Machine Learning – Lesson 1

**Linear Regression – Single Variable**

Hypothesis function

Cost function

*m* = number of data points  
*x(i)*= *ith* input data point  
*y(i)* = *ith* result data point

Gradient Descent - General

= learning rate (selected by user)

Gradient Descent - Linear Regression

For j = 0:

For j = 1:

Therefore:

Gradient Descent Points

* Each “step” of gradient descent will decrease as it converges, even if *α* is kept constant, because the partial derivative gets smaller at each step.
* If α is too large, no convergence. If α is too small, will take forever to converge.
* This is called “batch” gradient descent, because each step of gradient descent uses all training examples . There are other methods that look at a subset rather than the whole batch.
* Make sure to update & simultaneously!

Machine Learning – Lesson 2

**Linear Regression – Multiple Variables**

Hypothesis function

*n* = number of features

Vectorized format:

= feature vector of size n + 1  
 *θ* = parameter vector of size n + 1

Cost function

*x(i)*= input feature vector of *ith* training example  
*xj(i)*= feature value *j* in the vector of *ith* training example

Gradient Descent - General

Gradient Descent – Linear Regression

Feature Scaling

* Get every feature in a -1 ≤ *xi* ≤ 1 range
* Scale by: or
* Mean normalization: or

Polynomial Regression

(Quadratic)  
 (Cubic)  
 (Square Root)

Feature Scaling with Polynomial Regression

* The feature range becomes the *featuren* where *n* is the highest order polynomial

Normal Equation

*X* = matrix of training examples with feature values (*m x n + 1* matrix)

Make sure to include a column vector of ones in *X* to represent *x0*

Important Points for Normal Equation

* Analytical “closed-form” solution rather than using an iterative gradient descent algorithm
* Above is for linear least squares regression cost function only
* Advantages: No need to choose α, No need to iterate
* Disadvantages: Slow if number of features *n* is really large (need to invert a large matrix)

Machine Learning – Lesson 3

**Classification (Logistic Regression)**

Hypothesis function

Cost function

:

* should equal 1 if y = 1.
* Cost →0 if = 1, Cost →∞ if = 0

:

* should equal 0 if y = 0
* Cost →0 if = 0, Cost →∞ if = 1

Gradient Descent

Advanced Optimization

Built-in optimization algorithm in Octave

* *[optTheta, functionVal, exitFlag] = fminunc(@costFunction, initialTheta, options)*
* @costFunction is the cost function that you have defined as an input
* exitFlag = 1 when it converges

Multiclass Classification

* Instead of *y = 0* or *y = 1*, you have *y = 0, 1, 2, …n* classes
* Run it as several classification problems: for each class, with y = 0 or y=1
* On a new input *x*, to make a prediction, pick the class I that maximizes max

Regularization

* Used for controlling over-fitting or under-fitting data

Least-squares Regression Regularization

* *λ* is the regularization parameter set by user
* The larger *λ*, the more higher-order polynomials are surpressed
* Typically don’t regularize first parameter

Least-squares Gradient Descent

* *(1 – αλ/m)* is a factor < 1, so it reduces the contribution of . Everything else is same.

Normal Equation Regularization

* *I\** is the identity matrix without the first element.

Logistic Regression Cost Function with Regularization

Logistic Regression Gradient Descent with Regularization

* Looks the same as linear regression, but is different

Machine Learning – Lesson 4

Neural Networks

For a neural network with 1 hidden layer:

Hidden Layer

Output Layer

* x is a n+1 vector of features (n is the number of features)
* is a matrix of parameters for layer j.
* Rows: Parameter associated with activation unit *i*
* Columns: Parameter associated with input feature *n*
* If network has *sj* units in layer *j*, *sj+1*units in layer *j+1*, then

Dimension of