Machine Learning Crash Course

Week 5

I don't think that artificial intelligence means doomsday, and I think many new jobs will be created, too. However, it is becoming increasingly unlikely that these new types of jobs will favor low-income demographics. We need to address the needs of those who will be left out of the new job market." ~ Jens Martin Skibsted

Nothing is more important, certainly during these times of artificial intelligence, than our public education. And as it continues to grow and evolve, I think you and I know this is going to be critical that we are constantly training and retraining and creating these next-generation jobs.” ~ Marc Benioff

Topics

- Neural Networks
  - Cost Function
  - Overview
  - Backpropagation Algorithm
  - Intuition
- Implementation
  - Unrolling Parameters
  - Gradient Checking
  - Initializations
- Summary

1 Neural Networks

$$J(\theta) = -\frac{1}{m}(\sum_{i=1}^{m} y^i log h_\theta(x^i) + (1 - y^i) log(1 - h_\theta(x^i))) + \frac{\lambda}{2m}\sum_{j=1}^{n} \theta_j^2$$

Cost Function

- Neural Network cost can be conceptualized in 2 parts:
  - For ∀ “training” data (1 -> m):
    - Sum(each element of the “output” vector)
  - Calculate the weight decay term

- **DEMO**

Overview

- Forward Propagation
  - Pushes the input through the NN
- Back Propagation
  - Receives the NN output
  - Receives the H(θ)-y from a given layer L
  - Partial Derivatives & gradient descent to minimise the cost function J(θ)

Backpropagation

- Goal to find parameters θ that min(J(θ))
  - Calculate the **error** of the nodes in each layer
  - Calculate the **activation** of the nodes in each layer
  - Work backwards from the output layer
  - Obtain the **deltas** to calculate the partial derivative (**gradient**) of the cost function
- **(DEMO)**

2 Implementation

Unrolling Parameters

- Advanced Optimization Technique
  - Unroll the matrices into vectors for the algorithm that will min(cost)

- **DEMO**

Gradient Checking

- Verification that the cost function is decreasing over time
- Additional methodology to check for correct implementation
- Disable when training a given model due to time complexity

- **DEMO**

Initializations

- Initializing θ
  - Random Initializations
- Initializing α
  - Using an Advanced Optimization Algorithm

- **DEMO**

3 Summary

Summary

- Init weights randomly
- Implement forward propagation
- Implement cost function
- Implement backpropagation
- Utilize gradient checking
- Utilize advanced optimization

- **“CODE” DEMO**
- **https://github.com/TensorFlow-ML-Architectures/TensorFlow-Multilayer-Perceptron**

Thanks!

Any questions?

You can find me at:

http://cs.oswego.edu/~kzeller

Sources

- https://developers.google.com/machine-learning/crash-course/introduction-to-neural-networks/video-lecture
- https://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec3.pdf
- https://www.ritchieng.com/
- https://www.uio.no/studier/emner/matnat/ifi/INF5860/v17/undervisningsmateriale/in5860_lecture7_nnet2.pdf
- https://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example/