Info 251: Applied Machine Learning
Lab 10
4/1/2020

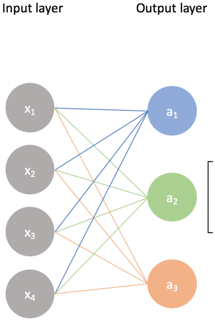
Topics

- Random Forests
- Neural networks
- Tensorflow

Neural Networks

- est 1940s
- Great success in AI
- Image recognition, speech processing etc

Neural Networks



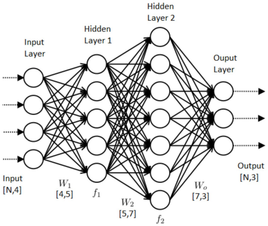
A simple neural network

$$\begin{bmatrix} w_1 & w_2 & w_3 & w_4 \\ w_1 & w_2 & w_3 & w_4 \\ w_1 & w_2 & w_3 & w_4 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}+\begin{bmatrix} b \\ b \\ b \end{bmatrix}=\begin{bmatrix} w_1x_1+w_2x_2+w_3x_3+w_4x_4+b \\ w_1x_1+w_2x_2+w_3x_3+w_4x_4+b \\ w_1x_1+w_2x_2+w_3x_3+w_4x_4+b \end{bmatrix}\xrightarrow[\text{activation}]{}\begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix}$$

- layer 1$\rightarrow$ input:x, output $a = g(W_1 x + b_1)$, $W \in \mathbb{R}^{3\times 4}$, $b \in \mathbb{R}^3$
- layer 2$\rightarrow$ input:$g(W_1 x + b_1)$, output $g(W_2 g(W_1 x + b_1) + b_2)$
- ...

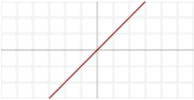
Neural Networks

- Universal function approximation theorem
- However, we don't know how many layers or their size
- How choose size then?
- Input layer
- Output layer
- Hidden layers
  - If data linearly separable need none!
  - Usually one is enough
  - Extra layers usually improve but add computational cost
  - Size of hidden layers
  - ROT: in between the size of input and output.
- Start with the above and keep iterating
- If we have unlimited computational power why not increase the dimensions a lot?

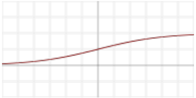


Neural Networks

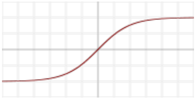
- How about activation functions?
- "Identity" $g(x) = x$



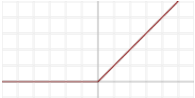
- "sigmoid" $g(x) = 1/(1 + e^{-z})$ *if* $x > 0$ *and* 0 *o.w.*



- "TanH" $g(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}$



- "ReLU" $g(x) = x$ *if* $x > 0$ *and* 0 *o.w.*



Neural Networks

- Choice of activation function?

Neural Networks

- Choice of activation function? Trial and error

Neural Networks

- Choice of activation function? Trial and error
- Objective function?
- Commonly, Regression $\rightarrow$ MSE $\frac{1}{n}\sum_{i=1}^{n}(y_i - \tilde{y}_i)^2$, Classification $\rightarrow$ CE $\sum_{i=1}^{n} -y_i log \tilde{y}_i - (1 - y_i) log(1 - \tilde{y}_i)$

Neural Networks

- Choice of activation function? Trial and error
- Objective function?
- Commonly, Regression $\rightarrow$ MSE $\frac{1}{n}\sum_{i=1}^{n}(y_i - \tilde{y}_i)^2$, Classification $\rightarrow$ CE $\sum_{i=1}^{n} -y_i log \tilde{y}_i - (1 - y_i) log(1 - \tilde{y}_i)$
- Data pre-processing
- Normalization?

Neural Networks

- Choice of activation function? Trial and error
- Objective function?
- Commonly, Regression $\rightarrow$ MSE $\frac{1}{n}\sum_{i=1}^{n}(y_i - \tilde{y}_i)^2$, Classification $\rightarrow$ CE $\sum_{i=1}^{n} -y_i log \tilde{y}_i - (1 - y_i) log(1 - \tilde{y}_i)$
- Data pre-processing
- Normalization?Yes! Better for optimization
- Categorical variables?
- Regularization?

Neural Networks

- How do we optimize
- GD?
- Tough composition of functions
- Backpropagation

NN: Implementation

- Tensorflow
- Python and C++ under the hood
- Create dataflow graphs-structures that describe how data moves through a graph
- Nodes represent mathematical operations
- Connections or edges between nodes are a multidimensional data arrays, or tensors
- Can use GPUs for computations
- (Remove) the feed dictionary specifies the placeholder values for that computation
- Other choices PyTorch, Caffe etc

- Notebook