathbf{I}$$



$$\mathbf{X_{hidden}} = \begin{pmatrix} 0.9 & 0.3 & 0.4 \\ 0.2 & 0.8 & 0.2 \\ 0.1 & 0.5 & 0.6 \end{pmatrix} . \begin{pmatrix} 0.9 \\ 0.1 \\ 0.8 \end{pmatrix}$$

$$\mathbf{X_{hidden}} = \begin{pmatrix} 1.16 \\ 0.42 \\ 0.62 \end{pmatrix}$$

A Three Layer Example



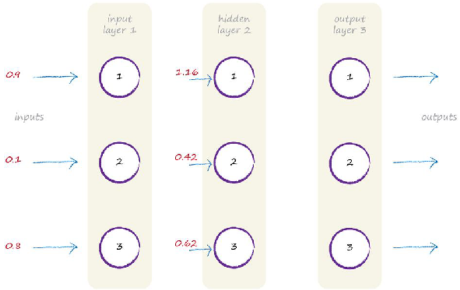
$$\mathbf{X_{hidden}} = \mathbf{W_{input\_hidden}} . \mathbf{I}$$

$$\mathbf{X_{hidden}} = \begin{pmatrix} 0.9 & 0.3 & 0.4 \\ 0.2 & 0.8 & 0.2 \\ 0.1 & 0.5 & 0.6 \end{pmatrix} . \begin{pmatrix} 0.9 \\ 0.1 \\ 0.8 \end{pmatrix}$$

$$\mathbf{X_{hidden}} = \begin{pmatrix} 1.16 \\ 0.42 \\ 0.62 \end{pmatrix}$$

A Three Layer Example

$$\mathbf{X_{hidden}} = \mathbf{W_{input\_hidden}}.\mathbf{I}$$

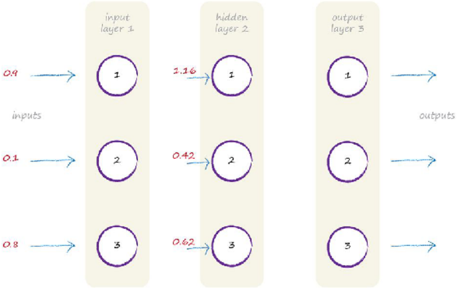


$$\mathbf{X_{hidden}} = \begin{pmatrix} 0.9 & 0.3 & 0.4 \\ 0.2 & 0.8 & 0.2 \\ 0.1 & 0.5 & 0.6 \end{pmatrix} . \begin{pmatrix} 0.9 \\ 0.1 \\ 0.8 \end{pmatrix}$$

$$\mathbf{X_{hidden}} = \begin{pmatrix} 1.16 \\ 0.42 \\ 0.62 \end{pmatrix}$$

A Three Layer Example

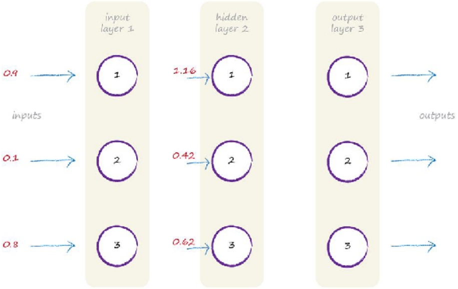
$$\mathbf{O}_{\mathbf{hidden}} = \text{sigmoid}(\mathbf{X}_{\mathbf{hidden}})$$



$$\mathbf{O}_{\mathbf{hidden}} = \text{sigmoid}\begin{pmatrix} 1.16 \\ 0.42 \\ 0.62 \end{pmatrix}$$

$$\mathbf{O}_{\mathbf{hidden}} = \begin{pmatrix} 0.761 \\ 0.603 \\ 0.650 \end{pmatrix}$$

A Three Layer Example

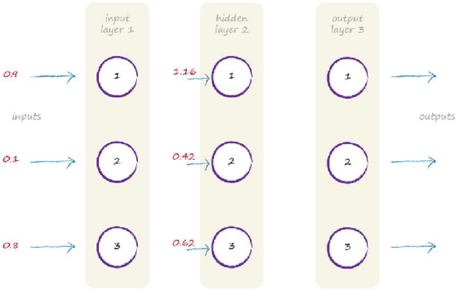


$$\mathbf{O_{hidden}} = \text{sigmoid}(\mathbf{X_{hidden}})$$

$$\mathbf{O_{hidden}} = \text{sigmoid}\begin{pmatrix} 1.16 \\ 0.42 \\ 0.62 \end{pmatrix}$$

$$\mathbf{O_{hidden}} = \begin{pmatrix} 0.761 \\ 0.603 \\ 0.650 \end{pmatrix}$$

A Three Layer Example

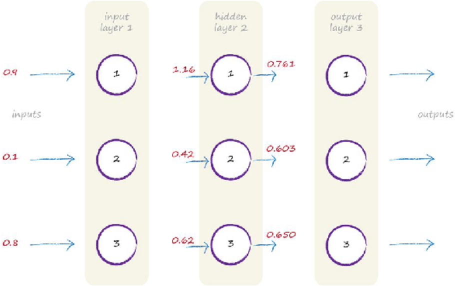


$$\mathbf{O}_{\mathbf{hidden}} = \mathsf{sigmoid}(\mathbf{X}_{\mathbf{hidden}})$$

$$\mathbf{O}_{\mathbf{hidden}} = \mathsf{sigmoid}\begin{pmatrix} 1.16 \\ 0.42 \\ 0.62 \end{pmatrix}$$

$$\mathbf{O}_{\mathbf{hidden}} = \begin{pmatrix} 0.761 \\ 0.603 \\ 0.650 \end{pmatrix}$$

A Three Layer Example



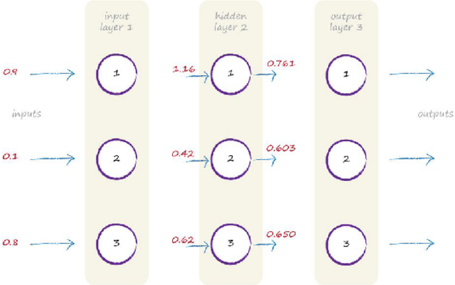
$$\mathbf{O_{hidden}} = \mathsf{sigmoid}(\mathbf{X_{hidden}})$$

$$\mathbf{O_{hidden}} = \mathsf{sigmoid}\begin{pmatrix} 1.16 \\ 0.42 \\ 0.62 \end{pmatrix}$$

$$\mathbf{O_{hidden}} = \begin{pmatrix} 0.761 \\ 0.603 \\ 0.650 \end{pmatrix}$$

A Three Layer Example

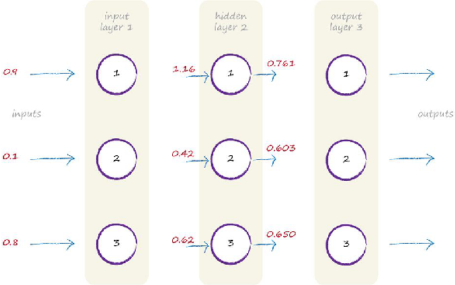
$$\mathbf{X_{output}} = \mathbf{W_{hidden\_output}} \cdot \mathbf{O_{hidden}}$$



