

Neural Networks, with Demo

Data Mining
Prof. Dawn Woodard
School of ORIE
Cornell University

Announcements

- Reading: Chap. 11 in SPB

Neural Networks

What if you don't like the fact that the prediction \hat{Y} in regression trees is a discontinuous function of the predictors?

Neural Networks (NNs):

- A method for predicting a continuous or categorical outcome, given a vector of continuous or categorical predictors
- Allows interactions and nonlinearities, like Classification and Regression Trees.

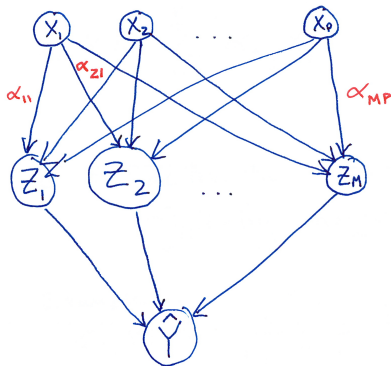



Neural Networks

- NNs are named after neurons in the brain, implying that they mimic human learning
- Have been very successful for:
 - predicting bankruptcy
 - detecting fraud (e.g. credit card)
 - autonomous driving at highway speeds
 - handwriting recognition

Neural Networks

Diagram of a “feed-forward” neural net with a single “hidden layer”, for a continuous outcome variable Y :



Inputs 

Hidden Layer

Output

The arrows indicate that the hidden layer nodes are defined as functions of the inputs, and the output is defined to be a function of the hidden layer nodes.

Neural Networks

Case of a categorical outcome with $K > 2$ categories <sketch>:

I.e., have K output nodes called Y_1, \dots, Y_K . They will be real-valued; to predict the category we will take the value of k for which Y_k is largest.

Neural Networks

- For the case of a categorical outcome with 2 categories, we use a single output node Y_1 and predict $Y = 1$ if Y_1 is above some threshold



- Let X_1, \dots, X_p indicate the continuous or binary (0/1) predictor variables (next time we will discuss how to handle an unordered categorical predictor with > 2 categories in linear regression/regression trees/neural nets). These are called the **“inputs”**

Neural Networks

The nodes in the **hidden layer** are:

$$Z_m \equiv \sigma(\alpha_{m0} + \sum_{j=1}^p \alpha_{mj} X_j) \quad m = 1, \dots, M$$

Here, α_{mj} for $m \in \{1, \dots, M\}$ and $j \in \{0, \dots, p\}$ are unknown parameters

σ is the known (specified) “activation function”, which is typically nonlinear

Neural Networks

activation function examples:

1. Logistic function:

<Draw>

2. Gaussian (“radial basis”) function:

Neural Networks

Output layer:

$$Y_k \equiv \tilde{\sigma}(\beta_{k0} + \sum_{m=1}^M \beta_{km} Z_m)$$

where $\tilde{\sigma}$ is another activation function and $\beta_{km} : k \in \{1, \dots, K\}, m \in \{0, \dots, M\}$ are more unknown parameters.

By adding up nonlinear functions of X_1, \dots, X_p and applying another transformation, we have created an extremely flexible model!

Neural Networks

Common choices of $\tilde{\sigma}$ include:



Neural Networks

Neural nets are more flexible models than multiple linear regression and logistic regression models (if no interaction / higher-order polynomial terms are used in those models), because:



Small Cheese Example

Small data set on cheese tasting:

Table 9.1: Tiny Example on Tasting Scores for Six Consumers and Two Predictors

<i>Obs.</i>	<i>Fat Score</i>	<i>Salt Score</i>	<i>Acceptance</i>
1	0.2	0.9	1
2	0.1	0.1	0
3	0.2	0.4	0
4	0.2	0.5	0
5	0.4	0.5	1
6	0.3	0.8	1

Example is from Shmueli et al. 2007

Small Cheese Example

- Predictor vars are the amount of fat and salt in the cheese sample, and response is whether or not the taster liked the cheese
- Since the outcome is binary, use the logistic function as the activation functions σ and $\tilde{\sigma}$

Small Cheese Example

A neural network for the cheese data, with arbitrary parameter values:

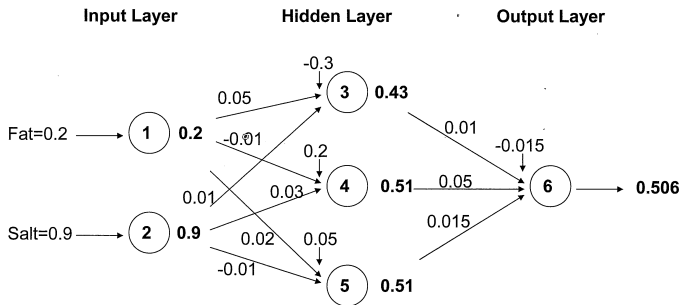


Figure 9.3: Computing node outputs (in boldface type) using the first observation in the tiny example and a logistic function.

