# Linear regression:

## Cost function

## Hypothesis

## Gradient descent algorithm

Repeat until convergence {

}

α: Learning rate

Plotting J(θ) as a function of number of iterations will show us how good is our chosen α

# Logistic regression:

## Cost function

## Hypothesis

## Notes

# Support vector machine

## Hypothesis

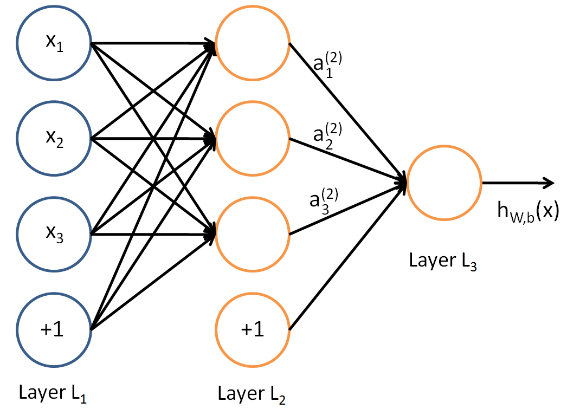
## Cost function

## Notes

1. It’s called large margin classifier because:

# Neural Network

## Feed-forward propagation



L = 4, s1 = 4, s2 = 4, s3 = 1

in ai(j), j is layer number and i is index

L: total number of layers in network

sl: number of units (not consisting bias unit) in layer l

In binary classification there will be one output unit, y = 0 or 1

In multi-class classification there will be k (> 2) output units, where y is in RK.

E.g.

## Cost function

## Backpropagation algorithm

Given training set {(x(1), y(1)), …, (x(m), y(m))}:

Set for all (l, i, j)

For training example i=1 to m {

Set

Perform forward propagation to compute a(l) for l = 2, 3, …, L

Using y(i), compute

Compute δ(L-1), δ(L-2), …, δ(2) using

or with vectorization

}

### Notes:

The capital-delta matrix is used as an "accumulator" to add up our values as we go along and eventually compute our partial derivative.

The actual proof is quite involved, but, the  terms are the partial derivatives and the results we are looking for

# Appendix

m: Number of training examples

n: Number of features

x: Training set

h: Hypothesis

θ: parameters, weights

σ: standard deviation

λ: regularization parameter

## Feature scaling

### Mean normalization

### Rescaling/Min-max scaling

### Standardization/Z-score normalization

* Gives us mean of 0 and variance of 1

## Sigmoid function

## Optimization algorithms

These algorithms provide optimized solution to calculate cost function and its derivitives.

### Gradient descent

Repeat until convergence {

}

α: Learning rate

Plotting J(θ) as a function of number of iterations will show us how good is our chosen α

Slower compared to other optimizers

### Conjugate descent

### BFGS

### L-BFGS

## Normal equation

Solve for θ analytically

Slow if n is very large