$$\mathbf{X_{output}} = \begin{pmatrix} 0.3 & 0.7 & 0.5 \\ 0.6 & 0.5 & 0.2 \\ 0.8 & 0.1 & 0.9 \end{pmatrix} \cdot \begin{pmatrix} 0.761 \\ 0.603 \\ 0.650 \end{pmatrix}$$

$$\mathbf{X_{output}} = \begin{pmatrix} 0.975 \\ 0.888 \\ 1.254 \end{pmatrix}$$

A Three Layer Example

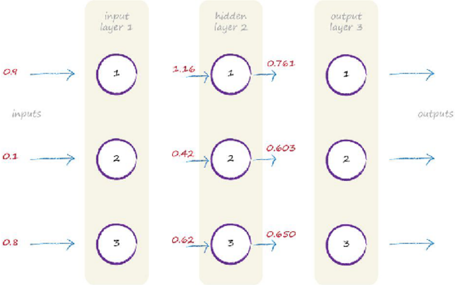


$$\mathbf{X}_{\mathbf{output}} = \mathbf{W}_{\mathbf{hidden\_output}} \cdot \mathbf{O}_{\mathbf{hidden}}$$

$$\mathbf{X}_{\mathbf{output}} = \begin{pmatrix} 0.3 & 0.7 & 0.5 \\ 0.6 & 0.5 & 0.2 \\ 0.8 & 0.1 & 0.9 \end{pmatrix} \cdot \begin{pmatrix} 0.761 \\ 0.603 \\ 0.650 \end{pmatrix}$$

$$\mathbf{X}_{\mathbf{output}} = \begin{pmatrix} 0.975 \\ 0.888 \\ 1.254 \end{pmatrix}$$

A Three Layer Example



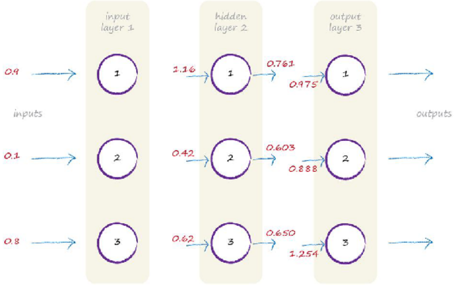
$$\mathbf{X}_{\mathbf{output}} = \mathbf{W}_{\mathbf{hidden\_output}} \cdot \mathbf{O}_{\mathbf{hidden}}$$

$$\mathbf{X}_{\mathbf{output}} = \begin{pmatrix} 0.3 & 0.7 & 0.5 \\ 0.6 & 0.5 & 0.2 \\ 0.8 & 0.1 & 0.9 \end{pmatrix} \cdot \begin{pmatrix} 0.761 \\ 0.603 \\ 0.650 \end{pmatrix}$$

$$\mathbf{X}_{\mathbf{output}} = \begin{pmatrix} 0.975 \\ 0.888 \\ 1.254 \end{pmatrix}$$

A Three Layer Example

$$\mathbf{X}_{\text{output}} = \mathbf{W}_{\text{hidden\_output}} \cdot \mathbf{O}_{\text{hidden}}$$

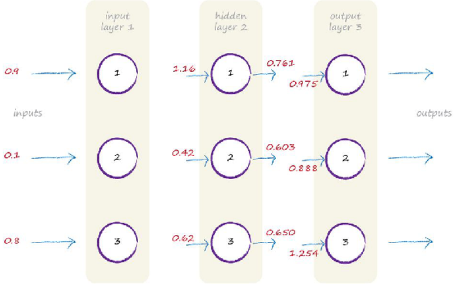


$$\mathbf{X}_{\text{output}} = \begin{pmatrix} 0.3 & 0.7 & 0.5 \\ 0.6 & 0.5 & 0.2 \\ 0.8 & 0.1 & 0.9 \end{pmatrix} \cdot \begin{pmatrix} 0.761 \\ 0.603 \\ 0.650 \end{pmatrix}$$

$$\mathbf{X}_{\text{output}} = \begin{pmatrix} 0.975 \\ 0.888 \\ 1.254 \end{pmatrix}$$

A Three Layer Example

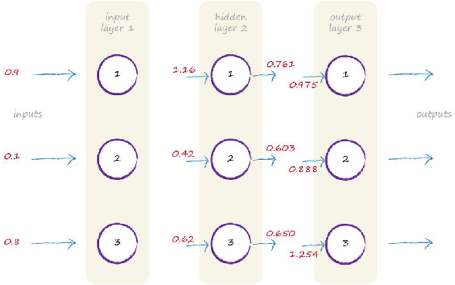
$$\mathbf{O_{output}} = \text{sigmoid}(\mathbf{X_{output}})$$



$$\mathbf{O_{output}} = \text{sigmoid}\begin{pmatrix} 0.975 \\ 0.888 \\ 1.254 \end{pmatrix}$$

$$\mathbf{O_{output}} = \begin{pmatrix} 0.726 \\ 0.708 \\ 0.778 \end{pmatrix}$$

A Three Layer Example

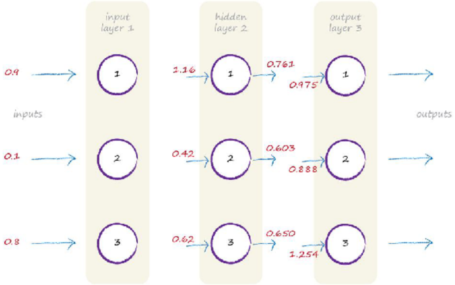


$$\mathbf{O}_{\mathbf{output}} = \text{sigmoid}(\mathbf{X}_{\mathbf{output}})$$

$$\mathbf{O}_{\mathbf{output}} = \text{sigmoid}\begin{pmatrix} 0.975 \\ 0.888 \\ 1.254 \end{pmatrix}$$

$$\mathbf{O}_{\mathbf{output}} = \begin{pmatrix} 0.726 \\ 0.708 \\ 0.778 \end{pmatrix}$$

A Three Layer Example

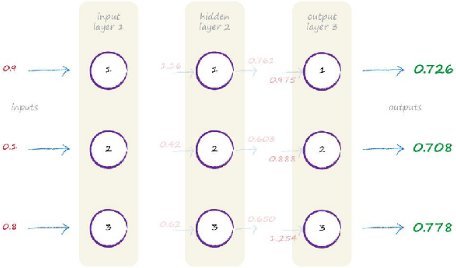


$$\mathbf{O}_{\mathbf{output}} = \text{sigmoid}(\mathbf{X}_{\mathbf{output}})$$

$$\mathbf{O}_{\mathbf{output}} = \text{sigmoid}\begin{pmatrix} 0.975 \\ 0.888 \\ 1.254 \end{pmatrix}$$

$$\mathbf{O}_{\mathbf{output}} = \begin{pmatrix} 0.726 \\ 0.708 \\ 0.778 \end{pmatrix}$$

