10주차 - Large Scale Machine Learning

- ## Gradient Descent with Large Datasets
  - Learning with Large Dataset

데이터 크기(m)가 100,000,000정도 되면 일반적인 경사하강법으로는 학습 시간이 너무 오래 걸린다.
아래와 같이 한 단계 Theta를 학습시키기 위해서는 sigma over m을 계산해야하는데, iteration = i라고 한다면
학습시간은 대략 O(mi)가 될 것이다. 따라서 큰 데이터에 대해서는 좀 더 효율적인 학습 방법을 도입하게 된다.
한편, 이렇게 모든 데이터를 확인하여 경사하강을 시도하는 방법을 'Batch Gradient Descent'라고 한다.

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

  - Stochastic Gradient Descent

확률 경사 하강(Stochastic Gradient Descent)은 한 단계 움직일 때 마다 모든 데이터를 다 확인하는 것이 아니라 한개의 데이터만 확인하여 최적 theta를 구하는 방법이다.
아래 그림과 같이 선형회귀에 대한 경사하강법을 예로 들면,

Linear regression with gradient descent

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

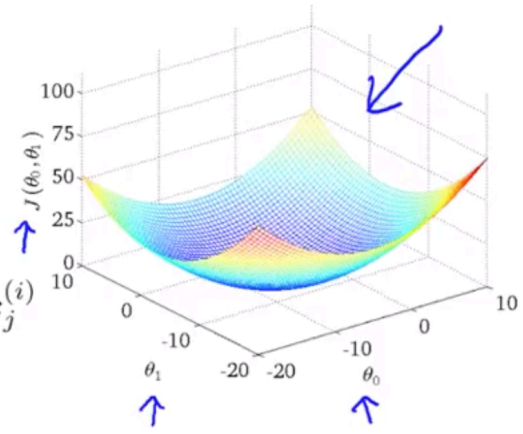
$$J_{train}(\theta) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2$$

Repeat {

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

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

}



Batch gradient descent와 다르게 비용함수(J)가 아니라 새로운 비용함수(cost)의 도함수를 이용하는 것을 볼 수 있다. 함수를 새로 정의 했지만 결국은 i번째 데이터 하나만 본다는 뜻이다. 확률 경사하강은 우측 하단 파란 글씨와 같이 2가지 과정으로 진행된다.

