Neural networks (Ch. 18.7)
You try Bat on this: \{WB=0.5, LE=1, CH=1\}

If Output(Node 5) > 0, guess mammal
Neural network: feed-forward

Output is -2, so bats are not a mammal... Oops!

if $\text{Output(Node 5)} > 0$, guess mammal
Neural network: feed-forward

One commonly used function is the sigmoid:

\[ S(x) = \frac{1}{1 + e^{-x}} \]
Eventually we get: \( \text{out}_1 = 0.7513, \text{out}_2 = 0.7729 \)

Suppose wanted:

\[
\begin{align*}
\text{out}_1 &= 0.01, \\
\text{out}_2 &= 0.99
\end{align*}
\]
Back-propagation

We will define the error as: \( \frac{\sum_i{(\text{correct}_i - \text{output}_i)^2}}{2} \)
(you will see why shortly)

Suppose we want to find how much \( w_5 \) is to blame for our incorrectness

We then need to find: \( \frac{\partial \text{Error}}{\partial w_5} \)

Apply the chain rule:

\[
\frac{\partial \text{Error}}{\partial \text{out}_1} \cdot \frac{\partial \text{S}(\text{In}(N_3))}{\partial \text{In}(N_3)} \cdot \frac{\partial \text{In}(N_3)}{\partial w_5}
\]
Back-propagation

\[
Error = \sum_i (\text{correct}_i - \text{output}_i)^2
\]

\[
\frac{\partial Error}{\partial \text{out}_1} = -(\text{correct}_1 - \text{out}_1) = -(0.01 - 0.7513) = 0.7413
\]

As \( S'(x) = S(x) \cdot (1 - S(x)) \)

\[
\frac{\partial S(\text{In}(N_3))}{\partial \text{In}(N_3)} = S(\text{In}(N_3)) \cdot (1 - S(\text{In}(N_3))) = 0.7513 \cdot (1 - 0.7513) = 0.1868
\]

\[
\frac{\partial \text{In}(N_3)}{\partial \omega_5} = \frac{\partial \text{out}(N_1) + \omega_6 \cdot \text{out}(N_2) + b_2 \cdot 1}{\partial \omega_5}
\]

\[
= \text{Out}(N_1) = 0.5932
\]

Thus, \( \frac{\partial Error}{\partial \omega_5} = 0.7413 \cdot 0.1868 \cdot 0.5932 = 0.08217 \)
Back-propagation

In a picture we did this:

Now that we know $w_5$ is 0.08217 part responsible, we update the weight by:

$$w_5 \leftarrow w_5 - \alpha \times 0.08217 = 0.3589 \text{ (from 0.4)}$$

$\alpha$ is learning rate, set to 0.5
Back-propagation

For $w_1$ it would look like:

(book describes how to dynamic program this)
Back-propagation

Specifically for $w_1$ you would get:

$$\frac{\partial \text{Error}}{\partial S(In(N_1))} = \frac{\partial \text{Error}_1}{\partial S(In(N_1))} + \frac{\partial \text{Error}_2}{\partial S(In(N_1))}$$

$$\frac{\partial S(In(N_1))}{\partial In(N_1)} = S(In(N_1)) \cdot (1 - S(In(N_1)))$$

$$= 0.5933 \cdot (1 - 0.5933) = 0.2413$$

$$\frac{\partial In(N_3)}{\partial w_5} = \frac{\partial w_1 \cdot In_1 + w_2 \cdot In_2 + b_1 \cdot 1}{\partial w_5}$$

$$= In_1 = 0.05$$

Next we have to break down the top equation...
Back-propagation

\[
\frac{\partial \text{Error}}{\partial S(\text{In}(N_1))} = \frac{\partial \text{Error}_1}{\partial S(\text{In}(N_1))} + \frac{\partial \text{Error}_2}{\partial S(\text{In}(N_1))}
\]

\[
\frac{\partial \text{Error}_1}{\partial S(\text{In}(N_1))} = \frac{\partial \text{Error}_1}{\partial S(\text{In}(N_3))} \cdot \frac{\partial S(\text{In}(N_3))}{\partial \text{In}(N_3)} \cdot \frac{\partial \text{In}(N_3)}{\partial S(\text{In}(N_1))}
\]

From before...

\[
\frac{\partial \text{Error}_1}{\partial S(\text{In}(N_3))} \cdot \frac{\partial S(\text{In}(N_3))}{\partial \text{In}(N_3)} = 0.7414 \cdot 0.1868 = 0.1385
\]

\[
\frac{\partial \text{In}(N_3)}{\partial S(\text{In}(N_1))} = \frac{\partial w_5 \cdot S(\text{In}(N_1)) + w_6 \cdot S(\text{In}(N_2)) + b_1 \cdot 1}{\partial S(\text{In}(N_1))}
\]

\[
= w_5 = 0.4
\]

Thus,

\[
\frac{\partial \text{Error}_1}{\partial S(\text{In}(N_1))} = 0.1385 \cdot 0.4 = 0.055540
\]
Back-propagation

Similarly for Error\textsubscript{2} we get:

\[
\frac{\partial \text{Error}}{\partial S(In(N_1))} = \frac{\partial \text{Error}_1}{\partial S(In(N_1))} + \frac{\partial \text{Error}_2}{\partial S(In(N_1))} = 0.05540 + (-0.01905) = 0.03635
\]

Thus, \[
\frac{\partial \text{Error}}{\partial w_1} = 0.03635 \cdot 0.2413 \cdot 0.05 = 0.0004386
\]

Update \( w_1 \leftarrow w_1 - \alpha \frac{\partial \text{Error}}{\partial w_1} = 0.15 - 0.5 \cdot 0.0004386 = 0.1498 \)

Whew, done!
You try Bat on this: \(\{\text{WB}=0.5, \text{LE}=1, \text{CH}=1\}\)
What will the top line of 2 weights become?

The output is -2, you can assume the correct output is 1.
Back-propagation

In the example I did, the earlier weights only shifted a very small amount: 0.15 to 0.1498

In your problem, the weights shifted a large amount: WB's weight went from 2 to 12

This is a common problem with neural networks: back propagation does not update nodes far away from the output well
NN examples

Despite this learning shortcoming, NN are useful in a wide range of applications:
  Reading handwriting
  Playing games
  Face detection
  Economic predictions

Neural networks can also be very powerful when combined with other techniques (genetic algorithms, search techniques, ...)

NN examples

Examples:

https://www.youtube.com/watch?v=umRdt3zGgpU
https://www.youtube.com/watch?v=qv6UVOQ0F44
https://www.youtube.com/watch?v=xclBoPuNliw
https://www.youtube.com/watch?v=0Str0Rdkxxo
https://www.youtube.com/watch?v=l2_CPB0uBkc
https://www.youtube.com/watch?v=0VT1BBLydE
AlphaGo has been in the news recently, and is also based on neural networks.

AlphaGo uses Monte-Carlo tree search guided by the neural network to prune useless parts.

Often limiting Monte-Carlo in a static way reduces the effectiveness, much like mid-state evaluations can limit algorithm effectiveness.