SAND2020-4639PE

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

Looking into NN "Interpretation" Methods

Feature Visualization

Katherine Goode

2020-04-02

NN Interpretation Methods

Started reading about "interpretation" methods for neural networks

Methods

- *Feature visualization* [1, 2] (focusing on this method in these slides)
- Network dissection [1]
- Testing with Concept Activation Vectors (TCAV) [3]

Note

- Implemented in practice with image data and convolutional neural networks
- Could apply to tabular and text data (with modifications?)

[1] Interpretable Machine Learning - Molnar 2019 - book available here

[2] Feature Visualization - Olah, Mordvintsev, and Schubert (all at Google) 2017 - article available here

[3] Interpretability Beyond Feature Attribution: Quantitative Testing with Concept Activation Vectors - Kim et. al. 2018 (includes Viegas ans Wattenberg) - paper available here]

Overview of Slides

1. Background on feature visualization
2. Simple Example of Feature Visualization with Tabular Data (iris data)
3. HE Simulated Data
4. Feature Visualization with HE Simulated Data (first attempt)
5. Feature Visualization with HE Simulated Data (PC Features)
6. Feature Visualization with HE Simulated Data (Adding Layers)
7. Feature Visualization with HE Simulated Data (Adding Responses)

Feature Visualization

Goal

Visualize the derived features from the neural network at specific locations in the network (neuron, layer, etc.)

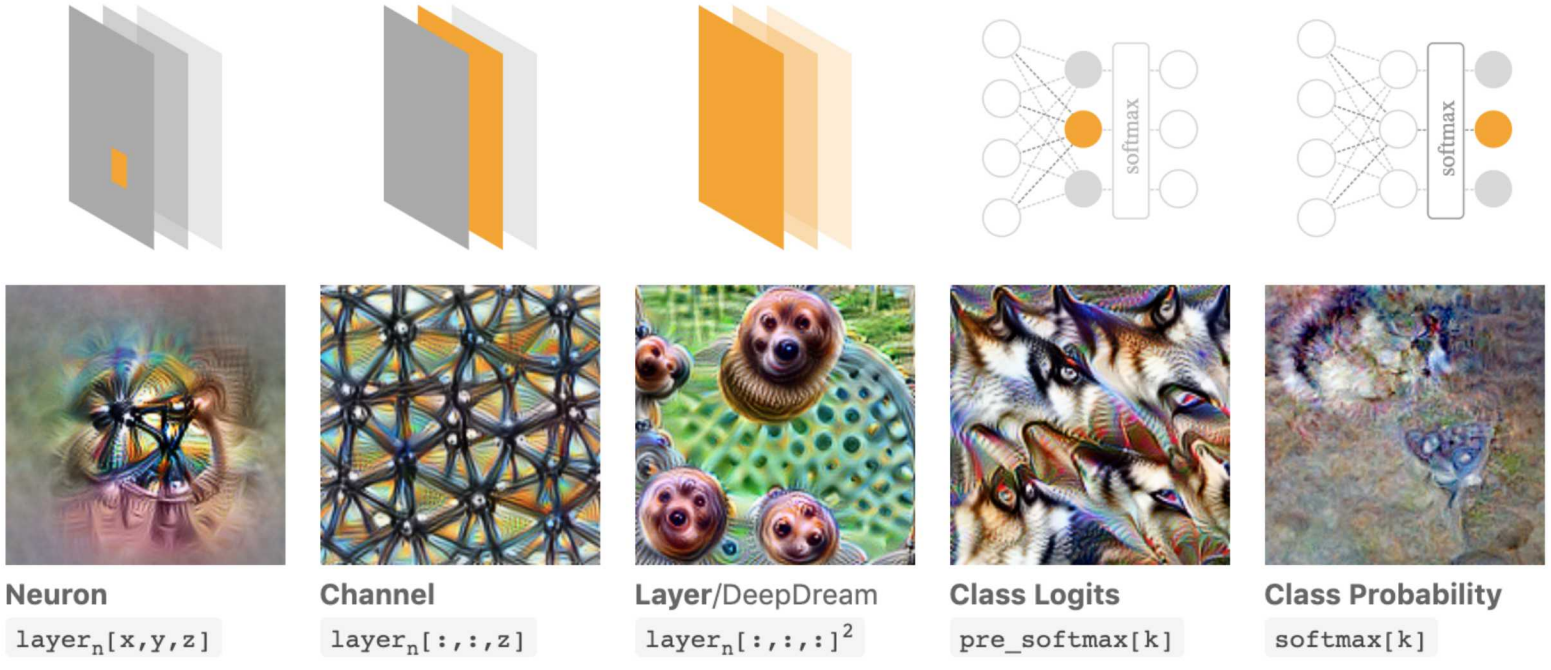


Image from https://distill.pub/2017/feature-visualization/

Implementation

Idea

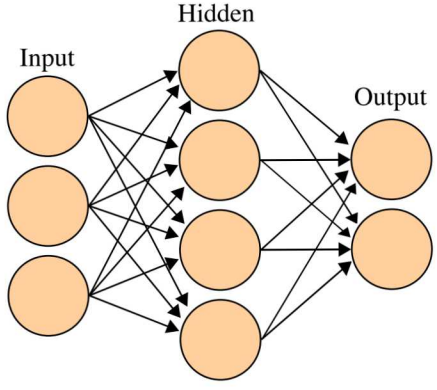
Determine input values that maximize the activation function for a unit (e.g. neuron) in the neural network

Steps

1. Train model
2. Set parameters at the estimated values
3. Maximize activation function at chosen unit over inputs

$\boldsymbol{x}$ = feature vector
$\hat{\beta}$ = estimated parameters
$\boldsymbol{h_u}$ = activation function at "unit" $\boldsymbol{u}$

$$\arg\max_{x} h_u\left(\hat{\beta}, x\right)$$



Issues with Optimization

I'm still working to understand the optimization details. Here is my understanding so far...

- From Olah, Mordvintsev, and Schubert it sounds like
  - this is not simple
  - gradient descent
  - involves regularization
- They say:

> Dealing with this high frequency noise has been one of the primary challenges and overarching threads of feature visualization research. **If you want to get useful visualizations, you need to impose a more natural structure using some kind of prior, regularizer, or constraint.**

Different Optimization Approaches

		Weak Regularization avoids misleading correlations, but is less connected to real use.			**Strong Regularization** gives more realistic examples at risk of misleading correlations.	
		Unregularized	**Frequency Penalization**	**Transformation Robustness**	**Learned Prior**	**Dataset Examples**
	**Erhan, *et al.*, 2009** [3] Introduced core idea. Minimal regularization.	■				
	**Szegedy, *et al.*, 2013** [11] Adversarial examples. Visualizes with dataset examples.	■				■
	**Mahendran & Vedaldi, 2015** [7] Introduces total variation regularizer. Reconstructs input from representation.		■			
	**Nguyen, *et al.*, 2015** [14] Explores counterexamples. Introduces image blurring.		■			
	**Mordvintsev, *et al.*, 2015** [4] Introduced jitter & multi-scale. Explored GMM priors for classes.			■	■	
	**Øygard, *et al.*, 2015** [15] Introduces gradient blurring. (Also uses jitter.)		■	■		
	**Tyka, *et al.*, 2016** [16] Regularizes with bilateral filters. (Also uses jitter.)		■	■		
	**Mordvintsev, *et al.*, 2016** [17] Normalizes gradient frequencies. (Also uses jitter.)		■	■		
	**Nguyen, *et al.*, 2016** [18] Paramaterizes images with GAN generator.				■	
	**Nguyen, *et al.*, 2016** [10] Uses denoising autoencoder prior to make a generative model.				■	

Image from https://distill.pub/2017/feature-visualization/

Lucid

- feature visualization package in Python
- GitHub Repo
- tutorial

Simple Example of Feature Visualization with Tabular Data

Iris Data Neural Network

response = setosa? (Yes/No)

features = petal length, petal width

structure = 1 hidden layer, 3 neurons

activation = logistic function

```
set.seed(384724029)
nn ← neuralnet(Species == "setosa" ~ Petal.Length + Petal.Width,
               data = iris,
               hidden = 3,
               act.fct = "logistic",
               linear.output = FALSE)
```

Formulas

Notation

$$\hat{\beta}_{i,j} = \text{estimated weight for neuron } i \text{ and feature } j$$

$$\sigma(v) = \text{logistic activation function} = \frac{1}{1 + e^{-v}}$$

$$z_i = \text{derived feature at neuron } i$$

Activation Functions

$$z_1 = \sigma\left(\hat{\beta}_{0,1} + \hat{\beta}_{1,1} \times \text{petal length} + \hat{\beta}_{2,1} \times \text{petal width}\right)$$

$$z_2 = \sigma\left(\hat{\beta}_{0,2} + \hat{\beta}_{1,2} \times \text{petal length} + \hat{\beta}_{2,2} \times \text{petal width}\right)$$

