Deep Learning A-Z: Hands-On AI

Luis Quintanilla

September 3, 2017

Contents

0.1 Section 1: Welcome to the Course

8/21/2017

0.1.1 What is Deep Learning

- Geoffrey Hinton - Godfather of DL. Check out youtube videos.
- Artificial Neural Nets
  - Made up of nodes
    - Input Nodes - Input Layer (known values)
    - Output Nodes - Output Layer (values to predict)
    - Hidden Layer - Between input and output layer.
  - Input -> Hidden -> Output
  - Deep Learning comes in when there are multiple hidden layers.

0.1.2 Installing Python

- Install Anaconda Python (version 3.5)
- Launch Spyder through Anaconda Navigator

0.1.3 Getting The Dataset

`http://superdatascience.com/deep-learning`

- Download Deep Learning A-Z Folder Template
- Download the dataset for each part of the course
- Look into Additional Reading mentioned throughout course

0.2 Section 2: Artificial Neural Networks

1 Part 1

1.1 Section 3: ANN Intuition

1.1.1 Plan of Attack

- What we will learn in this section:
  - The Neuron
  - The Activation Function
  - How do neural networks work? (example)

  - How do neural networks learn?
  - Gradient Descent
  - Stochastic Gradient Descent
  - Backprogation

1.1.2 The Neuron

- Basic building block of Artificial Neural Networks
- Neurons by themselves are useless. With lots of neurons, they work together.
- Dendrites are receivers of the signal and Axon is transmitter.
- Synapses - Term for connector that passes signal.
- Recreating on machine
  - Neuron (node)
    - Receives input (from input layer or other hidden layer neurons)
    - Generates output (to output layer or other hidden layer neurons)
    - Connected via synapses
  - Input Values
    - Independent variables
    - All for one single observation (one row in your database)
    - Need to be standadized (mean 0; variance: 1) or normalized (get values between 0 and 1)
      - Easier for neural network to process
  - Output Values
    - Continous, Binary, Categorical (several output values)
    - All for one single observation
  - Synapses
    - All assigned weights.
    - Weights is how the neural net works learn. By adjusting weights, neural net decides which signal gets passed along or not

    - ∗ Signals are sent into the neuron where they are added (weighted sum) of all input values and applies an activation function
    - ∗ Decision is made based on function whether signal will pass on or not

1.1.3 The Activation Function

- Threshold Function
  - 1 if x >= 0 | 0 if x < 0 (yes or no)
- Sigmoid Function
  - $\frac{1}{1+e^{-x}}$ (used in logistic regression)
  - Smooth gradual progression
  - Very useful in output layer
- Rectifier Function
  - $max(x, 0)$
  - One of most used
- Hyperbolic Tangent (tahn)
  - $\frac{1-e^{-2x}}{1+e^{-2x}}$
  - Can go below 0
- Exercise 1
  - Assuming DV is binary (y = 0 or 1)
    - ∗ Threshold Function
    - ∗ Sigmoid Function (probability of y being 1 or not)
- Exercise 2
  - In hidden layer apply rectifier
  - Output layer applies sigmoid function

8/22/2017

1.1.4 How do Neural Networks work?

Example: Property Valuations

- Input Layer
  - Area
  - Bedrooms
  - Distance to city
  - Age
- Output Layer
  - Price
- Hidden Layer
  - Connected with input layer
  - Some weights will have 0 value and others will not
  - Allows you to increase the flexibility or your neural network and look for specific things

1.1.5 How do Neural Networks learn?

Not rule-based. Neural network deciphers answer by itself given the inputs and a structure

- Single Layer Feedforward Neural Network (perceptron)
  - Input: x
  - Output: $\hat{y}$
  - Cost function: C $= \frac{1}{2}(\hat{y} - y)^2$ (most common)
    - ∗ What is the error that you have in your prediction
    - ∗ Goal is to minimize the cost function
  - Information is fed back through the network and weights are updated
  - Process is repeated with same observation until it converges
  - One epoch is when we go through a whole dataset and train our neural net on all observations
  - Cost function for multiple rows is $C = \sum \frac{1}{2}(\hat{y} - y)^2$. All rows share the same weights
  - Process of feedback is called backpropagation

1.1.6 Gradient Descent

- How do we reduce the cost function?
- The brute force method is to try out infinitely many weights to see which one is best. This is extremely inefficient due to the curse of dimensionality
- Gradient Descent starts somewhere and look at the slope is positive or negative (down or uphill)
- Then a "step" is taken and slope is recalculated to find out the next step using the new weight as the starting position

1.1.7 Stochastic Gradient Descent

- Gradient Descent is a way to help solve optimization problem of minimizing cost function
- However, Gradient Descent requires that the cost function is convex (parabola)
- If the cost function is not convex, Gradient Descent finds local minimum instead of global
- Stochastic Gradient Descent does not require a convex cost function
- With Stochastic Gradient Descent, the weights are updated on an observation basis as opposed to the aggregate after processing all observations
- Stochastic Gradient Descent allows you to avoid getting stuck at local minima
- Stochastic Gradient Descent has greater fluctuations
- Stochastic Gradient Descent is faster because all weights are not updated at the same time and data is not in memory
- Gradient Descent is deterministic algorithm and Stochastic Gradient Descent is Stochastic
- MiniBatch Gradient Descent method is a combination of both Gradient Desent and Stochastic Gradient Descent

1.1.8 Backpropagation

1.2 Section 4: Building an ANN

1.3 Section 5: Homework Challenge - Should we say goodbye to that customer?

1.4 Section 6: Evaluation, Improving and Tuning the ANN

1.5 Section 7: Homework Challenge - Put me one step down on the podium

1.6 Section 8: Convolutional Neural Networks