COMP3308/COMP3608 Introduction to Artificial Intelligence

Week 8 Tutorial exercises
Perceptrons. Multilayer Neural Networks 1.

***Exercise 1**. Perceptron learning (Homework)*

The homework for this week consists of 5 multiple-choice questions with two possible answers – True and False. To complete the homework, please go to Canvas -> Quizzes->w8-homework-submission. Remember to press the "Submit" button at the end. **Only 1 attempt is allowed.**

For each question below, select the correct answer:

1. Given are the following 2-dimensional examples from two classes (class I and class II).
   class I: p1 = [1 1], p2 = [-1 -1]
   class II: p3 = [2 2]

Can a perceptron learn to distinguish between them?

Yes No

2. (The same question as above but for different examples.) Given are the following 2-dimensional examples from two classes (class I and class II).
   class I: p1 = [1 1], p2 = [1 -1]
   class II: p3 = [1 0]

Yes No

Can a perceptron learn to distinguish between them?

3. If the data is linearly separable, the perceptron will optimize the decision boundary and is guaranteed to find the best possible boundary that separates the examples in a finite number of steps.

True False

4. A single perceptron can solve the XOR problem.

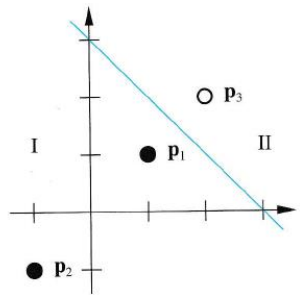
True False

5. The perceptron's learning rule reflects exactly the operation of the human neurons.

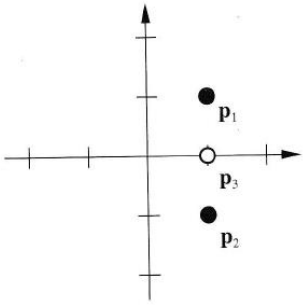
True False

Answers:

1. Yes, since the data is linearly separable. The diagram below shows a possible decision boundary that can be learned by the perceptron:



2. No, since the data is not linearly separable as the diagram shows:



3. False. The perceptron is guaranteed to find a linear boundary in a finite number of steps but it will not optimize this boundary and find the best possible boundary – it will simply stop when a separating boundary is found.

4 False. The XOR problem is linearly inseparable.

5. False. The perceptrons and their learning rule are simplifications of the biological neurons. They are only inspired by the operation of the biological neurons.

Exercise 2*. *Perceptron learning rule

Given is the following training set:

	input	output
ex. 1:	1 0 0	1
ex. 2:	0 1 1	0
ex. 3:	1 1 0	1
ex. 4:	1 1 1	0
ex. 5:	0 0 1	0
ex. 6:	1 0 1	1

a) Train by hand a perceptron **with a bias** on this training set. Assume that all initial weights (including the bias of the neuron) are 0. Show the set of weights (including the bias) at the end of each iteration. Apply the examples in the given order.

Stopping criterion: all examples are correctly classified or a maximum number of 2 epochs is reached. Reminder: The stopping criterion is checked at the end of each epoch; to check the first part, apply each training example, check if it is correctly classified and stop if all examples are correctly classified. Note that when doing this, there is no weight change just calculating the actual output and comparing it with the target output.

Use the step function and note that **step(0)=1.**

step(n) = 1, if n >= 0

= 0, otherwise.

b) Which stopping condition was satisfied? If it was the first one, how many epochs were needed?

Solution:

a) starting point: w=[0 0 0], b=0

iteration 1:

applying ex.1: p_1= [1 0 0]^T,t_1=1

a=step([0 0 0][1 0 0]+0)=step(0)=1

e=t_1-a=1-1=0

w^{new}=[0 0 0]+0[1 0 0] =[0 0 0]

b^{new}= 0+0=0

i.e. no change in w and b as e=0

iteration 2:

applying ex.2: p_2= [0 1 1]^T,t_2=0

a=step([0 0 0][0 1 1]+0)=step(0)=1

e=t_2-a=0-1=-1

w^{new}=[0 0 0]+(-1)[0 1 1] =[0 -1 -1]

b^{new}= 0+(-1)=-1

i.e. w an b have been updated

Similarly:

end of iteration 3: w^{new}=[1 0 –1], b^{new}=0

end of iteration 4: w^{new} =[0 -1, -2], b^{new} =-1

end of iteration 5: no change

end of iteration 6: w^{new} =[1 -1 –1], b^{new} =0

end of epoch 1

b) The first stopping condition. 1 epoch is needed.

Note: As discussed in class an epoch is one pass through the training set (i.e. each training example is passed once) involving weight adaptation. At the end of each epoch all examples are passed again to check if the stopping criterion is satisfied but this does not count for another epoch as there is no weight change.

Perceptron – Matlab exercise

Some of the exercises below are based on the Matlab's Neural Network toolbox. This toolbox allows to build and test neural network prototypes quickly. The goal of these exercises is to illustrate some important neural network concepts. You are not expected to know how to program in Matlab, we will use Matlab only for demonstration, and only this and next week.

You need to install both Matlab and the Deep Learning toolbox.