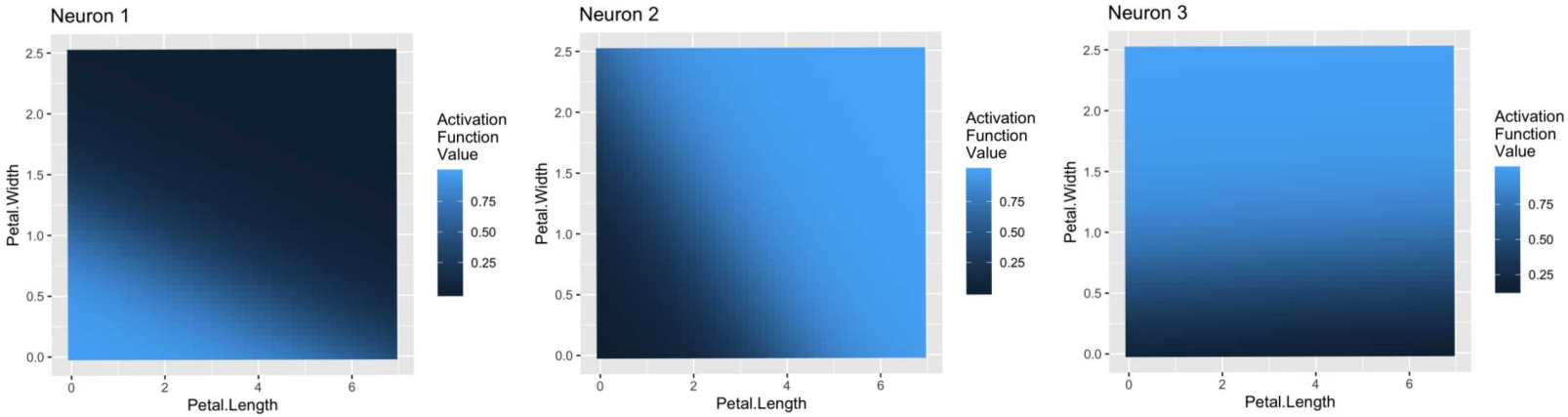
$$z_3 = \sigma\left(\hat{\beta}_{0,3} + \hat{\beta}_{1,3} \times \text{petal length} + \hat{\beta}_{2,3} \times \text{petal width}\right)$$

Visualizing Activation Functions

Goal of Feature Visualization

Determine which values of petal length and petal width optimize an activation function at a specific neuron

Activation Function Values over a Grid of Feature Values

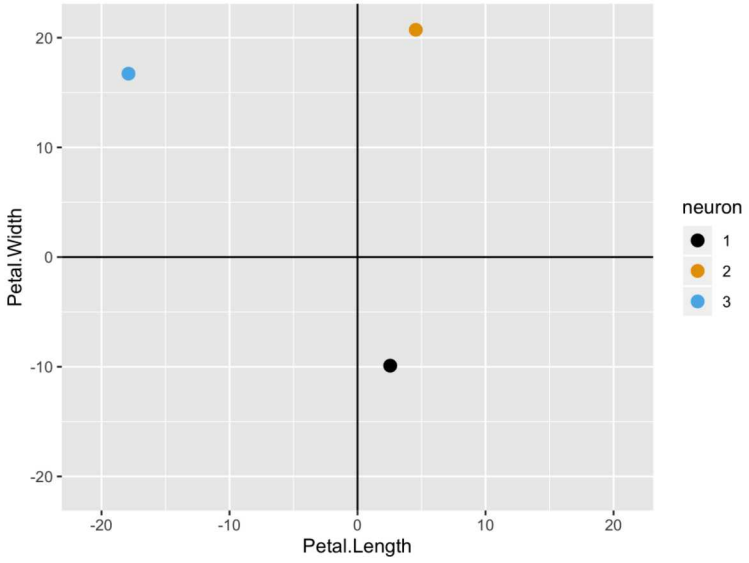


Optimized Values

Method

- for each of the three neurons
- determined feature values that maximized activation functions
- used `optim` in R (default settings, Nelder and Mead method)

Plot of Maximum Feature Values (colored by neuron number)

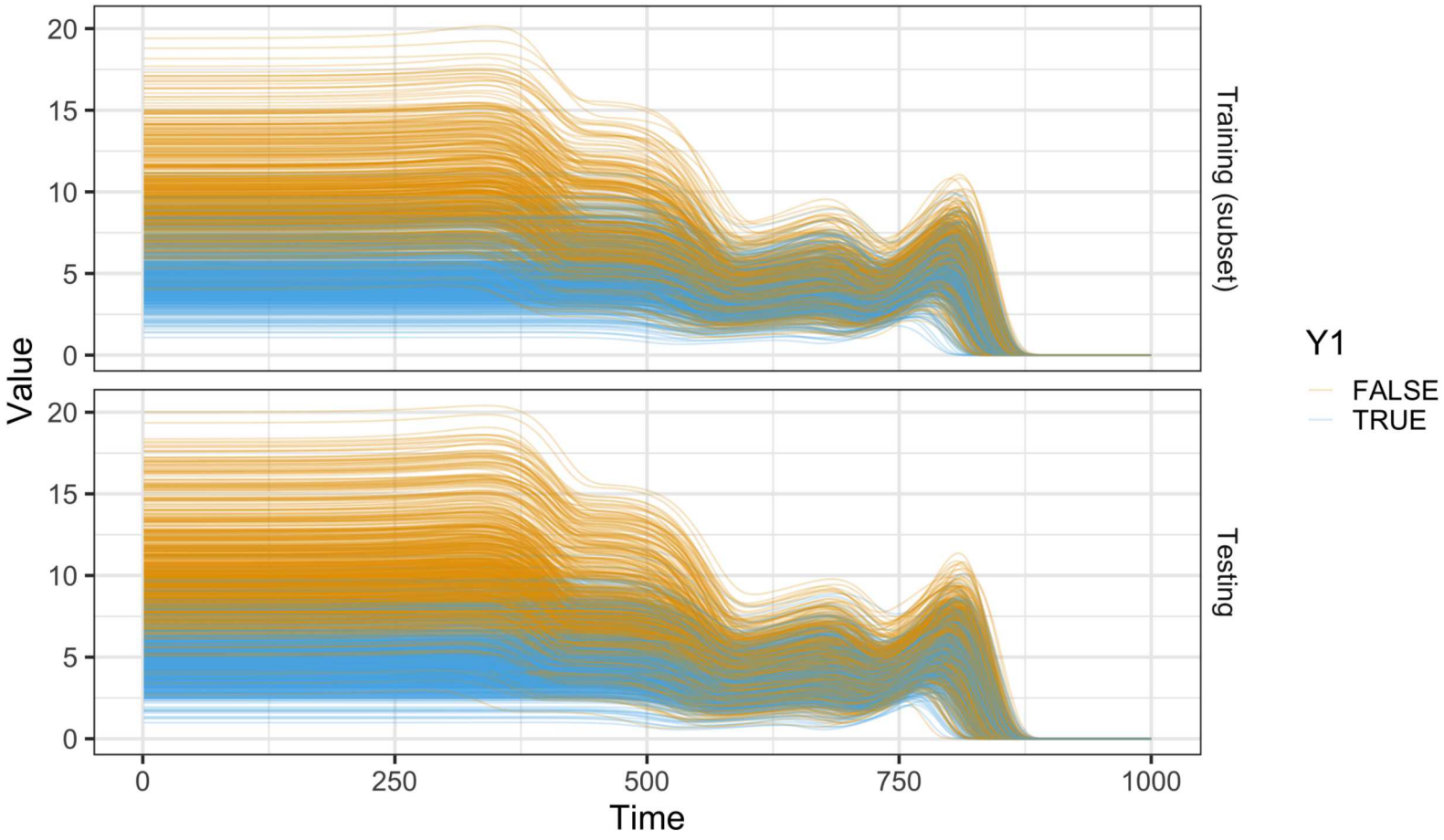


- interpret as an observation that activates a neuron (to the maximum extent)
- note that negative values not meaningful in this context
- maybe this is where constraints may play a role?

