ECE-9603A Assignment 2: Neural Networks

Joe Abley – jabley@uwo.ca 2018-11-07

Contents

Introduction

This document is a submission for Assignment 2, course ECE-9603A, Fall 2018, Western University Faculty of Engineering, Department of Electrical and Computer Engineering. It has been written in R Markdown[1]; the code used to produce the output is included in the document source[2].

In order to provide a coherent treatment of the core subject matter of the assignment, that of the construction and tuning of a neutral network to address a forecasting problem, much of the R code used to build the model and execute it has been suppressed from the main body of the text; the intention is a more language-neutral treatment which is sufficient (for example) to produce a clean-room implementation in some other language. The document source should be consulted to review the code.

Forecasting Problem

Given a dataset that describes various features of individual houses along with details of their sale, identify a forecasting model that can predict the prices realised by the sale of other houses based on their specific features.

The practical applications of this model include

- assessing whether the asking price of a house is reasonable, as a buyer;
- deciding at what price it is reaasonable to list a house, as a seller;
- deciding whether or not particular renovations might be reflected in improvements in sale price, as an owner.

In the previous assignment significant work was done with the same source data to provide a model for the forecasting problem. That work included selection and scaling of individual features from the source data

[1] https://rmarkdown.rstudio.com
[2] https://github.com/ableyjoe/uwo-mesc/tree/master/ECE-9603A-001-GF18/assignment2

and elimination of significant outliers to produce a training data set for a variety of algorithms. The output of each model was compared through cross-validation using a test data set.

This assignment will follow a similar process using multi-layer neural networks. While the focus of the previous assignment was feature analysis and data preparation, this assignment will concentrate on finding an appropriate network architecture and tuning it best address the forecasting problem. The feature analysis carried out on the source data in assignment 1 will be used as the starting point in this assignment; however, scaling and transformation of individual inputs will be considered separately and we will attempt to produce models without discarding outliers.

Data Preparation

Review of Features Isolated in Assignment 1

The input variables carried over from assignment 1 were as follows:

Input Variable	Description
`newBathrooms`	Total number of bathrooms of all kinds (derived)
`LotArea`	Lot size in square feet
`TotalBsmtSF`	Total square feet of basement area
`GrLivArea`	Above grade (ground) living area square feet
`TotRmsAbvGrd`	Total rooms above grade (does not include bathrooms)
`Fireplaces`	Number of fireplaces
`GarageArea`	Size of garage in square feet

The data set includes 1460 observations, each with an output variable `SalePrice`. From this source data we extract 60% of observations into a training set (876), 20% into a cross-validation set (292) and the last 20% into a test set (292).

Normalisation

As might be expected, each of the input variables (and the output variables) have dramatically different scales due to the units they represent, ranging from sale price dollar values in the hundreds of thousands to the number of bathrooms which are small integers. Each of these input variables x is normalised using min-max scaling to a fixed range $0 \leq x_{norm} \leq 1$ using

$$x_{norm} = \frac{x - x_{min}}{x_{max} - x_{min}}$$

No other transformations of input variables is carried out.

Design of Model

Network Architectures

We need a network architecture that is useful to act as a forecasting model; that is, one that is suitable for use in a regression problem. Although we might imagine other similar forecasting problems that have a time base, the data set in this case is static and contains no time information, a consequence of both the limited source data considered in assignment 1 and the feature reduction carried out at that time.

Since the problem at hand does not involve classification or feature extraction, the use of a convolutional neural network does not seem appropriate. The lack of a time series for the source data does not suggest a recurrant neural network; the lack of likely structure in the source data does not indicate a good application for a recursive neural network. The forecasting problem at hand requires a supervised learning model, and we have training, cross-validation and test datasets available to produce one.

A feedforward neural network, although simple, seems like a plausible basis to construct a forecasting model. We know from simple observation and experimentation with data transforms in assignment 1 that at least two of the variables in the observation set may well be non-linear, and hence we will build multi-layer networks of varying depths and attempt to use the cross-validation data set to identify the best network topology without overfitting. The best model identified using that methodology will then be tested against the test set.

