6. Regression
2019년 가을 학기

2019년 08월 24일
https://www.github.com/KU-BIG/KUBIG_2019_Autumn

앞의 개론 강의에서 전반적으로 머신러닝이 어떻게 진행되는가를 살펴보았다. 이번 강의부터는 머신러닝의 대표적인 회귀, 지도, 비지도 학습에 속해있는 대표적인 기법 몇 가지를 다룰 것이다. 이번 강의에서는 회귀분석과 기본적인 회귀 모형들을 살펴볼 것이다.

1. Introduction

회귀분석(Regression Analysis)은 어떤 결과값이 존재할 때, 그 결과값을 결정할 것이라고 추정되는 입력값과 결과값의 연관관계를 찾아 결과값을 예측하는 기법을 말한다. 회귀분석에서는 데이터를 아래와 같은 모델로 수식화할 수 있다고 본다.

$$y = h(x_1, x_2, \cdots, x_k\ ;\ \beta_1, \beta_2, \cdots, \beta_k) + e$$

회귀모델의 본래의 정의는 '어떤 자료에서 그 값에 영향을 주는 조건들을 고려하여 구한 평균'이다. 따라서, 위 수식에서 h()는 평균을 구하는 함수를 의미한다. 각 조건 $x_1, x_2, \ldots, x_k$ 에 대하여, 그들 각각의 영향력 $\beta_1, \beta_2, \ldots, \beta_k$ 을 구하여 평균을 계산한다. e는 오차항(error)을 의미하는데, 측정 상의 오차나 현실적인 한계로 인해 발생하는 불확실성('잡음')을 의미한다.

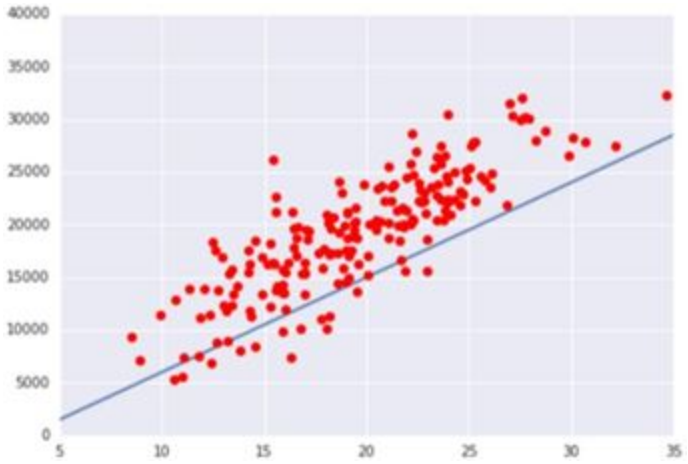
회귀분석에서는 데이터에 대해 여러가지 가정을 필요로 한다. 특정 조건이 만족되는지의 여부에 따라 사용하는 회귀모델이 달라지게 된다. 이 자료에서는 회귀모델의 두 종류인 기본 선형 회귀모형(Linear Regression)과 GLM(Generalized Linear Model)에 대해 다루고자 한다.

2. 회귀모델의 종류

2.1 선형 회귀모델 (Linear Regression)

선형 회귀모형이란 선형(Linear)이라는 표현에서 유추해볼 수 있듯이 종속 변수 y와 한 개 이상의 독립변수 x와의 선형 관계를 통해 결과값을 예측하는 방법을 말한다. 예를 들어, 종속변수가 '택시 요금', 독립변수가 '이동 거리' 라고 하자. 거리와 요금이 서로 비례하기 때문에 거리(x)와 요금(y)간의 상관관계는 다음과 같이 일차함수의 형태로 나타낼 수 있다.

$$y_i = \alpha + \beta x_i + \varepsilon_i.$$



이제 독립변수의 개수가 p, 데이터의 개수가 n이라고 하자. 다음과 같이 Matrix Notation으로 표현할 수 있다.

$$\mathbf{y} = X\boldsymbol{\beta} + \boldsymbol{\varepsilon}$$

where $\mathbf{y} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}$, $X = \begin{pmatrix} \mathbf{x}_1^{\mathsf{T}} \\ \mathbf{x}_2^{\mathsf{T}} \\ \vdots \\ \mathbf{x}_n^{\mathsf{T}} \end{pmatrix} = \begin{pmatrix} 1 & x_{11} & \cdots & x_{1p} \\ 1 & x_{21} & \cdots & x_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n1} & \cdots & x_{np} \end{pmatrix}$, $\boldsymbol{\beta} = \begin{pmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \vdots \\ \beta_p \end{pmatrix}$, $\boldsymbol{\varepsilon} = \begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{pmatrix}$

위의 식을 가지고 있을 때, $S(\boldsymbol{\beta}) = (\mathbf{y} - \mathbf{X}\boldsymbol{\beta})^T(\mathbf{y} - \mathbf{X}\boldsymbol{\beta})$를 β에 대해서 Minimize하는 것이 목표이다.

