Data Mining Lab 6: Introduction to Neural Networks

1 Introduction

In this lab we are going to have a look at some very basic neural networks on a new data set which relates various covariates about cheese samples to a taste response.

If you need to refer to previous labs or to download the data set, they will be on the course labs website:

`http://www.maths.tcd.ie/~louis/DataMining/`

2 Cheese Data Set

2.1 Getting Setup

Exercise: Download the cheese data from the course website.
Exercise: Load the cheese data into a variable called `cheese` in your workspace. (Hint: see lab 2, §2.1 if you've forgotten)

```
>
```

Exercise: Load the `nnet` package, which contains the functions to build neural networks in `R`. (Hint: see lab 2, §3.1 if you've forgotten)

```
>
```

2.2 Explore the Data

Since we're looking at a new data set it's good practice to get a feel for what we're looking at.

```
> cheese
> names(cheese)
> summary(cheese)
```

These two commands tell us respectively the names of the variables and some summary statistics about the data contained therein.
Exercise: Do a box plot and a pairs plot of all the data. (Hint: see lab 1, §5.3 if you've forgotten)

```
>
>
```

Exercise: To reinforce the new programming learned in the last lab and lecture, adapt the for-loop code from lab 4, §2.2 (top of page 2) to do a grid of histograms for all 4 variables in the cheese data set.

```
> par(mfrow=c(2, 2))
>
```

2.3 Train, Validate and Test Subsets

You might expect that we would next split the data into cross-validation sets. However, one of the first things that probably struck you about the data when looking at it is how few observations there are (only 30). Consequently, it is questionable whether it would be sensible to split the data three ways to get train, validate and test sets: we would end up with so little data in each set that our analysis would lack any power.

In lectures you will see/have seen two possible options for assessing models without splitting the data: the Akaike Information Criterion (AIC) and Schwarz Bayesian Information Criterion (SBIC). Under the assumption of normally distributed errors, these can be written:

$$\text{AIC} = 2k + n \log\left(\tfrac{\text{SSE}}{n}\right)$$

$$\text{SBIC} = k \log n + n \log\left(\tfrac{\text{SSE}}{n}\right)$$

where k is the number of parameters being estimated, n is the number of observations in the full data set and $\text{SSE} = \sum \varepsilon_i^2 = \sum (\hat{y}_i - y_i)^2$. We will use these instead of train, test and validate splits for this small data set.

3 Neural Networks

3.1 Simple Neural Net, Linear Activation Fn, No Hidden Layer

3.1.1 Fitting the Neural Net

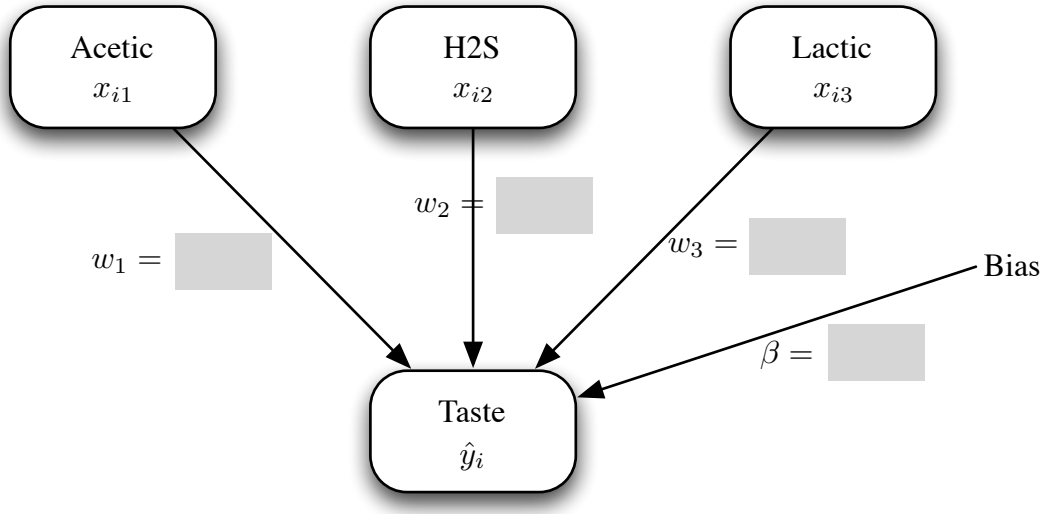
We are going to first fit the simplest possible neural network to the cheese data, to predict taste from acetic, H2S and lactic. This is the neural network with the input layer directly connected to the output.

```
> fitnn1 = nnet(taste ~ Acetic + H2S + Lactic, cheese, size=0,
                skip=TRUE, linout=TRUE)
> summary(fitnn1)
```

