Name:	
Course:	CAP 4601
Semester:	Summer 2013
Assignment:	Assignment 03
Date:	10 JUN 2013

Complete the following written problems:

1. Parametric vs. Nonparametric Methods (40 Points).

For the methods below, indicate whether the method is parametric or nonparametric:

Method	**Parametric**	**Nonparametric**
Linear Regression	X	
k-Nearest Neighbor		X
Bayesian Network		
Multivariate Linear Regression		
Logistic Regression		
Perceptron		
Multilayer Feed-Forward Neural Network		
Locally Weighted Regression		
Support Vector Machines		
Kernel Density Estimation		

2. Logistic Function (100 Points).

From Page 726 of AIMA, the Logistic Function (also known as the Sigmoid Function) is:

$$h_{\mathbf{w}}(\mathbf{x}) = \frac{1}{1+e^{-\mathbf{w}\bullet\mathbf{x}}}$$

where $\mathbf{w}$ are the weights attached to the independent variables in $\mathbf{x}$ such that $\mathbf{w} = (w_0, w_1, w_2, \ldots, w_n)$ and $\mathbf{x} = (1, x_1, x_2, \ldots, x_n)$. Hence, the dot product is:

$$\mathbf{w}\bullet\mathbf{x}$$

$$(w_0, w_1, w_2, \ldots, w_n)\bullet(1, x_1, x_2, \ldots, x_n)$$

$$w_0 \cdot 1 + w_1 \cdot x_1 + w_2 \cdot x_2 + \cdots + w_n \cdot x_n$$

$$w_0 + w_1 \cdot x_1 + w_2 \cdot x_2 + \cdots + w_n \cdot x_n$$

In other words, $\mathbf{w}\bullet\mathbf{x}$ is just a "plane" … possibly a steep "plane". We have seen how to rotate and translate a plane in 2 dimensions in order to classify points. From the Introduction to the Mathematics of Classification, Part 1, we saw that we could combine rotation and translation such that:

$$f(x, y) = \cos(\theta)(x - x_0) + \sin(\theta)(y - y_0)$$

where θ is the angle that "points" towards the positive class from the point (x_0, y_0).

If we also wanted to adjust the steepness of our plane (aka its "slope"), we could simply add a slope m out in front such that we would have:

$$f_m(x,y) = m\left(\cos(\theta)(x-x_0)+\sin(\theta)(y-y_0)\right)$$

Since "slope" is just the change in the "y" over the change in "x", we can represent this "slope" as:

$$m = \frac{\sin(\phi)}{\cos(\phi)} = \tan(\phi).$$

Therefore, we have:

$$f_\phi(x,y) = \tan(\phi)\left(\cos(\theta)(x-x_0)+\sin(\theta)(y-y_0)\right).$$

If we wanted to use this plane $f_\phi(x,y)$ in a version the logistic function $h_{\mathbf{w}}(\mathbf{x})$ above for two dimensions, then we would have:

$$h_{\mathbf{w}}(\mathbf{x}) = \frac{1}{1+e^{-\mathbf{w}\bullet\mathbf{x}}}$$

$$h_{(w_0,w_1,w_2)}(x,y) = \frac{1}{1+e^{-(w_0,w_1,w_2)\bullet(1,x,y)}}$$

$$= \frac{1}{1+e^{-(w_0+w_1\cdot x+w_2\cdot y)}}$$

$$= \frac{1}{1+e^{-f_\phi(x,y)}}$$

$$h_{x_0,y_0,\theta,\phi}(x,y) = \frac{1}{1+e^{-\tan(\phi)(\cos(\theta)(x-x_0)+\sin(\theta)(y-y_0))}}$$

However, let's just use the following:

$$h(x,y) = \frac{1}{1+e^{-\tan(\phi)(\cos(\theta)(x-x_0)+\sin(\theta)(y-y_0))}}$$

where the (x_0, y_0) is where we place this logistic function and θ is the direction in which this logistic function "points" toward the positive class for a "slope" ϕ such that $0<\phi<\frac{\pi}{2}$.

For instance, given $(x_0,y_0)=(0,0)$, $\theta = 0° = 0$, and $\phi = 45° = \frac{\pi}{4}$, we have:

$$h(x,y) = \frac{1}{1+e^{-\tan(\phi)(\cos(\theta)(x-x_0)+\sin(\theta)(y-y_0))}}$$

$$= \frac{1}{1+e^{-\tan\left(\frac{\pi}{4}\right)(\cos(0)(x-0)+\sin(0)(y-0))}}$$

$$= \frac{1}{1+e^{-1(1(x)+0(y))}}$$

$$= \frac{1}{1+e^{-x}}$$

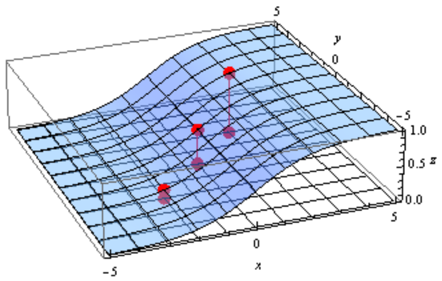
For the following three points:

$$\{(-2,-2),(0,0),(2,2)\}$$

We would have the following values:

$h(-2,-2)$	$h(0,0)$	$h(2,2)$
$\frac{1}{1+e^{-(-2)}}$	$\frac{1}{1+e^{-(0)}}$	$\frac{1}{1+e^{-(2)}}$
0.119203	0.5	0.880797

Hence, we would have the following plots:

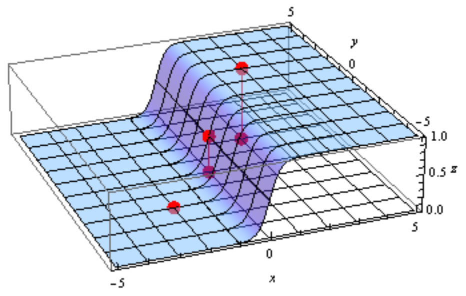


If we increased the "slope" to $\phi = 80° = 80° \frac{\pi}{180°} = \frac{4\pi}{9}$, then we have:

$$\begin{aligned} h(x,y) &= \frac{1}{1+e^{-\tan(\phi)(\cos(\theta)(x-x_0)+\sin(\theta)(y-y_0))}} \\ &= \frac{1}{1+e^{-\tan\left(\frac{4\pi}{9}\right)(\cos(0)(x-0)+\sin(0)(y-0))}} \\ &= \frac{1}{1+e^{-\tan\left(\frac{4\pi}{9}\right)(1(x)+0(y))}} \\ &= \frac{1}{1+e^{-\tan\left(\frac{4\pi}{9}\right)x}} \end{aligned}$$

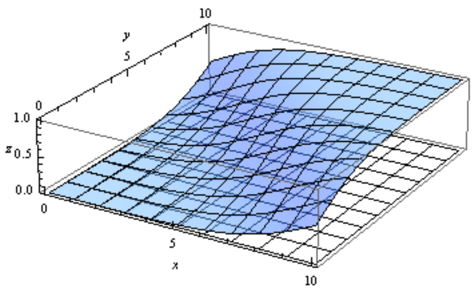
For those same three points, we would have the following values:

$h(-2,-2)$	$h(0,0)$	$h(2,2)$
$\dfrac{1}{1+e^{-\tan\left(\frac{4\pi}{9}\right)(-2)}}$	$\dfrac{1}{1+e^{-\tan\left(\frac{4\pi}{9}\right)(0)}}$	$\dfrac{1}{1+e^{-\tan\left(\frac{4\pi}{9}\right)(2)}}$
0.0000118572	0.5	0.999988

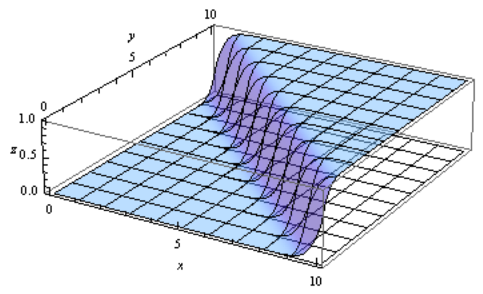


Similar to what we did above and using the points from Written Problem 4 of Assignment 01, do the following:

a) Write the equation of the logistic function $h_a(x,y)$ such that it is placed at $(5,5)$ and is oriented at $\theta = 45°$. In others words, oriented such that it points toward $(10,10)$. Use $\phi = 45°$. The plot for this logistic function would be:



b) Write the equation of another logistic function $h_b(x,y)$ such that it is placed at $(5,5)$ and is oriented at $\theta = 45°$, but this time use $\phi = 80°$. The plot for this logistic function would be:



c) Compute the values of $h_a(x,y)$ and $h_b(x,y)$ for the following points from Written Problem 4 of Assignment 01:

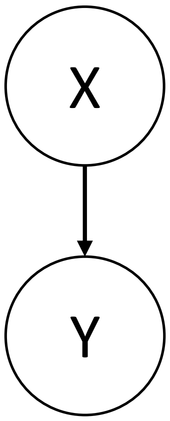
Class 1		
Point (x,y)	**Value of** $h_a(x,y)$	**Value of** $h_b(x,y)$
$(0,1)$		
$(1,0)$		
$(0,0)$		

Class 2		
Point (x,y)	**Value of** $h_a(x,y)$	**Value of** $h_b(x,y)$
$(10,10)$		
$(10,9)$		
$(9,10)$		

d) Which class (Class 1 or Class 2) has values for $h_a(x,y)$ and $h_b(x,y)$ that are over 0.5? And which class has values under 0.5?

3. Bayes Network (60 Points):

Given the following Bayes Network:



With the following probabilities:

$$P(X) = 0.25$$

$$P(Y|X) = 0.4$$

$$P(Y|\neg X) = 0.6$$

a) What is $P(X|Y)$? Show all work.

b) Show that $P(Y) + P(\neg Y) = 1$. Show all work.