선형 회귀모형은 그 형태가 단순한 만큼 데이터에 대해 많은 가정을 필요로 한다. 대표적으로 '종속변수와 독립변수는 선형관계이다', '오차항은 평균이 0이고, 분산이 일정한 정규분포를 따른다', '독립변수와 오차항은 서로 독립이다' 등이 있다. 이 조건들 중 두번째 오차항의 정규성 조건이 만족하지 않는 경우 사용할 수 있는 회귀모델이 GLM이다.

2.2 GLM (Generalized Linear Model)

GLM은 '오차항이 정규분포를 따르지 않는 경우를 포함하는 선형모형의 확장'이다. 예를 들어, 종속 변수가 0이나 1만의 값을 갖는 이항 반응 변수이거나, 어떤 것의 개수(도수)를 의미하는 변수라면, 오차항이 정규분포를 따르지 않게 되므로 Linear Regression에서와 같이 종속변수와 독립변수를 바로 선형결합할 수가 없다. 이때 GLM은 각 반응변수의 특성에 맞는 연결함수(link function)를 이용하여 종속변수와 독립변수를 선형 결합한다. 이로써 종속변수가(혹은 오차항이) 정규분포를 따르지 않는 회귀모형이 선형모형으로의 일반화가 가능하게 된다.

GLM을 구성하는 3가지 요소는 다음과 같다.

- Random Component : response variable Y
- Systematic Component : X
- Link function : a function of E(Y) related to X, 정확히는 $\mu = E(Y)$를 다음과 같이 linear predictor와 연결시키는 함수를 말한다. $g(\mu) = \beta_0 + \beta_1 x_1 + ... + \beta_p x_p$

연결함수는 monotone하고 미분가능해야 한다는 조건이 있다. 대표적인 종류는 아래와 같다.

- $g(\mu) = \mu$: identity link function, $-\infty < \mu < \infty$

  $\mu = \beta_0 + \beta_1 x_1 + ... + \beta_p x_p$ => Ordinary Regression
- $g(\mu) = log(\mu)$: log link function, $0 \leq \mu < \infty$

  $log(\mu) = \beta_0 + \beta_1 x_1 + ... + \beta_p x_p$ => Poisson Regression
- $g(\mu) = log(\frac{\mu}{1-\mu})$: logit link function, $0 \leq \mu < \infty$
- $log(\frac{\mu}{1-\mu}) = \beta_0 + \beta_1 x_1 + ... + \beta_p x_p$ => Logistic Regression

GLM의 likelihood function을 계산하고, 그 likelihood를 maximize하는 방식으로 β를 추정한다. 만약 수식적으로 maximize가 힘들다면, Newton-Raphson이나 Fisher scoring algorithm을 사용해서 계산한다.

또 다른 특징으로는, GLM은 Exponential dispersion family를 가정한다. 즉, y의 밀도함수가 아래와 같은 형태를 가지고 있는 상황에서 Generalized Linear Model을 사용할 수 있다.

$$f(y_i;\theta_i,\phi) = exp[\frac{y_i\theta_i - b(\theta_i)}{a(\phi)} + c(y_i,\phi)]$$

혹은, log-likelihood가 다음의 형태를 가지고 있을 때, 역시 Exponential dispersion family에 속한다고 볼 수 있다.

$$\sum_{i=1}^{n} logf(y_i;\theta_i,\phi) = \sum_{i=1}^{n}[\frac{y_i\theta_i - b(\theta_i)}{a(\phi)} + c(y_i,\phi)]$$

만약 $Y \sim f(y;\theta)$라면 아래의 식을 유도할 수 있다.

$$E\{\frac{\partial}{\partial\theta}logf(Y;\theta)\} = 0 \quad (3) \qquad -E\{\frac{\partial^2}{\partial\theta^2}logf(Y;\theta)\} = E[\{\frac{\partial}{\partial\theta}logf(Y;\theta)\}] \quad (4)$$

GLM density에 대해서 (3)으로부터 $\mu_i = E(Y_i) = b'(\theta)$를 유도할 수 있다.

또한, (4)로부터 $Var(Y_i) = b''(\theta_i)a(\phi)$ 식을 유도할 수 있다.