HE Simulated Data

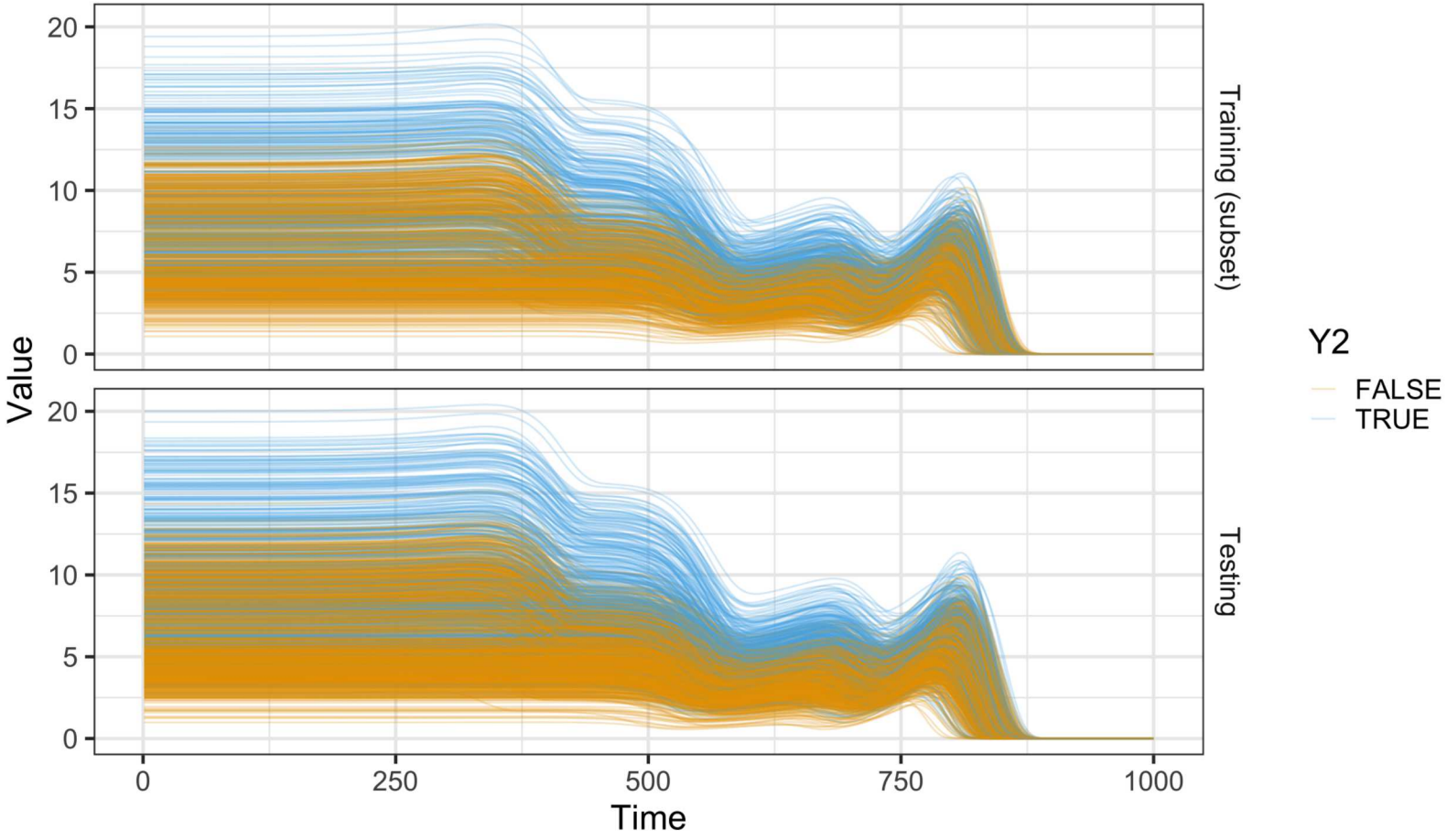
HE Simulated Data: Y1

Plots of the training (randomly selected subset) and testing data



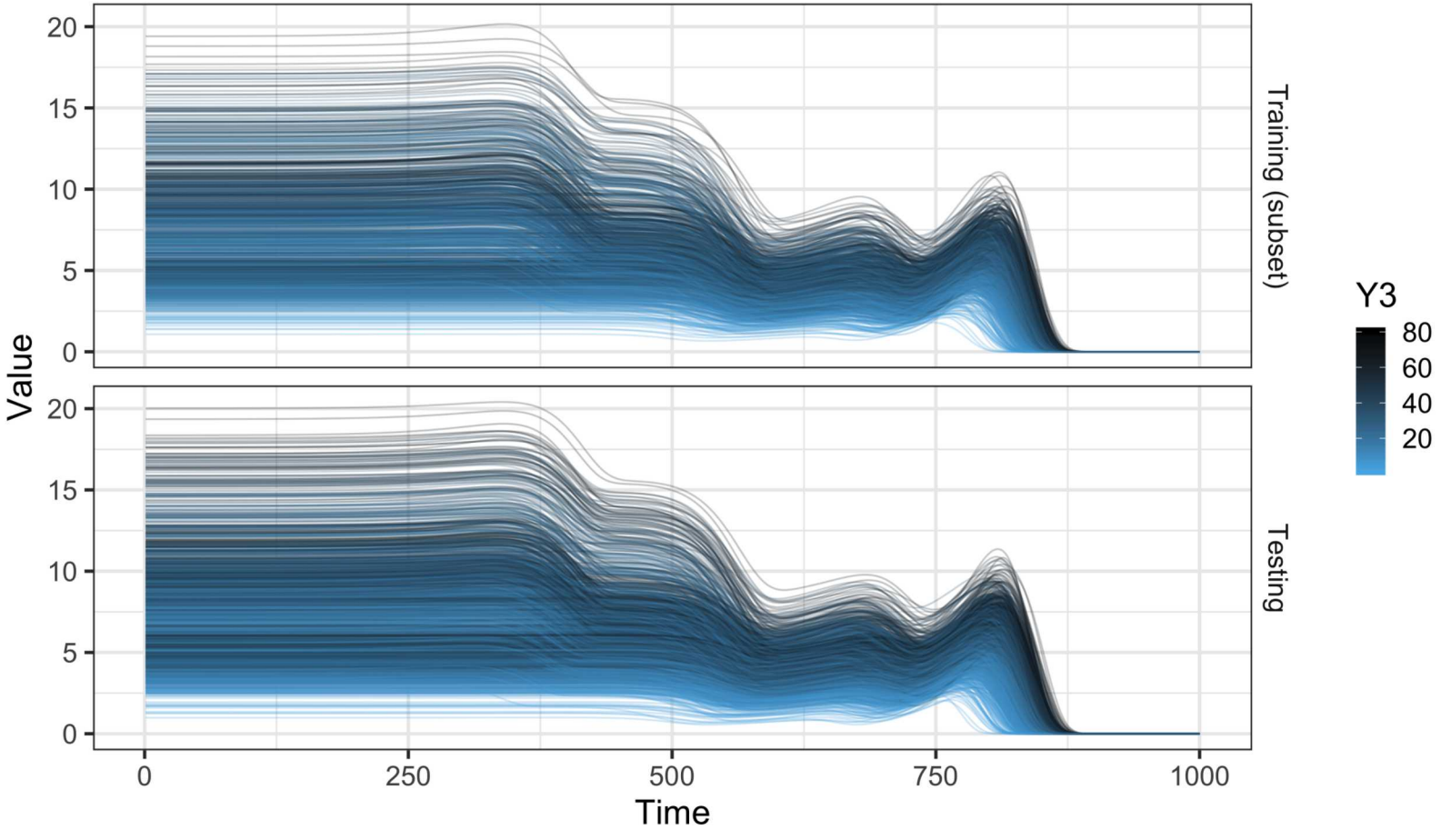
HE Simulated Data: Y2

Plots of the training (randomly selected subset) and testing data



HE Simulated Data: Y3

Plots of the training (randomly selected subset) and testing data



Feature Visualization with HE Simulated Data (first attempt)