A Three Layer Example

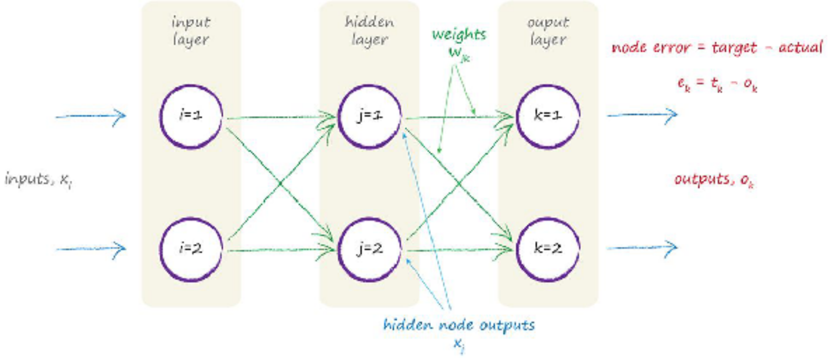


$$\mathbf{O}_{\mathbf{output}} = \text{sigmoid}(\mathbf{X}_{\mathbf{output}})$$

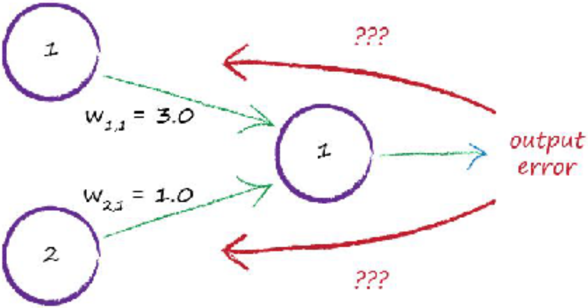
$$\mathbf{O}_{\mathbf{output}} = \text{sigmoid}\begin{pmatrix} 0.975 \\ 0.888 \\ 1.254 \end{pmatrix}$$

$$\mathbf{O}_{\mathbf{output}} = \begin{pmatrix} 0.726 \\ 0.708 \\ 0.778 \end{pmatrix}$$