GLM은 $\eta_i = g(\mu_i) = \boldsymbol{\beta}\mathbf{x}_i$와 같이, link function인 $g(\cdot)$를 사용해서 $\eta_i = \boldsymbol{\beta}\mathbf{x}_i$와 $\mu_i = E(Y_i)$를 연결한다. $\mu_i = b'(\theta_i), \quad hence \quad g = b'(\theta_i)^{-1}$이고, 이때 g를 canonical link라 부른다.

$$\sum_{i=1}^{n} logf(y_i;\theta_i,\phi) = \sum_{i=1}^{n}[\frac{y_i\theta_i - b(\theta_i)}{a(\phi)} + c(y_i,\phi)] \quad (5)$$

likelihood가 (5)와 같기 때문에, 이를 베타에 대해 미분한 결과를 0으로 두고 계산한다.

$$\frac{\partial L(\boldsymbol{\beta})}{\partial\beta_j} = \sum_i \frac{\partial L_i}{\partial\beta_j} = 0$$

$$\Rightarrow \sum_{i=1}^{N} \frac{(y_i-\mu_i)x_{ij}}{var(Y_i)}(\frac{\partial\mu_i}{\partial\eta_i}) = 0 \quad , \quad \mu_i = g^{-1}(\sum_j \beta_j x_{ij})$$

이 식을 푸는 방법은 위에서 한번 설명했듯이, 수식적으로 풀 수 있는 경우에는 전개하거나, 그러기 힘든 경우에는 iteratively Newton Raphson Method나 Fisher Scoring Method를 사용한다.

다음으로는 GLM의 대표적인 회귀모형 중 하나인 로지스틱 회귀(Logistic Regression)를 다루고자 한다.

2.2.1 로지스틱 회귀(Logistic Regression)

로지스틱 회귀는 종속변수가 0이나 1의 값만을 갖는 이항 반응변수일 때 사용하는 회귀모형이다. 로지스틱 회귀에서는 $\hat{y}$를 '$y=1$일 확률'로 보고, 이것이 특정값(cut-off) 이상이면 y=1로, 이하이면 y=0으로 예측하는 아이디어에서 출발한다. 그렇다면, 로지스틱 회귀에서는 y와 x를 어떻게 결합시킬까?

먼저, 선형 회귀에서와 같이 단순 선형 결합시킨다면 모델은 다음과 같이 될것이다.

$$y = p(\mathbf{x}) = \beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p, \quad 0 < p(\mathbf{x}) < 1$$

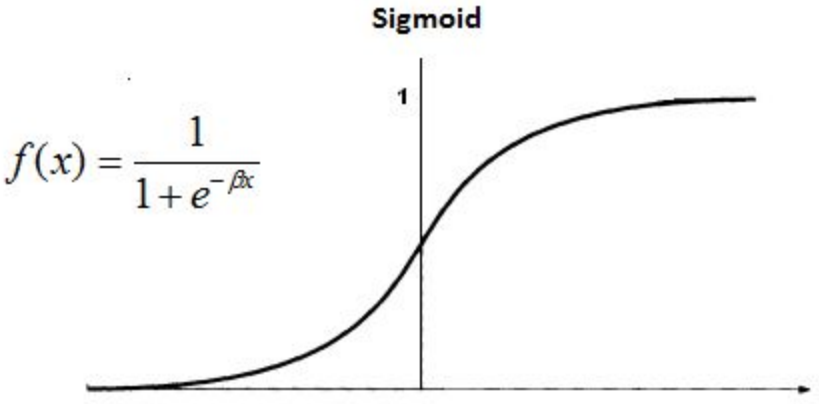
이때, 범위의 문제가 발생한다. $p(x)$의 경우 $(0,1)$사이의 값만을 갖지만, 우변 x의 선형결합은 $(-\infty,\infty)$사이의 값을 갖는다. 이를 해결하기 위해 $p(x)$에 적절한 함수를 취하여 범위가 $(-\infty,\infty)$이 되도록 만든다. 함수를 취하면 다음과 같다.

$$ln\frac{p(\mathbf{x})}{1-p(\mathbf{x})} = \beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p, \quad 0 < p(\mathbf{x}) < 1$$

이로써 좌변도 $(-\infty,\infty)$ 사이의 값을 갖게 되었다. 다시 $p(x)$에 대한 식으로 바꿔주면 다음과 같다.

$$p(\mathbf{x}) = \frac{1}{1+e^{-(\beta_0+\beta_1 x_1+\cdots+\beta_p x_p)}} = \frac{1}{1+e^{-\boldsymbol{\beta}\mathbf{x}}}, \quad 0 < p(\mathbf{x}) < 1$$

위와 같은 함수를 '로지스틱 함수', 분야에 따라서 '시그모이드 함수'로 불리우는 것이다. 이 함수는 아래 그림과 같이 0과 1 사이에서 S-커브 형태를 띤다.



http://byungjin-study.blogspot.com/2016/01/machine-learning-in-action-ch5-logistic.html

그러면 Logistic Regression이 왜 GLM에 속하는지 알아보도록 하자. Logistic Regression이 앞서 설명한 GLM의 정의에 해당하는지 증명해 보이면 된다.

