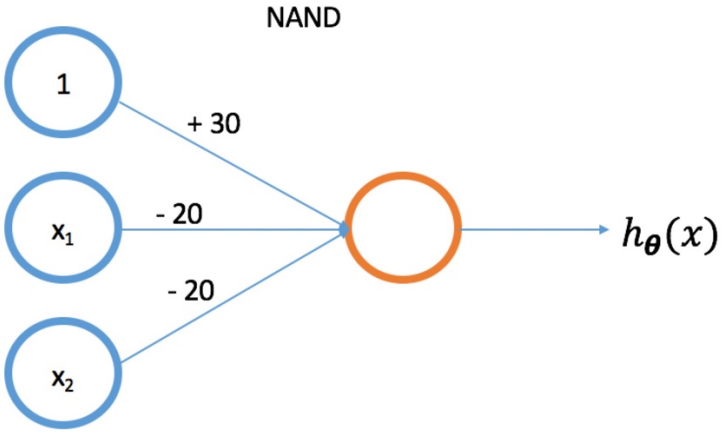
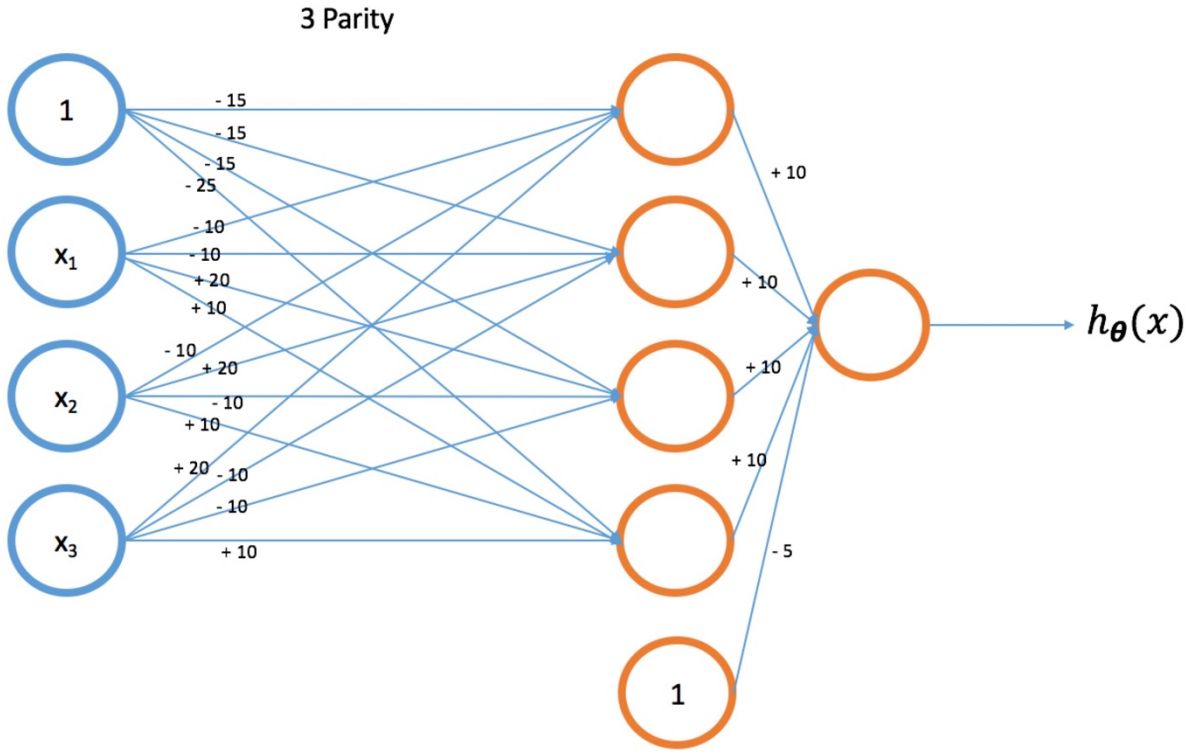
HW 4

Part I: Problem Set

1. Logistic functions with neural networks
   (a)



   (b)