HE Simple Neural Network

response = y1 (true/false)

features = observations at a time (1000 of these)

structure = 1 hidden layer, 5 neurons

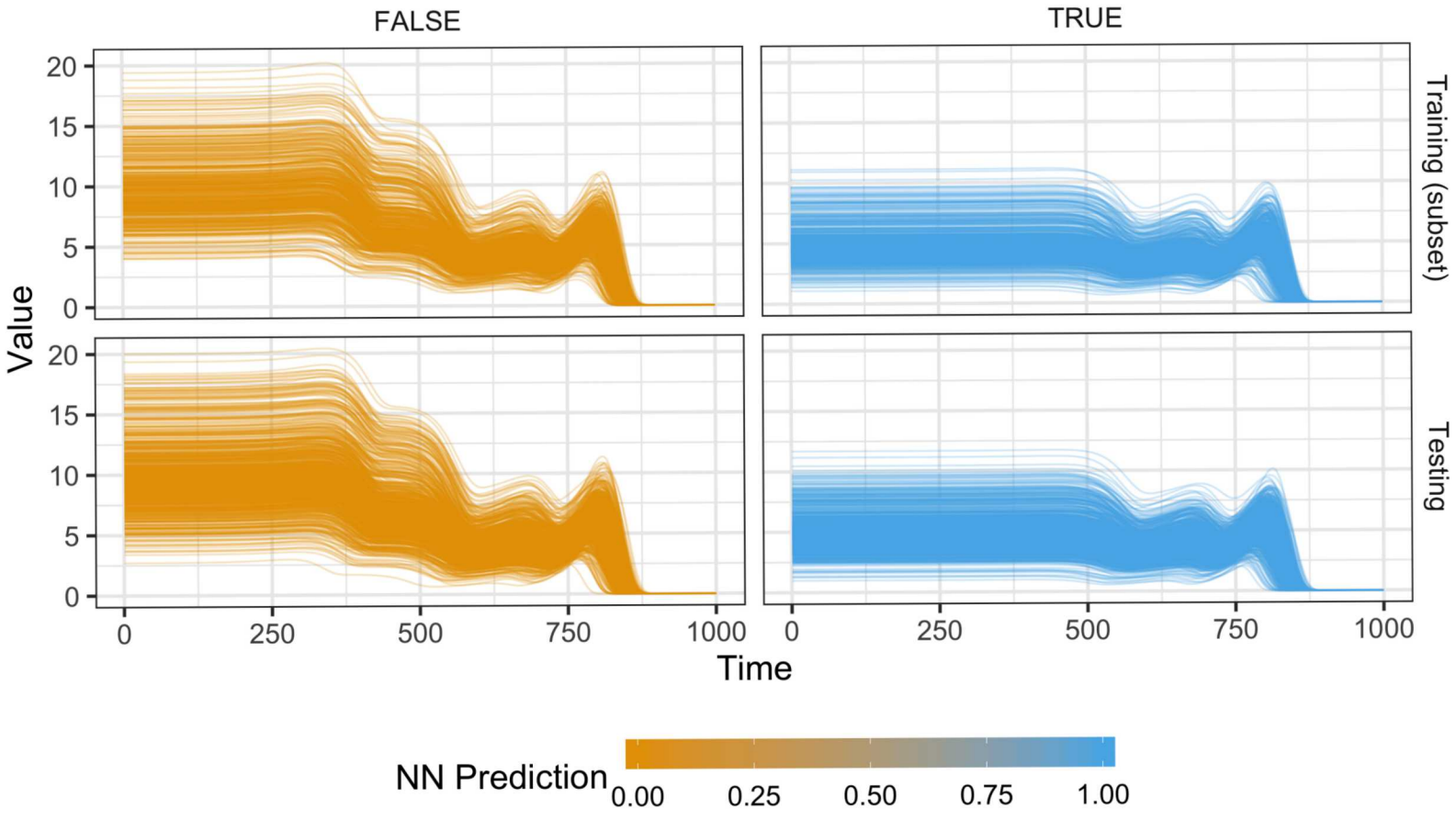
activation = logistic function

```
set.seed(2020030)
he_nn ←
  neuralnet(y1 ~ .,
            data = he_train %>% select(-obs, -y2, -y3),
            hidden = 5,
            act.fct = "logistic",
            linear.output = FALSE)
```

```
##     error
## 0.0128313
```

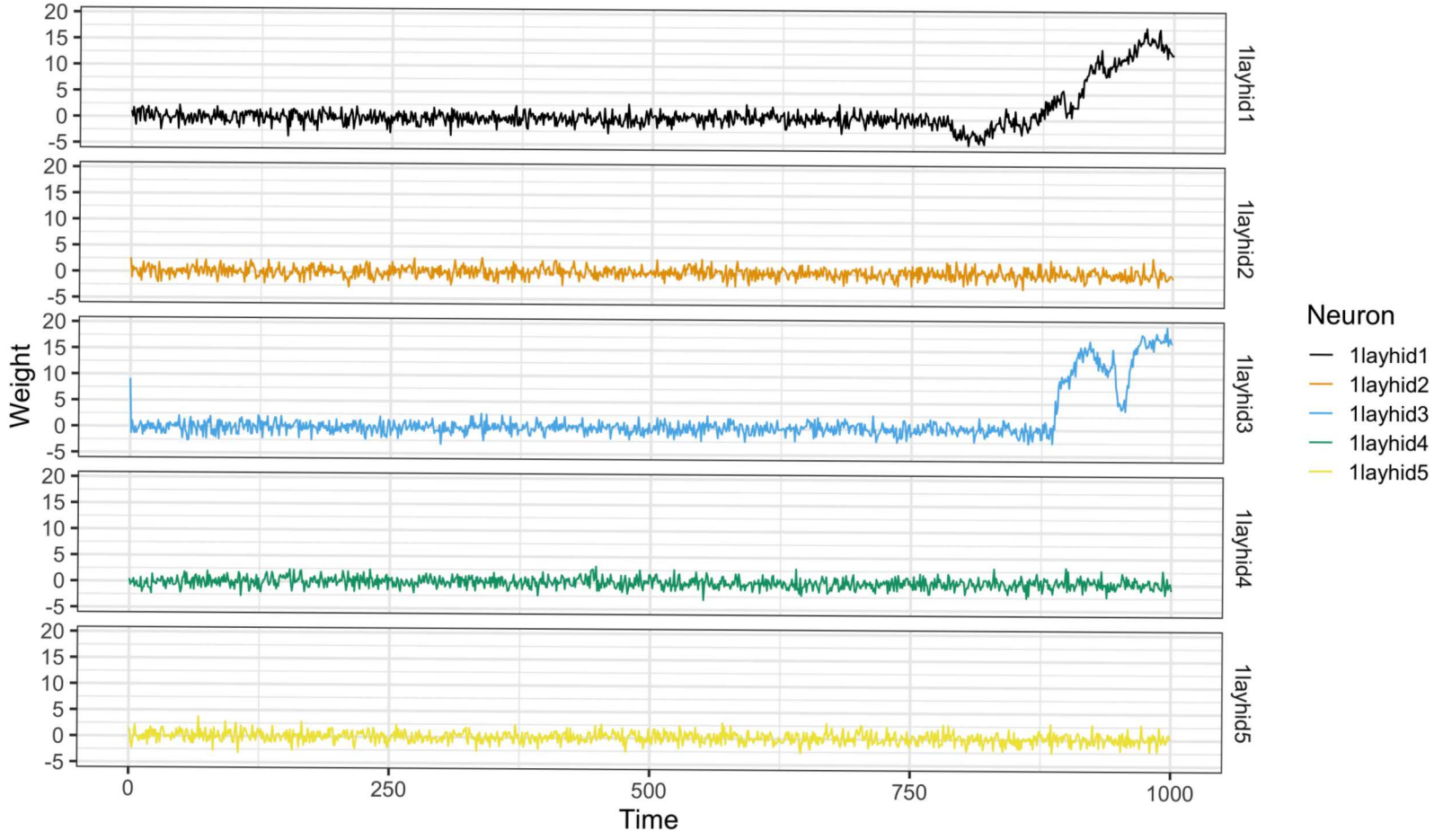
HE Neural Network Predictions

Plots of neural network predictions



HE Neural Network Weights

Weights from neural network plotted versus time



Activation Function Optimizations

Optimization equations for neurons 1 to 5:

$$\arg\max_X \sigma\left(\hat{\beta}_{0,1} + \hat{\beta}_{1,1}X_1 + \hat{\beta}_{2,1}X_2 + \cdots + \hat{\beta}_{1000,1}X_{1000}\right)$$

