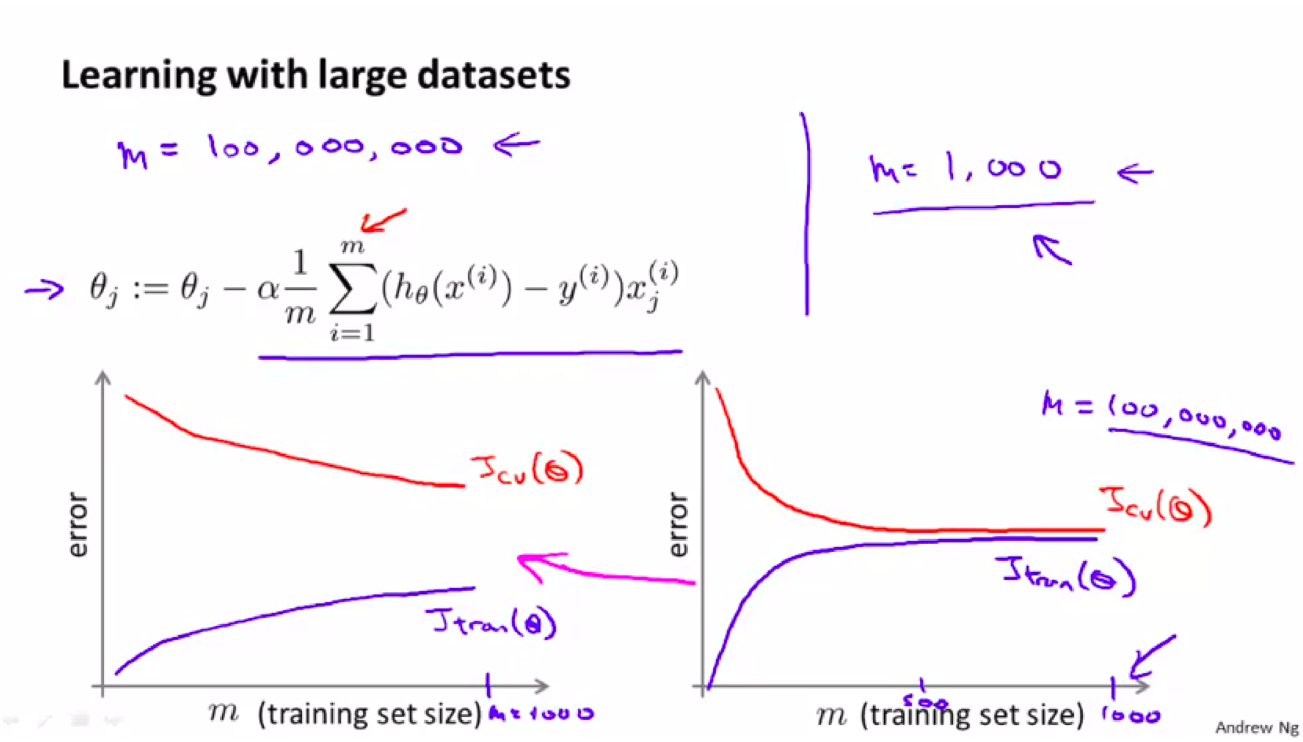
Week 10

- Gradient descent with large dataset
    - Learning with large dataset
    - Stochastic gradient descent
    - Mini-batch gradient descent
    - Stochastic gradient descent convergence
- Advanced topics
    - Online learning
    - MR

I. Gradient descent with large dataset

1. Learning with large dataset



2. Stochastic gradient descent

Linear regression with gradient descent

$$h_\theta(x) = \sum_{j=0}^{n} \theta_j x_j$$

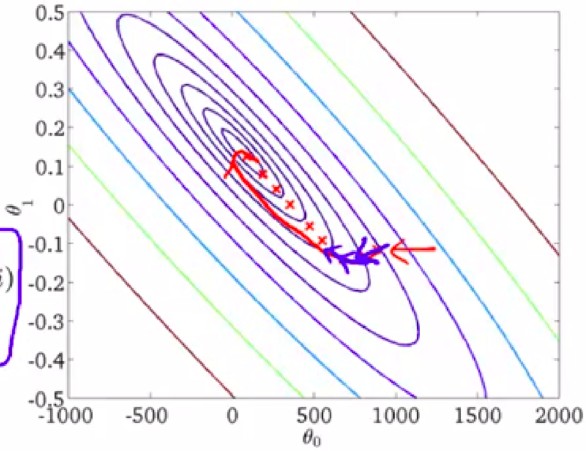
$$J_{train}(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2$$

Repeat {

$$\theta_j := \theta_j - \alpha \boxed{\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)}}$$

(for every $j = 0, \dots, n$)

}



$M = 300,000,000$

Batch gradient descent

Batch gradient descent

$$J_{train}(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2$$

Repeat {

$$\theta_j := \theta_j - \alpha \underbrace{\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)}}_{\frac{\partial}{\partial \theta_j} J_{train}(\theta)}$$

(for every $j = 0, \dots, n$)

}

Stochastic gradient descent

$$cost(\theta, (x^{(i)}, y^{(i)})) = \frac{1}{2}(h_\theta(x^{(i)}) - y^{(i)})^2$$

$$J_{train}(\theta) = \frac{1}{m}\sum_{i=1}^{m} cost(\theta, (x^{(i)}, y^{(i)}))$$

1. Randomly shuffle dataset.

2. Repeat {

for $i = 1, \dots, m$ {

$$\theta_j := \theta_j - \alpha \underbrace{(h_\theta(x^{(i)}) - y^{(i)}) \cdot x_j^{(i)}}_{\frac{\partial}{\partial \theta_j} cost(\theta, (x^{(i)}, y^{(i)}))}$$

(for $j = 0, \dots, n$)

}

}

$(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), (x^{(3)}, y^{(3)}), \dots$

Andrew Ng

Stochastic gradient descent

1. Randomly shuffle (reorder) training examples

2. Repeat {

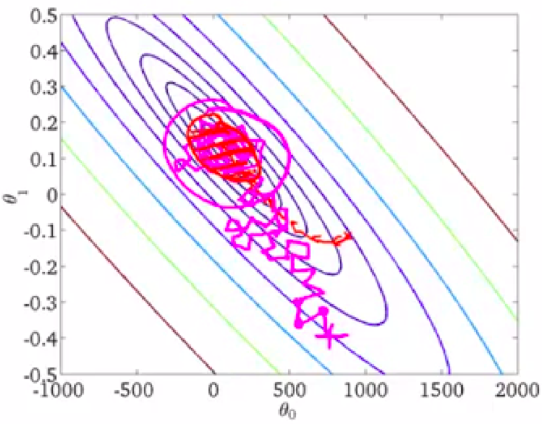
   for $i := 1, \ldots, m$ {

   $$\theta_j := \theta_j - \alpha(h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)}$$

   (for every $j = 0, \ldots, n$)

   }

}



m can be 3,000,000

3. Mini-Batch Gradient Descent

- Batch gradient descent: use all m examples in each iteration
- Stochastic gradient descent: use 1 example in each iteration
- Mini batch gradient descent: use b example in each iteration

Mini-batch gradient descent

Say $\boxed{b = 10}$, $m = 1000$.

Repeat {

for $i = 1, 11, 21, 31, \ldots, \boxed{991}$ {

$$\theta_j := \theta_j - \alpha \frac{1}{\boxed{10}} \sum_{k=i}^{\boxed{i+9}} (h_\theta(x^{(k)}) - y^{(k)})x_j^{(k)}$$

(for every $j = 0, \ldots, n$)

}

}

Why mini batch gradient look at b examples and stochastic gradient look at 1 example?

$\Rightarrow$ the answer is the vectorization in the $\sum$ part.

4. Stochastic Gradient Descent Convergence

Checking for convergence

- Batch gradient descent:
  - Plot $J_{train}(\theta)$ as a function of the number of iterations of gradient descent.
  - $J_{train}(\theta) = \frac{1}{2m} \boxed{\sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2}$

$M = 300,000,000$

- Stochastic gradient descent:
  - $cost(\theta, (x^{(i)}, y^{(i)})) = \frac{1}{2}(h_\theta(x^{(i)}) - y^{(i)})^2$
  - During learning, compute $cost(\theta, (x^{(i)}, y^{(i)}))$ before updating θ using $(x^{(i)}, y^{(i)})$.
  - Every 1000 iterations (say), plot $cost(\theta, (x^{(i)}, y^{(i)}))$ averaged over the last 1000 examples processed by algorithm.

$\rightarrow (x^{(i)}, y^{(i)}), (x^{(i+1)}, y^{(i+1)}), \ldots$

Andrew Ng

Stochastic gradient descent

$$cost(\theta, (x^{(i)}, y^{(i)})) = \frac{1}{2}(h_\theta(x^{(i)}) - y^{(i)})^2$$

$$J_{train}(\theta) = \frac{1}{2m} \sum_{i=1}^{m} cost(\theta, (x^{(i)}, y^{(i)}))$$

1. Randomly shuffle dataset.
2. Repeat {

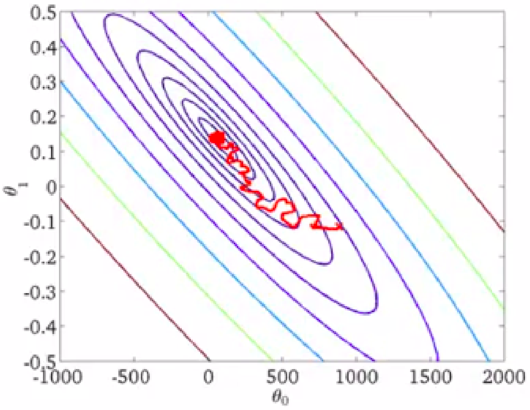
    for $i := 1, \ldots, m$ {

    $\theta_j := \theta_j - \alpha(h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)}$

    (for $j = 0, \ldots, n$)

    }

}



Learning rate α is typically held constant. Can slowly decrease α over time if we want θ to converge. (E.g. $\alpha = \frac{\texttt{const1}}{\boxed{\texttt{iterationNumber}} + \texttt{const2}}$) $\alpha \rightarrow 0$

Andrew Ng

II.Advanced topics

1. Online learning

Learn $p(y = 1|x; \theta)$

CTR

2. Map Reduce and Data Parallelism

$\boxed{m = 400}$ $\quad m = 400{,}000{,}000$

Map-reduce

Batch gradient descent: $\theta_j := \theta_j - \alpha \frac{1}{400} \sum_{i=1}^{400} (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)}$

$(x^{(1)}, y^{(1)})$
$\vdots$
$(x^{(m)}, y^{(m)})$

Machine 1: Use $(x^{(1)}, y^{(1)}), \ldots, (x^{(100)}, y^{(100)})$.

$temp_j^{(1)} = \sum_{i=1}^{100} (h_\theta(x^{(i)}) - y^{(i)}) \cdot x_j^{(i)}$

Machine 2: Use $(x^{(101)}, y^{(101)}), \ldots, (x^{(200)}, y^{(200)})$.

$temp_j^{(2)} = \sum_{i=101}^{200} (h_\theta(x^{(i)}) - y^{(i)}) \cdot x_j^{(i)}$

Machine 3: Use $(x^{(201)}, y^{(201)}), \ldots, (x^{(300)}, y^{(300)})$.

$temp_j^{(3)} = \sum_{i=201}^{300} (h_\theta(x^{(i)}) - y^{(i)}) \cdot x_j^{(i)}$

Machine 4: Use $(x^{(301)}, y^{(301)}), \ldots, (x^{(400)}, y^{(400)})$.

$temp_j^{(4)} = \sum_{i=301}^{400} (h_\theta(x^{(i)}) - y^{(i)}) \cdot x_j^{(i)}$

Combine:

$$\theta_j := \theta_j - \alpha \frac{1}{400} \left(temp_j^{(1)} + temp_j^{(2)} + temp_j^{(3)} + temp_j^{(4)} \right)$$

$(j = 0, \ldots, n)$