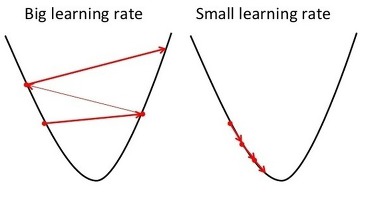
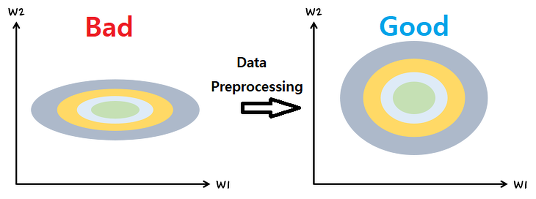
Gradient Descent Algorithm의 공식 **http://cfile9.uf.tistory.com/image/990980395A77F90612B0B1**을 보면 Cost Function의 미분값에 http://cfile21.uf.tistory.com/image/9982A93E5A77F94029A3FB를 곱한것을 볼 수 있습니다. 여기서 이 http://cfile21.uf.tistory.com/image/9982A93E5A77F94029A3FB가 바로 **Learning Rate(학습율)**를 의미합니다.



Learning Rate가 **너무 크게 잡혀있는 경우** Cost Function의 최소지점을 찾지 못하고 점점 **이상한 방향으로 학습을 하는 Overshooting**의 경우가 생기고, 만약 Learning Rate가 **너무 작게 잡혀있는 경우** Cost Function의 최소지점을 찾기까지의 **학습단계가 너무 많이 필요**한 경우가 생길 수 있기 때문입니다.

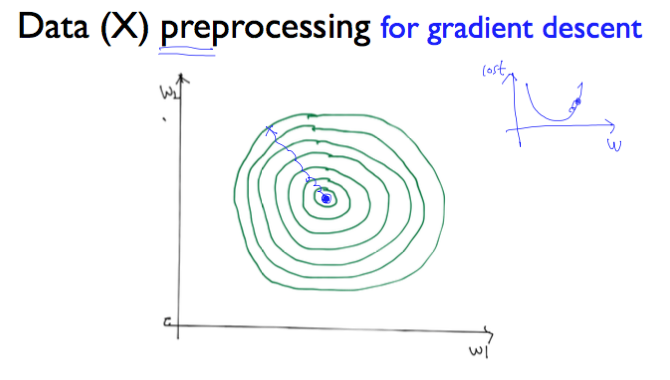
**일반적으로 Learning Rate는 0.01으로 많이 설정**

**Data Preprocessing(데이터 전처리)**

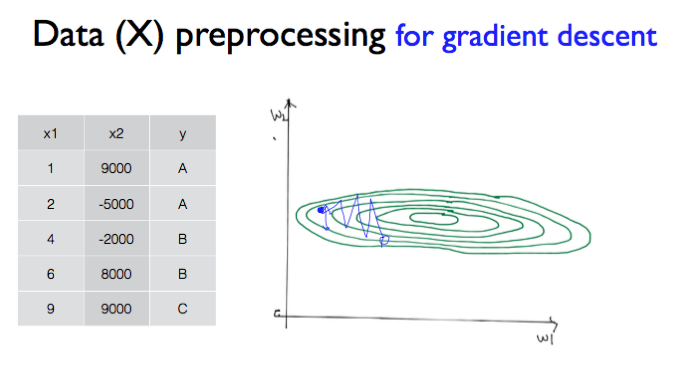


  Learning Rate를 적절하게 설정해준것 같은데 학습을 해보면 좋지 않은 결과가 나오는 경우가 있습니다. 이런 경우는 데이터에 조금 문제가 있을 때 발생할 수 있습니다. 이럴때 우리는 **Data Preprocessing(데이터 전처리)**를 해주어야 합니다.