$$\vdots$$

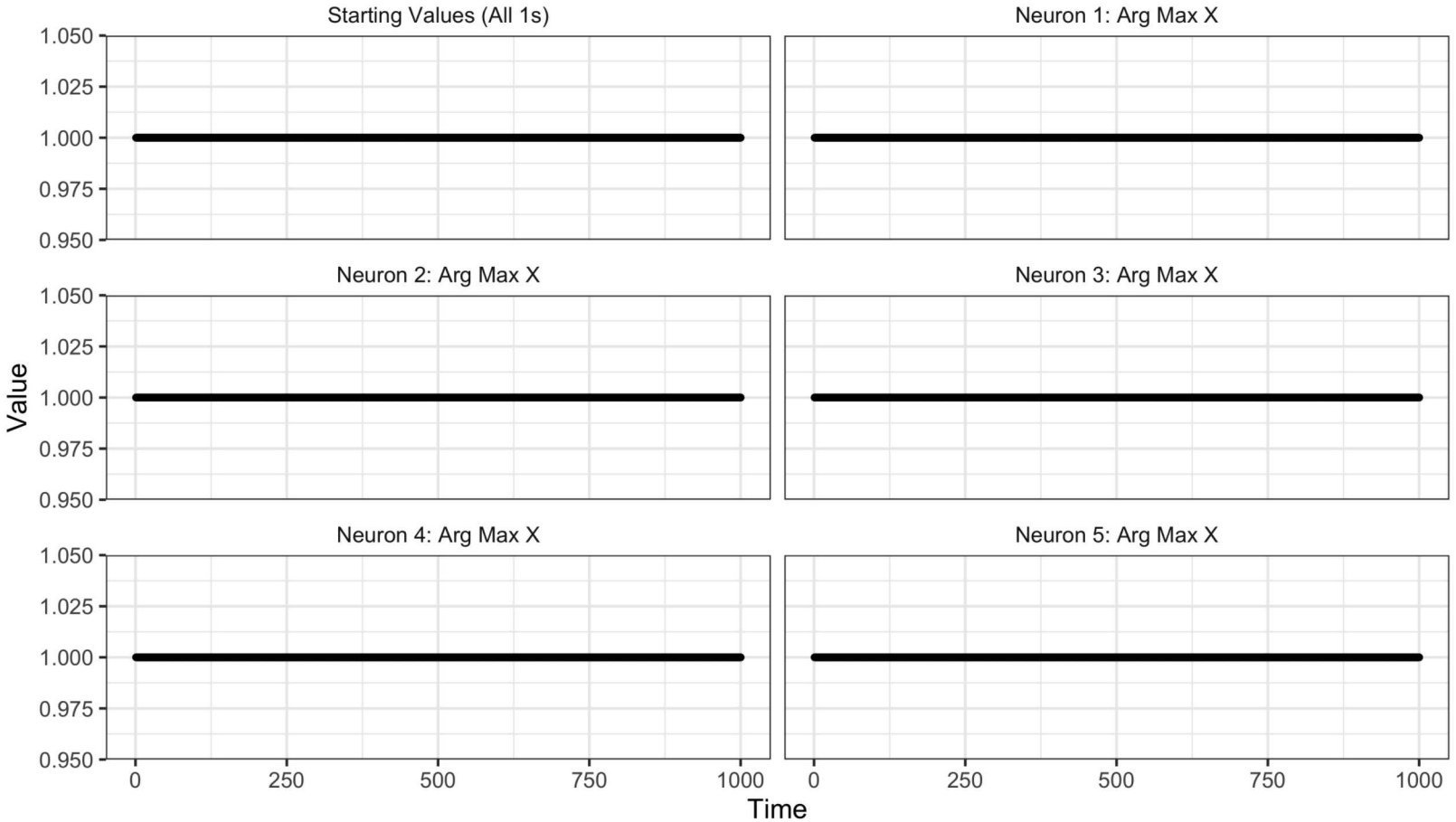
$$\arg\max_X \sigma\left(\hat{\beta}_{0,5} + \hat{\beta}_{1,5}X_1 + \hat{\beta}_{2,5}X_2 + \cdots + \hat{\beta}_{1000,5}X_{1000}\right)$$

where

- $\hat{\beta}$'s = estimated weights from the neural network
- z's = derived features

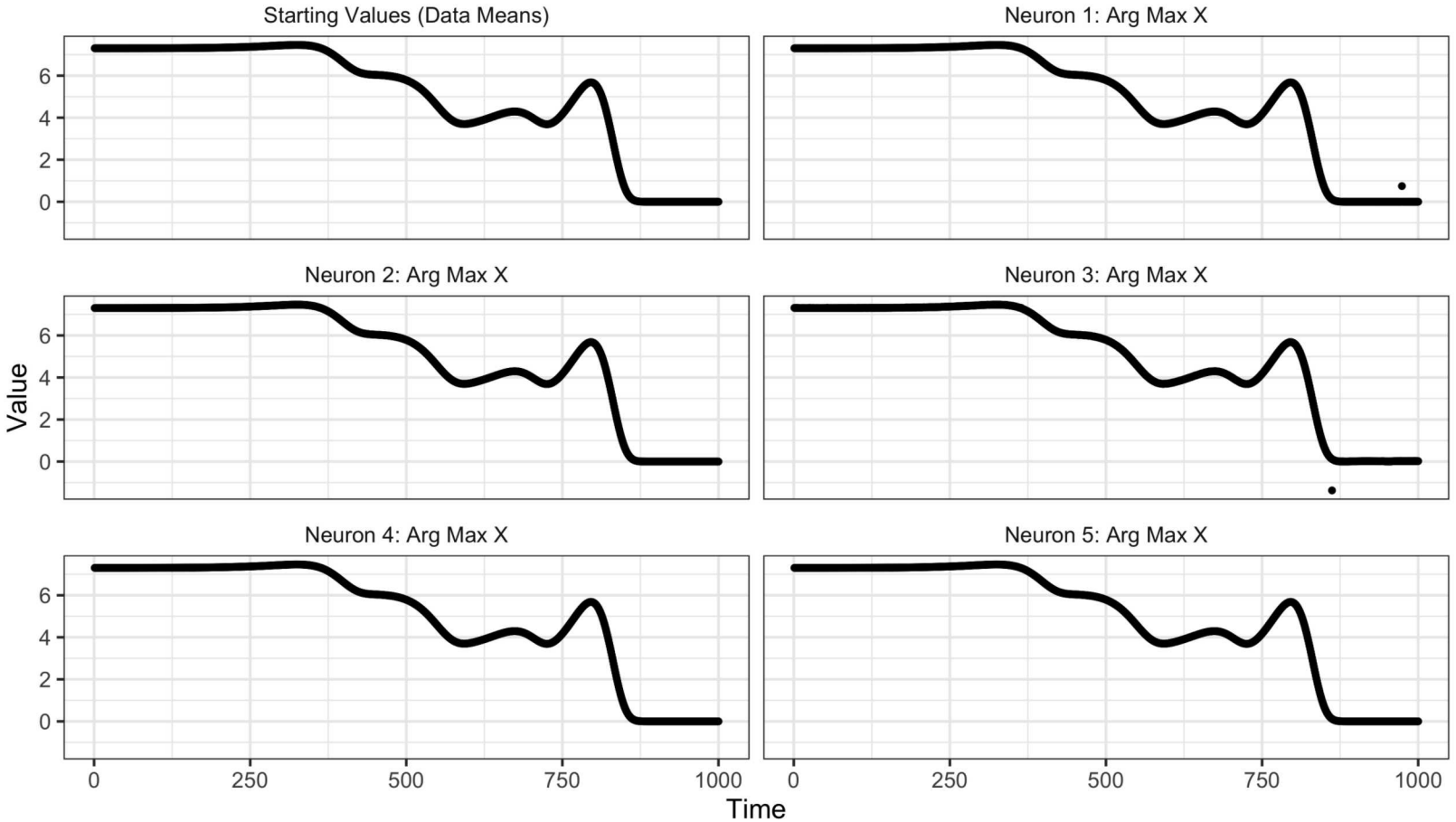
Attempt to Optimize

Not working...



Another Attempt

Still not working...



Feature Visualization with HE Simulated Data (PC Features)

HE NN with Principal Components

Attempt to fix the problem using PCs instead of raw observations...

response = y1 (true/false)
features = first 5 PCs (explain 99.83% of the variation)
structure = 1 hidden layer, 5 neurons
activation = logistic function

```
# Fit a neural network using the first five PCs
set.seed(2020030)
he_nn_pca ← neuralnet(y1 ~ PC1 + PC2 + PC3 + PC4 + PC5,
                      data = he_data_pcs,
                      hidden = 5,
                      act.fct = "logistic",
                      linear.output = FALSE,
                      err.fct = 'ce')
```

