Data, Environment and Society: Lecture 25: Neural Networks

Instructor: Duncan Callaway
GSI: Salma Elmallah

November 26, 2019

Announcements

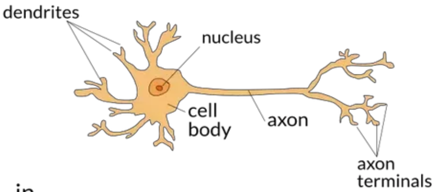
- HW10 due today
- Next today and tuesday: neural nets
  - Feel free to use on projects – but no HW here.
- Course evaluations available online; I will make time next Tuesday
- Thursday: Career panel
  - JP Dolphin, manager of Strategic Data Science at PG&E
  - Tanner Burke, senior data engineer, Streetlight Data
  - Jason Harville, assistant executive director of the Energy Data and Analytics Office, California Energy Commission

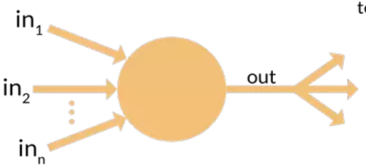
Today's outline

1. Neural networks (NN)
   - Brief introduction
   - Experiment with tensorflow playground – try fitting different classification problems
   - Objective: Understand the role of key parameters, what the hyperparameters are, and the model fitting process
2. Exam handout and discussion

Neural networks: Origins

- The name is due to analogy with brains
- First developed in 1943
- Inspired the development of the perceptron (see HW10) in the '50s
  - Here the purpose was just to remove noise on phone lines
  - Not to reproduce thought...
- Little research activity ∼1960-1990's due to computing limitations
  - Major exception: Werbos developed back-propogation in 1974. First effort to get NN to "learn" parameters
- Computing advances made "deep" NN possible in the last 20 years





Mathematics for a single "neuron"

In words, each neuron...

- Takes a vector of values as inputs
- Creates a scalar from a linear combination of the vector entries
- Passes the resulting scalar through an "activation function"
- Outputs a single value from that activation function

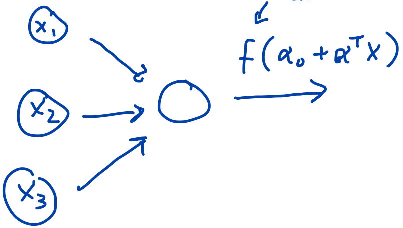
Terminology analogies:

- Electrical signal from other cells $\rightleftarrows$ input
- Neuron $\rightleftarrows$ Activation function
- Electrical signal to other cells $\rightleftarrows$ output

$$X = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \qquad \alpha = \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \end{bmatrix}$$

$$\begin{aligned} V &= \alpha_0 + \alpha^T X \quad \leftarrow \text{scalar value.} \\ &= \alpha_0 + \alpha_1 x_1 + \alpha_2 x_2 + \alpha_3 x_3 \end{aligned}$$

"activation function"



Mathematics for a single "neuron"

In words, each neuron...

- Takes a vector of values as inputs
- Creates a scalar from a linear combination of the vector entries
- Passes the resulting scalar through an "activation function"
- Outputs a single value from that activation function

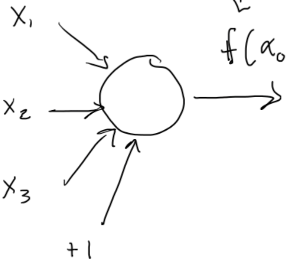
Terminology analogies:

- Electrical signal from other cells $\rightleftarrows$ input
- Neuron $\rightleftarrows$ Activation function
- Electrical signal to other cells $\rightleftarrows$ output

$$x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \qquad \alpha = \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \end{bmatrix}$$

activation f'n

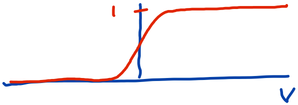
$$f(\alpha_0 + \alpha^T x)$$



What's f, the activation function?

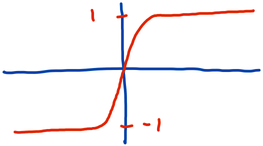
$$v = \alpha_0 + \alpha^T x$$

1. sigmoid



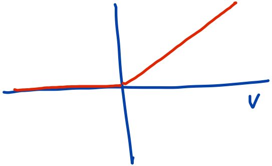
$$f(v) = \frac{1}{1 + e^{-v}}$$

2. tanh



$$f(v) = \frac{e^{v} - e^{-v}}{e^{v} + e^{-v}}$$

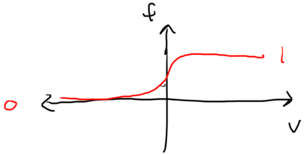
3. rectified linear (ReLU)



$$v = \max(0, v)$$

What's f, the activation function?