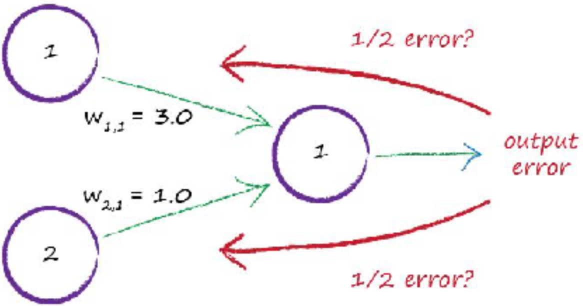
Backpropagating Errors



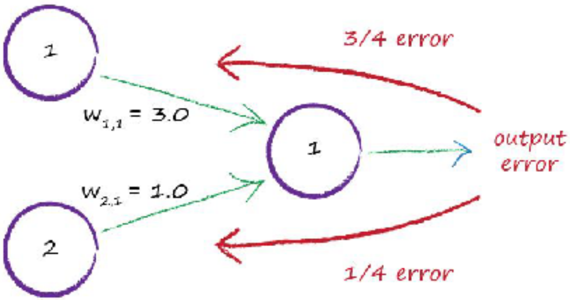
Backpropagating Errors



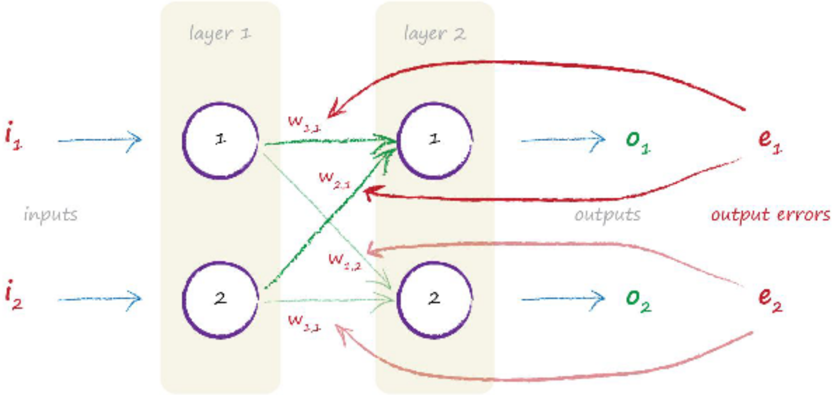
Backpropagating Errors



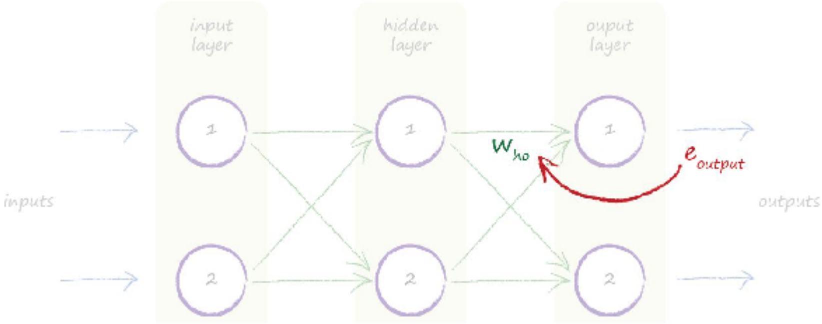
Backpropagating Errors



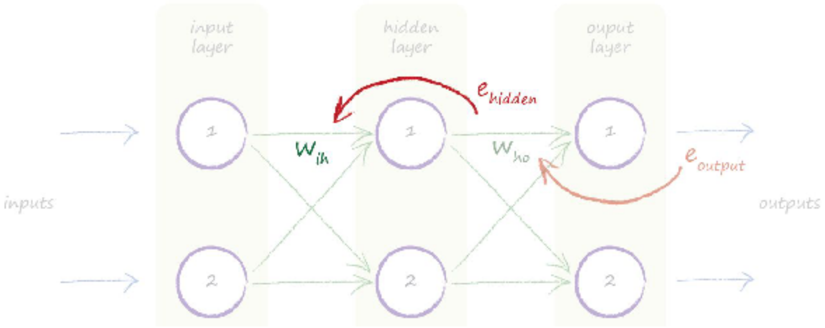
Backpropagating Errors



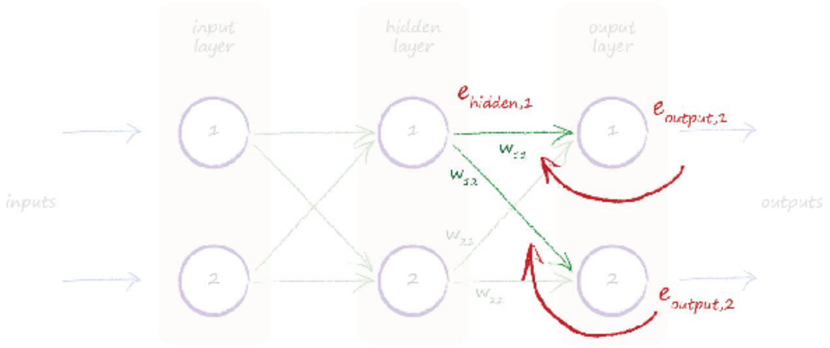
Backpropagating Errors



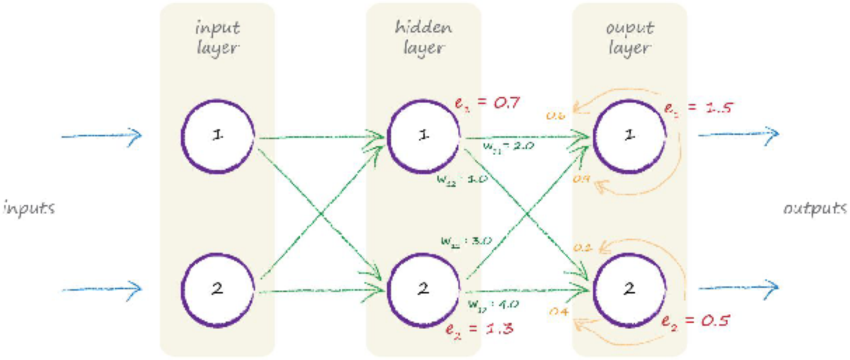
Backpropagating Errors



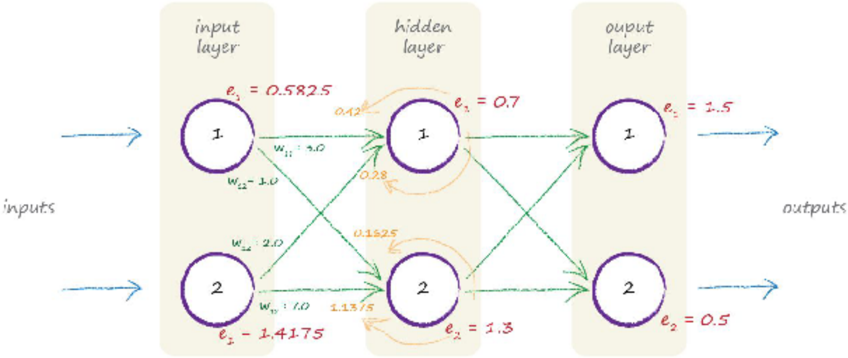
Backpropagating Errors



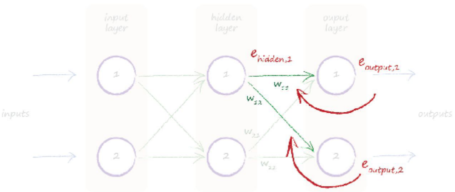
Backpropagating Errors



Backpropagating Errors



Errors with Matrix Multiplication



$$\mathbf{e}_{\mathbf{output}} = \begin{pmatrix} e_1 \\ e_2 \end{pmatrix}$$

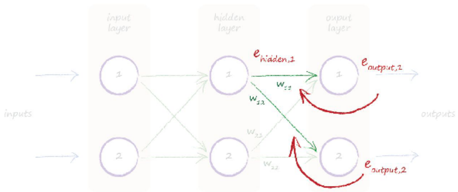
$$\mathbf{e}_{\mathbf{hidden}} = \begin{pmatrix} (e_1 * w_{1,1}) + (e2 * w_{1,2}) \\ (e_1 * w_{2,1}) + (e2 * w_{2,2}) \end{pmatrix}$$

$$\mathbf{e}_{\mathbf{hidden}} = \begin{pmatrix} w_{1,1} & w_{1,2} \\ w_{2,1} & w_{2,2} \end{pmatrix} \cdot \begin{pmatrix} e_1 \\ e_2 \end{pmatrix}$$

$$\mathbf{W}_{\mathbf{hidden\_output}} = \begin{pmatrix} w_{1,1} & w_{2,1} \\ w_{1,2} & w_{2,2} \end{pmatrix}$$

$$\mathbf{e}_{\mathbf{hidden}} = \mathbf{W}^{\mathbf{T}}_{\mathbf{hidden\_output}} . \mathbf{e}_{\mathbf{output}}$$

Errors with Matrix Multiplication



$$\mathbf{e_{output}} = \begin{pmatrix} e_1 \\ e_2 \end{pmatrix}$$

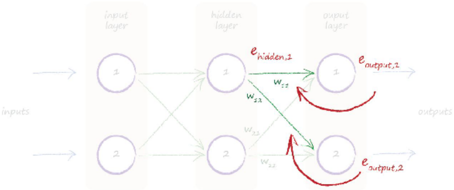
$$\mathbf{e_{hidden}} = \begin{pmatrix} (e_1 * w_{1,1}) + (e2 * w_{1,2}) \\ (e_1 * w_{2,1}) + (e2 * w_{2,2}) \end{pmatrix}$$

$$\mathbf{e_{hidden}} = \begin{pmatrix} w_{1,1} & w_{1,2} \\ w_{2,1} & w_{2,2} \end{pmatrix} . \begin{pmatrix} e_1 \\ e_2 \end{pmatrix}$$

$$\mathbf{W_{hidden\_output}} = \begin{pmatrix} w_{1,1} & w_{2,1} \\ w_{1,2} & w_{2,2} \end{pmatrix}$$

$$\mathbf{e_{hidden}} = \mathbf{W^T_{hidden\_output}} . \mathbf{e_{output}}$$

Errors with Matrix Multiplication



$$\mathbf{e_{output}} = \begin{pmatrix} e_1 \\ e_2 \end{pmatrix}$$

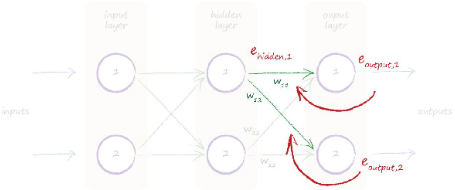
$$\mathbf{e_{hidden}} = \begin{pmatrix} (e_1 * w_{1,1}) + (e2 * w_{1,2}) \\ (e_1 * w_{2,1}) + (e2 * w_{2,2}) \end{pmatrix}$$

$$\mathbf{e_{hidden}} = \begin{pmatrix} w_{1,1} & w_{1,2} \\ w_{2,1} & w_{2,2} \end{pmatrix} . \begin{pmatrix} e_1 \\ e_2 \end{pmatrix}$$