2. Backpropabation with momentum

Epoch 1:

For $x_1 = 1, \quad x_2 = 0, \quad y_1 = 1$ and all θ are initialized to 0.1

$$a_2 = \frac{1}{1 + e^{-(0.1*1+0.1*1+0.1*0)}} = 0.55$$

$$a_3 = \frac{1}{1 + e^{-(0.1*1+0.1*0.55)}} = 0.539$$

$$\delta_3 = a_3 - y = 0.539 - 1 = -0.461$$

$$\delta_2 = 0.55 * (1 - 0.55) * 0.1 * (-0.461) = -0.0114$$

$$\Delta_{x_1,h} = \frac{1}{2} * (-0.0114 * 1) + 0.001 * 0.1 = -0.0056$$

$$\Delta_{x_2,h} = \frac{1}{2} * (-0.0114 * 0) + 0.001 * 0.1 = -0.0001$$

$$\Delta_{0,h} = \frac{1}{2} * (-0.0114) * 1 = -0.0057$$

$$\Delta_{h,y} = \frac{1}{2} * (-0.461 * 0.55) + 0.001 * 0.1 = -0.1267$$

$$\Delta_{0,y} = \frac{1}{2} * (-0.461 * 1) = -0.2305$$

For $x_1 = 0, \quad x_2 = 1, \quad y_1 = 0$ and all θ are initialized to 0.1

$$a_2 = \frac{1}{1 + e^{-(0.1*1+0.1*1+0.1*0)}} = 0.55$$

$$a_3 = \frac{1}{1 + e^{-(0.1*1+0.1*0.55)}} = 0.539$$

$$\delta_3 = a_3 - y = 0.539 - 1 = 0.539$$

$$\delta_2 = 0.55 * (1 - 0.55) * 0.1 * 0.539 = 0.0133$$

$$\Delta_{x_1,h} = \frac{1}{2} * (0.0133 * 0) = 0$$

$$\Delta_{x_2,h} = \frac{1}{2} * (0.0133 * 1) = 0.0066$$

$$\Delta_{0,h} = \frac{1}{2} * (0.0133) * 1 = 0.0066$$

$$\Delta_{h,y} = \frac{1}{2} * (0.539 * 0.55) = 0.1482$$

$$\Delta_{0,y} = \frac{1}{2} * (0.539 * 1) = 0.2695$$

$$D_{x_1,h} = 0 - 0.0056 = -0.0056$$

$D_{x_2,h} = 0.0066 + 0.0001 = 0.0067$
$D_{0,h} = -0.0057 + 0.0066 = 0.0009$
$D_{h,y} = 0.1482 - 0.1267 = 0.0215$
$D_{0,y} = 0.2695 - 0.2305 = 0.039$

Update Θ:
$\theta_{x_1,h} = 0.1 - 0.3 * (-0.0056) + 0.9 * 0 = 0.1017$
$\theta_{x_2,h} = 0.1 - 0.3 * 0.0067 + 0.9 * 0 = 0.098$
$\theta_{0,h} = 0.1 - 0.3 * 0.0009 + 0.9 * 0 = 0.0997$
$\theta_{h,y} = 0.1 - 0.3 * 0.0215 + 0.9 * 0 = 0.0936$
$\theta_{0,y} = 0.1 - 0.3 * 0.039 + 0.9 * 0 = 0.0884$

Epoch 2:
For $x_1 = 1, \quad x_2 = 0, \quad y_1 = 1$ and all θ are initialized to 0.1

$$a_2 = \frac{1}{1 + e^{-(0.1017*1+0.0997*1+0.098*0)}} = 0.55$$

$$a_3 = \frac{1}{1 + e^{-(0.0884*1+0.0936*0.55)}} = 0.5349$$

$\delta_3 = a_3 - y = 0.5349 - 1 = -0.4651$
$\delta_2 = 0.55 * (1 - 0.55) * 0.0936 * (-0.4651) = -0.0108$