Complete the following programming problem on `linprog4.cs.fsu.edu`:

Download the ZIP file containing the directory structure and files for this programming problem: assignment_03.zip

1. Logistic Regression (100 Points):

Recalling Logistic Regression from Machine Learning with Andrew Ng, program logistic regression using gradient descent on the data from Written Problem 4 of Assignment 01. Use the logistic function, cost function, and gradients provided in the video lectures to fit a logistic regression to those six points.

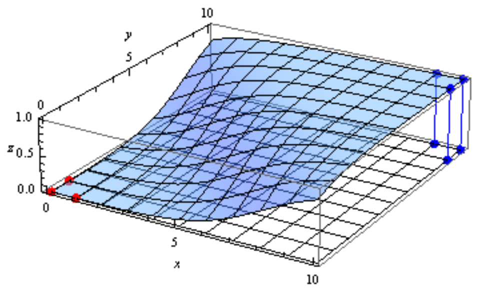
Using the notation from the Logistic Function in the Written Problems above, this logistic regression will iteratively find the locally best parameters $\mathbf{w} = (w_0, w_1, w_2, \ldots, w_n)$ for a given dimension n. Since we have two dimensional data, then $n = 2$; hence, we'll iteratively find the locally best parameters $\mathbf{w} = (w_0, w_1, w_2)$ for $\mathbf{x} = (1, x_1, x_2)$. Note: Andrew Ng used $\theta = (\theta_0, \theta_1, \ldots, \theta_j, \ldots, \theta_n)$ to represent these same parameters in the Machine Learning course.

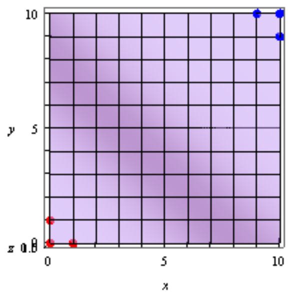
Start your gradient descent at with the parameters initially set to $\mathbf{w} = (0, 0, 0)$, use a learning rate of $\alpha = 0.1$, and ensure that the termination criteria for your gradient descent is set to the following:

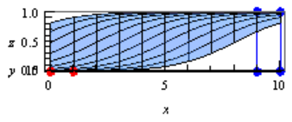
Stop the gradient descent when either:

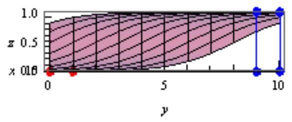
- The distance between consecutive iterations of parameters $\mathbf{w} = (w_0, w_1, w_2)$ is less than 0.000001. In other words, given the n-th and $n+1$-th iteration, $\|\mathbf{w}_{n+1} - \mathbf{w}_n\| < 0.000001$.
- The number of full gradient descent iterations exceeds 1024. In other words, don't do more than 1024 updates of gradient descent.

Given the Class 1 points (the red points below) and the Class 2 points (the blue points below), ensure your logistic regression approximately produces parameter values $\mathbf{w} = (w_0, w_1, w_2)$ consistent with the following plots:









HINT: Pay close attention to the $y^{(i)}$ terms that Andrew Ng mentions in the videos. You may need to manipulate the class labels in the data set.

HINT: Pay close attention to the $x_j^{(i)}$ terms that Andrew Ng mentions in the videos. Do not forget the 1 in those terms. Note: We used the notation $\mathbf{x} = (1, x_1, x_2)$ for this term above.

Use `std::cout` to output information exactly in the following format:

```
1: { 0, 0, 0 }
2: { 0, 1.4, 1.4 }
3: { -0.210437, 1.31978, 1.31978 }
4: { -0.405597, 1.24458, 1.24458 }
.
.
.
```

Note: The ellipses above should not be included in your output. The ellipses represent the rest of your properly formatted output for this gradient descent problem.

2. Cross Validation - Part 1 (100 Points):

Using OpenCV's CvMLData, create a program that reads in the Breast Cancer Wisconsin (Diagnostic) Data Set, ignores the ID number column from the dataset, uses the Diagnosis column as the class, and uses the remaining columns as the data.

Separate each class into its own dataset. Example: Place all the data whose Diagnosis column reads "M" (for malignant) into one container. Place all the data that reads "B" (for benign) into another container.

Split each container such that 95% of the data is placed into a training set and a 5% is placed into a testing set. Therefore, 95% of malignant data and 95% of benign data will be in the training dataset. Moreover, 5% of the malignant data and 5% of the benign data will be in the testing dataset.

Use `std::cout` to print the following:

- The number of malignant data points in the training dataset.
- The number of benign data points in the training dataset.
- The number of malignant data points in the testing dataset.
- The number of benign data points in the testing dataset.

Ensure your output is formatted exactly as follows:

```
Training Data
-------------
Malignant: AAA
Benign: BBB

Testing Data
------------
Malignant: CCC
Benign: DDD
```

… where `AAA`, `BBB`, `CCC`, and `DDD` represent the counts in their respective datasets.