STA 712 Challenge Assignment 2: Neural Networks and Logistic Regression

Due: By Friday, November 10 at 12:00pm (noon) on Canvas.

Instructions:

- Submit your work as a single PDF. Use TEXto type and format any math and equations, either with a LATEXeditor (Overleaf, Texmaker, etc.), or directly in an R Markdown or Quarto document. Include all R code needed to reproduce your results in your submission.
- You are welcome to work with others on this assignment, but you must submit your own work.
- The goal of this assignment is for you to learn about a topic beyond the core material covered in class. If you get stuck, I am happy to chat over email or in office hours.
- You can probably find the answers to many of these questions online. It is ok to use online resources! But make sure to show all your work in your final submission.

Introduction

Logistic regression is one of the most widely used tools for predicting a binary response. However, logistic regression is limited by the assumption that the log odds are a *linear* function of the predictors. What if we want to model nonlinear relationships? Transformations are one option, but it can be hard to choose the right transformations with high-dimensional data.

Another option is *neural networks*, a popular prediction/classification method which can be used to fit more complex relationships. It also turns out that logistic regression can be represented as a special case of a *feedforward* neural network (the simplest type), so neural networks provide a nice generalization of logistic regression.

The goal of this assignment is to introduce you to neural networks, show how logistic regression can be viewed as a simple network, and experiment with fitting more complicated models.

Background reading:

This assignment requires you to learn quite a bit of new information that isn't covered in class. Here I provide some references to get started, but you may need to do some research beyond these references.

- To get started and see an overview of neural networks, skim Chapter 1 of *Neural Networks and Deep Learning*, a free online book by Michael Nielsen available here: `http://neuralnetworksanddeeplearning.com/index.html`
- I also recommend the 3Blue1Brown YouTube videos on neural networks: `https://www.youtube.com/playlist?list=PLZHQObOWTQDNU6R1_67000Dx_ZCJB-3pi`
- You may also find these slides from a CMU course on machine learning helpful, in particular lecture 11 and lecture 14: `http://www.cs.cmu.edu/~mgormley/courses/10601-s18/slides/`
- For an introduction to performance metrics like sensitivity, specificity, and ROC curves, see *An Introduction to Statistical Learning* (James, Witten, Hastie, and Tibshirani), available for free online: `https://www.statlearning.com/`

Fitting a neural network:

There are many options for fitting a neural network. If you have never fit a network before, I suggest using one of the following:

- The `neuralnet` package in R (`https://cran.r-project.org/web/packages/neuralnet/`)
- TensorFlow for R, using Keras (`https://tensorflow.rstudio.com/`)
- PyTorch, in Python (`https://pytorch.org/tutorials/`)

Questions

Logistic regression assumptions satisfied

To begin, let's simulate data which satisfies the logistic regression assumptions, and fit both a logistic regression model and a neural network.

1. Simulate data $(\boldsymbol{X}_1, Y_1), ..., (\boldsymbol{X}_{1000}, Y_{1000})$ from the following model:

$$\begin{aligned} Y_i &\sim Bernoulli(p_i) \\ \log\left(\frac{p_i}{1-p_i}\right) &= -0.5 + 0.5X_{i,1} - 0.2X_{i,2} + X_{i,3} \\ (X_{i,1}, X_{i,2}, X_{i,3})^T &\overset{iid}{\sim} N(\mathbf{0}, \boldsymbol{I}) \end{aligned}$$

2. Using your simulated data from question 1, fit the following logistic regression model using the `glm` function in R, and report the estimated coefficients:

$$\begin{aligned} Y_i &\sim Bernoulli(p_i) \\ \log\left(\frac{p_i}{1-p_i}\right) &= \beta_0 + \beta_1 X_{i,1} + \beta_2 X_{i,2} + \beta_3 X_{i,3}. \end{aligned} \tag{1}$$

   (Because the model is correct, $\widehat{\boldsymbol{\beta}}$ should be close to $(-0.5, 0.5, -0.2, 1)^T$).

3. Now we want to fit the model with a neural network. Draw a network diagram (that is, draw the nodes in each layer of the network, and the connections between the nodes), showing how the logistic regression model from (1) can be represented as a feedforward neural network. Specify the input layer, output layer, the weights, any activation functions, and the loss function used in training.

4. Fit the neural network in question 3 (using software of your choice). Report the fitted weights and biases; these should be similar to the estimated coefficients from question 2.

5. What optimization method did you use to fit the network in question 4? Compare and contrast this optimization method with Fisher scoring.

6. Make a plot comparing the predicted probabilities $\widehat{p}_i$ from logistic regression to the true probabilities p_i for each point. Also plot the predicted probabilities from the neural network against the true probabilities p_i.

Logistic regression assumptions not satisfied

Now let's break the logistic regression assumptions!

7. Simulate data $(\boldsymbol{X}_1, Y_1), ..., (\boldsymbol{X}_{1000}, Y_{1000})$ from the following model:

$$\begin{aligned} Y_i &\sim Bernoulli(p_i) \\ \log\left(\frac{p_i}{1-p_i}\right) &= -0.5 + 0.2X_{i,1} + 0.1X_{i,1}^2 - 0.01X_{i,1}^3 - 0.2\sin(X_{i,2}) + \log(X_{i,3}^2) \\ (X_{i,1}, X_{i,2}, X_{i,3})^T &\overset{iid}{\sim} N(\mathbf{0}, \boldsymbol{I}) \end{aligned}$$

8. Using the simulated data from question 7, fit the logistic regression model from equation (1). (This logistic regression model is *not* the correct model). Plot the predicted probabilities $\widehat{p}_i$ from your fitted model against the true probabilities p_i. Does the model do a good job?

9. Now let's fit a neural network. But this time, add a hidden layer (hidden layers allow the network to better capture non-linearity). Draw a network diagram for your new network, and plot the predicted probabilities $\widehat{p}_i$ against the true probabilities p_i. Did adding the hidden layer improve your estimated probabilities?

10. Adding hidden layers makes the neural network more flexible. What are some downsides of making the network architecture more complex?

Working with real data

11. Choose a real dataset with a binary outcome (either a dataset we have previously used in 711/712, or another dataset you are interested in modeling). Briefly describe the data, the response variable, and the explanatory variables you will use.

12. Randomly split the data into a *training* set which contains 60% of the observations, and a *test* set which contains the remaining 40% of the observations. (We will fit our models on the training set, then evaluate their performance on the test set. This helps to avoid issues with overfitting).

13. Fit a linear regression model on the training data, then assess the performance of the fitted logistic regression model on the test data using an ROC curve. Report the area under the curve (AUC).

14. Fit a neural network on the training data (you may choose the architecture), then assess its performance on the test data using an ROC curve. Report the AUC, and compare with question 13. Which model performed better?