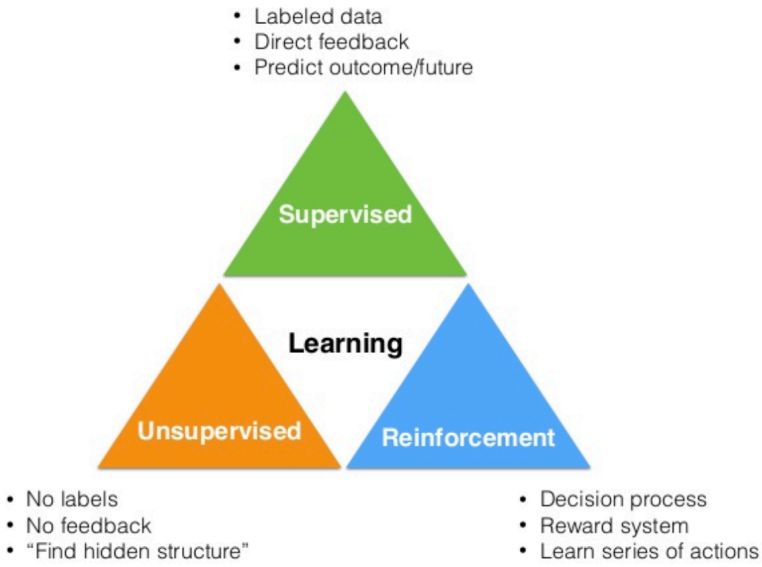
지도학습 - Regression

기계학습

기계학습(머신러닝, Machine Learning)이란 데이터를 이용해 컴퓨터를 학습시키는 방법론을 말한다. 이러한 기계학습 알고리즘은 일반적으로 지도 학습(supervised learning), 비지도 학습(unsupervised learning) 그리고 강화학습(reinforcement learning) 세 가지로 나누어진다. 여기에 지도학습과 비지도학습의 중간적인 성격을 가지는 (semi-supervised learning)을 추가적으로 분류하기도 한다.




지도학습

지도학습을 이해하는 가장 쉬운 방법은 지도학습과 비지도학습을 비교해보는 것이다. 비지도학습을 나누는 가장 큰 특징은 학습의 대상이 되는 데이터의 속성이라고 할 수 있다. 쉽게 말하면 지도학습은 정답을 주고 기계가 그것에 대한 피드백을 토대로 학습하는 것이고, 비지도학습은 정답을 주지 않고 기계가 비슷한 속성의 데이터를 모아가며(군집화) 학습을 진행하는 것을 말한다. 위의 그림을 보면 지도학습의 특성으로 labeled data 를 제시하고 있다. 반면 비지도학습은 'No labels'라고 되어 있는데, 이 때 label 이 정답을 의미한다.

예를들어 고양이, 개미, 사자, 사람 사진을 주며 이 중에서 고양이 사진을 분류해내는 알고리즘을 구현한다고 생각해보자. 지도학습에서는 고양이, 개미, 사자, 사람의

사진들과 함께 어떤 사진이 고양이 사진인지 알려주는 데이터를 함께 받는다. 이를 통해 지도학습은 고양이만이 가지는 특징을 찾아낸다. 반면 비지도학습은 사진만 받은채 각각의 사진이 가지고 있는 특징에 따라 사진들을 나누는 방식으로 진행하는 것이다.