만약 $n_i y_i \sim^{iid} Binomial(n_i, p_i)$ 이라면,

$$f(y_i; p_i, n_i) = \binom{n_i}{n_i y_i} \pi_i^{y_i} (1-\pi_i)^{n_i - n_i y_i} = exp[\frac{y_i \theta_i - log\{1+exp(\theta_i)\}}{1/n_i} + log\binom{n_i}{n_i y_i}]$$

where $\theta_i = log\{\frac{\pi_i}{1-\pi_i}\}$, $b(\theta_i) = log\{1+exp(\theta_i)\}, \quad and \quad a(\phi) = 1/n_i$

다음과 같이 정리할 수 있다. 따라서 exponential family에 속하기 때문에 GLM을 사용할 수 있다.

따라서, 바로 위의 mean과 Variance의 유도식을 그대로 적용할 수 있다.

$$E(Y_i) = b'(\theta_i) = \frac{\partial}{\partial \theta_i} log\{1+exp(\theta_i)\} = \frac{exp(\theta_i)}{1+exp(\theta_i)} = \pi_i$$

$$Var(Y_i) = b''(\theta_i)a(\phi) = \frac{exp(\theta_i)}{\{1+exp(\theta_i)\}^2 n_i} = \pi_i(1-\pi_i)/n_i$$

마지막으로, logistic regression의 link function은 다음과 같다.

$\eta_i = g(\mu_i) = \theta_i = (b')^{-1}(\mu_i) = log\frac{\pi_i}{1-\pi_i} = \boldsymbol{\beta}\mathbf{x}_i$ (link function)

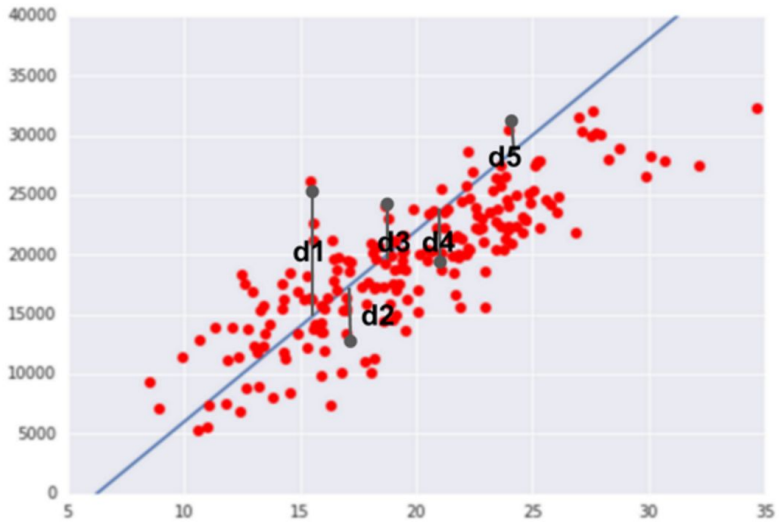
3. Cost function

회귀분석에서 우리의 목표는 가장 실제에 가깝게 모델을 만드는 것이다. 이를 위해서는, 실제 값과 모델이 예측한 값의 차이를 최소화해야 한다. 이들의 차이에 대한 함수를 Cost function이라 하고, 이것을 최소화하는 방향으로 모델을 학습시킴으로써 회귀계수를 추정할 수 있게 된다. 마찬가지로, 모델의 종류에 따라 cost function도 달라지게 된다. 앞서 설명한 선형회귀와 로지스틱회귀의 경우를 살펴보겠다.

3.1 선형 회귀모형 : MSE(Mean Square Error)

아래 그림과 같이 데이터가 흩뿌려져 있을 때, 이 데이터를 가장 잘 설명하는 회귀모형을 적합시키기 위해서는, 각 점들과 그래프와의 차이(residual)를 최소화해야 할 것이다.

