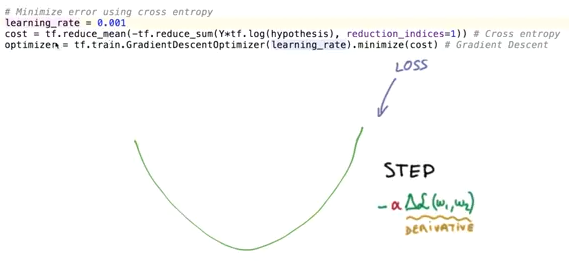
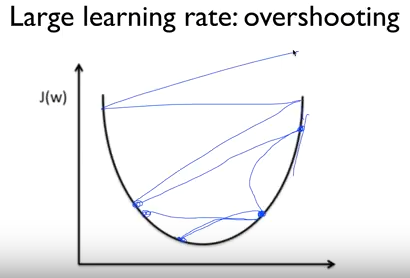
**7-1장. 학습 rate, Overfitting, 그리고 일반화(Regularization)**

1. **Learning rate**

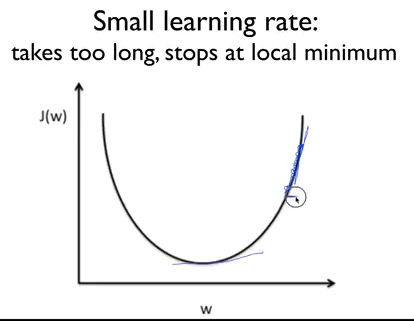
Gradient descent 알고리즘을 정할 때 α (learning rate)를 임의로 정했었다.

α (learning rate)는 경사면을 따라서 내려가는 한 step의 크기이다.





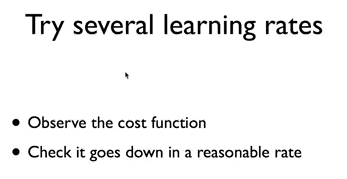
(\*) learning rate 가 커서 cost값이 줄어들지 않고, 더 커지는 경우를 overshooting 이라 한다.



(\*) learning rate 가 작아서 cost값이 거의 변하지 않으면 learning rate를 더 큰 값으로 변경해야 한다.

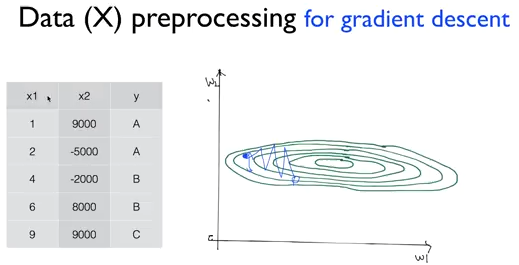
(\*) learning rate 는 환경에 따라 다르므로, 초기값을 0.01로 시작하여, cost값의 변화추이를 보고,

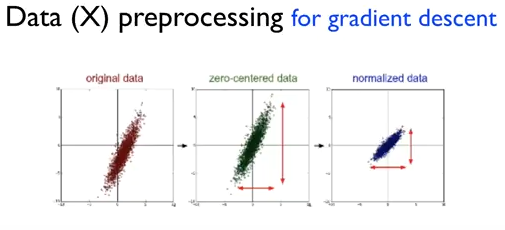
cost값이 점점 커지면 learning rate를 더 작은값으로, cost값이 너무 늦게 줄어들면 learning rate를 더 큰 값으로 변경해 가면서 테스트해 보도록 한다.



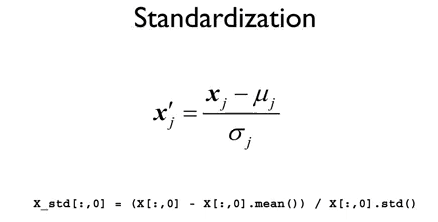
1. **Data preprocessing (데이터 선처리)**

: gradient descent 알고리즘에서 x1, x2 데이터간의 값 차이가 너무 커서 learning rate가 좋은 값임에도 불구하고, 최소값을 찾지 못하는 경우에는 데이터 선처리가 필요하다.



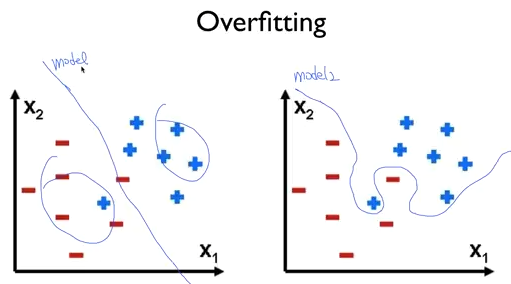


(\*) normalization 방법 (μ:평균, σ:분산)



1. **Overfitting**

학습 데이터에 너무 딱 맞는 모델을 만들게 되어, 실제 데이터에는 잘 맞지 않게 되는 것



(\*) 왼쪽의 model이 일반적인 model이고, 오른쪽의 model은 overfitting된 model이다.

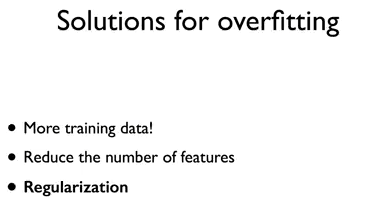
Training data에는 최적이지만, 다른 데이터에는 틀린 예측값을 리턴할 수 있게 된다.

(\*) overfitting 을 줄이기 위한 방법

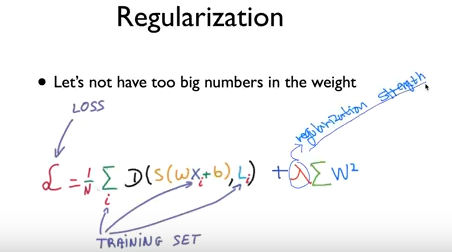
(1) 많은 Traning data

(2) 중복된 데이터(feature)가 있으면 줄인다

(3) 일반화 시킨다



1. **Regularization (일반화)**



λ 값으로 Regularization 을 사용할지 여부를 정할수 있다

(\*) 파이썬에서는 아래 문장으로 Regularization을 구현할 수 있다.