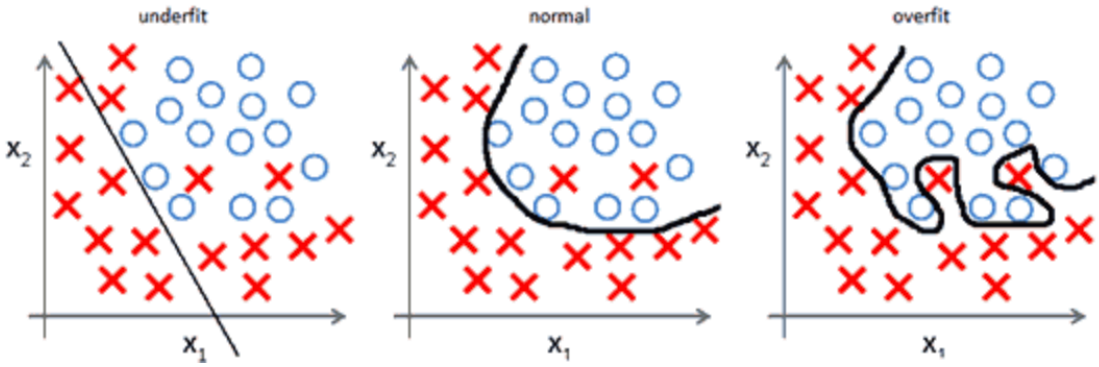
회귀와 분류

이러한 지도학습은 다시 목적에 따라 두 가지로 나누어진다. 먼저 회귀란 특정한 값을 예측하는 것을 말한다. 따라서 회귀의 결과는 연속적인 숫자 중 하나로 나온다. 반면 분류는 이름에서도 알 수 있듯이 무엇인지 나누는 것이다. 분류의 결과는 회귀와 달리 0, 1, 2 와 같이 정수(분류 클래스 이름)로 나오게 된다.

위의 지도학습에서 제시한 예시처럼 어떤 사진은 고양이 사진이고, 다른 어떤 사진은 사람 사진으로 구별하는 것은 분류이다. 반면 어떤 사진의 고양이의 수명을 17.2 년으로 예측하고 다른 고양이 사진의 수명은 15.3 년으로 예측하는 것처럼 구체적인 숫자로 결과값이 나오면 회귀라고 할 수 있다.

지도학습의 대표적인 알고리즘으로는 KNN, SVM, 로지스틱, 릿지, 라소, 엘라스틱 넷, 의사결정트리 등이 있다.

일반화, 과소적합, 과대적합




지도학습은 레이블이 있는 데이터를 학습하여 처음 보는 데이터에 대해서도 정확히 예측하는 모델을 만드는 것을 목표로 한다. 따라서 지도학습 모델의 성능을 결정하는 것은 새로운 데이터에 대한 예측능력이라 할 수 있다. 이때 새로운 데이터에 대해서도 학습 시에 사용한 데이터와 비슷한 수준의 정확도를 보이는 것을

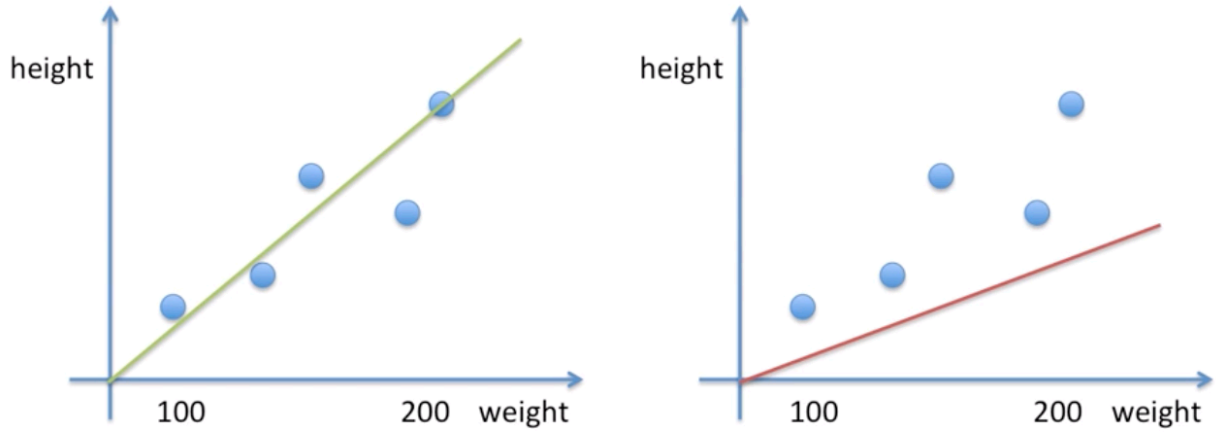
일반화(Generalization)라고 한다. 그리고 머신러닝의 목표 중 하나는 이러한 일반화 성능이 최대가 되는 모델을 만드는 것이라고 할 수 있다.