$$\text{residual} = y - \hat{y}$$



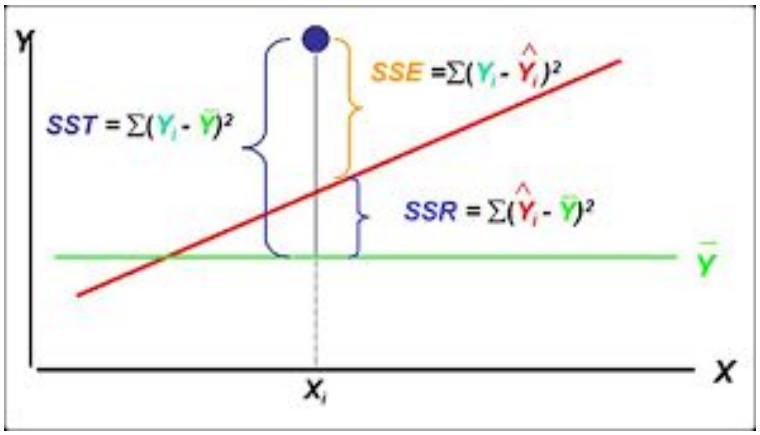
https://bcho.tistory.com/1139

이 때 쓸 수 있는 방법이 LSM(Least Square Method)이다. LSM은 모형의 parameter를 구하는 방법 중 하나로서, 이름 그대로 prediction 결과와 실제 target value 간의 차이인 잔차(residual)의 제곱 합을 최소화하는 방법으로 계산된다.

즉, LSM을 사용한다는 말은 다음의 식을 최소화하도록 f(x)의 파라미터를 결정한다는 뜻이다.

$$\sum_{i=1}^{n} r_i^2 = \sum_{i=1}^{n} (y_i - f(x_i))^2$$

아래 그림을 보자. 다시 말해서 우리의 목표는 예측값($\hat{y}$)과 실제값(y) 사이의 차이의 제곱합(SSE)을 최소화하는 것이다. 이 그림을 이해해보자.



https://wikidocs.net/images/page/20168/linear-regression.jpg

y의 전체 평균($\bar{y}$)은 아무런 정보가 주어지지 않았을 때 선택할 수 있는 가장 기본적인 모델이다. 그렇기 때문에 SST의 경우 각 y와 $\bar{y}$ 의 차이값들의 제곱합이며, 아무런 사전정보가 주어지지 않았을 때의 residual 제곱합이라고도 볼 수 있다(y의 전체 평균을 모델로 설정했을 경우). SSR은 y의 예측값($\hat{y}$)과 $\bar{y}$의 차이 값들의 제곱합을 의미한다.

위 그림에서 확인해 볼 수 있듯이, SST = SSR + SSE의 구조를 가지고 있다. SST는 고정되어있기 때문에 SSE가 작아질수록, 그리고 SSR이 커질수록 잔차는 줄어들었고 모델이 설명하는 영역이 $\bar{y}$를 사용했을 때 보다 증가했음을 의미한다. 참고로, 선형 회귀 모델에서 자주 쓰는 Evaluation Metric인 R^2 역시 이것을 바탕으로 한다. 추정한 모델의 설명력이 높아질수록 R^2의 값 역시 높아지게 된다.

$$\mathrm{R}^2 = \frac{SSR}{SST} = 1 - \frac{SSE}{SST}$$

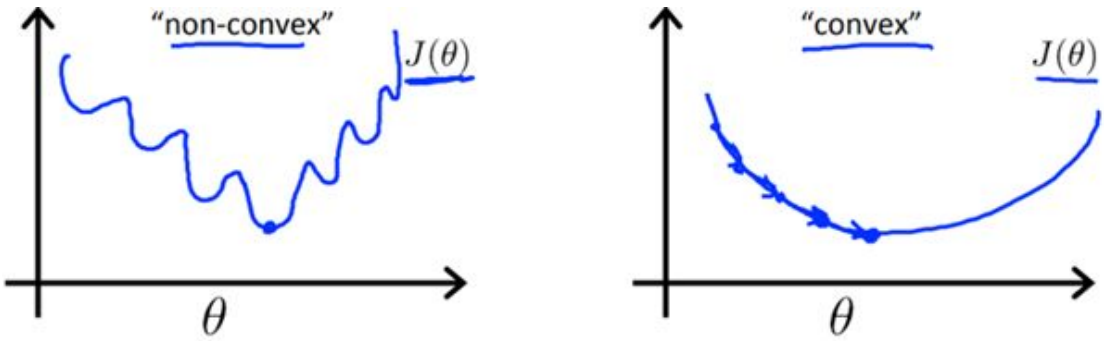
여기까지가 LSM, 즉 잔차의 제곱합을 최소화하는 방법에 대한 설명이었다. 주로 데이터가 총 m개 있다고 할 때 잔차의 제곱합을 m으로 나눈 것, 즉 잔차의 제곱합의 평균을 선형 회귀모델의 cost function으로 사용한다. 이를 MSE(Mean Square Error)라고 한다.

