Unit 1

회귀분석소개

6주차. 선형회귀분석

학습 내용

- 회귀분석이란?
- 비용함수, 경사하강법
- 경사하강법의 적용
- Analytic Solution of Linear Regression

학습 목표

- 회귀분석의 개념과 모형을 이해하고 설명할 수 있다.
- 회귀분석 모형을 실제 적용할 수 있다.

선형회귀분석이란?

회귀분석의 개념

선형회귀분석(Linear regression)

지도학습(supervised learning)의 일종으로,
input에 대한 실수값의 output을 예측하는 문제

- 주어진 데이터를 나타내는 최적의 직선을 찾아 input X와 output Y의 관계를 도출해내는 과정
- 가설 = input (feature)과 output (target) 의 관계를 나타내는 함수

Notation

Matrix Notation

N Number of features
(설명변수의 수 = feature의 수)

Number of training examples
(관측치, observation의 수) **M**

Notation

Matrix Notation

$x^{(i)}$ Input(features)의 i번째의 training example

$x_j^{(i)}$ feature ji번째의 traning example에 있는 feature J의 값

Notation

가설함수의 표현

가설함수

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$$

feature가 모두 1로 이루어진 절편 의미

절편

실제 feature의 수 : n+1

Unit 1 회귀분석소개

Notation

☑ column vector 표현

$x_0^{(i)}$

- 1로 구성된 column vector를 의미
- n+1차원

x와 θ의 모든 차원은 $n + 1$

$$x = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_n \end{bmatrix} \in R^{n+1} \qquad \theta = \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_n \end{bmatrix} \in R^{n+1}$$

Notation

비용함수의 column vector 표현

비용함수를 column vector로 표현하면

$$h_\theta(x) = [\theta_0\,\theta_1\,\theta_2 \cdots \theta_n] = \begin{bmatrix} \boldsymbol{x}_0 \\ \boldsymbol{x}_1 \\ \vdots \\ \boldsymbol{x}_n \end{bmatrix} = \theta^T \boldsymbol{x}$$

$$= \theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots \theta_n x_n$$

» θ과 x는 n+1 column vector

Notation

☑ M개의 example

m 개의 example → $m \times (n + 1)$

$$X = \begin{bmatrix} x_0^{(1)} \ x_1^{(1)} \cdots x_n^{(1)} \\ x_0^{(2)} \ x_1^{(2)} \cdots x_n^{(2)} \\ x_0^{(3)} \ x_1^{(3)} \cdots x_n^{(3)} \\ \vdots \\ x_0^{(m)} \ x_1^{(m)} \cdots x_n^{(m)} \end{bmatrix}$$

Notation

가설함수의 표현

가설함수

$$h_\theta(X) = X\theta$$

$$X = \begin{bmatrix} x_0^{(1)} & x_1^{(1)} & \cdots & x_n^{(1)} \\ x_0^{(2)} & x_1^{(2)} & \cdots & x_n^{(2)} \\ x_0^{(3)} & x_1^{(3)} & \cdots & x_n^{(3)} \\ & \vdots & & \\ x_0^{(m)} & x_1^{(m)} & \cdots & x_n^{(m)} \end{bmatrix}, \quad \theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_n \end{bmatrix}, \quad y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix}$$

$$\underset{m \times 1}{X\theta} = \begin{bmatrix} \theta_0 x_0^{(1)} + \theta_1 x_1^{(1)} \cdots \theta_n x_n^{(1)} \\ \theta_0 x_0^{(2)} + \theta_1 x_1^{(2)} \cdots \theta_n x_n^{(2)} \\ \vdots \\ \theta_0 x_0^{(m)} + \theta_1 x_1^{(m)} \cdots \theta_n x_n^{(m)} \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix} = y$$

비용함수

☑ 비용함수의 개념

비용함수 (cost function)

최적화 과정에서 **비용을 최소화** 하고자 하는 목적함수로 사용

비용함수

비용함수

- Parameter vector θ
- $\theta \in R^{n+1}$

$$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}\underbrace{(h_\theta(x^{(i)}) - y^{(i)})^2}_{\text{residual}}$$

➤ 평균과정에서 편의를 위해

➤ 비용함수와 실제 데이터 차이 : 편차

➤ ' 거리'로 표현가능

➤ 실제 함수를 추정했을 때 벗어나는 정도

➤ (작을수록 좋음)