이러한 일반화 성능과 관련해서 과소적합(under fitting)과 과대적합(over fitting) 문제가 있다. 과소적합이란 모델의 복잡도 수준이 너무 낮아서 새로운 데이터 뿐만 아니라 훈련 데이터에 대해서도 정확도가 낮은 상태를 말한다. 반면 과대적합은 모델의 복잡도 수준이 너무 높아서 훈련 데이터에 대해서는 매우 높은 정확도 수준을 보이지만 새로운 데이터에 대한 정확도가 크게 떨어지는 것이다.

과소적합과 과대적합 문제는 모델의 복잡도와 데이터의 수 등과 높은 관련성을 보인다. 데이터의 수가 많고, 또 데이터의 수에 따른 적절한 수준의 모델 복잡도를 가지면 일반적으로 일반화 성능이 높다.

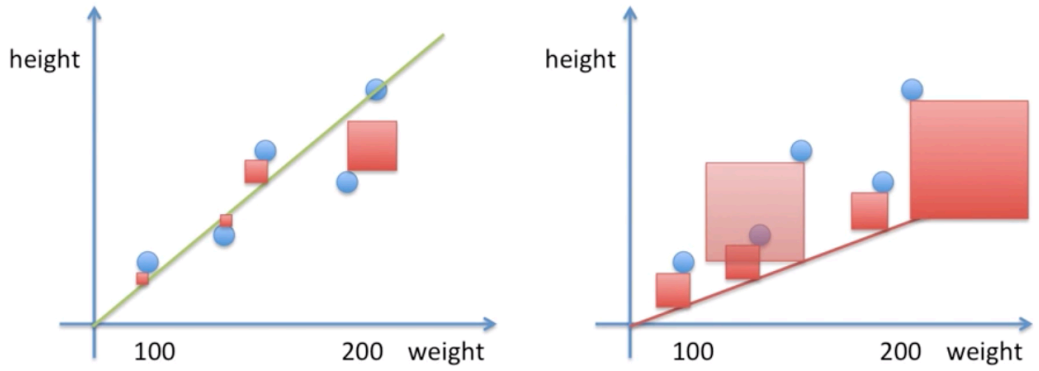
선형회귀

선형 회귀란 선형(Linear)라는 표현에서 유추해볼 수 있듯이 $y = ax + b$ 의 1 차 함수를 이용해 결과값을 예측하는 방법을 말한다. 선형회귀의 결과를 보면 일반적으로 함수를 그릴 때 사용하는 x 축 y 축 평면에 많은 점들이 찍혀있고, 그 사이를 가로지르는 1 차 함수가 있는 꼴로 나타난다.




위의 두 그림은 동일한 데이터에 대한 선형 회귀의 결과를 보여주고 있다. 이 중 어떤 모델이 정확도가 더 높을까? 단순히 그래프의 위치만 보더라도 왼쪽 모델이 보다 정확해 보인다. 그렇다면 이러한 모델의 정확도 차이는 어떻게 나타낼 수 있을까. 가장 많이 사용되는 방법은 평균제곱오차(Mean square error)이다.

평균제곱오차




위의 그림에서 각각의 점과 선형 함수의 y 값 차를 오차(Error)라고 할 수 있고 평균제곱오차는 이러한 오차의 제곱, 즉 빨간색 사각형의 넓이로 오차의 수준을 보여준다. 이러한 평균제곱오차를 이용하면 왼쪽 모델이 오른쪽 모델이 정확하다는 것뿐만 아니라 얼마나 더 정확한지까지도 알 수 있다. 이러한 내용을 수식으로 표현하면 다음과 같다.