$$\mathbf{W_{hidden\_output}} = \begin{pmatrix} w_{1,1} & w_{2,1} \\ w_{1,2} & w_{2,2} \end{pmatrix}$$

$$\mathbf{e_{hidden}} = \mathbf{W^T_{hidden\_output}} . \mathbf{e_{output}}$$

Errors with Matrix Multiplication



$$\mathbf{e_{output}} = \begin{pmatrix} e_1 \\ e_2 \end{pmatrix}$$

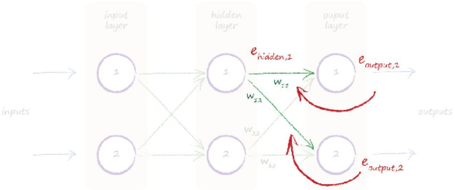
$$\mathbf{e_{hidden}} = \begin{pmatrix} (e_1 * w_{1,1}) + (e2 * w_{1,2}) \\ (e_1 * w_{2,1}) + (e2 * w_{2,2}) \end{pmatrix}$$

$$\mathbf{e_{hidden}} = \begin{pmatrix} w_{1,1} & w_{1,2} \\ w_{2,1} & w_{2,2} \end{pmatrix} . \begin{pmatrix} e_1 \\ e_2 \end{pmatrix}$$

$$\mathbf{W_{hidden\_output}} = \begin{pmatrix} w_{1,1} & w_{2,1} \\ w_{1,2} & w_{2,2} \end{pmatrix}$$

$$\mathbf{e_{hidden}} = \mathbf{W^T_{hidden\_output}} . \mathbf{e_{output}}$$

Errors with Matrix Multiplication



$$\mathbf{e_{output}} = \begin{pmatrix} e_1 \\ e_2 \end{pmatrix}$$

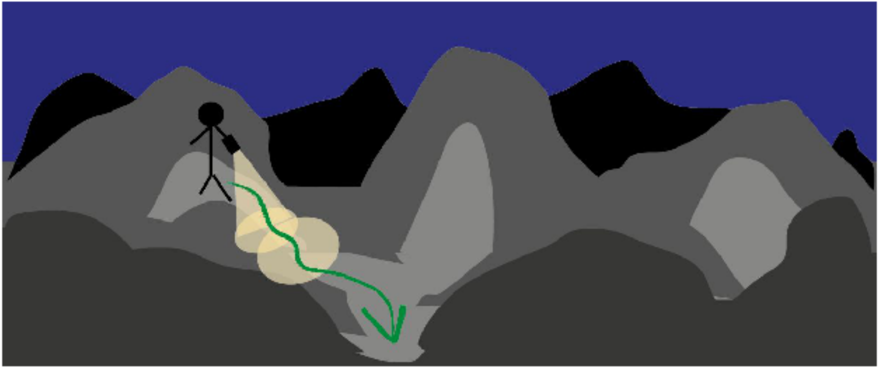
$$\mathbf{e_{hidden}} = \begin{pmatrix} (e_1 * w_{1,1}) + (e2 * w_{1,2}) \\ (e_1 * w_{2,1}) + (e2 * w_{2,2}) \end{pmatrix}$$

$$\mathbf{e_{hidden}} = \begin{pmatrix} w_{1,1} & w_{1,2} \\ w_{2,1} & w_{2,2} \end{pmatrix} . \begin{pmatrix} e_1 \\ e_2 \end{pmatrix}$$

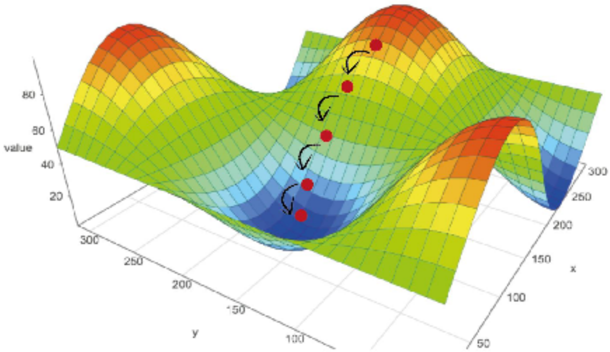
$$\mathbf{W_{hidden\_output}} = \begin{pmatrix} w_{1,1} & w_{2,1} \\ w_{1,2} & w_{2,2} \end{pmatrix}$$

$$\mathbf{e_{hidden}} = \mathbf{W^{T}_{hidden\_output}} . \mathbf{e_{output}}$$

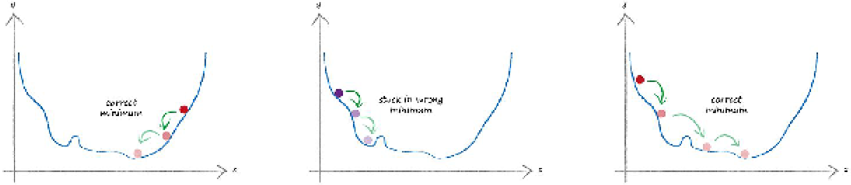
Update Weights: Gradient Descent



Update Weights: Gradient Descent



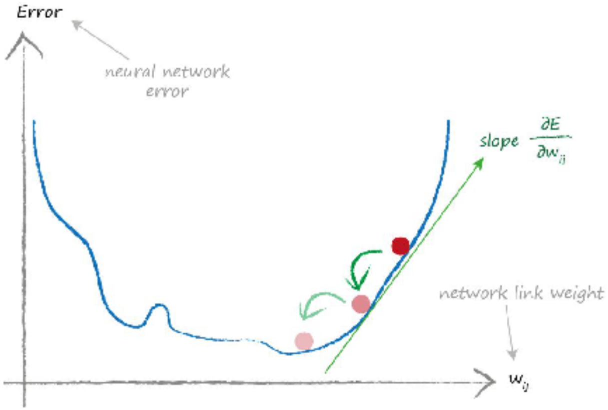
Update Weights: Gradient Descent



Update Weights: Error Function

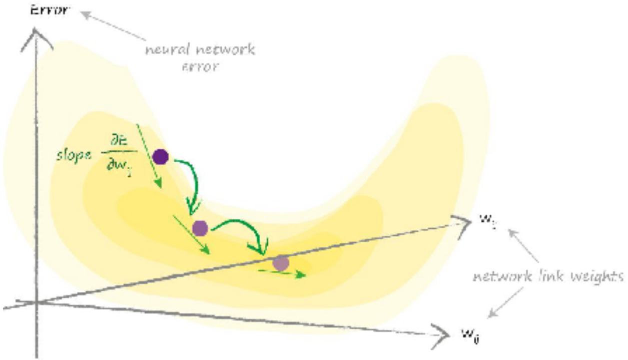
Network Output	Target Output	Error (target - actual)	Error |target - actual|	Error $(\text{target - actual})^2$
0.4	0.5	0.1	0.1	0.01
0.8	0.7	-0.1	0.1	0.01
1.0	1.0	0	0	0
Sum		0	0.2	0.02

Update Weights



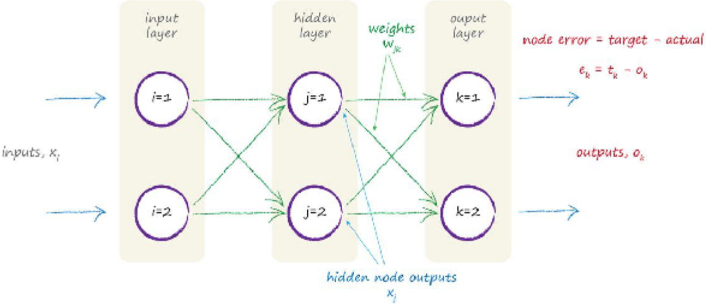
- One link weight.
- Two link weights.

