Solution to Tutorial Week 7

1. Note: ‘<filename>’ refers to the name of the data file you are working with.

(a) Extract the first 10 lines

head -10 <filename>

(b) Extract all lines from line 10 onwards

tail -n +10 <filename>

(c) Extract the sign column

cut -d, -f4 <filename>

(d) Extract parking events that occurred in Chinatown

grep -F “Chinatown” <filename>

[Note: fgrep “Chinatown” is equivalent]

(e) Extract parking events that occurred in April 2012

grep -F '/04/2012' <filename>

(f) Extract parking events that occurred in Chinatown in April 2012

grep -F “Chinatown” <filename> | grep -F '/04/2012'

(g) Extract parking events that occurred in a 2-hour zone

grep -F ',2P' <filename>

(2P after a comma)

(h) Extract events for which there was a violation (ie a 1 in the 'violation?' column)

grep -E -e',1$' <filename>

(Needs grep -E to catch 1 at the end of a line)

[Note: egrep -e',1$' <filename> is equivalent]

2. For any node, the output is given by,

$$O = transfer\_function\ (\sum_{i=0}^{n} w_i X_i)$$

Where X_0 is defined to be the threshold on that node and w_0 to be 1. In this exercise, for example, X_0 of the leftmost node is 0.5. The transfer function in this question is the sigmoid function.

a) Let’s name the leftmost node (with 0.5 threshold) H1, the one to the right of it H2, and the rightmost node H3. And we call the node at the bottom Out1.

X1 = 0

X2 = 0

O(H1) = logistic ((0 * -3.1) + (0 * -3.7) + 0.5) = logistic (0.5)
O(H2) = logistic ((0 * 2.5) + (0 * 0.5) + 1.0) = logistic (1)
O(H3) = logistic ((0 * -1.2) + (0 * 3.5) + 1.5) = logistic (1.5)
O(Out1) = logistic ((O(H1) * 1.8) + (O(H2) * -2.2) + (O(H3) * 8.1) + 5.0)
The output of the logistic function can be looked up from the chart attached.
O(Out1) = logistic ((0.62 *1.8) + (0.73 * -2.2) + (0.82 * 8.1) + 5.0) = logistic (11.152), and this value is very close to 1.

b) X1 = 1
X2 = 0
O(H1) = logistic ((1 * -3.1) + (0 * -3.7) + 0.5) = logistic (-2.6)
O(H2) = logistic ((1 * 2.5) + (0 * 0.5) + 1) = logistic (3.5)
O(H3) = logistic ((1 * -1.2) + (0 * 3.5) + 1.5) = logistic (0.3)
O(Out1) = logistic ((0.7 * 1.8) + (0.96 * -2.2) + (0.56 * 8.1) + 5.0) = logistic (4.18), this is close to 1.

c) X1 = 0.7
X2 = 0.2
O(H1) = logistic ((0.7 * -3.1) + (0.2 * -3.7) + 0.5) = logistic (-2.41)
O(H2) = logistic ((0.7 *2.5) + (0.2 * 0.5) + 1) = logistic (2.85)
O(H3) = logistic ((0.7 * -1.2) + (0.2 * 3.5) + 1.5) = logistic (3.04)
O(Out1) = logistic ((0.075 * 1.8) + (0.94 * -2.2) + (0.95 * 8.1) + 5.0) = logistic (10.76), this is again very close to 1.

d) X1 = -5
X2 = 5
O(H1) = logistic ((-5 * -3.1) + (5 * -3.7) + 0.5) = logistic (-2.5)
O(H2) = logistic ((-5 *2.5) + (5 * 0.5) + 1) = logistic (-9)
O(H3) = logistic ((-5 * -1.2) + (5 * 3.5) + 1.5) = logistic (25)
O(Out1) = logistic ((0.08 * 1.8) + (0 * -2.2) + (1 * 8.1) + 5.0) = logistic (13.24), this is again very close to 1.

3. With the **step function** as the transfer function:

If $\sum_{i=0}^{n} w_i X_i$ is greater than 0 then output is 1, otherwise 0.

So with (0,0) as input, O(H1) = 1, O(H2)=1, O(H3)=1, O(Out1)=1.
Similarly, the output for other input patterns can be calculated.

4.

Inputs					Desired Outputs				Actual Outputs			
0.0	1.0	0.5	1.0	0.0	1.0	0.5	0.0	0.0	0.23	0.74	0.19	0.99
0.0	0.3	0.5	1.0	0.0	1.0	0.5	0.0	0.0	0.23	0.80	0.19	0.99
0.0	1.0	0.5	1.0	0.0	0.0	0.5	0.0	0.0	0.23	0.74	0.19	0.88

a) Row 1:

$$TSS(for\ row\ 1) = \sum_{i=1}^{4} (O_{actual_i} - O_{desired_i})^2$$

$$(1 - 0.23)^2 + (0.5 - 0.74)^2 + (0 - 0.19)^2 + (0 - 0.99)^2$$

=0.9801 + 0.0361+ 0.0576+ 0.5929

= 1.6667

Using the same formula for the next rows (or instances), and the output of each row would be:

Row 2: **1.6991**

Row 3: **0.921**

$\mathbf{TSS = Row1 + Row2 + Row3 = 4.2868}$

b)

$\mathbf{MSE} = \frac{TSS}{n} = \mathbf{1.4289}$, where n is the number of instances or rows which in this case is 3.