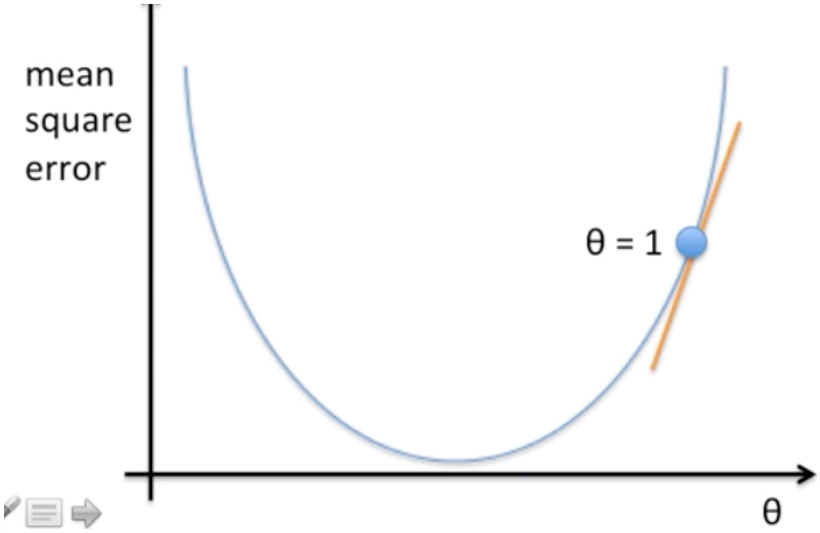
$E = h(x) - y$
$\text{Square } E = (h(x) - y)^2$
$\text{Mean Square } E = 1/n * \Sigma(h(x) - y)^2$

우리의 목표는 새로운 데이터의 결과값을 정확하게 예측하는 것, 즉 오차를 최소화하는 것이라고 할 수 있다. 이는 곧 위의 Mean square E 의 값이 최소화되는 h(x)를 구하는 문제로 전환된다. 그리고 이때의 h(x)를 구하는 방법으로 대표적인 것이 경사하강법(Gradient descent)이다

경사하강법

경사하강법이란 최적화의 대상이 되는 함수의 기울기가 0 이 되는 지점을 구하는 방법으로 기울기의 크기에 비례하여 x 축의 위치를 조정하여 기울기가 0 이 되는 x 값에 최대한 가깝게 다가가는 방법이다. 이를 수식과 그림으로 표현하면 다음과 같다

$\Theta = \Theta - a\ (\ d(\text{mean square } E) /\ d\ \Theta)$




여기서 d(mean square E) / d Θ 는 평균제곱오차를 x 축의 값으로 미분한 것으로, 기울기를 나타내며, a 는 학습률, 즉 다음 x 축 값의 변동폭을 결정한다. 학습률은 경사하강법의 속도와 밀접한 관련이 있는데, 학습률이 너무 크거나 작으면 속도가 느려진다. 상황에 따라 다르지만 학습률은 일반적으로 0.01 에서 0.1 로 설정한다.

로지스틱 회귀

지도학습 알고리즘 중 하나인 로지스틱 회귀(Logistic regression)은 선형 분류에 주로 사용된다. 로지스틱 회귀는 이름에 회귀라는 단어가 들어가지만 결과가 범주형인 경우, 즉 분류에 사용되는 알고리즘이다. 이것이 일반적인 선형 회귀분석과 가장 큰 차이점이다. 즉, 로지스틱 회귀 분석은 선형 회귀 분석과 마찬가지로 종속변수와 독립변수의 관계를 구체적인 함수로 나타내고 이를 통해 예측한다는 점에서는 동일하지만 종속변수의 값이 연속형이 아닌 분류형인 것이다.

로지스틱 회귀는 선형회귀에서 결과값 y 를 'A 일 확률'로 보고, 그 확률이 특정 임계치 이상이면 A 로, 아닐 경우에는 B 로 분류하는 방법에서 출발한다. 하지만 이 경우 y 의 범위 문제가 발생하게 된다. y=ax + b 와 같은 선형회귀에서는 y 의 값이 제한이 없다. 수학적으로 표현하면 y 의 범위는 - ∞에서 ∞ 사이가 된다. 하지만 분류를 목적으로 하는 로지스틱 회귀의 y 값은 0, 1 으로 한정되어야 한다. 이러한 문제를 해결하기 위해 로지스틱 회귀에서는 y = ax + b 의 선형함수를 일련의 과정을 거쳐 변형한다.