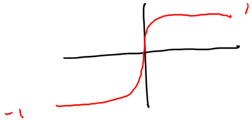
1. sigmoid



$$v = \alpha_0 + \alpha^T x$$

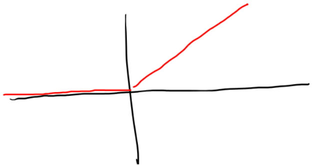
$$f(v) = \frac{1}{1+e^{-v}}$$

2. tanh



$$f(v) = \frac{e^{v} - e^{-v}}{e^{v} + e^{-v}}$$

3. rectified linear (ReLU)



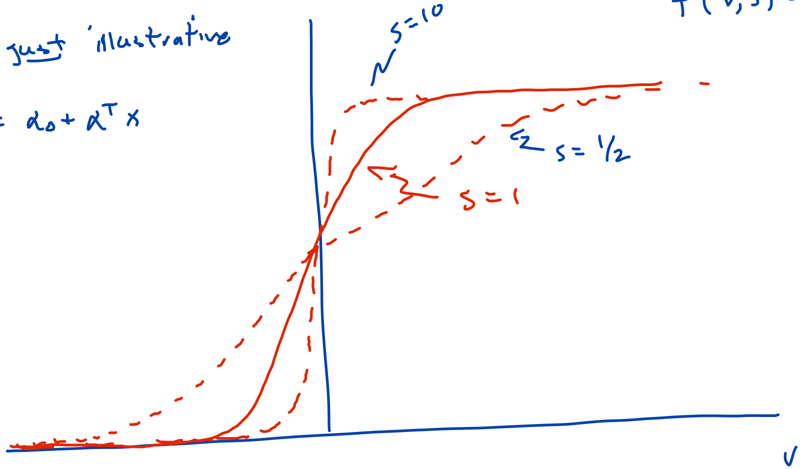
$$\max(0, v)$$

How the sigmoid function works

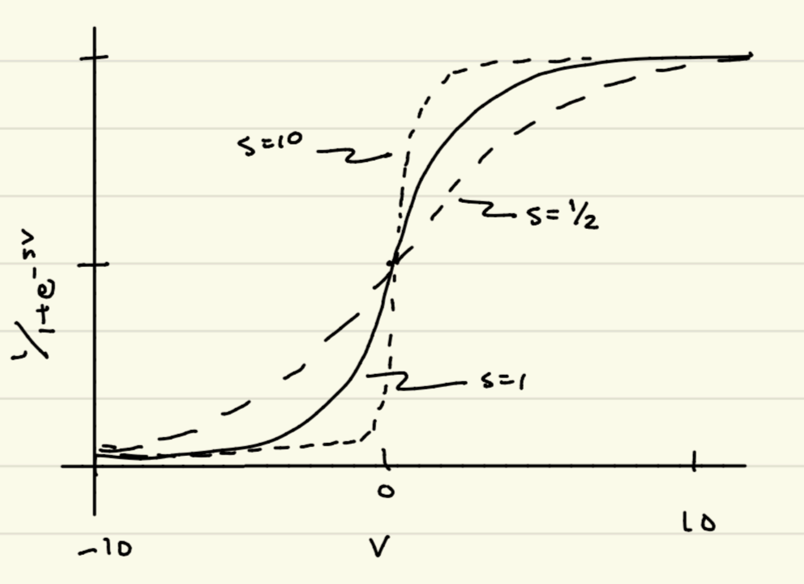
$$f(v,s) = \frac{1}{1+e^{-sv}}$$

s here is <u>just</u> 'illustrative'

<u>In</u> fact $v = \alpha_0 + \alpha^T x$

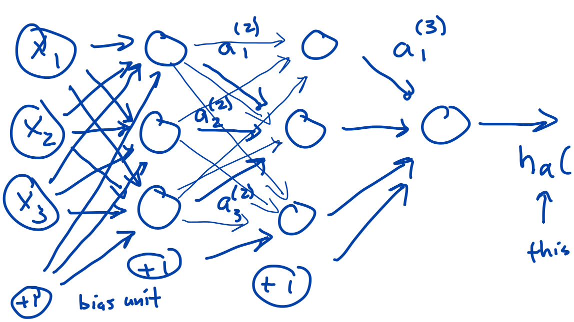


How the sigmoid function works



Remember: $v = \alpha_0 + \alpha^T \mathbf{x}$

Neural network: just gang the neurons together



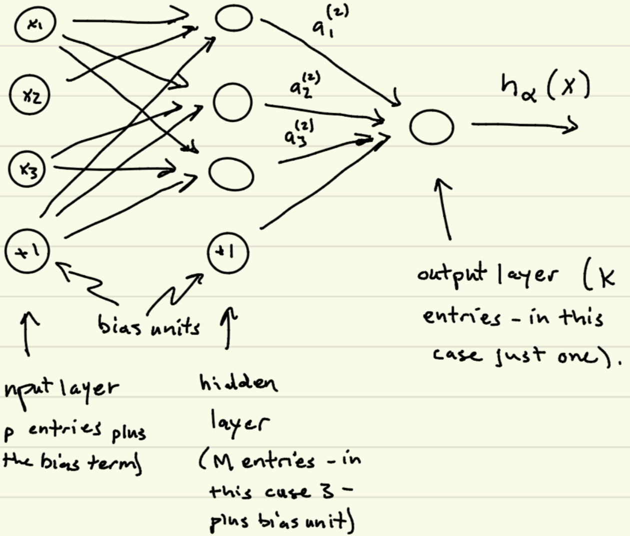
$h_\alpha(x)$

this means all α of network

convention: $a_i^{(\ell)}$ output from node i in layer ℓ.

$v = \alpha_0 + \alpha^T x$

Neural network: just gang the neurons together



Convention: $a_i^{(\ell)}$ is the output of the i^{th} neuron in the ℓ^{th} layer.

Mathematical merging of neurons

Convention:

- $\alpha_{ij}^{(l)} \rightarrow$ weight from node j in layer l to node i in $l+1$ layer.
- $a_i^{(l)} \rightarrow$ output of node i in layer l.

$$a_1^{(2)} = f\left(\alpha_{10}^{(1)} + \alpha_{11}^{(1)} x_1 + \alpha_{12}^{(1)} x_2 + \alpha_{13}^{(1)} x_3\right)$$

$$a_1^{(3)} = f\left(\alpha_{10}^{(2)} + \sum_{j=1}^{M_2} \alpha_{1j}^{(2)} a_j^{(2)}\right)$$

Note that I used x in the first equation because I'm calling the features (inputs to the model) the first "layer" of the network

Question: What are the parameters of a neural network model?

Mathematical merging of neurons

Convention:

- $\alpha_{ij}^{(l)} \rightarrow$ weight from node j in layer l to node i in $l+1$ layer.
- $a_i^{(l)} \rightarrow$ output of node i in layer l.

$$a_1^{(2)} = f(\alpha_{10}^{(1)} + \alpha_{11}^{(1)} x_1^{(1)} + \alpha_{12}^{(1)} x_2^{(1)} + \alpha_{13}^{(1)} x_3^{(1)})$$

$$a_1^{(3)} = f(\alpha_{10}^{(2)} + \sum_{j=1}^{M} \alpha_{1j}^{(2)} a_j^{(2)})$$

Note that I used x in the first equation because I'm calling the features (inputs to the model) the first "layer" of the network

Question: What are the parameters of a neural network model?

Just the α values. a values are outputs from internal nodes or neurons. We call these "hidden states" because they depend on the input values x.

Thinking about the features and target

Let's watch this video. It uses graphics in a nice way to explain what NNs are doing.

`https://www.youtube.com/watch?v=aircAruvnKk`

Start the video at 2:05. We'll stop watching around 5:30.

Compact notation motivates a name...

Compact notation motivates a name...

$$a_i^{(2)} = f(\alpha_{i0}^{(1)} + \alpha_{i1}^{(1)} x_1^{(1)} + \alpha_{i2}^{(1)} x_2^{(1)} + \alpha_{i3}^{(1)} x_3^{(1)})$$

$$a_i^{(3)} = f(\alpha_{i0}^{(2)} + \sum_{j=1}^{M_2} \alpha_{ij}^{(2)} a_j^{(2)}) \quad (M_j \text{ is the number of neurons in layer } j)$$

$$a_i^{(4)} = f(\alpha_{i0}^{(3)} + \sum_{j=1}^{M_3} \alpha_{ij}^{(3)} a_j^{(3)})$$

$$\vdots$$

$$h_\alpha(x) = f(a^{(\ell)}, \alpha^{(\ell)}) \quad \text{Final output of NN. } \ell \text{ is the number of layers}$$

- The $_\alpha$ subscript means h is a function of ALL the α values of the network
- We dropped subscripts on a, meaning a is a vector of inputs to the final layer

Because each layer informs the next, we call this a **feedforward** neural network.