Parameters

Networks are constructed using the default activation functions and learning algorithms provided by the *neuralnet* library in R.

Network Topology

The network topology will include an input layer with seven nodes, corresponding to the seven normalised input variables in the data set, and an output layer with one node, corresponding to the normalised output variable `SalePrice`. We will add some number of hidden layers each containing some number of neurons, and we expect some of these topologies to yield better results than others.

Whilst some work has been done to build genetic algorithms for selection of the number of hidden layers and neurons in a (deep) feedforward neural network, common practice appears to be based on various rules of thumb. A commonly-held guideline is that in a neutral network with N_i input neurons, N_o output neurons, N_s samples in the training set, it is often found that the upper bound of the number of hidden neurons to avoid overfitting is of the form

$$N_h = \frac{N_s}{\alpha(N_i + N_o)}$$

where α is some arbitrary scaling factor, often chosen such that $2 \leq \alpha \leq 10$. For our model we have $N_i + N_o = 8$; with $N_s = 876$ and choosing $\alpha = 2$ to give the largest advisable number of hidden neurons we obtain $N_h = 54.75$.

The number of possible arrangements of up to 54 neurons in an unconstrained number of hidden layers is unhelpfully large. For the purposes of choosing a manageable number of meaningfully-different topologies to compare, we shall arbitrarily select six simple topologies of between one and six hidden layers, each containing between four and eight nodes.