P = ax + b
Odds = (P / 1 - P) = ax + b  // P 값의 범위는 0 ~ 무한대  ← 여전히 문제가 있다
ln(Odds) = ln(P/1-P) = ax + b  // P 값의 범위는 0 ~ 1 로 제한  ← 해결

이를 바탕으로 식을 정리하면,

$\ln(P/1-P) = ax + b$

$P / 1-P = e^y$ // y = ax + b, 역수 취하기

$1 - P / P = 1 / e^y$

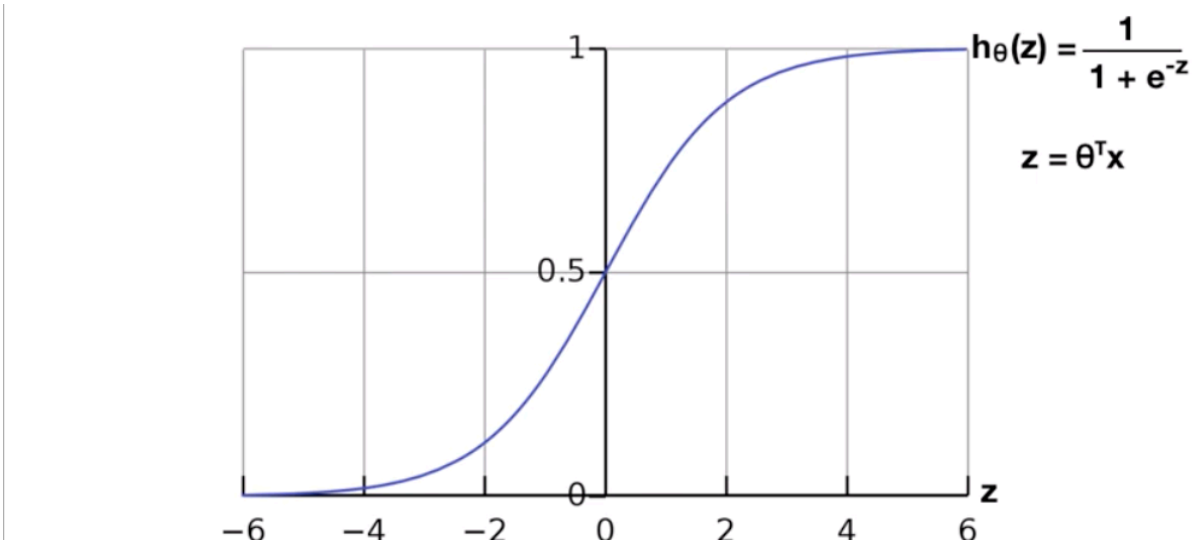
$1 / P - 1 = 1 / e^y$

$1 / P = 1 / e^y + 1$

$1 / P = 1 + e^y / e^y$

$P = e^y / (1 + e^y)$ // 로지스틱 곡선

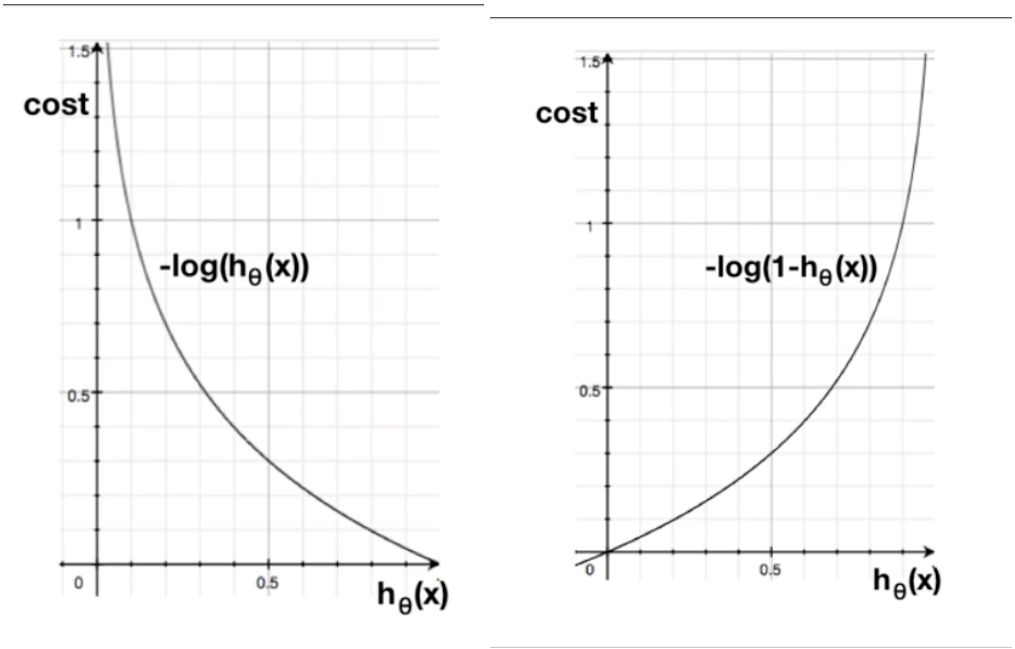
이러한 과정을 통해 $P = e^y / (1 + e^y)$ 이라는 로지스틱 곡선이 도출된다. 자연상수 e 를 취하며 이와같이 복잡한 수식으로 변환한 이유는 시그모이드 함수와 관련된다.
아래의 그림으로 나타나는 시그모이드 함수는 그림에서도 알 수 있듯이 y 값이 0 과 1 사이를 유지한다. 이러한 특성 때문에 시그모이드 함수는 출력값의 설정과 관련되는 딥러닝과 같은 분야에서도 자주 쓰이고 있다.




