

## Machine Learning Simulation Worksheet

You will be using the Netlogo platform for this activity. Take 1-2 minutes to familiarize yourself with the platform. The picture below is the first simulation platform you will use which is a single-layered Perceptron. **You will want to reference this throughout the activity.**

The screenshot shows the Netlogo interface for the 'Artificial Neural Net - Perceptron' model. The interface includes a control panel on the left with buttons for 'setup', 'train', and 'train once', and sliders for 'learning-rate' (set to 0.5000) and 'examples-per-epoch' (set to 500). A graph titled 'Error vs. Epochs' shows the error decreasing over 10 epochs. The central workspace displays a neural network diagram with two input nodes (one white, one black), a bias node (white), and a summation node (Σ). Weights are shown as lines between nodes: 0.0357 between the white input and the summation node, -0.0160 between the black input and the summation node, and -0.0298 between the bias node and the summation node. The output node shows a value of -1. On the right, there is a 'Testing Section' with input fields for 'input-1' (set to 1) and 'input-2' (set to -1), and a 'test' button. Below the testing section is a 'Rule Learned' graph showing a boundary line. A 'show-weights?' switch is set to 'On'. Annotations with blue arrows point to various parts of the interface, explaining their functions.

Input nodes can be 1 or -1 and are on the left. White represents 1 while black represents -1.

Bias node which is constantly set to 1.

Setup and reset the simulation

Continually train until clicking a 2<sup>nd</sup> time.

Train once

This refers to the amount of change given to the weights.

Error vs. Epochs shows how well the model is performing during training.

Weights are shown in the field and are turned on using this switch. The line width corresponds to the weights as well.

Select the function

The output from the model during training.

Testing Section: Inputs can be changed and tested. A dialog box will show the result of the test.

Rule Learned: A boundary line graph will appear here as the model learns a rule.

Work in pairs and have one partner change the parameters on the simulation while the other partner uses this document to help navigate.

1. Open Netlogo from your PC.
2. Go to File - Models Library and choose "Artificial Neural Net - Perceptron" from the "Computer Science" folder.
3. Click Open at the bottom.

Name:

Date:

Class:

### OR Function

1. Click on the drop-down under "Target function" and choose "or"
2. Click on the Setup button
3. Click "On" on the "show weights?" button.
4. Adjust:
  - a. The learning rate to a value somewhere between 0 and 0.1
  - b. The examples per epoch to 1. An epoch is one set of training data. By making these adjustments, you are giving the model one set of inputs and a very low learning rate.
5. Notice the weights and thickness of the lines.

Write down the weights \_\_\_\_\_

On the right-hand side, you will see "3. Test" this area allows you to test the model after it has been trained. Even though you have not trained the model, click on test 5-10 times. You can change the inputs around a few times as well. Keep track of your results here.

# of tests: \_\_\_\_\_ # of correct answers: \_\_\_\_\_

6. Click the "train once" button.

Write down the weights \_\_\_\_\_

Did these change? \_\_\_\_\_

The rule learned - this is the Perceptron's attempt at finding the boundary line and to divide true outputs and false outputs.

You may not see anything in the "Error vs. Epochs" graph just yet but keep an eye on this. This is an indicator of how well the model is learning.

Click on train once again.

Write down the weights \_\_\_\_\_

Did these change? \_\_\_\_\_

Did the graph of the rule learned change? \_\_\_\_\_

Click on train once again.

Write down the weights \_\_\_\_\_

Did these change? \_\_\_\_\_

Did the graph of the rule learned change? \_\_\_\_\_

Name:

Date:

Class:

Click on the “train” button. This will continually train the model. Let this run for about 10-15 seconds and keep watching the “Error vs. Epochs” graph.

What do you notice about the “rule learned?” How did this change? Does this make sense with what you know about this function?

What do you notice about the “Error vs. Epochs” graph?

Did the model learn the correct rule? Hint: Check the change in the boundary line in the “rule learned” window and look at the “Error vs. Epoch” graph.

