Analytics
Assignment 2:
Neural Networks.

Statistics Honours

March 22, 2019

General Notes and Instructions

- Where questions require algebra, you may provide hand-written answers provided that they are properly labelled though it is suggested that you typeset answers properly — Report writing is a skill you should develop as soon as possible. You may use any typesetting software you wish, but I would encourage you to use R-Markdown or LaTeX if you have sufficient experience. See the primers on R-Markdown on Vula and make use of the cheat-sheets for reference.
- Unless stated otherwise, where you are required to write R-code, **DO NOT** give blocks of code and output interspersed between your answers (you may still refer to R-functions in-text). Instead, provide the complete code for each of the relevant questions under separate headings as an **appendix** to your write-up. For example, for Question 2, provide all code pertaining to this question as a single script in the appendix under the subsection heading 'R-Code: Question 2'. When you typeset R code use `courier` or an equivalent 'typewriter'-like font.
- Plots are very useful but use them sparingly! Make sure that a given plot is relevant to the question and pertains to text in your answer. Figures are meant to enrich your analysis, not clutter the analysis and be left for the reader to disentangle. Make sure that your plots are square and have appropriate legends.
- You may write as much R code as you like, but the body of your text (i.e., excluding the appendix with your R code) **may not exceed 8 pages**.
- All assignments must be accompanied by a signed plagiarism declaration. No declaration, no mark.
- Hand-in dates will be announced on Vula.

Problem Set

Question 1 (25 marks)

Breast Cancer Data: The following dataset is taken from 'Breast Cancer Prediction Dataset', a Kaggle dataset created for "AI for Social Good: Women Coders' Bootcamp". The description reads:

"Worldwide, breast cancer is the most common type of cancer in women and the second highest in terms of mortality rates.Diagnosis of breast cancer is performed when an abnormal lump is found (from self-examination or x-ray) or a tiny speck of calcium is seen (on an x-ray). After a suspicious lump is found, the doctor will conduct a diagnosis to determine whether it is cancerous and, if so, whether it has spread to other parts of the body.

This breast cancer dataset was obtained from the University of Wisconsin Hospitals, Madison from Dr. William H. Wolberg."

You are provided with two datasets, `Breast Cancer Dataset A.txt`, containing the data you should use to answer the questions below, and `Breast Cancer Dataset B.txt`, containing observations on the following variables: "Features are computed from a digitized image of a fine needle aspirate (FNA) of a breast mass. They describe characteristics of the cell nuclei present in the image."

Variable	Description
`Response`	Diagnosis indicator 1 if 'benign', 0 if 'malignant'.
`Radius`	Mean radius.
`Texture`	Mean texture.
`Perimeter`	Mean perimeter.
`Area`	Mean area.
`Smoothness`	Mean smoothness .

```
R> rm(list = ls(all = TRUE))
R> dat = read.table('Breast Cancer Dataset A.txt', h = TRUE)
R> head(dat, 10)
```

```
  Radius Texture Perimeter Area Smoothness Response
1   20.60   29.33    140.10 1265.0  0.11780        0
2   11.47   16.03     73.02  402.7  0.09076        1
3   17.47   24.68    116.10  984.6  0.10490        0
4   14.26   18.17     91.22  633.1  0.06576        1
5   13.85   15.18     88.99  587.4  0.09516        1
6   14.11   12.88     90.03  616.5  0.09309        1
7   12.99   14.23     84.08  514.3  0.09462        1
8   13.74   17.91     88.12  585.0  0.07944        1
9   14.97   16.95     96.22  685.9  0.09855        1
10  13.61   24.69     87.76  572.6  0.09258        0
```

(a) Write an R function that calculates the number of parameters in a neural network with an arbitrary number of hidden layers based on the size of the input vector (`p`), the size of the output vector (`q`), and the number of units in each hidden layer (the number of nodes for the l-th layer corresponding to the l-th element of the vector `d`). Outline the calculation and show the relevant R code in your report. [2]

(b) The updating equation which characterises the standard feedforward neural network is given in **vectorised** form by: [2]

$$\mathbf{a}^l = \sigma_l((\mathbf{W}^l)^T\mathbf{a}^{l-1} + \mathbf{b}^l)$$

for $l = 1, 2, \ldots, L$ subject to the initial condition $\mathbf{a}^0 = \mathbf{x}$. When fitting a neural network, this equation needs to be evaluated for each training example. I.e., starting from $\mathbf{a}^0 = \mathbf{x}_i$ for $i = 1, 2, \ldots, N$ where N is the number of training examples. Give an expression for the updating equation in **matrix** form. Hint: The design matrix is $\mathbf{X} = (\mathbf{x}_i^T : i = 1, 2, \ldots, N)_{N\times p}$.