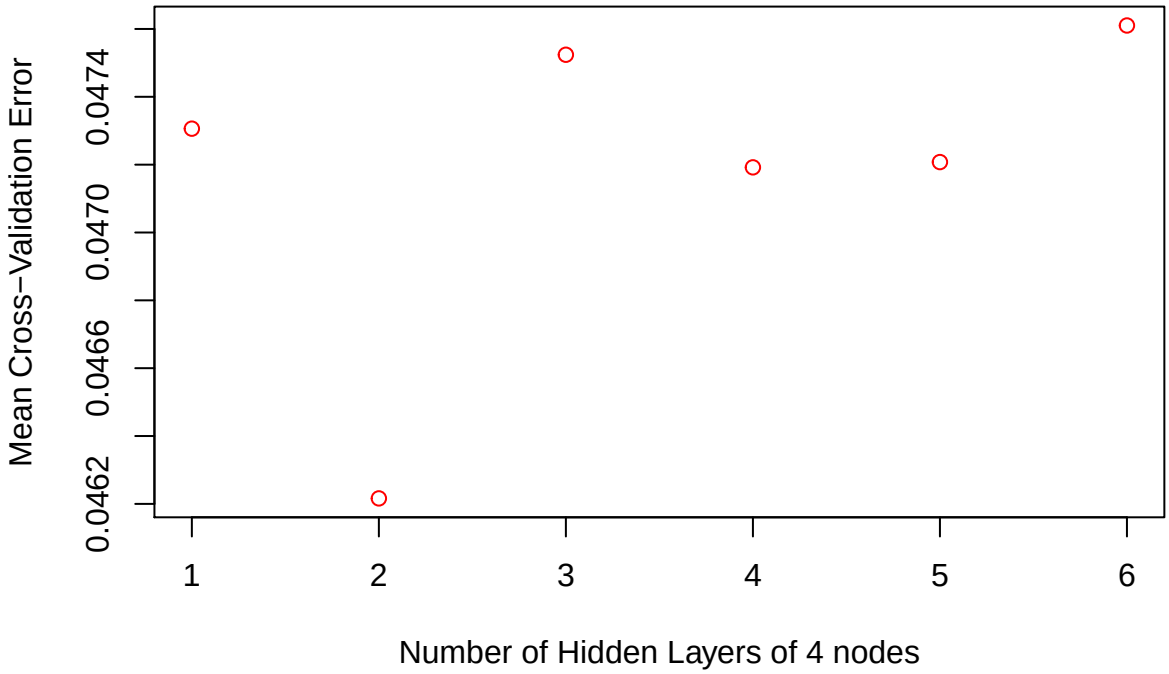
Validation of Each Topology

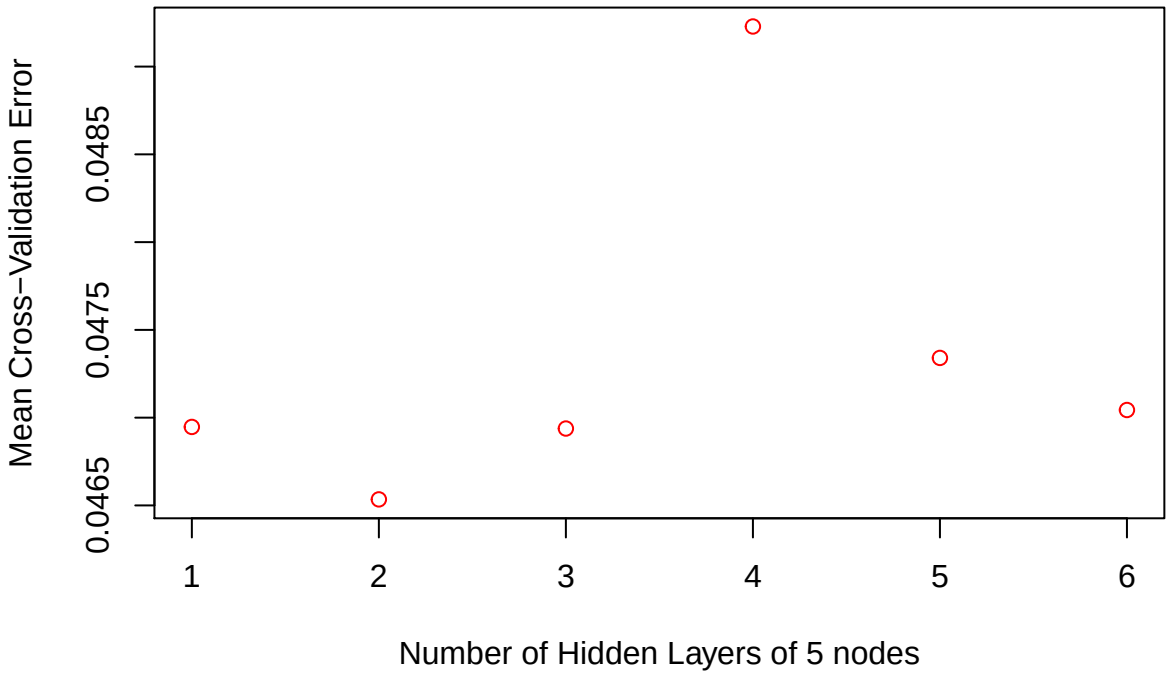
For $1 \leq h \leq 6$, $5 \leq n \leq 10$ and $1 \leq i \leq 10$ we create feedforward neural network $N_{h,n,i}$ with h hidden layers of n nodes and train it using the same training set of 876 observations. Multiple networks are consequently created for each h, since each neural network is initialised with random weights on each edge.

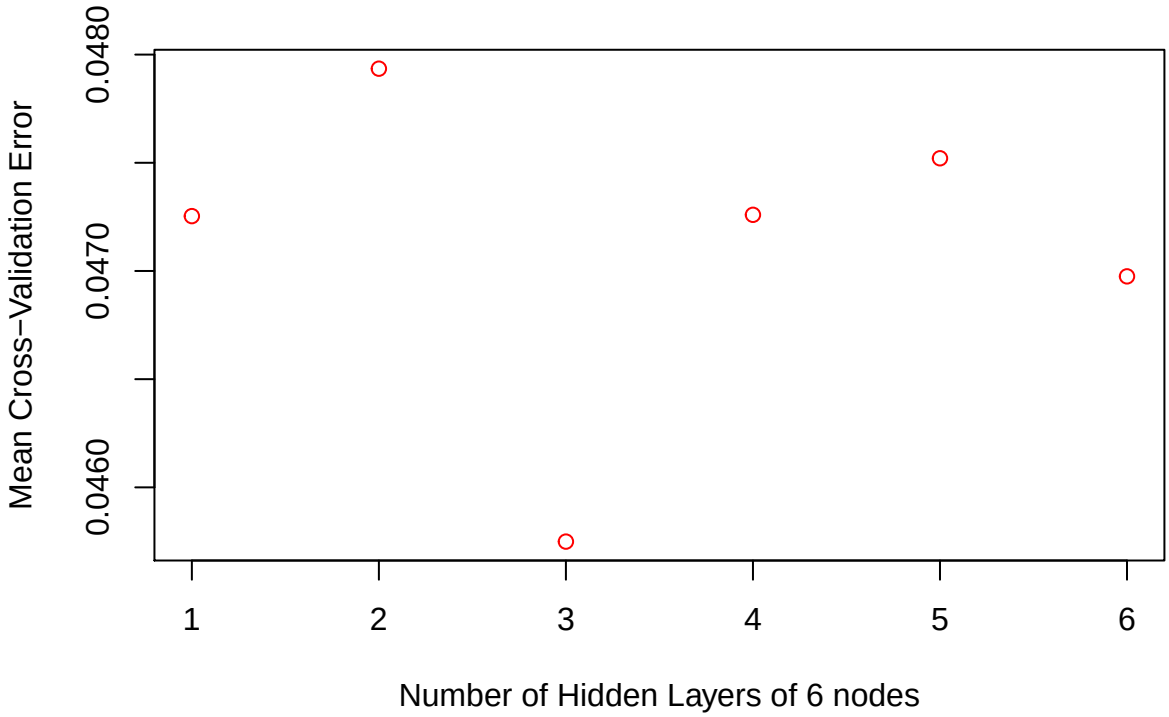
The accuracy of each $N_{h,n,i}$ is calculated by using it to predict the sale price for of the 292 observations in the validation set and calculating $E_{h,i}$, the root mean square difference between the predicted sales price and the actual sales price, as obtained from the known actual value (`SalePrice`). The mean error E_h is then calculated $\forall i$, and we plot h vs. E_h to observe the effect of different values of h on the mean error observed in the model.