- a. Describe how you know the model learned the correct rule.

Just like item #6: On the right-hand side, you will see “3. Test” this area allows you to test the model after it has been trained. Click on this the same number of times you did in item 6 above and keep track of the results here.

# of tests: \_\_\_\_\_ # of correct answers: \_\_\_\_\_

How did your results here compare to the results in item #6?  
What does this tell you about the process of machine learning?

Name:

Date:

Class:

### AND Function

Change the “target function” to “And” and repeat the same process as above.

1. Click on the Setup button
2. Click “On” on the “show weights?” button.
3. Adjust:
  - a. The learning rate to a value somewhere between 0 and 0.1
  - b. The examples per epoch to 1. An epoch is one set of training data. By making these adjustments, you are giving the model one set of inputs and a very low learning rate.
4. Notice the weights and thickness of the lines.

Write down the weights \_\_\_\_\_

On the right-hand side, you will see “3. Test” this area allows you to test the model after it has been trained. Even though you have not trained the model, click on test 5-10 times. You can change the inputs around a few times as well. Keep track of your results here.

# of tests: \_\_\_\_\_ # of correct answers: \_\_\_\_\_

5. Click the “train once” button.

Write down the weights \_\_\_\_\_

Did these change? \_\_\_\_\_

The rule learned - this is the Perceptron’s attempt at finding the boundary line and to divide true outputs and false outputs.

You may not see anything in the “Error vs. Epochs” graph just yet but keep an eye on this. This is an indicator of how well the model is learning.

Click on train once again.

Write down the weights \_\_\_\_\_

Did these change? \_\_\_\_\_

Did the graph of the rule learned change? \_\_\_\_\_

Click on train once again.

Write down the weights \_\_\_\_\_

Did these change? \_\_\_\_\_

Did the graph of the rule learned change? \_\_\_\_\_

Name:

Date:

Class:

Click on the “train” button. This will continually train the model. Let this run for about 10-15 seconds and keep watching the “Error vs. Epochs” graph.

What do you notice about the “rule learned?” How did this change? Does this make sense with what you know about this function?

What do you notice about the “Error vs. Epochs” graph?

Did the model learn the correct rule? Hint: Check the change in the boundary line in the “rule learned” window and look at the “Error vs. Epoch” graph.

- a. Describe how you know the model learned the correct rule.

Just like item #6: On the right-hand side, you will see “3. Test” this area allows you to test the model after it has been trained. Click on this the same number of times you did in item 6 above and keep track of the results here.

# of tests: \_\_\_\_\_ # of correct answers: \_\_\_\_\_

How did your results here compare to the results in item #6?

Name:

Date:

Class:

### XOR Function

Now Change the “target function” to “XOR” and repeat the same process as above. You should notice something very interesting about XOR.

1. Click on the Setup button
2. Click “On” on the “show weights?” button.
3. Adjust:
  - a. The learning rate to a value somewhere between 0 and 0.1
  - b. The examples per epoch to 1. An epoch is one set of training data. By making these adjustments, you are giving the model one set of inputs and a very low learning rate.
4. Notice the weights and thickness of the lines.

Write down the weights \_\_\_\_\_

On the right-hand side, you will see “3. Test” this area allows you to test the model after it has been trained. Even though you have not trained the model, click on test 5-10 times. You can change the inputs around a few times as well. Keep track of your results here.

# of tests: \_\_\_\_\_ # of correct answers: \_\_\_\_\_

5. Click the “train once” button.

Write down the weights \_\_\_\_\_

Did these change? \_\_\_\_\_

The rule learned - this is the Perceptron’s attempt at finding the boundary line and to divide true outputs and false outputs.

You may not see anything in the “Error vs. Epochs” graph just yet but keep an eye on this. This is an indicator of how well the model is learning.

Click on train once again.

Write down the weights \_\_\_\_\_

Did these change? \_\_\_\_\_

Did the graph of the rule learned change? \_\_\_\_\_

Click on train once again.