Fitting the model - regression

Training data: $\{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$

$h_\alpha(x)$

$x \in \mathbb{R}^p$ (p features), single output, y

Objective function:

\# layers

\#nodes in $l+1$ layer

$$J(\alpha, X, Y) = \sum_{i=1}^{n} \left(h_\alpha(x_i) - Y_i \right)^2 + \lambda \sum_{l=1}^{n_l} \sum_{i=1}^{s_{l+1}} \sum_{j=1}^{s_l} \left(\alpha_{ij}^{(l)} \right)^2$$

can also use

L0, L1 norms.

Fitting the model - regression

Training data: $\{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$

$x \in \mathbb{R}^p$ (p features), single output, y

Objective function:

layers

nodes in $\ell+1$ layer

nodes in ℓ

$$J(\alpha, x, y) = \sum_{i=1}^{n} \left(h_\alpha(x_i) - y_i\right)^2 + \lambda \sum_{\ell=1}^{n_\ell} \sum_{i=1}^{s_{\ell+1}} \sum_{j=1}^{s_\ell} \alpha_{ij}^{(\ell)}$$

Quick notes on objective function and finding parameters

- Form is amenable to classification, just one-hot encode the output and use classification error rate as your objective
- For regression, be sure to scale the *output* variables to lie in the range of the activation function.
  - For sigmoid, scale to: $[0, 1]$

Quick notes on objective function and finding parameters

- Form is amenable to classification, just one-hot encode the output and use classification error rate as your objective
- For regression, be sure to scale the *output* variables to lie in the range of the activation function.
  - For sigmoid, scale to: $[0, 1]$
  - Tanh: $[-1, 1]$

Quick notes on objective function and finding parameters

- Form is amenable to classification, just one-hot encode the output and use classification error rate as your objective
- For regression, be sure to scale the *output* variables to lie in the range of the activation function.
  - For sigmoid, scale to: $[0, 1]$
  - Tanh: $[-1, 1]$

Quick notes on objective function and finding parameters

- Form is amenable to classification, just one-hot encode the output and use classification error rate as your objective
- For regression, be sure to scale the *output* variables to lie in the range of the activation function.
  - For sigmoid, scale to: $[0, 1]$
  - Tanh: $[-1, 1]$
  - (I *believe* ReLU requires shifting output to be non-negative; textbook does not address.)

- Solving the objective function involves a form of gradient search
  - The partial derivatives are found via a technique called backpropogation

Tensorflow playground

On **this website** you'll find a cool interactive tool that allows you to play with NN for classification.

1. What are the hyperparameters of the model? Can you explain what each one does?
2. Try fitting the "exclusive or" (choose on top left) data set.
3. Also try fitting the "Spiral" data set.
4. Possible spiral solution:

Tensorflow playground

On **this website** you'll find a cool interactive tool that allows you to play with NN for classification.

1. What are the hyperparameters of the model? Can you explain what each one does?
2. Try fitting the "exclusive or" (choose on top left) data set.
3. Also try fitting the "Spiral" data set.
4. Possible spiral solution:
    1. Learning rate 0.03
    2. Two hidden layers, six and four neurons each
    3. Tanh activation
    4. Include all but X_1X_2 features.
    5. L1 regularization, regularization rate $= 0.001$

Tensorflow playground

On **this website** you'll find a cool interactive tool that allows you to play with NN for classification.

1. What are the hyperparameters of the model? Can you explain what each one does?
2. Try fitting the "exclusive or" (choose on top left) data set.
3. Also try fitting the "Spiral" data set.
4. Possible spiral solution:
    1. Learning rate 0.03
    2. Two hidden layers, six and four neurons each
    3. Tanh activation
    4. Include all but X_1X_2 features.
    5. L1 regularization, regularization rate $= 0.001$
5. You got close by trial and errror. What's another way?

Tensorflow playground

On **this website** you'll find a cool interactive tool that allows you to play with NN for classification.

1. What are the hyperparameters of the model? Can you explain what each one does?
2. Try fitting the "exclusive or" (choose on top left) data set.
3. Also try fitting the "Spiral" data set.
4. Possible spiral solution:
   1. Learning rate 0.03
   2. Two hidden layers, six and four neurons each
   3. Tanh activation
   4. Include all but $X_1 X_2$ features.
   5. L1 regularization, regularization rate $= 0.001$
5. You got close by trial and errror. What's another way?
   - Cross validation! Grid search, randomized search
   - But everything is computationally intense.

What's going on in the hidden layers?

Hover over the hidden layers in the tensorflow playground.

Q: What are we looking at?

What's going on in the hidden layers?

Hover over the hidden layers in the tensorflow playground.

Q: What are we looking at?

Ans: The scalar output of that neuron's activation function at each point in the feature space.

These can have interesting (but sometimes dubious) interpretations. More next time!