$$MSE = \frac{1}{m}\sum_{i=1}^{m}(y - \hat{y})^2$$

한편, 잔차를 제곱하는 방법은 이론적으로 중요한 장점을 가지게 된다. 바로 개론 때 한번 접했던 최적화 알고리즘인 Gradient Descent를 적용할 때 계산이 용이해진다는 것이다. 이에 대해 뒤에서 한번 더 설명할 예정이지만, 간단히 말하자면 선형회귀의 cost function을 그래프로 나타내면 아래로 볼록('convex')한 2차 함수 꼴이 되어 Gradient Descent 알고리즘을 통해 최저점을 찾을 수 있다. 이는 Logistic Regression의 cost function을 정할 때에도 중요한 기준이 된다. 이제 Logistic Regression의 경우를 보자.

3.2 로지스틱 회귀: Cross Entropy

로지스틱 회귀에서는 선형 회귀에서와 같은 cost function을 사용하지 않는다. 그 이유는 로지스틱 회귀에서 MSE 방법을 적용하면 아래 그림과 같이 local minimum이 다수 존재하는 'non-convex'한 그래프가 되기 때문이다. 직관적으로 선형회귀에서는 직선을 구부리기 때문에 'convex'한 그래프가 되고, 로지스틱 회귀에서는 시그모이드 함수를 구부리기 때문에 굴곡이 많은 'non-convex'한 그래프가 된다고 이해하면 편하다. 따라서 그대로 최적화 알고리즘을 사용하게 되면, global minimum이 아닌 local minimum을 최저점으로 간주하는 문제점이 발생할 수 있다.



http://mlwithdata.blogspot.com/2015/05/logistic-regression-cost-function.html

따라서 로지스틱 회귀에서는 Cross Entropy라고 불리는 다음과 같은 cost function을 사용한다.

$$Cross\ Entropy = -\frac{1}{m}\left[\sum_{i=1}^{m} y^i \cdot log\widehat{y^i} + (1-y^i)\cdot lo\,g(1-\widehat{y^i})\right]$$

전체 training data에 대해 평균을 내는 아이디어는 같으므로, 내부 수식을 이해해보자.

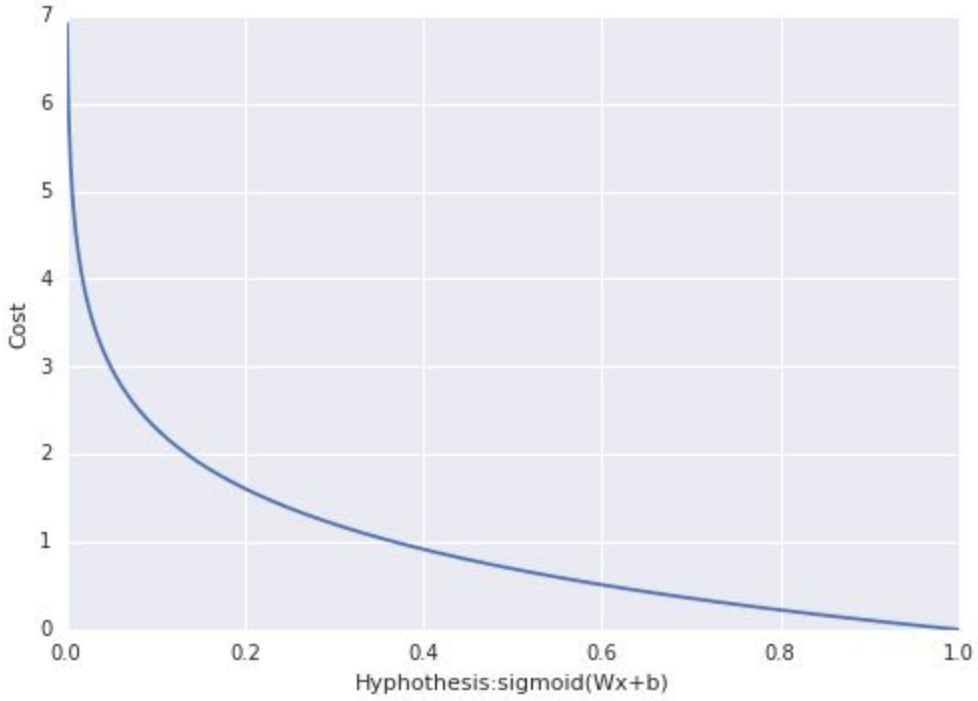
$$L(\mathrm{y},\hat{y}) = -[\mathrm{y}\log(\hat{y}) + (1-y)\log(1-\hat{y})]$$

이때, y는 0또는 1의 값만을 가지므로, y가 0일 때와 1일 때로 나누어보겠다.