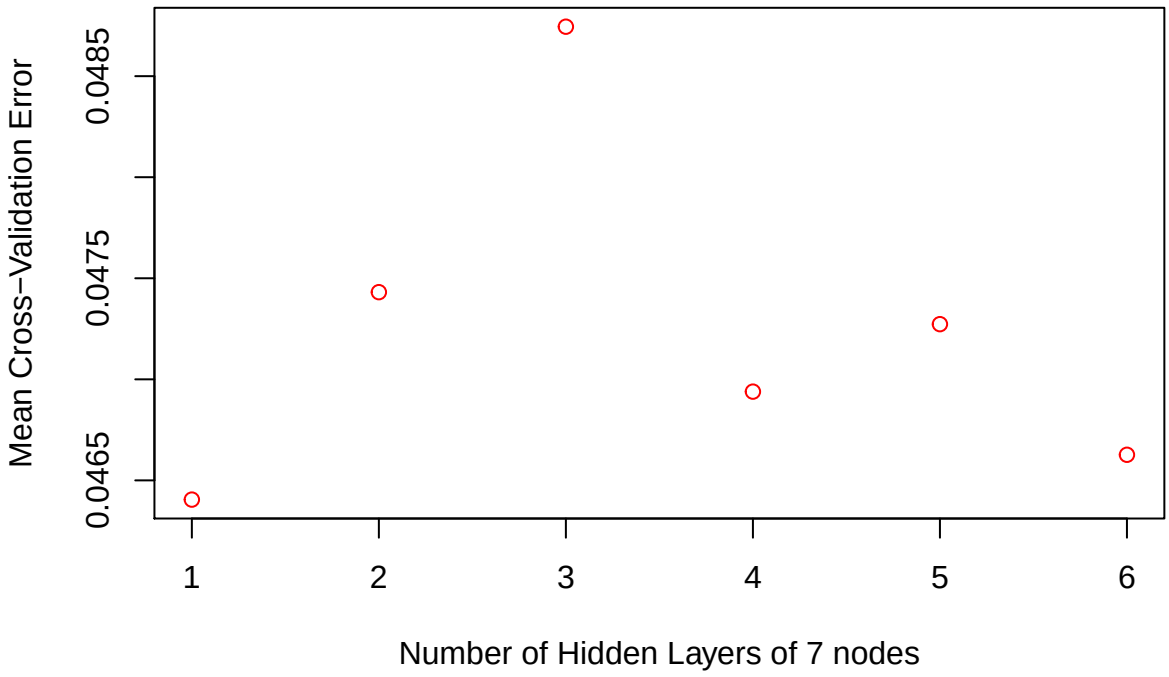
Results

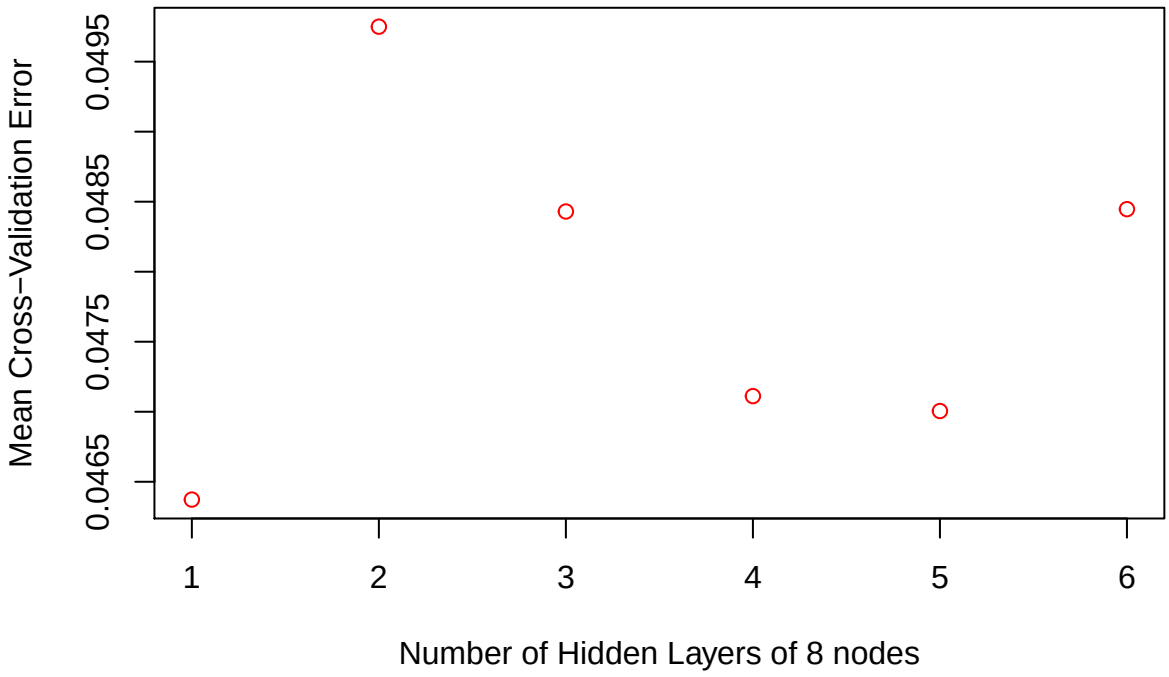
The following graphs show the mean root mean square error (measured against actual values in the validation set) for networks with different numbers of hidden layers and nodes per hidden layer.






Mean Cross−Validation Error
0.0460
0.0470
0.0480
1
2
3
4
5
6
Number of Hidden Layers of 6 nodes


Mean Cross−Validation Error
0.0465
0.0475
0.0485
1
2
3
4
5
6
Number of Hidden Layers of 7 nodes


Mean Cross−Validation Error
0.0465
0.0475
0.0485
0.0495
1
2
3
4
5
6
Number of Hidden Layers of 8 nodes