Write down the weights \_\_\_\_\_

Did these change? \_\_\_\_\_

Did the graph of the rule learned change? \_\_\_\_\_

Name:

Date:

Class:

Click on the “train” button. This will continually train the model. Let this run for about 10-15 seconds and keep watching the “Error vs. Epochs” graph. What do you notice about the “rule learned?” How did this change? Does this make sense with what you know about this function?

What do you notice about the “Error vs. Epochs” graph?

Did the model learn the correct rule? Hint: Check the change in the boundary line in the “rule learned” window and look at the “Error vs. Epoch” graph.

- a. Describe how you know the model learned the correct rule.

Just like item #6: On the right-hand side, you will see “3. Test” this area allows you to test the model after it has been trained. Click on this the same number of times you did in item 6 above and keep track of the results here.

# of tests: \_\_\_\_\_ # of correct answers: \_\_\_\_\_

How did your results here compare to the results in item #6?

Change the learning rate to around 0.5 and the examples per epoch to 500 (default values) and click on “train.”

- a. What do you notice about the XOR function?  
Look at the “Rule Learned” and “Errors vs. Epochs” windows. What is happening?

Name:

Date:

Class:

## Multi-layer

The screenshot shows a neural network simulation interface. At the top, there are buttons for 'Edit', 'Delete', 'Add', and 'Button', along with a 'normal speed' slider and a 'View updates on ticks' checkbox. Below this, there are three main sections: '1. Setup neural net:', '2. Train neural net:', and '3. Test neural net:'. The '1. Setup neural net:' section has 'setup' and 'train' buttons. The '2. Train neural net:' section has 'train once' and 'train' buttons, and sliders for 'learning-rate' (0.5000) and 'examples-per-epoch' (500). The '3. Test neural net:' section has 'input-1' (1), 'input-2' (0), and 'test' buttons. A central diagram shows a neural network with three layers: an input layer with two nodes (one white, one black), a hidden layer with two nodes, and an output layer with one node. Weights are shown on the connections between nodes. A 'target-function' dropdown is set to 'xor', and a 'show-weights?' checkbox is checked. An 'output' box shows the value 0.89. An 'Error vs. Epochs' graph shows the error decreasing over 22.5 epochs. Annotations with blue arrows point to various parts of the interface, explaining the function of different elements.

Input nodes can be 1 or -1 and are on the left. White represents 1 while black represents -1.

Bias node which is constantly set to 1.

Setup and reset the simulation

Continually train until clicking a 2<sup>nd</sup> time.

Train once

This refers to the amount of change given to the weights.

Error vs. Epochs shows how well the model is performing during training.

Weights are shown in the field and are turned on using this switch. The line width corresponds to the weights as well.

Select the function

The output from the model during training. The output value represents a probability.

Testing Section: Inputs can be changed and tested. A dialog box will show the result of the test.

There is no rule learned graph for this model.

Notice the 2<sup>nd</sup> layer with 2 nodes. This is called the "hidden layer."

1. Click on File, Models Library and choose artificial neural net- multi-layer.
2. Choose "OR" under target function but do not make any adjustments to the learning rate.
3. Click "On" in the "Show weights" box.
4. Click on train and hit the train button once you see the "Error vs. Epochs" graph settle to no error. Notice how quickly this happens.
5. Click test several times and note any incorrect answers.
6. Click on setup to reset.
7. Click on test several times and note any incorrect answers. Change the inputs in the test section and try each combination a few times.
8. Choose "XOR" under the target function and repeat the above process.

What do you notice about the XOR function? In terms of the XOR function, what is different about the multi-layered Perceptron vs. the single-layered?

### Additional things to try:

Manipulate the **learning-rate** parameter. Can you speed up or slow down the training?

Switch back and forth between OR and XOR several times during a run. Why does it take less time for the network to return to 0 error the longer the network runs?



Name:

Date:

Class: