Topic 4. Neural Networks

Case 3: Donor Recapture

using Transaction, Overlay, and Census Data

Reading Assignment

Berry and Linoff (2000)

- Pages 112–128. Neural networks (reviews).

The Plan

1. Review and augment the previous discussion of neural nets.

2. Use boosting to combine tools.

3. Fit nets to the donor data.

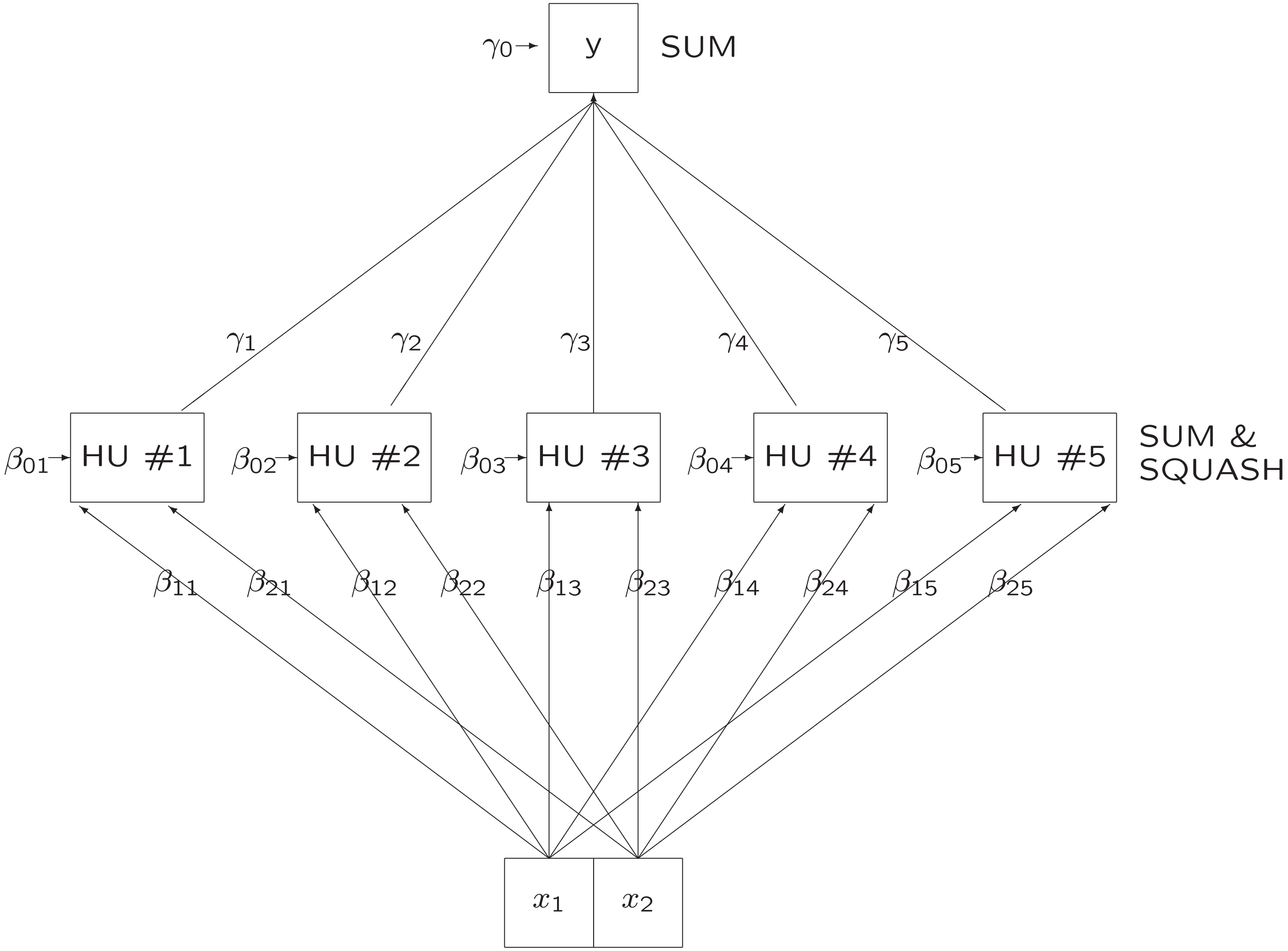
4. Analyze results.