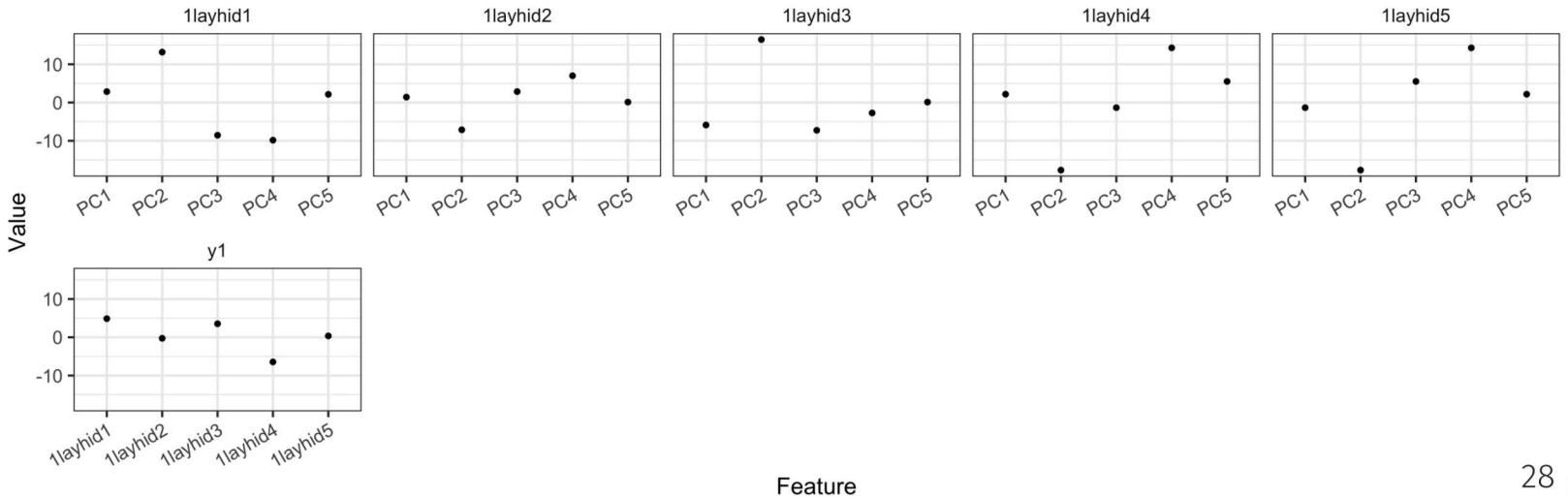
```
##      error
## 0.01435877
```

Activation Function Optimizations

Optimization equation for neurons i = 1,...,5

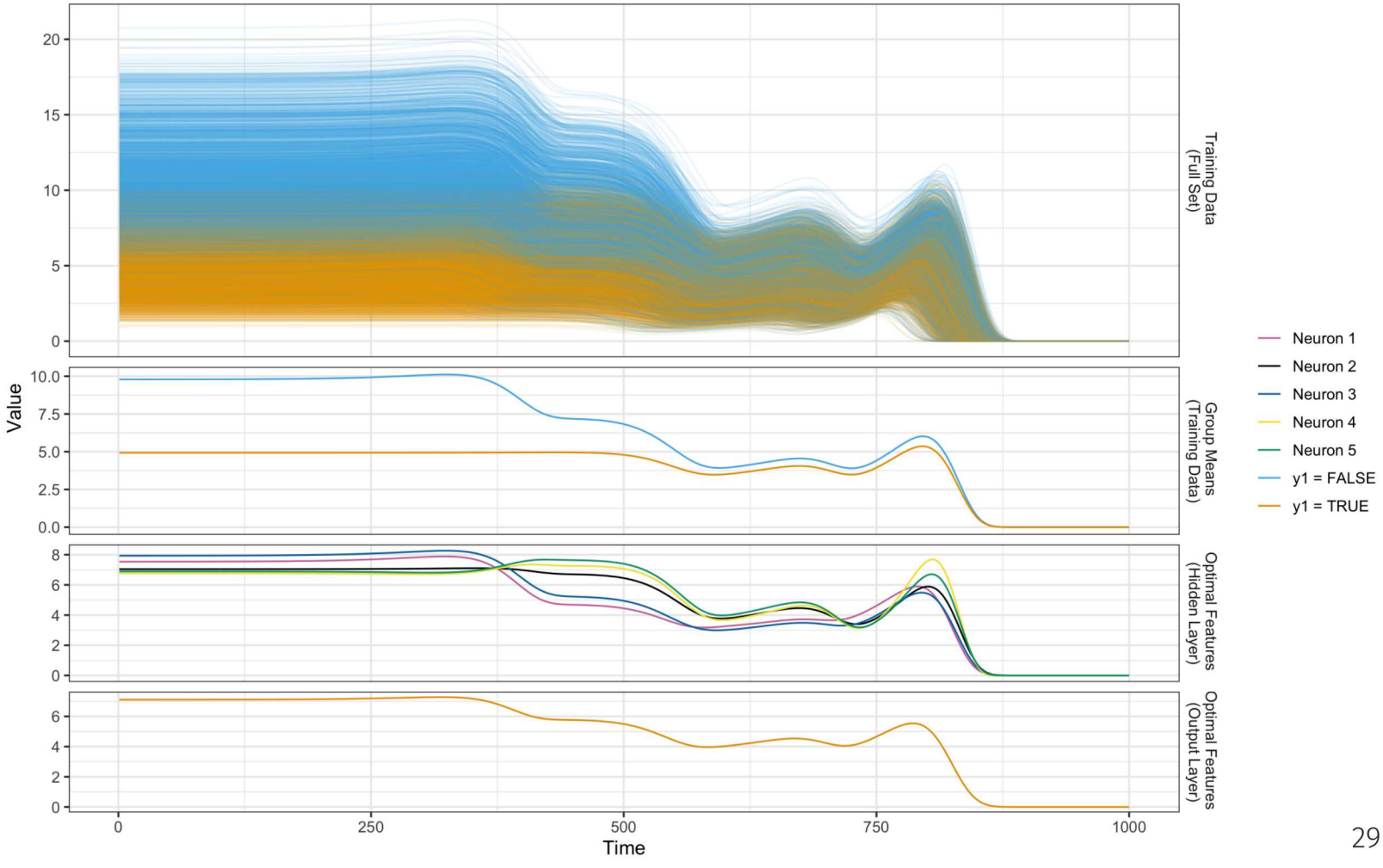
$$\arg\max_{PC} \sigma\left(\hat{\beta}_{0,i} + \hat{\beta}_{1,i}PC_1 + \hat{\beta}_{2,i}PC_2 + \cdots + \hat{\beta}_{5,i}PC_5\right)$$

```
# Perform feature visualization
he_pca_opt = feature_viz(nn = he_nn_pca,
                         act_form = "logistic",
                         start = c(1,1,1,1,1))
```