|  |  |
| --- | --- |
|  |  |

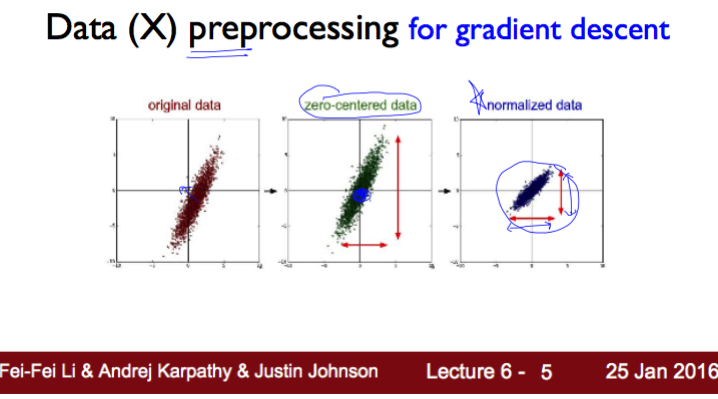


multi-variables 모델에서, 변수라고 부르는 feature가 여러 개 있는 경우의 그래프는 위의 그림처럼 등고선 내지는 3차원 입체 이상으로 표현되어질 수밖에 없다. 등고선의 한 점에서 gradient descent 알고리듬으로 등고선의 중심인 최저점을 향해 간다고 할 때 매끄럽게 보이는 직선처럼 이동하는 것은 매우 어려울 수밖에 없다.

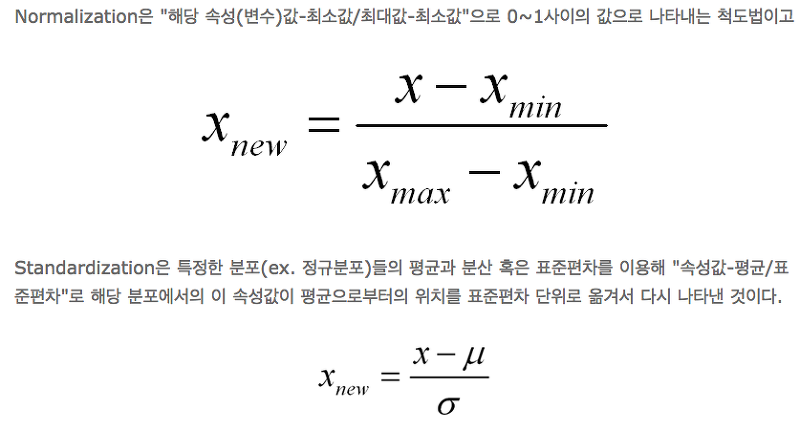


x1 변수는 10보다 작은 숫자, x2 변수는 -5000에서 9000까지의 숫자라면 진짜 동그랗게 생긴 원의 모양이 아니라 한쪽으로 길게 늘어진 타원 모양이 된다. 이렇게 된다면 수평으로 이동할 때와 수직으로 이동할 때 엄청난 불균형이 발생하게 되어 gradient descent 알고리즘을 적용하기 어려운 상황이 될 수 있다.

등고선으로 표현할 때, 가장 좋은 형태는 완벽하게 둥근 원(circle)이다.

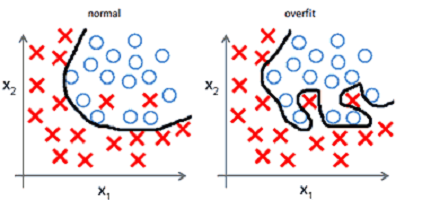


Feature Scaling은 변수의 범위를 일정하게 혹은 비교할 수 있도록 만드는 것을 말한다. Normalization(Re-scaling)과 Standardization의 두 가지가 있다.

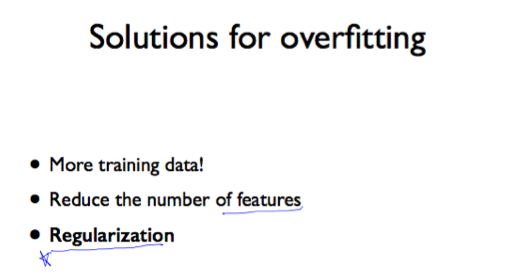


  **Normalization**  
  수식 : (요소값 - 최소값) / (최대값 - 최소값)  
  설명 : 전체 구간을 0~100으로 설정하여 데이터를 관찰하는 방법으로, 특정 데이터의 위치를 확인할 수 있게 해줌  
  