(c) Write an R-function that evaluates the updating equation for a feed-forward neural net with **two** hidden layers of `m` units each (identical number of nodes for each layer). Your function should also take as input a vector of parameters, `theta`, which concatenates all weights and biases in the function. Outline the algorithm[1] and show the relevant R code in your report (just the function). You may use the following template: [4]

```
> # X     - Input matrix (N x p)
> # Y     - Output matrix (N x q)
> # d     - Vector with number of hidden units for hidden layers
> # theta - Parameters of the NN
> neural_net = function(X, Y, theta, d = c(m, m), lambda)
+ {
+   # Your work here
+ }
```

Hint: Use your answer in (b) to improve the speed with which the network is evaluated over multiple training examples.

(d) Modify the R-function in part (b) by applying an appropriate cost function to the output layer of the neural network.[2] Give a short motivation for your choice of cost function. [2]

(e) Modify the R-function in part (b) by applying an appropriate regularisation mechanism to the model[2]. Give a short explanation of the function of the regularization mechanism in fitting a neural network and motivate your choice of regularisation mechanism. [2]

(f) Using the R-function developed in parts (c)–(e), fit neural networks with a two hidden layers of dimension $m = 2, 3, \ldots, 5$ to set A for the ——— data (you may use general purpose optimisers such as `nlm()` or `optim()` to minimize the relevant objective function—the function written in (a) should be useful here. For each architecture, fit the model using a range of values for the regularisation parameter. That is, create a sequence of values for the regularisation parameter and plot the corresponding training and test error using an appropriate validation split (e.g., 80% train, 20% validation) for each value in the sequence[3]. Use **one figure** to plot the in sample for all m and another for the out-of sample error for all architectures. [7]

Hints: `iterlim`, `scale()`, `smooth.spline()`.

[1]The outline may be in numbered list form/pseudo code.
[2]You may add the modification to your code in (b).
[3]You may add a smoother to generate a legible curve from the data points.

(g) Based on the results in (e), give your analysis as to whether there is an optimal number of nodes for the hidden layers for the present dataset and choose a configuration from the set of networks that performs 'best', clearly motivating your choice. [4]

(h) Write your predictions (use 1 to indicate 'benign' and 0 to indicate 'malignant') to a **.csv** file using the following naming convention (replace 'STUDENTNO' with your student number): [2]

```
R> pred = data.frame(cbind(predictions))
R> write.table(pred,'Cancer_Pred_STUDENTNO.csv', quote = F, row.names = F, sep = ',')
```

and upload your predictions along with your write-up to Vula.

Question 2 (15 marks)

What's in a Neural Network? The following questions pertain to the `UnderTheHood.txt` data.

(a) Modify your `neural_net()` function in Question 1 to evaluate neural networks with two hidden layers but with differing numbers of nodes. Use the revised function to fit a $(6, 2)$-network to the `UnderTheHood.txt` data (6 nodes on the first hidden layer and 2 on the second hidden layer). [7]

(b) Further modify your `neural_net()` function to return a the hidden nodes for each observation along with the output nodes (i.e., `return(list(a1 = a1, a2= a2, out = out,...))`. Use this to evaluate and plot the intensity of **each node** in the network over the input space. To do this, create a 2D lattice of equispaced points on $[-1, 1] \times [-1, 1]$ and colour each point according to the value of the particular node you are interested in (evaluate the network for the using the coordinate of the lattice as the input). You may use the following function to assign colours to intensity: [4]

```
# Takes input x, assigns colour according to level in range(x)
  colfunc = function(x, n = 100)
{
   xx = seq(min(x),max(x), length =n) # Create n bins
   ii = findInterval(x,xx)         # Which bin does each element x goo to
   colr = colorRampPalette(c('grey','black','blue'))
   cols = colr(n) # A colour from the ramp above for each bin
   return(cols[ii])
}
```

Notes: Create a 3 by 2 grid of plots for the first layer, and then a 2 by 1 grid for the second layer.

(c) By interpreting the figures generated in (b), provide an explanation of what the neural network is *doing*. [4]

Useful resources

- Matrix Algebra in R
- Matrix Computations in R
- How to write and debug an R function