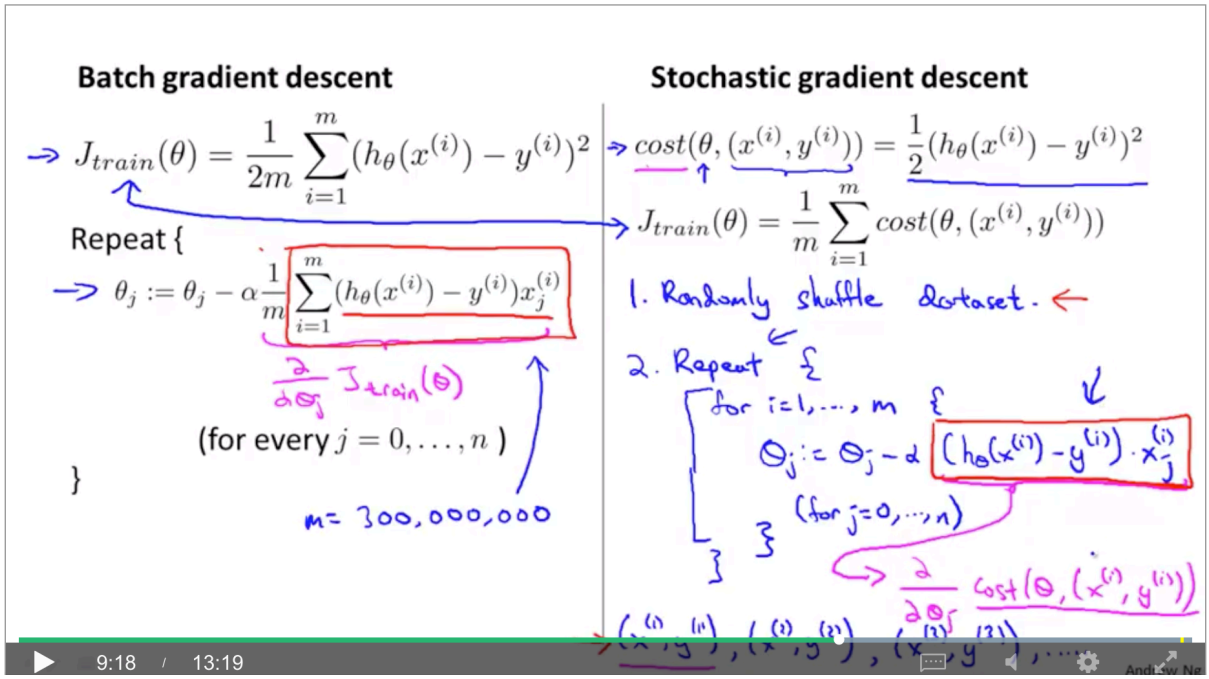
1. 데이터를 무작위로 섞는다

섞지 않는다면 학습 순서 자체가 파라미터가 되어버린다. (학습 순서에 편향될 수 있다.)

2. 한 단계 진행할 때 하나의 데이터에 대한 경사하강을 시도한다.(모든 데이터에 대하여 2번 반복)

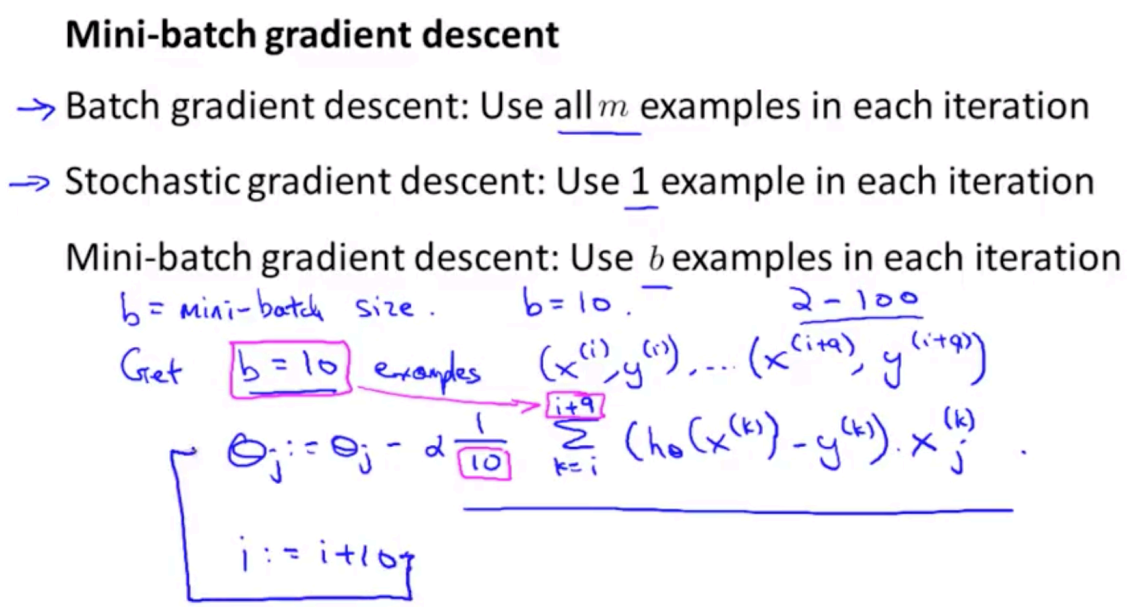
-> 비용함수가 일정 이하가 아니라면 다시 첫번째 데이터부터 2번 시작. (이 과정은 주로 1~10번정도 진행된다)

Stochastic gradient descent의 장점은 큰 데이터에 대해서도 빠르게 작동한다는 것이지만, 비용함수를 모든 test dataset에 대해 고려하지 않으므로 local minimum이 아니라 그 인근 area에서 계속 진동할 수 있다. 즉 local minimum을 정확히 보장하지는 않는다. 하지만 일반적인 경우에 local minimum에 정확히 맞출 필요가 없으므로 대부분 괜찮다.

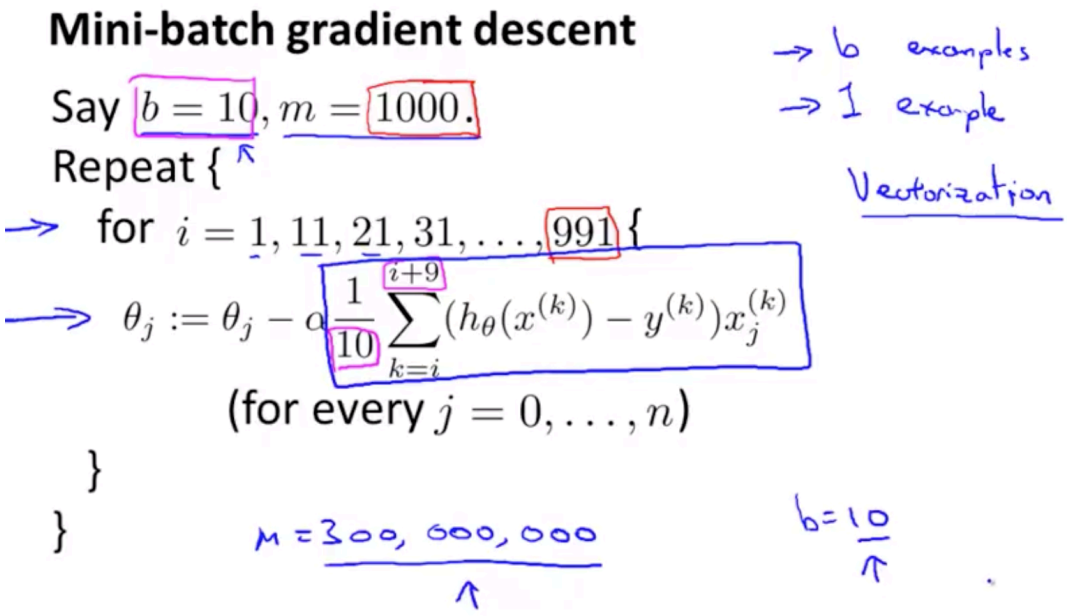


- Mini-Batch Gradient Descent

이번에는 Batch 경사하강보다는 덜 엄격하고 확률 경사하강보다는 엄격한 경사하강법이다.
b개의 example(일반적으로 2~100개를 이용)을 보고 도함수를 구하는 것이다. 확률 경사하강법과 크게 다른 컨셉은 아니다. 확률 경사하강법이 매 데이터마다 한 단계 theta를 업데이트한다면 mini-batch 경사하강법은 b개의 데이터마다 한 단계 업데이트한다.



mini-batch 경사하강법은 경우에 따라서 확률 경사하강보다 효율적일 수 있는데, 그 이유는 적절한 b의 크기와 vectorization된 경사하강법 계산이 가능하다면 좀 더 빨리 최적값에 도달할 수 있기 때문이다. 그리고 마찬가지로 모든 데이터를 사용하지 않기 때문에 local minimum을 정확히 구할 수 는 없다.



- Stochastic Gradient Descent Convergence

이전에 Stochastic과 mini-batch 경사하강법은 local minimum을 정확히 알아낼 수 없다고 했다. 따라서 위 알고리즘이 수렴하는지 알 수 있는 방법이 필요하다.
기존 batch 경사하강법에 경우에는 단순히 매 iteration마다 J를 그리면 되었다.
하지만 Stochastic을 도입하는 목적이 효율을 위함인데 매 iteration마다 J를 계산할 수 없으므로 Stochastic 경사하강법의 경우에는 이전에 처리된 데이터 m'(예를들어 1000)에 대한 비용함수(cost)를 평균을 내서 표현한다. m'의 크기가 커질수록 좀 더 정교하고 구체적인 비용함수 양상을 볼 수 있다.

아래 그림을 보면 비용함수의 양상을 총 4가지로 나눌 수 있음을 볼 수 있다. 우측 상단부터 ㄷ모양으로 1,2,3,4 그래프라고 해보자. (파란그래프는 공통적으로 1000개의 데이터마다 평균 비용(cost)를 그린 것이다.

1. 각 데이터마다 그림을 그려도 local minimum으로 수렴하는 것을 쉽게 볼 수 있다.

-> 빨간 그래프는 데이터 간격을 5000개로 바꾼 것이다. 조금 더 정교하다

2. learning rate(alpha)를 달리하여 그래프를 그린것이다.

-> 빨간 그래프가 더 learning rate가 작다, 학습은 느리지만 좀 더 local minimum에 가까워진다. (진동이 작아짐)

3. 파란 그래프만 보고는 수렴 양상을 파악하기 힘들다.

-> 평균을 내는 데이터를 5000개로 늘려서 좀 더 자세히 볼 수 있다.

4. 데이터가 발산한다.

-> learning rate가 너무 높은 것에 기인한다.