5. Compare to regression results.

Neural Nets

Let’s have another look at their diagrammatic and mathematical representations . . .

Fig 56. Single Hidden Layer Neural Net, Five Hidden Units



What the Diagram Represents

Boxes are neurons and lines are dendrites.

The two boxes at the lowest level represent sensory neurons. They send signals of varying strength to the neurons above them. Signal strength is represented by the β_{ij}.

Each of these second level neurons additively combine the weighted signals, adding to them a bias represented by the β_{0j} . If the sum exceeds a threshold, it is passed on to the next higher level with varying strengths represented by the γ_j.

The top neuron additively combines these weighted signals, adding a bias γ_0. This sum may or may not be thresholded.

Shown is a single hidden layer feed forward neural net. One can have more hidden layers, feedback, etc. But for data analysis it can be proved that a single hidden layer feed forward net is adequate.

Single Hidden Layer Neural Net, Five Hidden Units

Mathematical Representation:

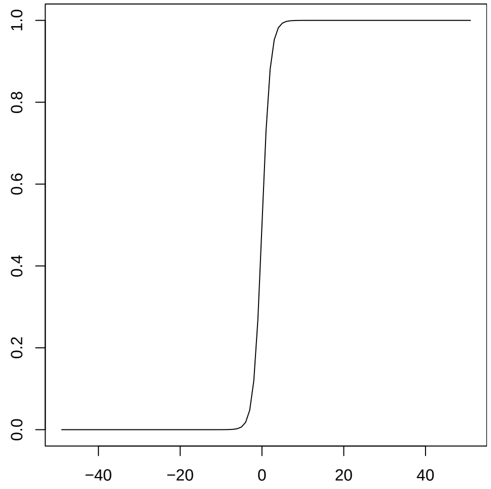
$$y = \gamma_0 + \sum_{j=1}^{5} \gamma_j \, S\big(\beta_{0j} + \beta_{1j} x_{1j} + \beta_{2j} x_{2j}\big)$$

Weights:

$$\gamma_0,\, \gamma_1,\, \beta_{01},\, \beta_{11},\, \beta_{21},\, \ldots, \gamma_5,\, \beta_{05},\, \beta_{15},\, \beta_{25}$$

Squasher:

$$S(x) = \frac{\exp(x)}{1 + \exp(x)}$$



What the Mathematics Represents

The mathematical representation shows the summation and thresholding.

The threshold function is called a squasher in the neural net literature and is usually chosen to be a differentiable function as shown in the slide.

From a statistical perspective, a neural net can be viewed as a nonlinear regression that can fit by least squares using standard optimization algorithms.

They are very difficult to fit!

What the Mathematics Reveals

The squasher expects numeric inputs, not categorical inputs.

This defeats many implementations of neural nets and most menu driven software.

But it shall not defeat us!

Table 9. Features Available to Net

File	Feature	Type	Dummies Required
464	LASTGIFT	num	
75	PEPSTRFL	chr	1
4	STATE	chr	31
11	RECP3	chr	1
8	DOB	num	
6	MAILCODE	chr	1
359	MHUC2	num	
465	LASTDATE	num	
460	MINRAMNT	num	

The Categorical Features Problem

Table 9 contains categorical features whereas, as the math reveals, neural networks expect numerical variables.

One can convert the categorical variables to dummy variables, which are numerical, and then apply a neural net.

This doesn't work well unless weights are hand coded to make the squasher effectively pass the dummy through untouched.

In our case hand coding is not a pleasant prospect because it is tedious and error prone, especially with as many dummies as we have.

The Solution to the Categorical Features Problem

Fit a dictionary model that is the sum of a linear regression in the dummies and a neural net in the quantitative variables.