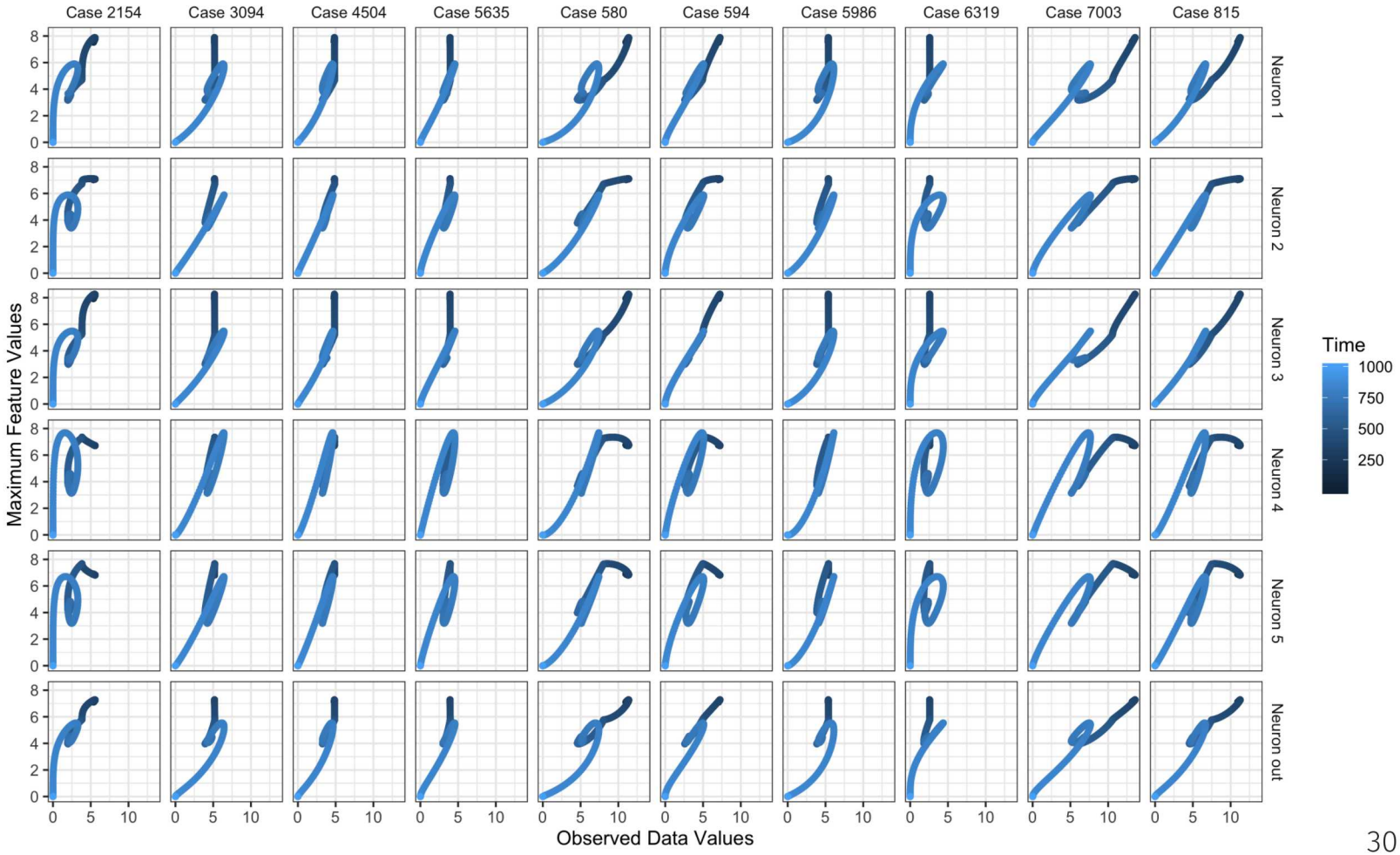
Optimal Features and Observed Data

Comparing back-transformed optimal features to the observed data



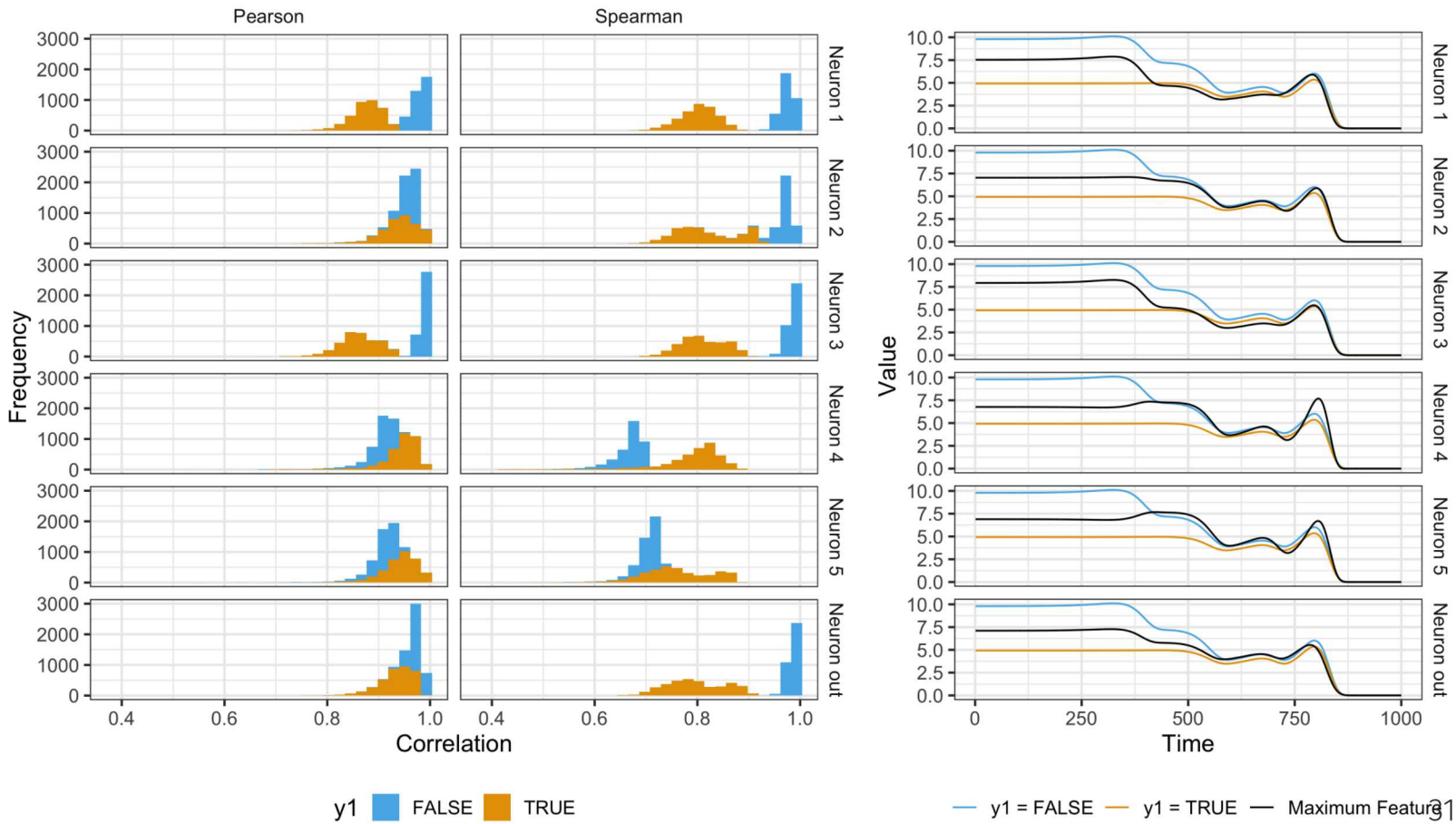
Correlating Max Features to Observed

Maximum feature versus data values for 10 random observations



Correlating Max Features to Observed

Histograms of correlations between optimal features and observed data (left) and maximum features with training data group means (right)



Feature Visualization with HE Simulated Data (Adding Layers)

HE NN with Three Layers

response = y1 (true/false)

features = first 5 PCs (explain 99.83% of the variation)

structure = 3 hidden layer3, 5 neurons at each layer

activation = logistic function

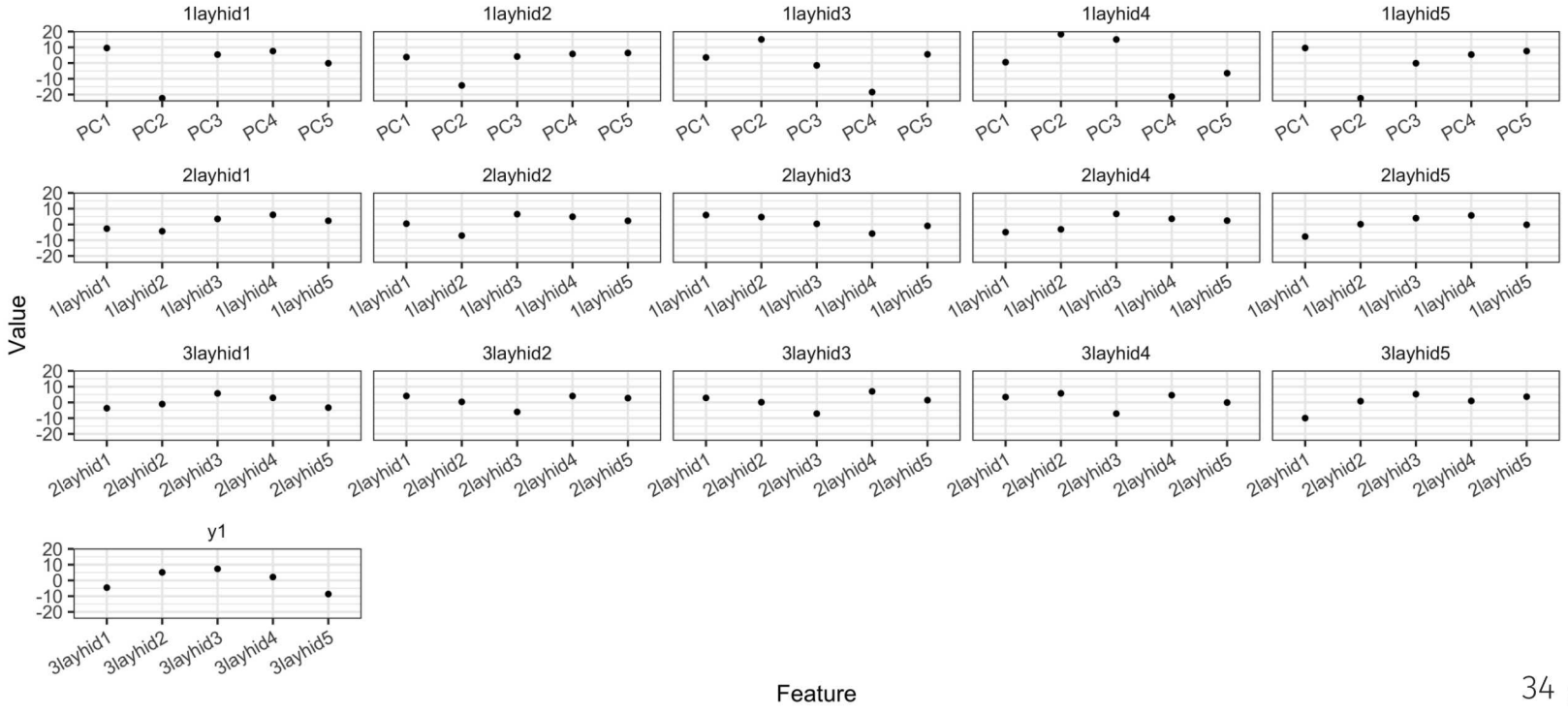
```
# Fit a neural network using the first five PCs
set.seed(20200318)
he_nn3l ← neuralnet(y1 ~ PC1 + PC2 + PC3 + PC4 + PC5,
                    data = he_data_pcs,
                    hidden = c(5, 5, 5),
                    act.fct = "logistic",
                    linear.output = FALSE)
```

```
##       error
## 0.006929095
```