비용함수

비용함수 vector로 표현

$$J(\theta) = \frac{1}{2m}(X\theta - y)^T(X\theta - y)$$

- 이때의 y는 모든 y(i) 값을 포함하는 벡터

경사하강법

경사하강법의 개념

경사하강법 (gradient descent)

1차 편미분 값(기울기)을 활용하여 최적값을 발견하는 알고리즘

» 함수의 기울기(경사)를 구해서 낮은 쪽으로 계속 이동시켜 최적값에 이를 때까지 반복해서 찾아가는 과정

$$
\begin{aligned}
&h_\theta(x) = \theta^T x \\
&= \theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots \theta_n x_n) \\
&\quad \text{where, } x_0 = 1.
\end{aligned}
$$

경사하강법

☑ 경사하강법

비용함수를 최소화 하는 Parameter

Parameter θ_0부터 θ_n까지
각각 편미분을 하여 기울기 도출

$$\left[\ \theta = [\theta_0\, \theta_1 \theta_2\ \cdots\ \theta_n]^T \ \in\ R^{n+1}\ \right]$$

경사하강법

경사하강법

Feature의 개수가 하나인 경우 (n=1)

θ 대해서 편미분

$$\theta_1 := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_0, \theta_1)$$

$$= \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x^{(i)}$$

학습속도
(learning rate)

비용함수를
θ에 대해 편미분

경사하강법

경사하강법

비용함수 값이 아주 작을 때,

즉, parameter값이 변해도 비용 함수 값이 거의 변하지 않을 때

" minimum 지점에 도달했다고 간주 "

Parameter의 추정치

경사하강법

경사하강법

Feature의 개수가 하나 이상인 경우 (n≥1)

비용함수 값이 어느 숫자보다 **작을 때 stop** → **Parameter의 추정치**

경사하강법

Matrix Notation

기울기 하강을 Matrix notation으로 표현

$$\left[\ \theta := \theta - \alpha \nabla J(\theta)\ \right]$$

nabla (편미분)

학습속도 (learning ratio)

비용함수

경사하강법

Matrix Notation

편미분의 matrix notation

$$\left[\theta \coloneqq \theta - \frac{\alpha}{m} X^{T}(X\theta - y) = \theta - \alpha \nabla J(\theta)\right]$$

경사하강법의 적용

Feature Scaling

gradient descent의 실제 적용

Feature Scaling

- Feature값의 단위가 서로 다를 수 있음
- 미분하는 과정에서 움직임의 폭이 불규칙

모든 feature들의 값의 범위를 아주 비슷하게 축소
(동일할 필요는 없음)

모든 feature가 비슷한 범위에 있으면 반복하는 과정에서 안정적으로 수렴 가능

경사하강법의 적용

Feature Scaling

표준화(Standardization)

Feature 샘플 평균 = 0

$$\left[\, Z = \frac{X - \mu_X}{\sigma_x} \rightarrow Z \sim (\,0, 1\,) \,\right]$$

평균이 0이고, 분산이 1인 standard normal 분포를 따르는 random variable

경사하강법의 적용

Feature Scaling

표준화(Standardization)

Feature 샘플 평균 = 0

$$\left[Z = \frac{X - \mu_X}{\sigma_x} \rightarrow Z \sim (0, 1) \right]$$

- 평균 0, 분산 1의 범위 내에서 모든 feature들이 움직임
- 0과1 사이의 값이 아님
- 0을 중심으로 ±σ 값 : 68% ±2σ : 95%, ±3σ 99%의 range