I.e. fit a dictionary model of the following form

$$\begin{aligned} y &= \text{Dummies}(\textsc{mailcode}, \textsc{pepstrfl}, \textsc{state}, \textsc{recp3}) \\ &+ \text{Net}(\textsc{lastgift}, \textsc{dob}, \textsc{mhuc2}, \textsc{lastdate}, \textsc{minramnt}) \end{aligned}$$

Implementation

Although a few implementations allow the model to be fit as posed —

$$
\begin{aligned}
y &= \text{Dummies}(\textsc{mailcode}, \textsc{pepstrfl}, \textsc{state}, \textsc{recp3}) \\
&+ \text{Net}(\textsc{lastgift}, \textsc{dob}, \textsc{mhuc2}, \textsc{lastdate}, \textsc{minramnt})
\end{aligned}
$$

— we shall fit it using a boosting technique that is a useful idea in general for two reasons:

- It allows dictionary models to be built from different tools.

- It allows one to trade computer time for computer memory.

Boosting Strategy: The Idea

Initialize by putting the predictions to zero and the residuals to the target.

Fit a model to the residuals to get new predictions and new residuals. Add the new predictions to the previous predictions. Replace the previous residuals with the new residuals.

Repeat until MSE quits changing.

The tools do not need to be the same at each step, which allows tools to be combined.

Boosting Strategy: The Algorithm

Step 0 Set $\widehat{y} = 0$

Step 1 Fit

$$y - \widehat{y} = \text{Net}(\textsc{lastgift, dob, mhuc2, lastdate, minramnt})$$

to get predicted values $\widehat{y}_{(1)}$ and replace $\widehat{y}$ with $\widehat{y}_{\text{new}} = \widehat{y} + \widehat{y}_{(1)}$

Step 2 Fit

$$y - \widehat{y} = \text{Dummies}(\textsc{mailcode, pepstrfl, state, recp3})$$

to get predicted values $\widehat{y}_{(2)}$ and replace $\widehat{y}$ with $\widehat{y}_{\text{new}} = \widehat{y} + \widehat{y}_{(2)}$

Step n Repeat Steps 1 and 2 until the validation MSE stabilizes.

Notice That

The fitting is done in the learning sample.

The MSE is computed in the validation sample.

Net Fitting Strategy

The most popular fitting strategy for nets is back propagation, which is a sequential steepest descent algorithm known as Robbins-Monroe in the statistical literature.

In my experience, back propagation does not work well.

Much better is to use a standard nonlinear optimization algorithm such as BFGS (Broyden-Fletcher-Goldfarb-Shanno)

with

numerous (hundreds or thousands) of random starts over balls of increasing radius.

The Implementation

Implementing boosting with random starts over concentric balls for the nonlinear optimization requires the looping and control structures of a scripting language like R — this strategy is beyond the reach of menu driven software.

Neural Net Fit: Results

charity/nnet/cty_net_05.r.Rout

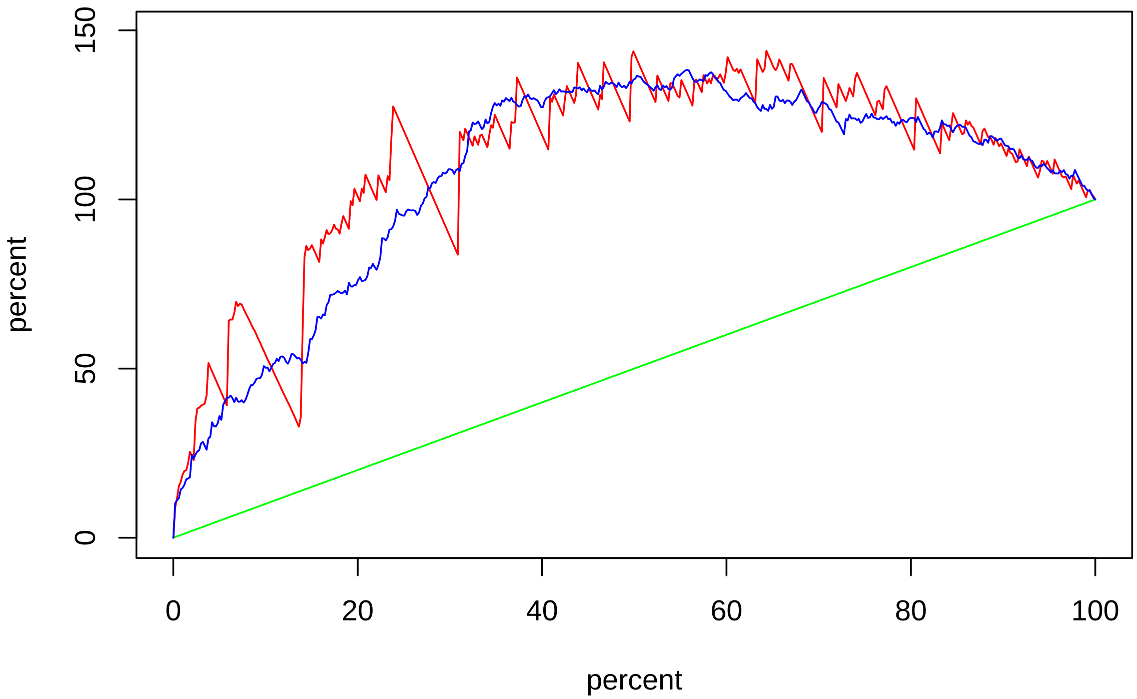
```
iter     =  1.1
mse.lrn =  20.0726176720395
mse.val =  18.7985310354253
mse.tst =  17.8527745362471

iter     =  1.2
mse.lrn =  20.026999125388
mse.val =  18.7681067564899
mse.tst =  17.8314805001506
.
.
.
iter     =  6.1
mse.lrn =  19.9773126090124
mse.val =  18.7259439837969
mse.tst =  17.8525822478034

iter     =  6.2
mse.lrn =  19.9773084292726
mse.val =  18.7259454732097
mse.tst =  17.8527514363572
```

Analysis of Results

As with linear regression, we shall summarize results with lift charts and mean squared error performance measures.

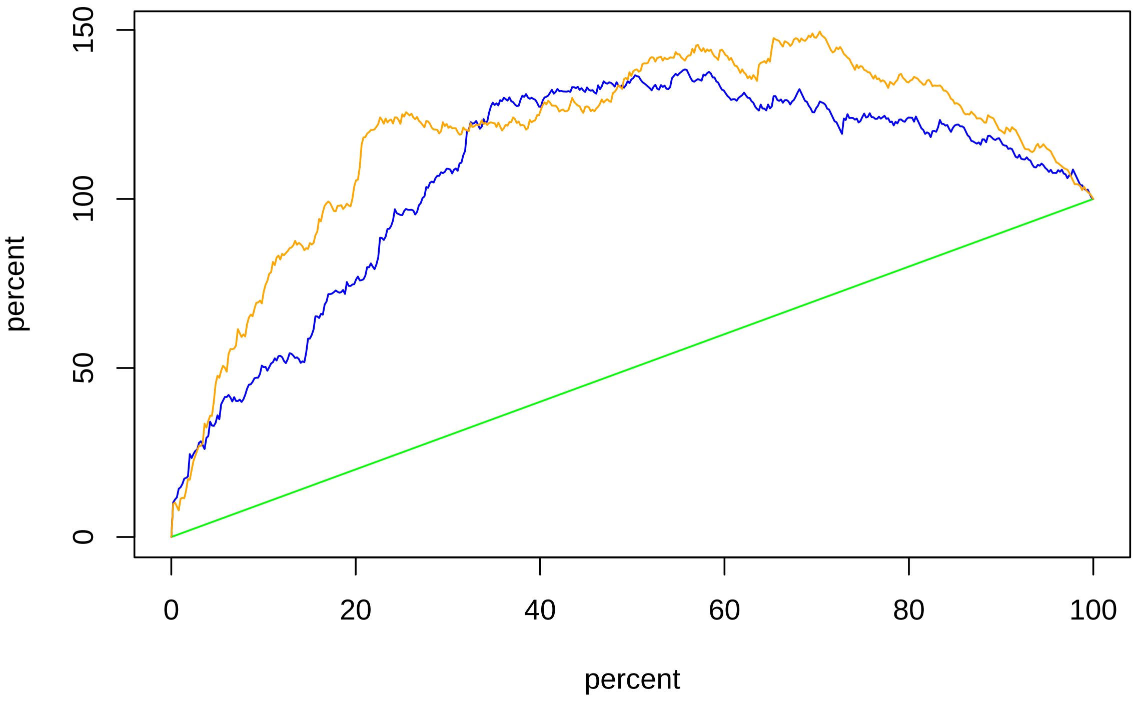
Nets with 2 and 10 hidden units were also tried. But they did not do as well as the 5 hidden unit nets and are therefore dismissed from consideration.