Activation Function Optimizations

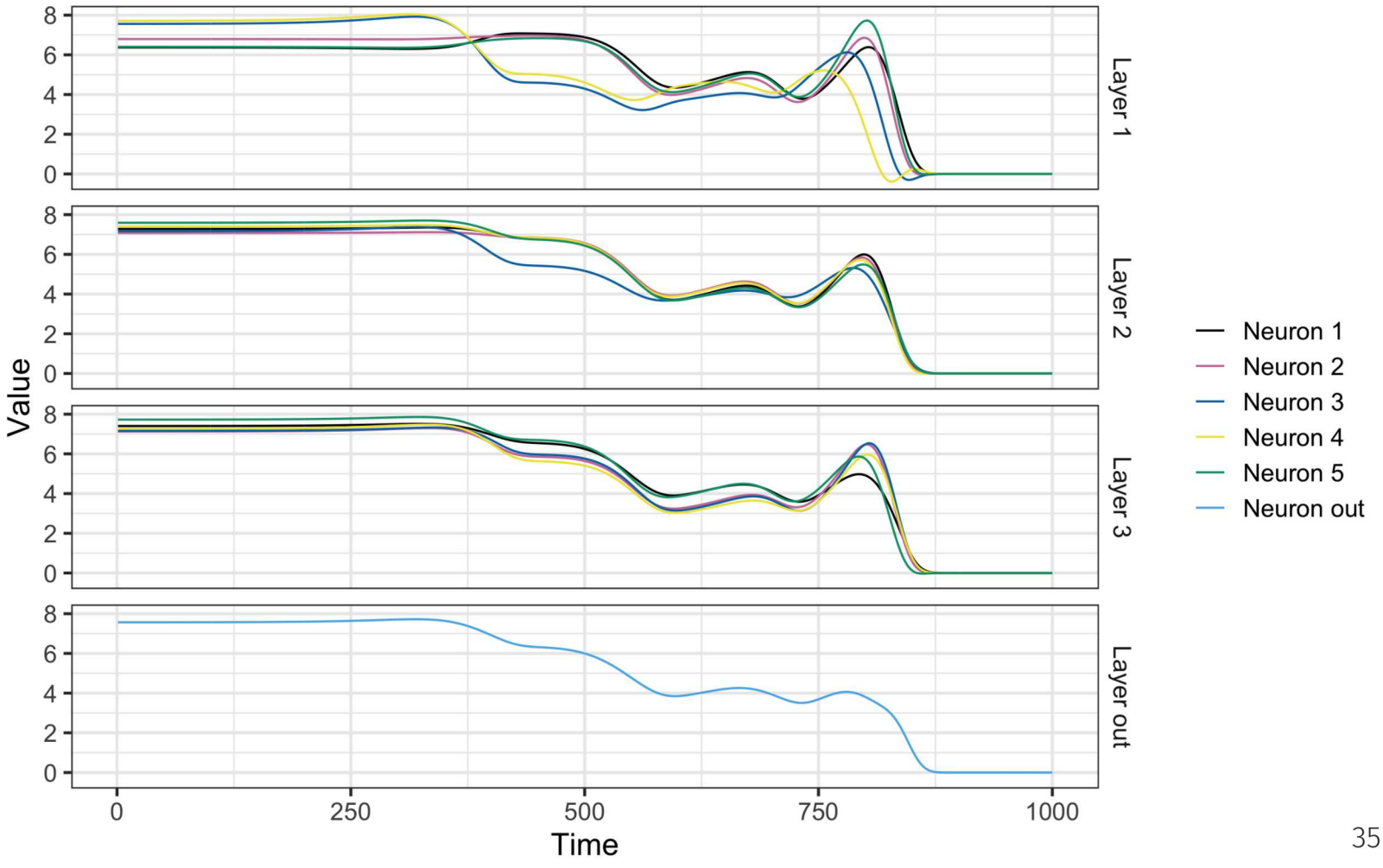
```
# Perform feature visualization
he_3l_opt = feature_viz(nn = he_nn3l, act_form = "logistic",
                        start = c(1,1,1,1,1))
```

Values that optimized the activation functions at each neuron



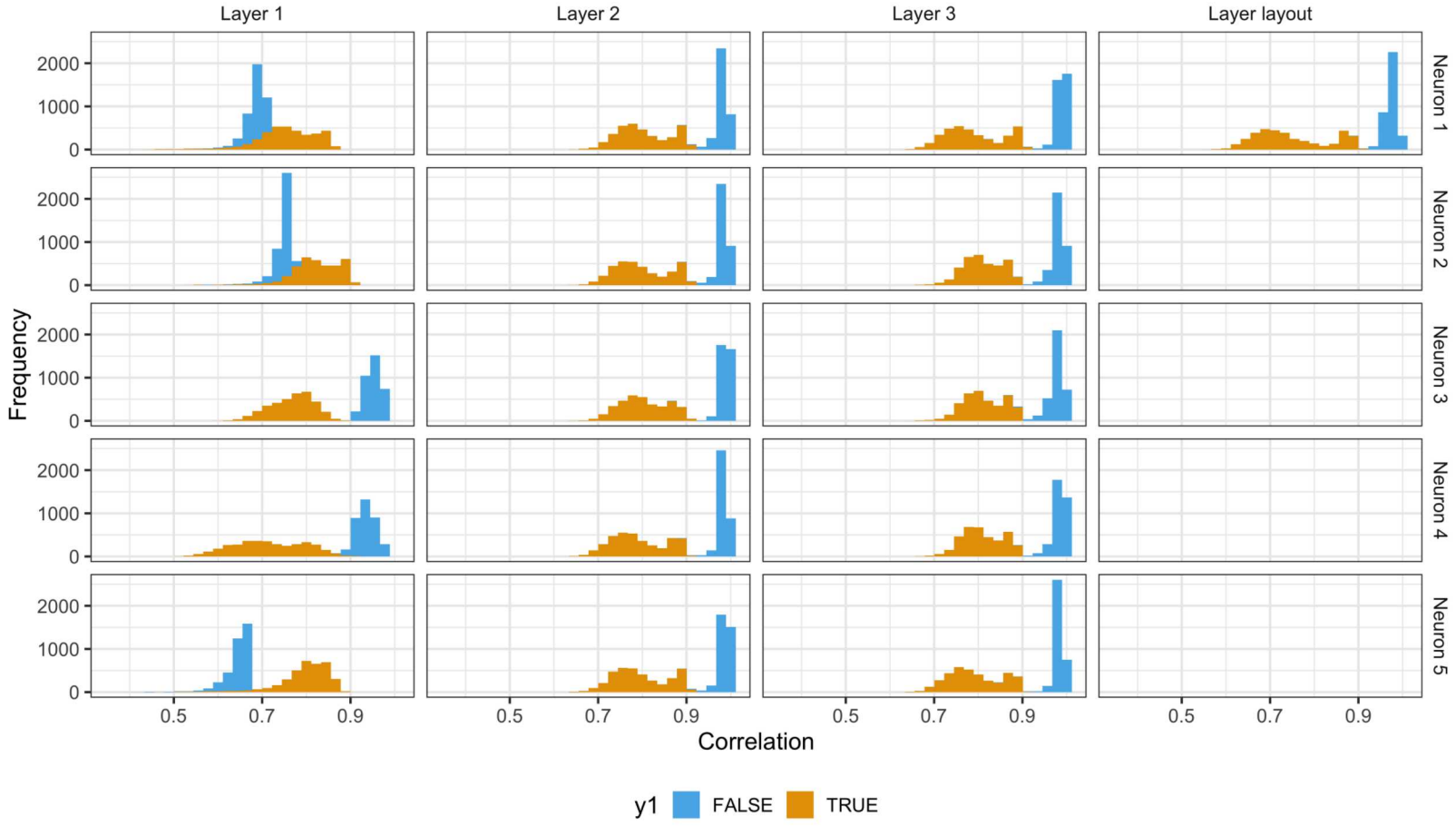
Back-Transformed Features

Plots of the back transformed PCs that optimized the activation functions



Correlating Max Features to Observed

Spearman correlations between optimal features and observed data



Feature Visualization with HE Simulated Data (Adding Responses)

HE NN: 2 Responses and PC Features

responses = y1 (true/false) and y2 (true/false)

features = first 5 PCs (explain 99.83% of the variation)

structure = 1 hidden layer, 4 neurons

activation = logistic function

```
# Fit a neural network using the first five PCs
set.seed(20200327)
he_nn_pca2r ← neuralnet(y2 ~ PC1 + PC2 + PC3 + PC4 + PC5,
                        data = he_data_pcs,
                        hidden = 4,
                        act.fct = "logistic",
                        linear.output = FALSE,
                        err.fct = 'ce',
                        rep = 3)
he_nn_pca2r$call
```