How to install Matlab on your computer (it is free for students) – 2 download options:

Option 1. Via the University's webpage

Step 1: Go to the University's Matlab page to download Matlab (VPN needed)

http://softserv.usyd.edu.au/data/MatLab/

Option 2. Via Matlab's webpage (no VPN needed)

Step 1: Go to the Matlab's webpage:

https://protect-au.mimecast.com/s/A05wCk81N9t48DnmT255Br?domain=au.mathworks.com

Step 2: Create an account using your student email.

Step 3: Download the latest version of MATLAB and use your username and password to activate it during the installation process

Note on the neural networks implemented in Weka: Weka includes multilayer perceptron and other neural networks (e.g. RBF networks - not covered in this course). It doesn't include single perceptron but includes "Voted perceptrons", an algorithm developed by Freud and Schapire, which combines the decisions of several perceptrons by voting, i.e. it is an *ensemble* of classifiers. We will study ensembles of classifiers later in the course.

Exercise 3. Banana/orange classification using perceptron

Start Matlab. Type `nnd3pc` to run the demo. This is the banana/orange example from the lecture. A perceptron *has been already trained* to classify the fruits into 2 classes, the demo represents the classifications of new fruits. Follow the instructions of the demo and observe the calculations at each step during the classification. Make sure that you switch on the sound for this exercise!

Exercise 4. Perceptron decision boundaries

In Matlab, type `nnd4db` to run the demo.

Move the perceptron decision boundary by dragging its handles and try to separate the examples (white and black circles) into 2 classes, i.e. so that none of them has red edges. You can insert more examples by dragging the white and black circles from the left part. The weights and bias will take the values associated with the new boundary. What is the relation between the decision boundary and the weight vector?

Answer: The weight vector w is always orthogonal to the decision boundary

Exercise 5. Perceptron learning

In Matlab, type rule `nnd4pr` to run the demo.

1) Press Train to apply the perceptron rule 5 times. Each time an example is applied, the weights are adapted (their values are shown) and the new decision boundary is plotted.

2) Learning linearly separable and linearly non-separable patterns

a) Add by dragging some new white and black points to define a linearly separable problem. Train the perceptron. Will the perceptron be able to separate the patterns into 2 classes?

Answer: Yes, the perceptron can be trained to separate them.

b) Now create a linearly non-separable problem. The same question as above.

Answer: No matter how long the training is, the perceptron can not separate linearly non-separable data points.

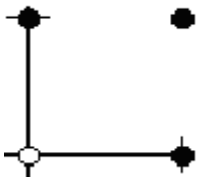
3) Try the learning with and without bias. What is the influence of the bias? What happens if there is no bias and if b=0?

Answer: If there is no bias or b=0 (at some learning stage), the decision boundary passes via the coordinate beginning.

Exercise 6. Perceptron – the importance of the bias weight

A follow-up from the previous exercise, here we again focus on the meaning and importance of the bias weight.

Suppose that you need to solve the classification problem shown below using a perceptron. The target of 0 is represented with a white circle and target of 1 is represented with a black circle.



a) Draw a possible decision boundary.

b) Can this problem be solved with a perceptron with step transfer function without a bias? Why?

Reminder: $step(n) = \begin{cases} 1 & if \quad n \geq 0 \\ 0 & if \quad n < 0 \end{cases}$

Answer:

a) Any line separating the light and blue circles; an infinite number of such lines exist.

b) No, as the white circle will be misclassified.

Since there is no bias, the decision boundary passes through the origin where one of the points lays (the light circle). The light circle is [0,0]; the perceptron's output for it will be $step([0\ \ 0][w1\ w2]^T) = step(0) = 1$, however its target output is 0, i.e. it will be misclassified.

Exercise 7. Perceptron and the XOR problem (Advanced students only)

Show analytically that the perceptron cannot solve the XOR problem.

Hint: Use a system of inequalities.

Solution:

XOR truth table:

input1	input2	output
0	0	0
0	1	1
1	0	1
1	1	0

The perceptron will have 2 inputs and 1 output; the input weights are w1 and w2 and the bias is b. The system of linear inequalities is:

$w1*0 + w2*0 + b < 0 \quad \Rightarrow b < 0$ (1)

$w1*0 + w2*1 + b \geq 0 \quad \Rightarrow w2 \geq -b$

$w1*1 + w2*0 + b \geq 0 \quad \Rightarrow w1 \geq -b$

$\Rightarrow w1 + w2 \geq -2b$ (2)

$w1*1 + w2*1 + b < 0 \quad \Rightarrow w1 + w2 < -b$ (3)

There is a contradiction between (2) and (3), given (1). Hence, a single perceptron cannot solve the XOR problem.

Multilayer perceptron trained with the backpropagation algorithm – Matlab exercises

Exercise 8. ***Backpropagation calculation for 2-layer network***

In Matlab, type `nnd11bc` to run the demo.
Press the first button and observe the basic stages of the backpropagation algorithm – applying an example, output calculation, error calculation, error propagation and weights update.

Exercise 9: ***Error space and convergence; local and global minima***

In Matlab, type `nnd12sd1` or select "Steepest Descent Backpropagation" from Demos -> Toolboxes -> Neural Networks (upper left window)

Use the radio buttons to select two parameters. The corresponding error surface will be plotted. Select a starting point and observe the convergence process. Is the backpropagation algorithm guaranteed to find the global minimum?

Answer: No, it may reach a local minimum.