HW 12

1. Reading: Sections 11.3-11.5 from Elements of Statistical Learning. This is the Neural Net chapter.

2. In this problem you will implement a neural network to solve a classification problem. To keep things simple - since time is limited - the data will consist of covariates $x^{(i)} \in \mathbb{R}^2$ and a response $y_i \in \{0, 1\}$ for $i = 1, 2, \dots, N$ (notice y takes on only two possible values). The classification problem involves fitting the model $y \sim f(x)$ over functions $f(x)$ that can be parameterized by our neural net, which is described below. The attached file `nn.txt` contains the samples. Each sample, corresponding to a row in the file, gives the three values $(x_1^{(i)}, x_2^{(i)}, y_i)$.

   Your neural network should consist of three layers, as discussed in class. The input level should contain X_1 and X_2 nodes, which will be the two coordinates for each of the samples $x^{(i)}$ (i.e. $X_1 = x_1^{(i)}$ and $X_2 = x_2^{(i)}$). The middle (hidden) layer should contain at m nodes, Z_j for $j = 1, 2, \dots, m$. And an output layer consisting of single nodes T_1, T_2. Then Y_1, Y_2 are the neural net's estimate of the probability that the class of $x^{(i)}$ is 0 and 1, respectively. To repeat what we mentioned in class, each Z_j is parameterized as follows:

$$Z_j = \sigma(\alpha_0^{(j)} + \alpha^{(j)} \cdot x), \tag{1}$$

   where $x = (X_1, X_2)$, $\alpha_0^{(j)} \in \mathbb{R}$, $\alpha^{(j)} \in \mathbb{R}^2$, and $\sigma(w) = 1/(1 + \exp(-w))$. The node T_j is parameterized as follows:

$$T_j = \sigma(\beta_0^{(j)} + \beta^{(j)} \cdot z), \tag{2}$$

   where $z = (Z_1, Z_2, \dots, Z_m)$, $\beta_0^{(j)} \in \mathbb{R}$, $\beta^{(j)} \in \mathbb{R}^m$. Finally

$$Y_i = \frac{\exp[T_i]}{\exp[T_1] + \exp[T_2]} \tag{3}$$

   (a) Visualize the dataset by plotting it with different colors for the two classes of y.

(b) Let η be the parameters of the neural net (i.e. all the α's and β's). What is the dimension of η in terms of m?

(c) Write a function $NN(x, \eta, m)$ which takes a sample $x \in \mathbb{R}^2$ and a choice for η and returns the values of Y_1 and Y_2. (Hint: It may be helpful to write functions such as `get_alpha(eta, i)`, which given η and i returns $\alpha^{(i)}$, and `get_alpha_0(eta, i)`, which given η and i returns $\alpha_0^{(i)}$. Using such functions will greatly simplify your code.)

(d) Explain why the log likelihood function $\log L(\eta)$ for the neural net is given by

$$\log L(\eta) = \sum_{i=1}^{N} (1 - y_i) \log(Y_1) + y_i \log(Y_2). \tag{4}$$

(In class, when I wrote the log likelihood, I forgot the log on the Y_1 and Y_2!) Write a function that computes $\log L(\eta)$ (you will need to pass the data to the function). Write a function that uses finite difference to compute the gradient of $\log L(\eta)$.

(e) Set $m = 4$ and train your neural net by maximizing the $\log L(\eta)$ using steepest ascent. (It took me roughly 45 minutes of run time to get a good fit, roughly 3000 iterations. Your results may vary from this depending on implemenation and hardware.)

(f) Remember that a classifier in this case is a function $F(x) : \mathbb{R}^2 \to \{0, 1\}$, where $x \in \mathbb{R}^2$. Once you choose η by by computing the maximum likelihood in (e), choose a cutoff $p \in [0, 1]$. Set F by

$$F(x) = \begin{cases} 0 & \text{if } Y_1(x, \eta) < p \\ 1 & \text{if } Y_1(x, \eta) \geq p \end{cases} \tag{5}$$

Try different value of p and for each p, visualize your classifier. You can do this in any way you like, but here is one way. Generate many points in $\mathbb{R}^2$, determine the predicted class of each using your classifier, and then plot as in part (a). You can generate random coordinates within $[-2, 2]$, which is roughly where all the data points lie, using

```
x1 <- 4*runif(10000) - 2
x2 <- 4*runif(10000) - 2
```

`runif(10000)` generates 10000 numbers uniformly distributed on $[0, 1]$.

(g) Now repeat (e) and (f), but in your log likelihood, include a penalty term of the form,

$$\rho\left(\sum_{i=1}^{m}\|\alpha^{(i)}\|^2+\sum_{j=1}^{2}\|\beta^{(j)}\|^2\right). \tag{6}$$

Given that we are maximizing, explain why you should subtract the penalty term rather than add it onto the log-likelihood in (d). Train your net with several values of ρ, visualize the resulting classifier for differnt p, and compare your results.

3. Repeat problem 2, but now use a logistic regression to build your classifier. How does the logistic regression classifier compare to the neural net classifier?