Homework 13

1. In this problem you will implement a neural network to solve a classification problem. To keep things simple, the data will consist of covariates $x^{(i)} \in \mathbb{R}^2$ and a response $y_i \in \{0, 1\}$ for $i = 1, 2, \ldots, N$ (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 layer 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 m nodes, Z_j for $j = 1, 2, \ldots, m$. And an output layer consisting of nodes T_1, T_2. Then, the probability that the class of $x^{(i)}$ is 1 given by,

$$P(y = 1 \mid x^{(i)}, \alpha) = \frac{\exp[T_1]}{\exp[T_1] + \exp[T_2]}, \tag{1}$$

   where α is the vector of parameters of the neural net and T_1, T_2 are computed using the neural net with input $x^{(i)}$. To repeat what we mentioned in class, each Z_j is parameterized as follows:

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

   where $x = (X_1, X_2)$, $\beta_0^{(j)} \in \mathbb{R}$, $\beta^{(j)} \in \mathbb{R}^2$, and $\sigma(w) = 1/(1 + \exp(-w))$. The node T_j is parameterized as follows:

$$T_j = \sigma(\gamma_0^{(j)} + \gamma^{(j)} \cdot z), \tag{3}$$

   where $z = (Z_1, Z_2, \ldots, Z_m)$, $\gamma_0^{(j)} \in \mathbb{R}$, $\gamma^{(j)} \in \mathbb{R}^m$. Then α is the concatenation of all the parameters: $\beta_0^{(j)}$, $\beta^{(j)}$ for $j = 1, 2, \ldots, m$ and $\gamma_0^{(j)}$, $\gamma^{(j)}$ for $j = 1, 2$.

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

(b) What is the dimension of α in terms of m?

(c) Write a function $NN(x, \alpha, m)$ which takes a sample $x \in \mathbb{R}^2$ and a choice for α and returns the neural net estimate of $P(y = 1 \mid x, \alpha)$.. (Hint: It may be helpful to write functions such as `get_beta(alpha i)`, which given α and i returns $\beta^{(i)}$, and `get_beta_0(eta, i)`, which given α and i returns $\beta_0^{(i)}$. 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

$$\log L(\eta) = \sum_{i=1}^{N} (1-y_i) \log \left(\frac{\exp[T_1]}{\exp[T_1] + \exp[T_2]} \right) + y_i \log \left(1 - \frac{\exp[T_1]}{\exp[T_1] + \exp[T_2]} \right). \tag{4}$$

Write a function that computes $\log L(\alpha)$ (you will need to pass the data to the function).

(e) Write a function that uses finite difference to compute the stochastic gradient of $\log L(\alpha)$ based on a single or small number of samples (you can pick whether to use one sample or a few).

(f) Set $m = 4$ and train your neural net by maximizing the $\log L(\alpha)$ using **stochastic** steepest ascent.

(g) 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 α by computing the maximum likelihood in (e), choose a cut-off $p \in [0, 1]$. Set F by

$$F(x) = \begin{cases} 0 & \text{if } P(y = 1 \mid x, \alpha) > p \\ 1 & \text{if } P(y = 1 \mid x, \alpha) \leq p \end{cases} \tag{5}$$

Try different value of p and for each p, visualize your classifier. You can visualize your classifier in any way you like, but here is one way. 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
```

Then plot each pair `x1[i], x2[i]` and vary the color of the point depending on whether the classifier predicts 1 or 0.