위에서 도출한 로지스틱 곡선 또한 시그모이드 함수와 동일한 형태를 가진다.
로지스틱 회귀에서는 x 값에 따른 P 값이 임계치 이상일 경우에는 1 로, 임계치 이하일 경우에는 0 으로 분류한다.
로지스틱 회귀를 최적화하는 방법은 일반적인 1 차 함수 형태인 선형회귀와는 다르다. 특히 시그모이드 함수를 이용하는 분류의 문제인 로지스틱 회귀에서 평균제곱오차 방식을 통해 오차를 구하는 것은 좋은 방법이 아니다(local minimum 문제). 대신 교차 엔트로피 오차를 사용한다.

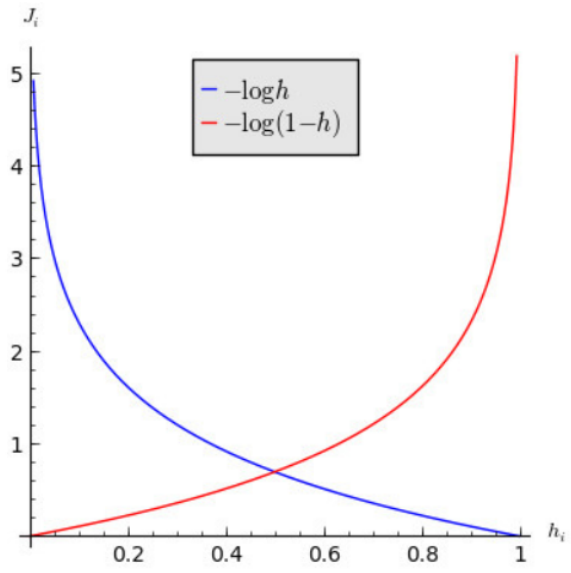
교차 엔트로피 오차

로지스틱 회귀의 결과값은 y = 0 인 경우와 y = 1 인 경우 두 가지로 나눌 수 있는데 교차 엔트로피 오차는 이러한 특성을 이용하는 것이다.




여기서 왼쪽 그림은 y = 1 인 경우에 로지스틱 회귀의 오차 크기이고, 오른쪽은 y = 0 인 경우의 오차 크기이다. 로지스틱 회귀는 두 가지 상황 모두 존재하므로 두 함수를 합쳐주게 된다. 그 수식과 함수의 모양은 다음과 같다.

$$C(H(x), y) = ylog(H(x)) - (1 - y)log(1 - H(x))$$




평균제곱오차가 아닌 교차 엔트로피 오차라는 새로운 비용함수를 사용하였지만 이를 통해 도출된 비용함수 또한 선형회귀의 비용함수와 마찬가지로 가운데가 들어간 모습을 하고 있으며, 이는 앞서 설명한 경사하강법을 통해 최적화를 달성할 수 있다는 것을 의미한다.