Fig 57. Lift Charts



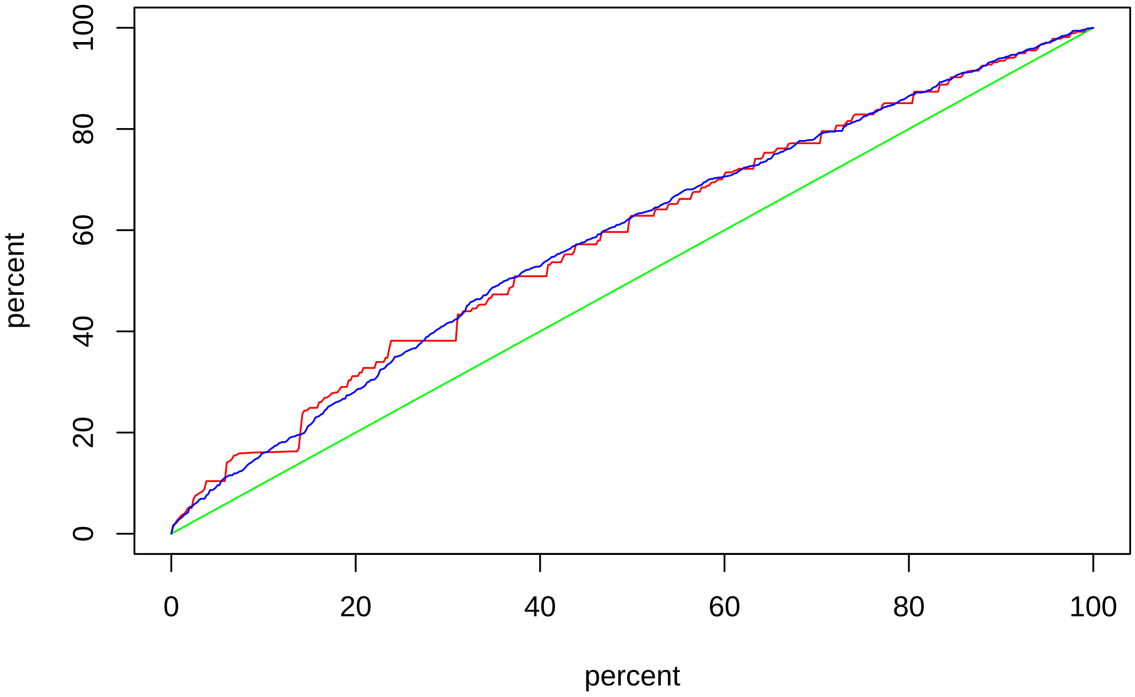
The green curve shows net revenue if persons in the learning sample were mailed solicitations in random order. The red curve shows net revenue in the learning sample if persons are sorted by their predicted gift and mailed solicitations in sorted order, highest first; blue is the same for the validation sample. The plots are normalized so endpoints plot at (100,100). Net revenue is the gift less a mailing cost of $0.68.