Small Cheese Example

Calculations are shown for the first data point:

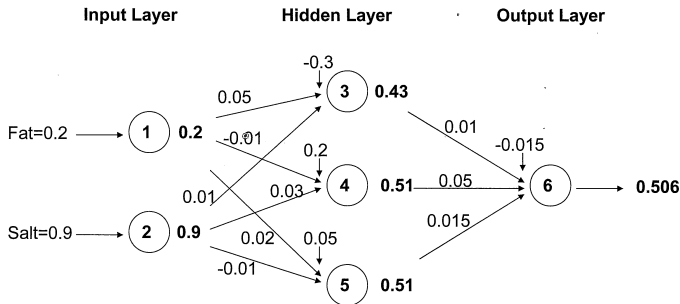


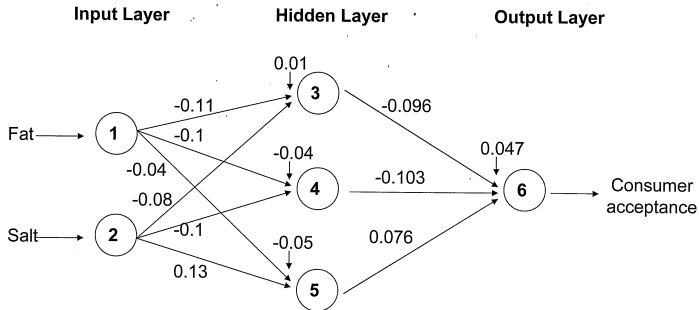
Figure 9.3: Computing node outputs (in boldface type) using the first observation in the tiny example and a logistic function.

Small Cheese Example

- Arbitrary values for the coefficients α and β are shown in the graph
- The value of the node labeled “3” is calculated as:
- Similarly for the other nodes
- The output is compared to the threshold .5, yielding a prediction of $\hat{Y} =$

Small Cheese Example

A trained neural network for the cheese data:



(Next time we'll discuss how the training is done)

Demo on Iris Data

- Goal: predict iris species based on
 - petal length
 - petal width
 - sepal length
 - sepal width
- Data: “iris” data in R. Call e.g. “iris[1:10,]” to view the first 10 observations.
- The variable iris\$Species gives the species. Find the names of the three species.

Demo on Iris Data

- install & load the “nnet” package, which implements:
- Choose a training set with half the observations:
- Fit a neural network with 1 hidden layer having 2 nodes:
`iris.net = nnet(Species ~ ., data = iris[samp,], size = 2,
rang = 0.1, decay = 5e-4, maxit = 200)`
 - Here, Species is a factor variable with 3 categories, so the output nodes are:

■ **size** specifies:



■ **rang** and **decay** are:



■ **maxit** gives:



Demo on Iris Data

- Apply your network to predict on the test data:

```
iris.preds = predict( object = iris.net, newdata = iris[-samp,], type =  
"class")
```

- Here, **type = "class"** says that



- Create a table of predicted vs. actual outcome values:

iClicker: How many misclassifications did you get?

- A. 0-10
- B. 11-20
- C. 21-30
- D. 31+

Demo on Iris Data

- The parameters α and β are output in **iris.net\$wts**. Are there the correct # parameters?
- iClicker: how many parameters?
 - A. 0-10
 - B. 11-20
 - C. 21-30
 - D. 31+

Categorical predictors

- Consider a predictor variable that has $Q > 2$ categories that have no natural ordering
- It is not correct to code these categories as 1, 2, 3, etc. and treat the predictor as continuous. *WHY?*
- The results of linear regression, regression trees, neural nets etc. would depend on the arbitrary order that we picked for the categories. Change order, get different predictions!

Categorical predictors

For linear regression or neural nets, first replace the categorical variable with $Q - 1$ binary variables. E.g., say we have categories $x_1 \in \{\text{married, single, divorced}\}$ (marital status).

- replace x_1 variable with

$$x^* = \mathbf{1}_{(x_1 = \text{"single"})}$$

$$x^{**} = \mathbf{1}_{(x_1 = \text{"divorced"})}$$

- then fit linear regression model / neural net as usual.
- For the linear regression case, there are now 2 regression coefficients associated with the variable "marital status" (β_*, β_{**})

Categorical predictors

Interpretation of coefficients in the linear regression case:

- 
-

Categorical predictors

- If the categorical predictor is coded as a factor variable in **R**, then when you fit a linear regression model using the `lm` function, **R** will automatically replace with $Q - 1$ binary variables when fitting the model.



If coded as a numeric variable will get wrong results!

- For the `nnet` function:

Categorical predictors

- Why don't we need $x^{***} = \mathbf{1}_{(x_1 = \text{"married"})}$?
 - all information from x_1 is already in x^* and x^{**} .
 - Including x^{***} would result in numerical singularities when calculating $\hat{\beta}$ in the linear regression model, for instance.

Categorical predictors

Categorical predictors in regression / classification trees:

