Machine Learning
September 26, 2015
Ali M. Bramson

Neural Networks

- Processing power was an issue until 2006…took longer but didn't necessarily get better results than SVMs
- Deep Nets and CNNs are by today standards the best algorithm for Image Pattern Recognition
- NN and Deep Nets: Geoffrey Hinton, Yann LeCun and Yoshua Bengio are the main people – with Google, Facebook and University of Toronto

What is a neural network?

- Main idea is to mimic how a real neuron works. It has a nucleus and pathways that connect it to other equally simple units. Neurons that react the same to different inputs relevant – fire when exposed to music, image… do similar here but with weights going into nucleus and the output is function of inputs.
- Sum of Wixi + b -> regression equation. xi = features, Weights tell us importance of each feature… not sending variations of signal. But output is just 1 or 0 -> mostly used as classifiers. Discretize the outputs….one of most common activation functions it he sigmoid function. Logistical regression equation. f(z) = 1/1+exp(z). Perceptron just does logistical regression.
- Instead of modify inputs, put many units together. Maybe one classifier divides space one way, and then another divides space another….get lots of layers with lots of classifiers…. Can divide whether space is linear or not. By connecting multiple classifiers together, can create a more complicated classifier… neural network does feature engineering for you.
- Need to regularize NN since they can also be prone to overfitting.
- First layer is input layer- each neuron gets inputs from each features and there are weights with each. F is activation function (sigmoid) but now have a next layer of classifier that gets outputs of the first layer of classifiers. Each layer classifying different things; finding different lines.
- X1 could be sex, x2 could be ticket price, x3 could be class (in Titanic example), etc…for features. Size of input goes all the way, say, to 64. Each input is each feature. In an image, each would be a pixel.
- How many neurons is enough? The more neurons and layers the more powerful, the more classes you can make. Start with blank network- random weights. Get output, operate again, get output… in first pass, unless you're very lucky you're gonna get bad results. Have labels. Figure out error. So do feedforward-backpropogation… last layer says error due to inputs coming in from layer before.

- Instead of using a single classifier, we are using a bundle of them. Each classifier, via different weights will emphasize different parts of the input signal. For example, in an image, that could be shadows, or shapes, if image is red or blue, etc.
- Tensorflow – neural network library from google. If you want to learn a neural network library, use this because has google power behind it so people will develop things and it has support. Problem though is it doesn't work on windows…just linux and mac. Need an nvidia not amd video card. Can use without video card but it will take ages. Uses the graphical processor in this.
    - Playground.tensorflow.org
    - How thick the line in the visualization is gives how big weight
- 1 neuron = logistical regression. Sigmoid activation. How much data to train vs test…how noisy… as you make it noisy, harder to classify. If add two feature inputs, can get diagonal line to separate two clusters of points. Add x^2 feature, can get polynomial rather than straight line. Or x1x2 ends up shaping mistake more and messing up….
- If you put way too many units, its going to converge but will take a lot of time. Don't use too much level of complexity. Now if space looks like checkerboard…. Squared terms allow parabolas…but because only 2nd degree change, cant go down and back again…. Now if add 2 neurons can do it. Not one can do it but together than can. Most of weights are from original ones. But without squared terms, can only find lines, not parabolas. So need squared terms and 2 neurons even though weights not that big for squared terms. If keep only original input but do 3 neurons…classifier doing ok job without parabola…now 4, 5…getting good job…and each is only learning line. Now if add a second layer with only 2 neurons each…. But still only learning lines. This how train neural networks…try different things…but want smallest, most powerful neural network you can create.
- Want neural network to try to go off original features- don't usually want to add more features or transformations because apriori you usually don't know because you don't always know how the space looks like in say 100D space.
- Or how about spiral dataset…
- If have tons of computational power- can stack lots of neurons and layers…. But if you don't have that much power, you're gonna have to add extra features.
- Before 2006, was useless to use more than 4 layers…