• y의 실제 레이블이 1일 때, 함수는 다음과 같다.

$$L(\mathrm{y},\hat{y}) = -log(\hat{y})$$

이 때, 예측값 $\hat{y}$ 은 그 값이 커질수록 실제 레이블에 가까워진다. 다시 말해서, $\hat{y}$ 의 값이 커질수록 예측을 잘 한 것이고, cost는 감소한다. 따라서, 그래프의 모양은 다음과 같이 된다.

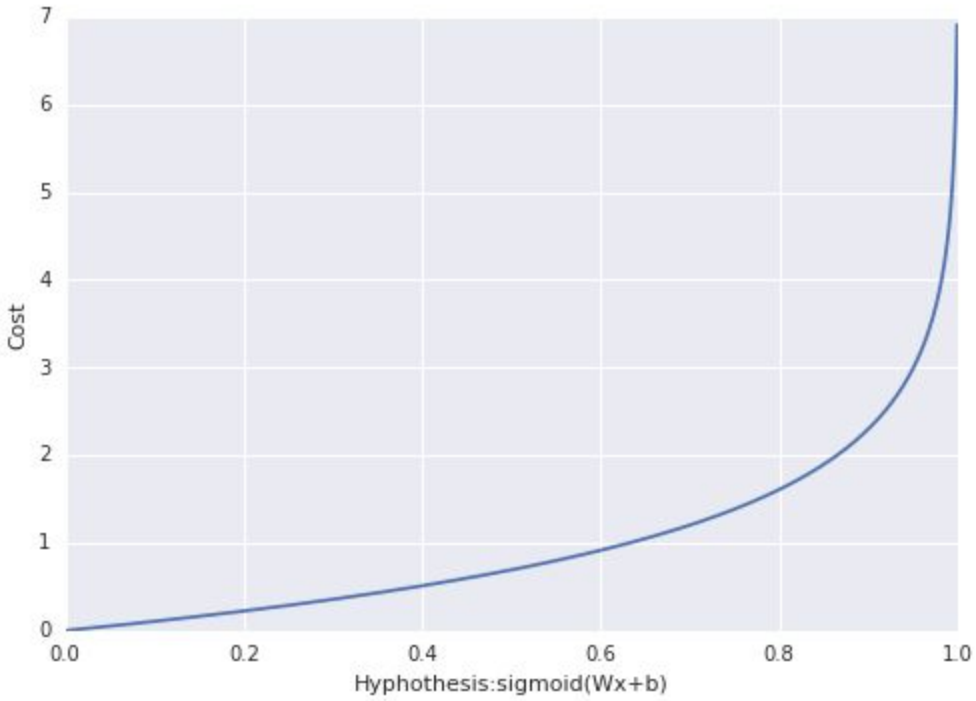


https://bcho.tistory.com/1142

• y의 실제 레이블이 0일 때, 함수는 다음과 같다.

$$L(y, \hat{y}) = -\log(1 - \hat{y})$$

이 때에는 예측값 $\hat{y}$은 그 값이 작아질수록 실제 레이블에 가까워진다. 다시 말해서 $\hat{y}$의 값이 작아질수록 예측을 잘한 것이고, cost는 감소하게 된다. 따라서 그래프의 모양은 다음과 같이 된다.



https://bcho.tistory.com/1142

이로써 두 종류의 회귀 모델의 cost function을 알아보았다. 이제 이들을 최소화(최적화)하는 회귀계수를 찾아야 할 것이다. 이는 최적화 알고리즘인 Gradient Descent를 이용하면 된다.