The `size` argument specifies how many nodes to have in the hidden layer, `skip` indicates that the input layer has a direct connection to the output layer and `linout` specifies the simple identity activation function.

The output from summary gives us the detail of the neural network. `i1`, `i2` and `i3` are the input nodes (so acetic, H2S and lactic respectively); `o` is the output node (so taste); and `b` is the bias.

Exercise: Fill in the shaded boxes on the neural network diagram below with the values which `R` has fitted.



$$\implies \hat{y}_i = \beta + w_1 x_{i1} + w_2 x_{i2} + w_3 x_{i3}$$

$$\implies \hat{y}_i =$$

We can compare this to the fit we get by doing standard least squares regression:

```
> lm(taste ~ Acetic + H2S + Lactic, cheese)
```

Exercise: Upto a reasonable level of accuracy, do you see any difference between the linear model and neural network fit?

3.1.2 Model Evaluation

We can now compute the AIC for this fitted model quite simply:

```
> SSE1 = sum(fitnn1$residuals^2)
> AIC1 = 2*5 + 30*log(SSE1/30)
> AIC1
```

Exercise: Using the SSE already calculated in `SSE1`, compute the SBIC and store it in a variable called `SBIC1`.

```
>
>
```

A scientist working on the project may believe there is reason to query whether acetic is a good linear predictor for taste. To test this conjecture, we can refit a linear model without acetic in and compare the AIC and SBIC of the two models.

```
> fitnn2 = nnet(taste ~ H2S + Lactic, cheese, size=0, skip=TRUE,
                linout=TRUE)
> summary(fitnn2)
> SSE2 = sum(fitnn2$residuals^2)
> AIC2 = 2*4 + 30*log(SSE2/30)
> AIC2
> AIC2 < AIC1
```

Exercise: On the basis of the above analysis, what is your advice to the scientist? Should the model with or without acetic be preferred?

3.2 Adding a Hidden Layer With a Single Node

We now add a hidden layer in between the input and output neurons, with a single node.

```
> fitnn3 = nnet(taste ~ Acetic + H2S + Lactic, cheese, size=1,
                linout=TRUE)
```

You may (or may not) see output with numbers of similar magnitude to the following:

```
# weights:  6
initial  value 25372.359476
final  value 7662.886667
converged
```

The numbers being printed here are the SSE $= \sum \varepsilon_i^2 = \sum (\hat{y}_i - y_i)^2$ for the fit on the current iteration of the neural network fitting algorithm. If you look back to the previous `nnet()` calls, you should see that the final SSE ≈ 2700. It seems very strange that the SSE has increased so dramatically when we are *increasing* the model complexity! If you keep rerunning the last command, you should find that sometimes you get SSE < 2700 and other times not (try this until you see it change). This is because the algorithm chooses random starting weights for the neural net and for some choices is getting stuck in a local minima. The writers of `nnet()` have designed the algorithm to be more robust to this kind of issue when scaled data is used ($\bar{x}_i = 0, s_{x_i}^2 = 1$).

So, we'll scale the data and refit:

```
> cheese2 = scale(cheese)
> fitnn3 = nnet(taste ~ Acetic + H2S + Lactic, cheese2, size=1,
                linout=TRUE)
> summary(fitnn3)
```

Aside: *Note that we could not now compare an AIC/SBIC from this model to the earlier ones because the data were on different scales which affects the SSE. We would have to refit the earlier model on the scaled data to do a comparison.*

Exercise: Use the space below to draw the full neural network we have fitted. Make sure you show all appropriate connections; all the weights for every connection; and write the full model equation after the diagram.