Feature scaling을 하게 되면 미분을 통한 수렴 용이

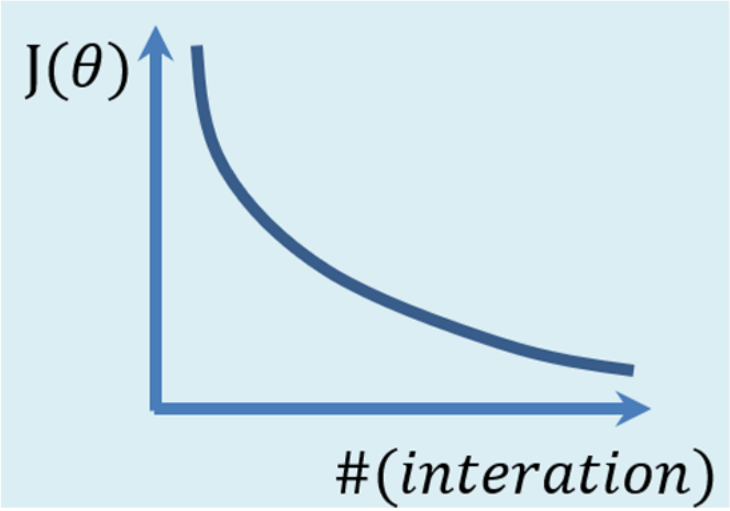
경사하강법의 적용

학습속도

학습속도 엔지니어가 선택해야 되는 파라미터

학습속도를 선택하는 기준

J(θ)는 iteration마다 감소



경사하강법의 적용

학습속도

알파값(학습속도)이 너무 작을 경우

수렴 속도 느림

알파값(학습속도)이 너무 클 경우

발산 가능성

알파값을 적절하게 선택하는 것이 매우 중요

경사하강법의 적용

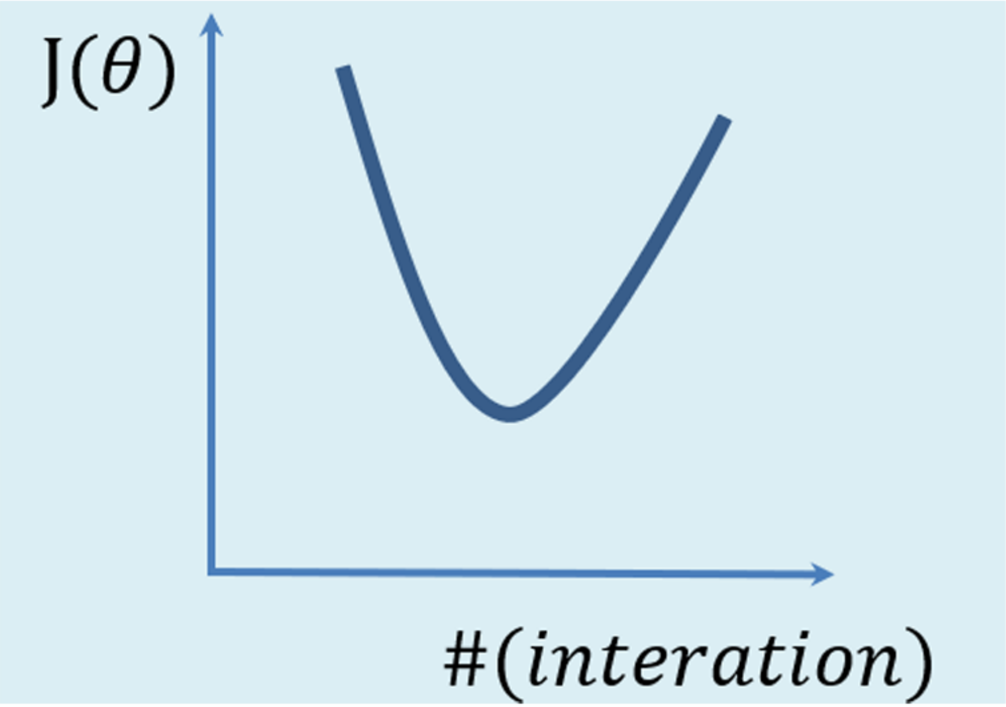
학습속도

[⋯ , 0.001, 0.003, 0.01, 0.03, 0.1, 0.3, 1, 3, ⋯]

- 알파값을 작은 값에서 조금씩 큰 값으로 변경

iteration 하는 과정에서
비용함수가 증가하는 구간이 있으면 안됨

➡ 그보다 작은 값을 선택



Analytic Solution of Linear Regression

OLS의 Analytic Solution

Analytic solution

$$[\; y = X\theta + \varepsilon \;]$$

$$[\; \varepsilon \perp X \;]$$

$$[\; \varepsilon \sim \mathrm{N}(0, \sigma^2 I) \;]$$

Analytic Solution of Linear Regression

OLS의 Analytic Solution

OLS 선형회귀의 파라미터를 구하는 philosophy

Residual의 최소화

$$\left[\sum_{i=1}^{m} e_j^{\,2} = e'e = (y - X\theta)'(y - X\theta) \right]$$

이 잔차제곱의 합을 최소화 하는 θ를

OLS 추정치로 함

Analytic Solution of Linear Regression

OLS의 Analytic Solution

목적함수를 수식으로 표현하면

$$\left[\frac{\partial (y - X\theta)'(y - X\theta)}{\partial \theta} = -2X'(y - X\theta) = 0 \right]$$