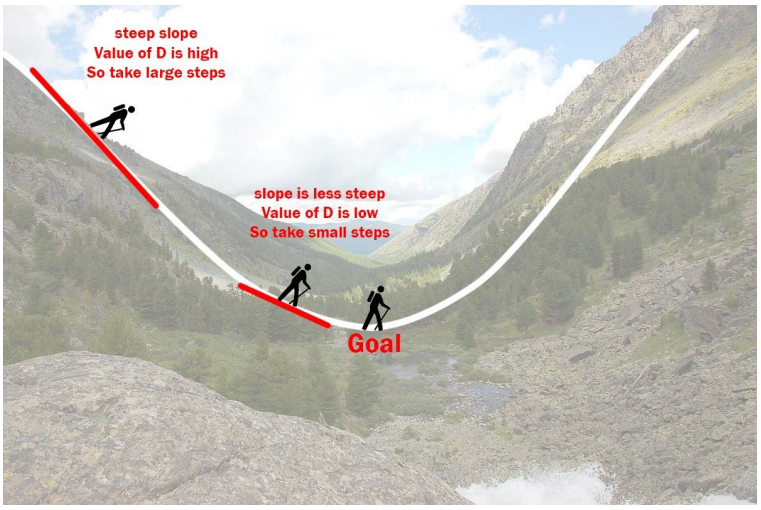
4. Optimization: Gradient Descent

Optimization이란 어떤 목적함수의 함숫값을 최적화(상황에 따라 최소화 또는 최대화)하는 parameter를 찾는 문제를 말한다. 일반적으로 Cost function을 최소화하는 parameter 조합을 추정하는 것을 말한다. 상황에 따라 여러가지 Optimizer가 존재하지만, 이 장에서는 Gradient Descent 알고리즘에 대해 다루고자 한다.

Gradient Descent는 한마디로 기울기가 0인 부분을 찾아가는 알고리즘이다. 'convex'한 함수에서 대체로 최저점은 기울기가 0인 부분이기 때문이다. Gradient Descent 알고리즘의 기본 수식은 다음과 같다. $J(\theta)$는 θ를 모수로 하는 cost function이고, α $(0 < \alpha < 1)$는 학습률(learning rate)을 의미한다.

$$
\begin{aligned}
&Repeat\ \{ \\
&\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta) \\
&(simultaneously\ update\ all\ \theta_J) \\
&\}
\end{aligned}
$$

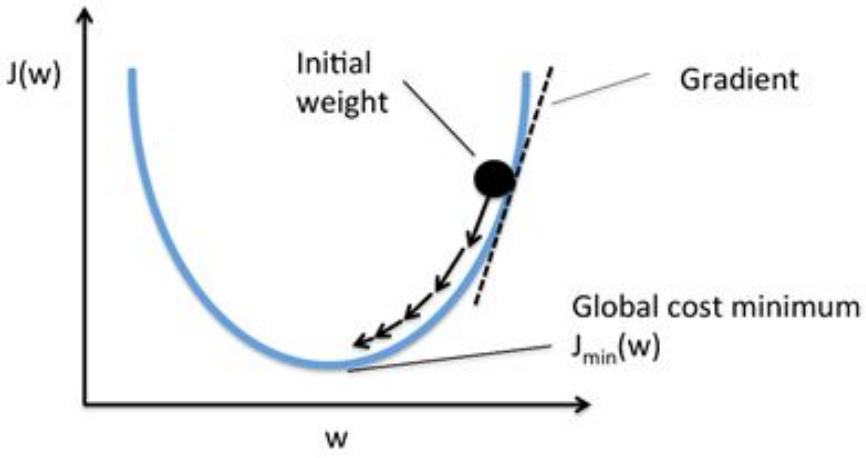
Gradient Descent 알고리즘은 산등성이에서 눈을 가리고 평평한 곳으로 한발씩 내딛어 가는 것으로 비유될 수 있다.



https://towardsdatascience.com/linear-regression-using-gradient-descent-97a6c8700931

이 때 현재 위치는 θ_j 이고, α 는 보폭, $f'(\theta_j) = \frac{\partial}{\partial \theta_j} J(\theta)$ 는 방향을 의미한다고 볼 수 있다. 새로운 위치는 현재 위치에서 보폭과 방향의 곱을 뺀 것으로 갱신이 된다. 이 절차를 반복하여 평평한 곳으로 나아가는 것이다. 만약 보폭이 너무 작으면 시간이 너무 오래 걸리고 보폭이 너무 크면 목표한 지점보다 더 가게 되듯이, 학습률이 너무 작으면 시간이 오래 걸리고, 학습률이 크면 최저점을 무질서하게 이탈하게 될 수 있다. 따라서, 상황에 따라 적절한 학습률을 정하는 것이 중요하다.

이 알고리즘을 선형 회귀의 cost function에 적용해보자. 선형 회귀의 cost function의 그래프는 다음과 같이 아래로 볼록한 2차 함수 형태이다. Gradient Descent를 적용하면, 그래프 상의 임의의 점에서 시작하여 기울기를 조금씩 갱신해 나가면서 결국 Global minimum에 도달하게 된다. 이때 시작점이 어느 위치이든 간에 같은 점에 도달하게 된다.



http://rasbt.github.io/mlxtend/user_guide/general_concepts/gradient-optimization/

로지스틱 회귀도 마찬가지로 Cross Entropy 함수에 Gradient Descent 알고리즘을 적용하면 된다. 로지스틱 회귀에 Gradient Descent적용은 딥러닝 파트에서 배우게 될 Backward Propagation개념과 함께 주로 설명되므로, 이 강의에서는 생략하겠다.