Update Weights



- One link weight.
- Two link weights.

Update Weights



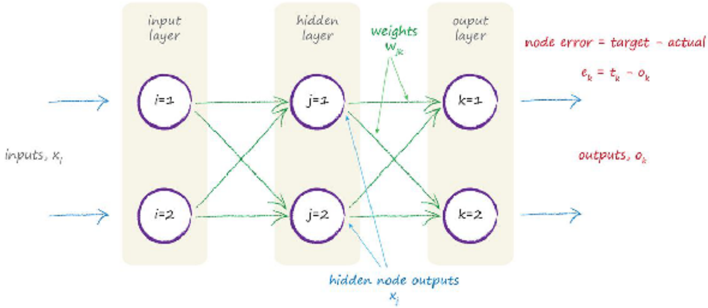
$$\frac{\delta E}{\delta w_{jk}}$$

$$\frac{\delta E}{\delta w_{jk}} = \frac{\delta}{\delta w_{jk}} \sum_n (t_n - o_n)^2$$

$$\frac{\delta E}{\delta w_{jk}} = \frac{\delta}{\delta w_{jk}} (t_k - o_k)^2$$

$$\frac{\delta E}{\delta w_{jk}} = -2(t_k - o_k).\frac{\delta o_k}{\delta w_{jk}}$$

Update Weights



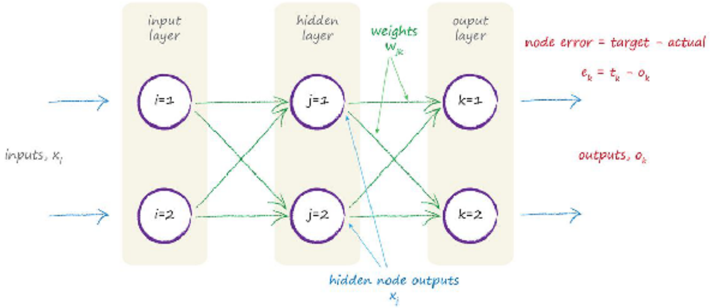
$$\frac{\delta E}{\delta w_{jk}}$$

$$\frac{\delta E}{\delta w_{jk}} = \frac{\delta}{\delta w_{jk}} \sum_n (t_n - o_n)^2$$

$$\frac{\delta E}{\delta w_{jk}} = \frac{\delta}{\delta w_{jk}} (t_k - o_k)^2$$

$$\frac{\delta E}{\delta w_{jk}} = -2(t_k - o_k).\frac{\delta o_k}{\delta w_{jk}}$$

Update Weights



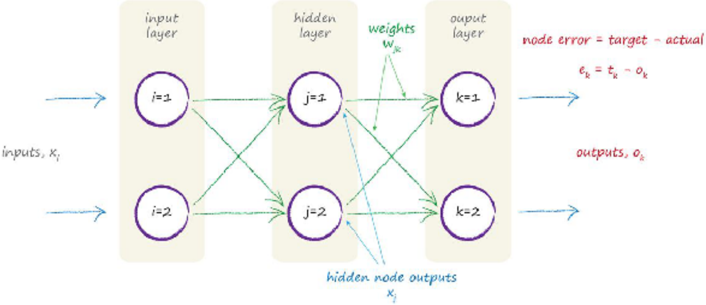
$$\frac{\delta E}{\delta w_{jk}}$$

$$\frac{\delta E}{\delta w_{jk}} = \frac{\delta}{\delta w_{jk}} \sum_n (t_n - o_n)^2$$

$$\frac{\delta E}{\delta w_{jk}} = \frac{\delta}{\delta w_{jk}} (t_k - o_k)^2$$

$$\frac{\delta E}{\delta w_{jk}} = -2(t_k - o_k) . \frac{\delta o_k}{\delta w_{jk}}$$

Update Weights



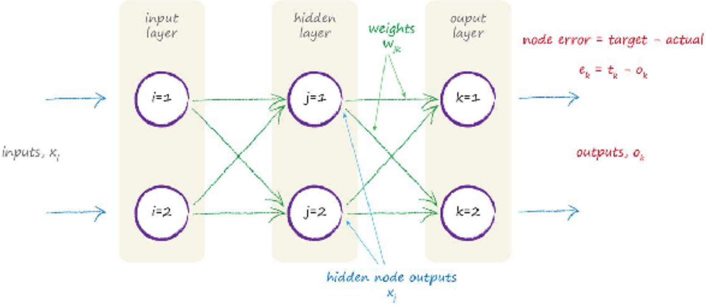
$$\frac{\delta E}{\delta w_{jk}}$$

$$\frac{\delta E}{\delta w_{jk}} = \frac{\delta}{\delta w_{jk}} \sum_n (t_n - o_n)^2$$

$$\frac{\delta E}{\delta w_{jk}} = \frac{\delta}{\delta w_{jk}} (t_k - o_k)^2$$

$$\frac{\delta E}{\delta w_{jk}} = -2(t_k - o_k) . \frac{\delta o_k}{\delta w_{jk}}$$

Update Weights



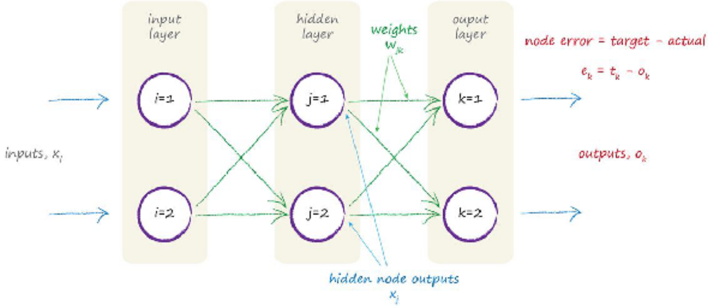
$$\frac{\delta E}{\delta w_{jk}}$$

$$\frac{\delta E}{\delta w_{jk}} = \frac{\delta}{\delta w_{jk}} \sum_n (t_n - o_n)^2$$

$$\frac{\delta E}{\delta w_{jk}} = \frac{\delta}{\delta w_{jk}} (t_k - o_k)^2$$

$$\frac{\delta E}{\delta w_{jk}} = -2(t_k - o_k).\frac{\delta o_k}{\delta w_{jk}}$$