θ에 대해서 잔차제곱합을 1차 편미분 하면

$$\left[\text{Normal Equation} : X'y = X'X\theta \right]$$

Analytic Solution of Linear Regression

OLS의 Analytic Solution

Normal Equation을 풀면 **OLS estimator**

$$\left[\ \hat{\theta}^{OLS} = (X'X)^{-1}X'y\ \right]$$

이렇게 Analytic solution이 존재하게 됨

기타 추정방법

overfitting 문제

OLS 추정방법의 실질적 문제

overfitting

- 되도록 많은 수의 feature를 포함시키려고 함
- 모형이 복잡해지고 커지는 경향이 나타남

기타 추정방법

overfitting 문제

overfitting

= high variance 문제

- out of sample, test set에 대해서는
  설명을 전혀 못 하게 되는 과적합 문제 발생

정규화를 통해서 OLS문제를 해결하는 접근법

→ **Ridge regression**

기타 추정방법

Ridge Regression

Ridge Regression

L2 정규화를 통해서 해결

L2 norm

- element의 제곱을 합해서 스퀘어 루트
- 스퀘어 루트는 안해도 제곱의 합이기 때문에

L2 Regularization

기타 추정방법

Ridge Regression

잔차제곱합 + 패널티 항

$$\left[\min_{\theta} \sum_{i=1}^{m} (y_i - \hat{y}_i)^2 + \lambda \sum_{j=0}^{n} \theta_i^2 \right]$$

파라미터가 크면 패널티가 크게 됨

Ridge Regression의 philosophy → **파라미터값의 최소화**

기타 추정방법

Ridge Regression

λ

- Shrinkage(축소)의 정도를 조정
- 통상 0보다 큼
- λ가 클 경우 계수들이 0으로 수축

L2 Regularization

- 예측의 정도, 정확도가 높아짐
- 0계수 추정량의 bias와 variance간의 상관관계 최적화
- Feature들 간의 다중공선성 문제 해결

Ridge regression은 Principal Component Analysis (PCA)와 상당한 연관성이 있음

기타 추정방법

Lasso Regression

Lasso Regression

- Lasso : Least Absolute Shrinkage and Selection Operator
- L1 정규화를 통해서 해결
  - L1행렬에서 L1 norm은 절대값의 합으로 거리를 표현
  - 회귀계수의 절대값 합을 패널티 항으로 사용

$$\left[\min_{\theta} \sum_{i=1}^{m} (y_i - \hat{y}_i)^2 + \lambda \sum_{j=0}^{n} |\theta_j| \right]$$

  - 패널티의 shrinkage를 조정하는 λ (lambda)가 조절

기타 추정방법

Lasso Regression

$$\left[\min_{\theta}\sum_{i=1}^{m}(y_i - \hat{y}_i)^2 + \lambda\sum_{j=0}^{n}|\theta_j|\right]$$

- 절대값이기 때문에 최적값이 **코너 솔루션**이 될 확률이 Ridge에 비해 상당히 높음
- parameter가 제로가 되면 그 feature는 없어지게 됨

→ **feature selection 효과**

정보 손실, 정확도 감소, 그러나 과적합 문제 해결

기타 추정방법

Ridge & Lasso

Ridge	Lasso
L2 정규화	L1 정규화
변수선택 불가능	변수 선택 가능
Closed-form solution 존재	Closed-form solution 존재 X

파라미터의 크기를 줄여주는 효과

코너 솔루션을 통해

➤ 수치해석을 통해서 구함

기타 추정방법

Lasso Regression

Ridge	Lasso
변수간 상관관계가 높은 상황에서 좋 은 예측성능	상황에서 Ridge에 비해 상대적으로 예측성능 저하
크기가 큰 변수를 우선적으로 줄이는 경향	

- 파라미터의 크기가 크면
- 패널티가 크기 때문에

기타 추정방법

Elastic Net Regression

Elastic Net Regression

Ridge와 Lasso를 동시에 포함

$$\left[\min_{\theta}\sum_{i=1}^{m}(y_i - \hat{y}_i)^2 + \lambda_1\sum_{j=0}^{n}|\theta_j| + \lambda_2\sum_{j=0}^{n}\theta_j^2\right]$$

Ridge와 Lasso의 장점을 모두 가지고 있어
변수의 수도 줄이고 variance도 줄이고 싶을 때
사용 가능