Fig 58. Lift Charts



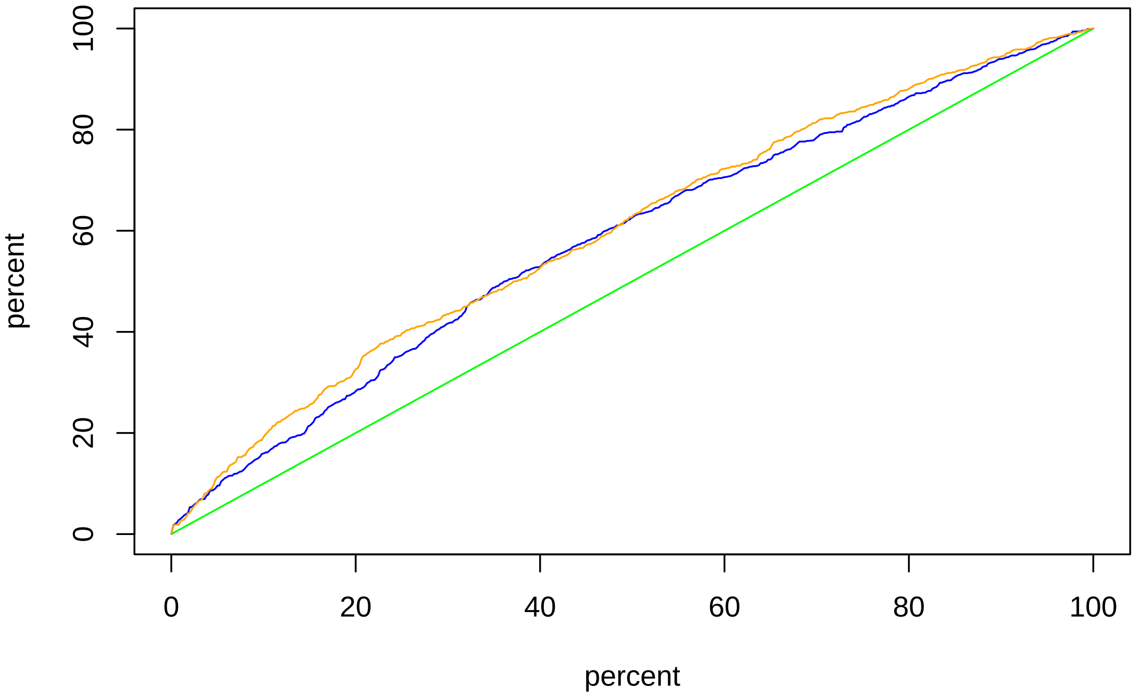
Same as Fig 57 except that the orange line is the blue line from Fig 54, which shows the lift of the regression model in the validation sample.

Fig 59. Conventional Lift Charts



Same as Fig 57 but gross revenue instead of net revenue.

Fig 60. Conventional Lift Charts



Same as Fig 59 except that the orange line is the blue line from Fig 55, which shows the lift from the regression model in the validation sample.

Comments on Lift Charts

The regression model and the neural net model certainly are different.

The regression model is predicting well in the validation sample up to about 40%.

The neural net model is making some horrid mistakes prior to 40%.

The neural net model is doing better than the regression model from 40% onward.

How do they compare on MSE? Next slide.

Table 10. Performance Measures

		Mean Squared Error		
Model	Specification	Learning	Validation	Test
Mean	learning sample	20.09922	18.82322	17.86605
Regr	selected model	19.96083	18.67709	17.80003
Nnet	Iter 1.1	20.07262	18.79853	17.85277
Nnet	Iter 1.2	20.02700	18.76811	17.83148
Nnet	Iter 6.1	19.97731	18.72594	17.85258

Note in Passing

Also, a purist would insist that we not be allowed to look at results in the test sample at this point in the analysis.

To the purest, that should be done only once at the end of the analysis as the final comparison of all models fitted.

My response is that I'm not going waste your time and mine to go over these tables now with one column less and then later with that column replaced to satisfy some picky purest.

Neural Networks Main Points

1. Neural nets as typically implemented cannot handle categorical features.

2. Boosting can be used to resolve this difficulty and, indeed, make a dictionary method out of any combination of tools.

3. Nets did not beat regression in this application, but might with more tinkering.

4. Balancing MSE in learning, validation, and test samples is a bad idea.