Update Weights



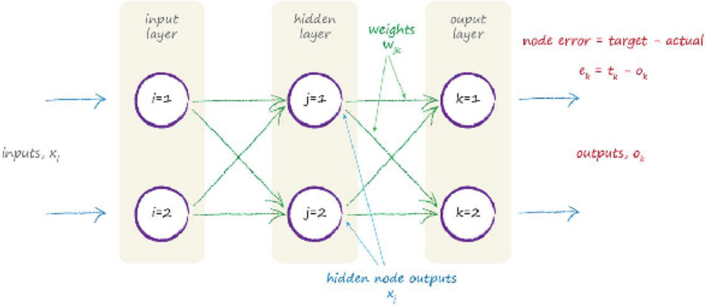
$$\frac{\delta E}{\delta w_{jk}} = -2(t_k - o_k).\frac{\delta o_k}{\delta w_{jk}}$$

$$\frac{\delta E}{\delta w_{jk}} = -2(t_k - o_k) \,.\, \frac{\delta}{\delta w_{jk}} \; sigmoid\left(\sum_j w_{jk} \,.\, o_j \right)$$

$$\frac{\delta E}{\delta w_{jk}} = -2(t_k - o_k).sigmoid(\sum_j w_{jk}.o_j)(1 - sigmoid(\sum_j w_{jk}.o_j)).\frac{\delta}{\delta w_{jk}}(\sum_j w_{jk}.o_j)$$

$$\frac{\delta E}{\delta w_{jk}} = -2(t_k - o_k) \,.\, sigmoid(\sum_j w_{jk} \,.\, o_j)\,(1 - sigmoid(\sum_j w_{jk} \,.\, o_j)) \,.\, o_j$$

Update Weights



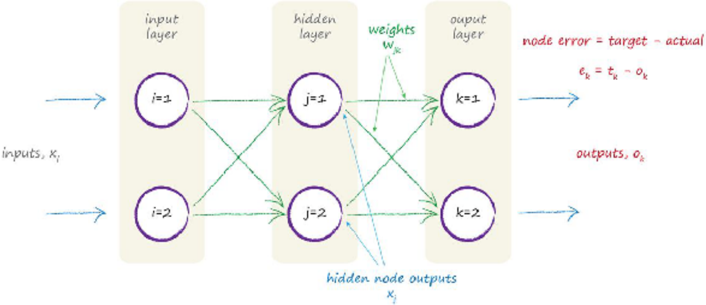
$$\frac{\delta E}{\delta w_{jk}} = -2(t_k - o_k).\frac{\delta o_k}{\delta w_{jk}}$$

$$\frac{\delta E}{\delta w_{jk}} = -2(t_k - o_k) \ . \ \frac{\delta}{\delta w_{jk}} \ sigmoid \left(\sum_j \ w_{jk} \ . \ o_j \right)$$

$$\frac{\delta E}{\delta w_{jk}} = -2(t_k - o_k).sigmoid(\sum_j w_{jk}.o_j)(1 - sigmoid(\sum_j w_{jk}.o_j)).\frac{\delta}{\delta w_{jk}}(\sum_j w_{jk}.o_j)$$

$$\frac{\delta E}{\delta w_{jk}} = -2(t_k - o_k) \ . \ sigmoid \left(\sum_j \ w_{jk} \ . \ o_j \right) \left(1 - \ sigmoid \left(\sum_j \ w_{jk} \ . \ o_j \right) \right) . \ o_j$$

Update Weights



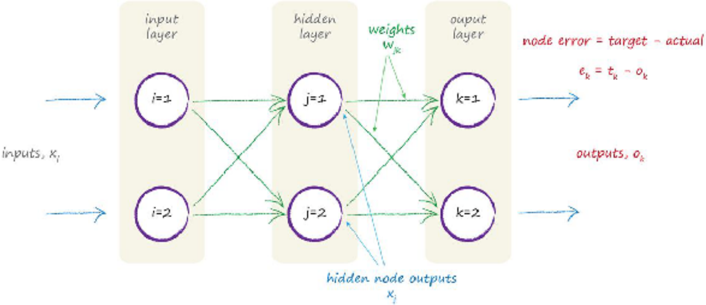
$$\frac{\delta E}{\delta w_{jk}} = -2(t_k - o_k).\frac{\delta o_k}{\delta w_{jk}}$$

$$\frac{\delta E}{\delta w_{jk}} = -2(t_k - o_k) \ . \ \frac{\delta}{\delta w_{jk}} \ sigmoid \ (\ \sum_j \ w_{jk} \ . \ o_j \ )$$

$$\frac{\delta E}{\delta w_{jk}} = -2(t_k - o_k).sigmoid(\sum_j w_{jk}.o_j)(1 - sigmoid(\sum_j w_{jk}.o_j)).\frac{\delta}{\delta w_{jk}}(\sum_j w_{jk}.o_j)$$

$$\frac{\delta E}{\delta w_{jk}} = -2(t_k - o_k) \ . \ sigmoid \ (\ \sum_j \ w_{jk} \ . \ o_j \ ) \ (\ 1 - \ sigmoid \ (\ \sum_j \ w_{jk} \ . \ o_j \ )) \ . \ o_j$$

Update Weights



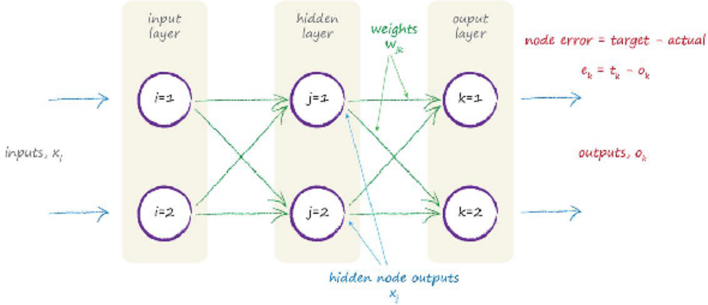
$$\frac{\delta E}{\delta w_{jk}} = -2(t_k - o_k).\frac{\delta o_k}{\delta w_{jk}}$$

$$\frac{\delta E}{\delta w_{jk}} = -2(t_k - o_k) \; . \; \frac{\delta}{\delta w_{jk}} \; sigmoid\left(\textstyle\sum_j \; w_{jk} \, . \, o_j \right)$$

$$\frac{\delta E}{\delta w_{jk}} = -2(t_k - o_k).sigmoid(\textstyle\sum_j w_{jk}.o_j)(1 - sigmoid(\sum_j w_{jk}.o_j)).\frac{\delta}{\delta w_{jk}}(\sum_j w_{jk}.o_j)$$

$$\frac{\delta E}{\delta w_{jk}} = -2(t_k - o_k)\, . \, sigmoid(\textstyle\sum_j \; w_{jk} \, . \, o_j)\,(1 \; - \; sigmoid(\sum_j \; w_{jk} \, . \, o_j))\, . \, o_j$$

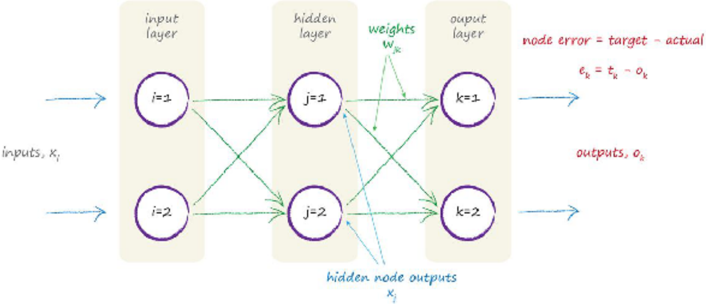
Update Weights



$$\frac{\delta E}{\delta w_{jk}} = -(t_k - o_k)\,.\,sigmoid(\textstyle\sum_j w_{jk}\,.\,o_j)\,(1 - sigmoid(\textstyle\sum_j w_{jk}\,.\,o_j))\,.\,o_j$$

$$\frac{\delta E}{\delta w_{ij}} = -(e_j)\,.\,sigmoid(\textstyle\sum_j w_{ij}\,.\,o_i)\,(1 - sigmoid(\textstyle\sum_j w_{ij}\,.\,o_i))\,.\,o_i$$

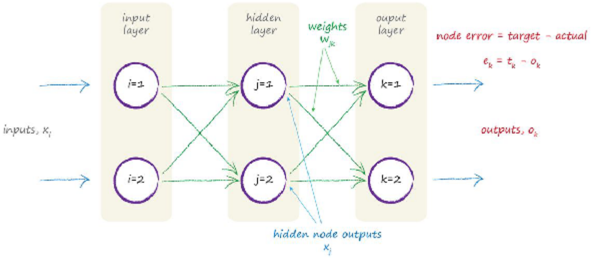
Update Weights



$$\frac{\delta E}{\delta w_{jk}} = -(t_k - o_k) \, . \, sigmoid\left(\sum_j w_{jk} \, . \, o_j\right)\left(1 - sigmoid\left(\sum_j w_{jk} \, . \, o_j\right)\right) . \, o_j$$

$$\frac{\delta E}{\delta w_{ij}} = -(e_j) \, . \, sigmoid\left(\sum_j w_{ij} \, . \, o_i\right)\left(1 - sigmoid\left(\sum_j w_{ij} \, . \, o_i\right)\right) . \, o_i$$

Update Weights: Matrix Multiplication

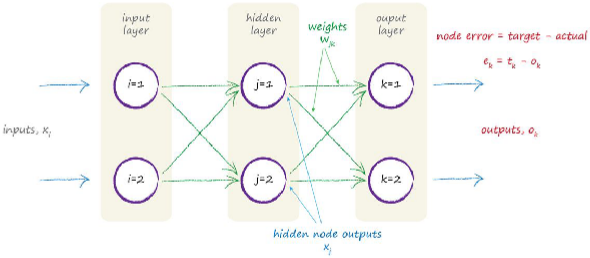


$$\text{new } w_{jk} = \text{old } w_{jk} - \alpha \cdot \frac{\delta E}{\delta w_{jk}}$$

$$\Delta w_{jk} = \alpha * E_k * sigmoid(i_k)(1 - sigmoid(i_k)) \cdot o_j^T$$

$$\begin{pmatrix} \Delta w_{1,1} & \Delta w_{2,1} & \Delta w_{3,1} & \dots \\ \Delta w_{1,2} & \Delta w_{2,2} & \Delta w_{3,2} & \dots \\ \Delta w_{1,3} & \Delta w_{2,3} & \Delta w_{j,k} & \dots \\ \dots & \dots & \dots & \dots \end{pmatrix} = \begin{pmatrix} E_1 * S_1(1 - S_1) \\ E_2 * S_2(1 - S_2) \\ E_k * S_k(1 - S_k) \\ \dots \end{pmatrix} \cdot \begin{pmatrix} o_1 & o_2 & o_j & \dots \end{pmatrix}$$

Update Weights: Matrix Multiplication

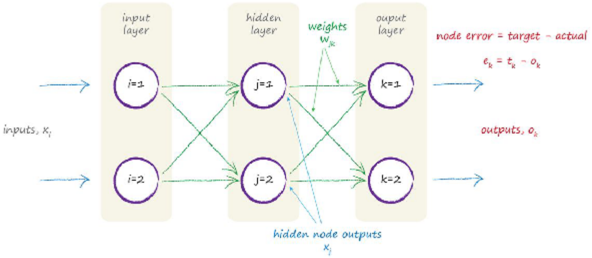


$$\text{new } w_{jk} = \text{old } w_{jk} - \alpha \,.\, \frac{\delta E}{\delta w_{jk}}$$

$$\Delta w_{jk} \;=\; \alpha * E_k * sigmoid(\, i_k \,)\,(1 \;-\; sigmoid(\, i_k\,))\,.\, o_j^T$$

$$\begin{pmatrix} \Delta w_{1,1} & \Delta w_{2,1} & \Delta w_{3,1} & \dots \\ \Delta w_{1,2} & \Delta w_{2,2} & \Delta w_{3,2} & \dots \\ \Delta w_{1,3} & \Delta w_{2,3} & \Delta w_{j,k} & \dots \\ \dots & \dots & \dots & \dots \end{pmatrix} = \begin{pmatrix} E_1 * S_1(1 - S_1) \\ E_2 * S_2(1 - S_2) \\ E_k * S_k(1 - S_k) \\ \dots \end{pmatrix} . \begin{pmatrix} o_1 & o_2 & o_j & \dots \end{pmatrix}$$

Update Weights: Matrix Multiplication



$$\text{new } w_{jk} = \text{old } w_{jk} - \alpha \cdot \frac{\delta E}{\delta w_{jk}}$$

$$\Delta w_{jk} = \alpha * E_k * sigmoid(i_k)(1 - sigmoid(i_k)) \cdot o_j^T$$

$$\begin{pmatrix} \Delta w_{1,1} & \Delta w_{2,1} & \Delta w_{3,1} & \dots \\ \Delta w_{1,2} & \Delta w_{2,2} & \Delta w_{3,2} & \dots \\ \Delta w_{1,3} & \Delta w_{2,3} & \Delta w_{j,k} & \dots \\ \dots & \dots & \dots & \dots \end{pmatrix} = \begin{pmatrix} E_1 * S_1(1 - S_1) \\ E_2 * S_2(1 - S_2) \\ E_k * S_k(1 - S_k) \\ \dots \end{pmatrix} \cdot \begin{pmatrix} o_1 & o_2 & o_j & \dots \end{pmatrix}$$

THANK YOU