$\Delta_{x_1,h} = \frac{1}{2} * (-0.0108 * 1) + 0.001 * 0.1017 = -0.0053$
$\Delta_{x_2,h} = \frac{1}{2} * (-0.0108 * 0) + 0.001 * 0.098 = -0.000098$
$\Delta_{0,h} = \frac{1}{2} * (-0.0108) * 1 = -0.0055$
$\Delta_{h,y} = \frac{1}{2} * (-0.461 * 0.55) + 0.001 * 0.0936 = -0.1281$
$\Delta_{0,y} = \frac{1}{2} * (-0.4651 * 1) = -0.2326$

For $x_1 = 0, \quad x_2 = 1, \quad y_1 = 0$ and all θ are initialized to 0.1

$$a_2 = \frac{1}{1 + e^{-(0.1017*0+0.0997*1+0.098*1)}} = 0.5493$$

$$a_3 = \frac{1}{1 + e^{-(0.0883*1+0.0936*0.5493)}} = 0.5349$$

$\delta_3 = a_3 - y = 0.5349 - 0 = 0.5349$
$\delta_2 = 0.5493 * (1 - 0.5493) * 0.0936 * 0.5349 = 0.0124$

$\Delta_{x_1,h} = \frac{1}{2} * (0.0124 * 0) = 0$

$$\Delta_{x_2,h} = \frac{1}{2} * (0.0124 * 1) = 0.0062$$
$$\Delta_{0,h} = \frac{1}{2} * (0.0124) * 1 = 0.0062$$
$$\Delta_{h,y} = \frac{1}{2} * (0.5349 * 0.5493) = 0.1469$$
$$\Delta_{0,y} = \frac{1}{2} * (0.5349 * 1) = 0.2675$$

$$D_{x_1,h} = 0 - 0.0053 = -0.0053$$
$$D_{x_2,h} = 0.0062 + 0.000098 = 0.006298$$
$$D_{0,h} = -0.0053 + 0.0062 = 0.0009$$
$$D_{h,y} = 0.1469 - 0.1281 = 0.0188$$
$$D_{0,y} = 0.2674 - 0.2325 = 0.0349$$

Update Θ:
$$\theta_{x_1,h} = 0.1017 - 0.3 * (-0.0053) + 0.9 * (-0.0056) = 0.0982$$
$$\theta_{x_2,h} = 0.098 - 0.3 * 0.006298 + 0.9 * 0.0067 = 0.1021$$
$$\theta_{0,h} = 0.0997 - 0.3 * 0.0009 + 0.9 * 0.0009 = 0.1003$$
$$\theta_{h,y} = 0.0936 - 0.3 * 0.0188 + 0.9 * 0.0215 = 0.1073$$
$$\theta_{0,y} = 0.0884 - 0.3 * 0.039 + 0.9 * 0.039 = 0.1125$$

3. TANH Neural Networks
   (a) Sigmoid function gives a value between (0,1), while Tanh function always gives a value between (-1, 1). Since the data is centered around 0, the derivatives are higher, thus tanh function gives stronger gradients.
   (b) $\tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}} = \frac{2}{1+e^{-2z}} - 1 = 2\sigma(2z) - 1$

   $$\sigma(z) = \frac{\tanh\left(\frac{z}{2}\right) + 1}{2}$$

   Thus the output function

   $$y_k(x,\theta) = \sigma\left(\sum_{j=1}^{M} \Theta_{jk}^{(2)} \sigma\left(\sum_{i=1}^{d} \Theta_{ij}^{(1)} x_i + \Theta_{0j}^{(1)}\right) + \Theta_{0k}^{(2)}\right)$$