  **Standardization**  
  수식 : (요소값 - 평균) / 표준편차  
  설명 : 평균까지의 거리로, 2개 이상의 대상이 단위가 다를 때, 대상 데이터를 같은 기준으로 볼 수 있게 해줌

**Overfitting(과적합)**

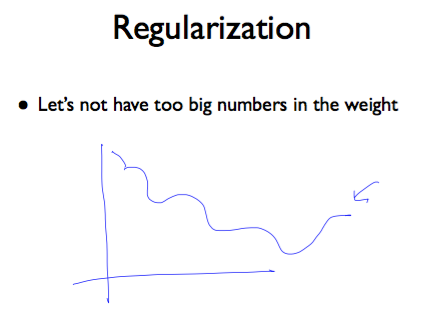


**Overfitting(과적합)**은 **Machine Learning에서의 가장 큰 문제**들 중 하나입니다. Overfitting은 말 그대로 **Training Data Set에 너무 적합**해진 현상을 의미합니다.

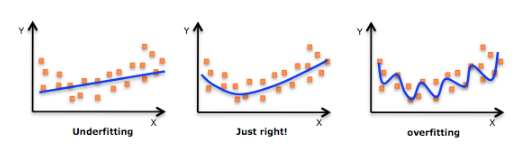


Overfitting의 해결법 3가지

**. training data가 많을 수록 좋다  
  . 입력으로 들어오는 변수(feature, x)의 개수를 줄여라  
  . Regularization을 사용해라**



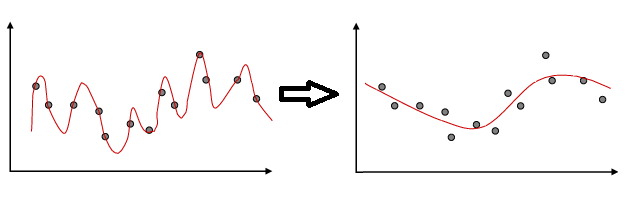
Regularization은 W(weight)가 너무 큰 값들을 갖지 않도록 하는 것을 말한다. 값이 커지면, 그림에서 보는 것처럼 구불구불한 형태의 cost 함수가 만들어지고 예측에 실패하게 된다. 머신러닝에서는 "데이터보다 모델의 복잡도(complexity)가 크다"라고 설명한다. 과도하게 복잡하기 때문에 발생하는 문제라고 보는 것이다. 다시 말하면, Regularization은 모델의 복잡도를 낮추기 위한 방법을 말한다.



모델을 구축했을 때의 3가지 경우.

W가 크면 예측하려는 값(y hat)이 정상적인 규칙으로부터 벗어나 있는 경우에도 예측이 가능하다. 그러나, 비정상적이거나 애매한 위치에 있는 데이터를 올바르게 예측하는 것을 '맞았다'라고 얘기할 수는 없다. 오히려 '틀렸다'라고 얘기하는 것이 더욱 좋을 수 있다. 이럴 경우 training dataset에 특화된 overfitting 현상이 발생한다. 가장 이상적인 경우는 최소한의 에러를 인정하는 'Just right'이다.

**Regularization(일반화)**



**Regularization(일반화)**은 데이터를 표현하는 모델을 최대한 덜 울퉁불퉁하게 표현하는 것을 의미합니다. 이러한 Regularization을 수식으로 표현하면 http://cfile22.uf.tistory.com/image/991444505A7810A3165F07와 같습니다. 여기서 http://cfile5.uf.tistory.com/image/9967BA455A7810DB021960는 **Regularization Strength**를 의미하는데 이 값을 0으로 설정하면 Regularization을 하지 않겠다는 의미이고 1으로 설정하면 Regularization을 많이 적용한다는 의미입니다. 이 값은 상황에 따라서 적당하게 설정해주면 되는 값입니다.

W에 대한 값이 클 경우에 penalty를 부여.

W에 대해 제곱을 한 합계를 cost 함수에 더하고, 람다(λ) 값을 사용해서 얼마나 penalty를 부여할 것인지 결정할 수 있다.

출처: <http://pythonkim.tistory.com/23> [파이쿵]