   Can be rewritten as:

$$y_k(x,\theta) = \sigma(\sum_{j=1}^{M} \Theta_{jk}^{(2)}[\frac{1}{2}tanh\left(\frac{1}{2}\sum_{i=1}^{d}\Theta_{ij}^{(1)}x_i + \frac{1}{2}\Theta_{0j}^{(1)}\right) + \frac{1}{2}] + \Theta_{0k}^{(2)})$$

$$= \sigma(\sum_{j=1}^{M}(\frac{1}{2}\Theta_{jk}^{(2)})tanh\left(\frac{1}{2}\sum_{i=1}^{d}\Theta_{ij}^{(1)}x_i + \frac{1}{2}\Theta_{0j}^{(1)}\right) + (\frac{1}{2}\sum_{j=1}^{M}\Theta_{jk}^{(2)} + \Theta_{0k}^{(2)}))$$

$$= \sigma(\sum_{j=1}^{M}\omega_{jk}^{(2)}tanh\left(\sum_{i=1}^{d}\omega_{ij}^{(1)}x_i + \omega_{0j}^{(1)}\right) + \omega_{0k}^{(2)})$$

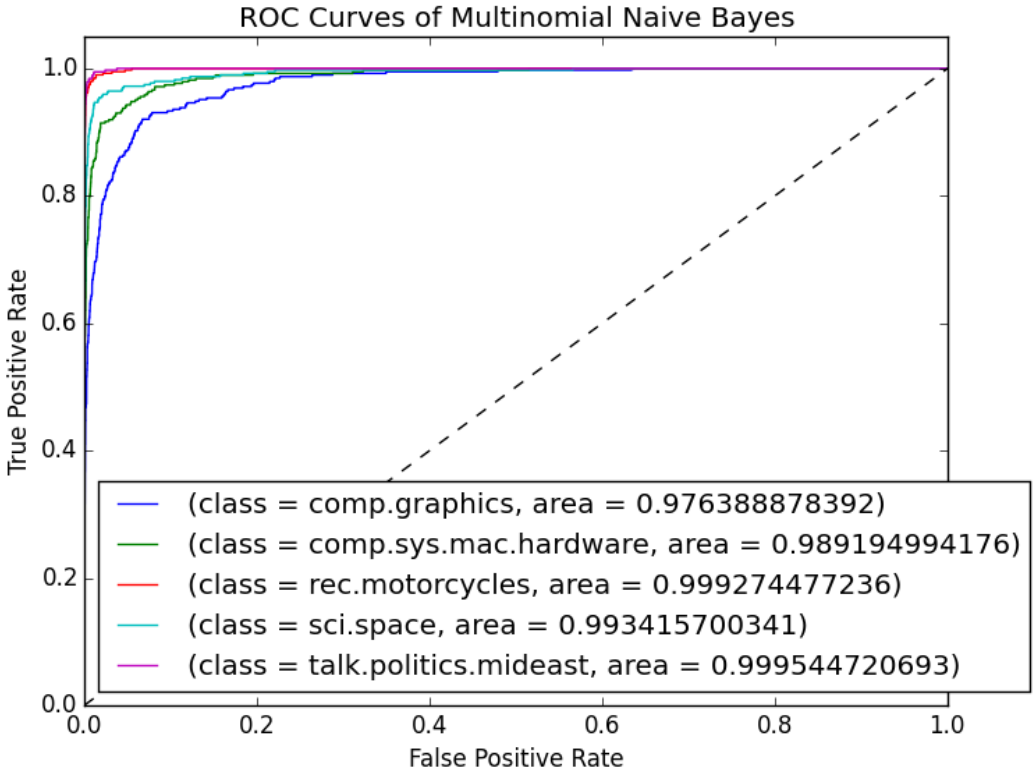
$$\omega_{jk}^{(2)} = \frac{1}{2}\Theta_{jk}^{(2)},\ \omega_{ij}^{(1)} = \frac{1}{2}\Theta_{ij}^{(1)},\ \ \omega_{0j}^{(1)} = \frac{1}{2}\Theta_{0j}^{(1)},\ \ \omega_{0k}^{(2)} = \frac{1}{2}\sum_{j=1}^{M}\Theta_{jk}^{(2)} + \Theta_{0k}^{(2)}$$

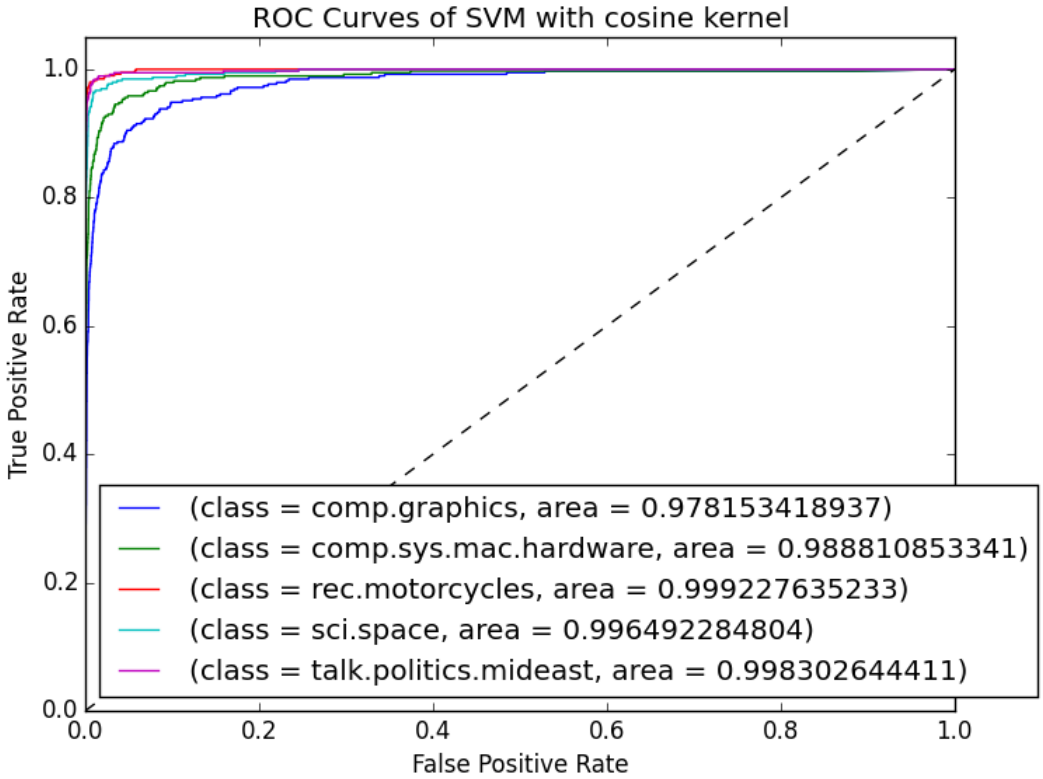
Part II Programming:

1.

Machine Learning Methods		accuracy	precision	recall	time
multinomial Naive Bayes	train	0.997790348241	0.997797640995	0.997790348241	0.152586
multinomial Naive Bayes	test	0.837360594796	0.838591349207	0.837360594796	0.152586
SVM & cosine similarity	train	0.996199398975	0.996221040367	0.996199398975	34.578064
SVM & cosine similarity	test	0.846255974509	0.851661750238	0.846255974509	34.578064

It can be seen from the table that multinomial Naïve Bayes method has a slightly higher training accuracy, precision and recall, but SVM with cosine similarity kernel has a slightly higher accuracy, precision and recall on testing data. As for running time, multinomial Naïve Bayes uses a lot less time than SVM with cosine similarity kernel. Thus I think multinomial Naïve Bayes is better than SVM because it's a lot cheaper in time compared to SVM with almost the same accuracy, precision and recall.


ROC Curves of Multinomial Naive Bayes
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
(class = comp.graphics, area = 0.976388878392)
(class = comp.sys.mac.hardware, area = 0.989194994176)
(class = rec.motorcycles, area = 0.999274477236)
(class = sci.space, area = 0.993415700341)
(class = talk.politics.mideast, area = 0.999544720693)


ROC Curves of SVM with cosine kernel
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
(class = comp.graphics, area = 0.978153418937)
(class = comp.sys.mac.hardware, area = 0.988810853341)
(class = rec.motorcycles, area = 0.999227635233)
(class = sci.space, area = 0.996492284804)
(class = talk.politics.mideast, area = 0.998302644411)

2.