

## 모델 훈련

박종민(jijupax@gmail.com)

May 19, 2020

- 선형 회귀를 알아봅니다.
  - 직접 계산할 수 있는 공식을 사용
  - 경사 하강법을 사용
- 비선형 데이터셋을 훈련시킬 수 있는 다항 회귀를 알아봅니다.
- 과대적합을 감지하는 방법을 알아봅니다.
- 과대적합을 감소시킬 수 있는 규제 방법을 알아봅니다.
- 로지스틱 회귀와 소프트맥스 회귀를 알아봅니다.

$$\begin{aligned}\hat{y} &= \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n \\ &= h_{\theta}(\mathbf{x}) = \theta \cdot \mathbf{x}\end{aligned}\tag{1}$$

- $\theta$ 는 선형 회귀 모델의 파라미터이고, 각 특성과 곱해집니다.
- 식 1과 같이 입력 특성의 가중치 합과 편향의 합으로 표현할 수 있습니다.

$$\text{MSE}(\mathbf{X}, h_{\theta}) = \frac{1}{m} \sum_{i=1}^m \left( \theta^T \mathbf{x}^{(i)} - y^{(i)} \right)^2 \quad (2)$$

- 회귀 모델을 훈련시키는 방법은 식 2와 같이 비용 함수인 평균 제곱 오차(MSE)를 구해 그 값을 최소화하는 것입니다.

$$\hat{\theta} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y} \quad (3)$$

- 비용 함수를 최소화하는  $\theta$ 를 찾기 위한 해석적인 방법을 식 3과 같이 정규방정식이라고 합니다.

```
1 theta_best = np.linalg.inv(X_b.T.dot(X_b)).dot(X_b.T).dot(y)
2
3 from sklearn.linear_model import LinearRegression
4 lin_reg = LinearRegression()
5 lin_reg.fit(X, y)
6
7 theta_best_svd, ... = np.linalg.lstsq(X_b, y, rcond=1e-6)
```

- numpy를 사용하여 식 3을 직접 계산 하는 것이 가능합니다.
- sklearn의 LinearRegression 또는 numpy.linalg.lstsq 함수를 사용해도 됩니다.

## Gradient Descent

비용 함수를 최소화하기 위해 반복해서 파라미터를 조정해가는 방법입니다.

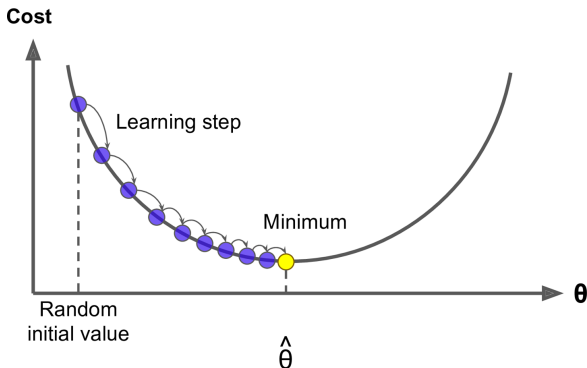


Figure 1: 경사 하강법

- 파라미터 벡터  $\theta$ 에 대해 비용 함수의 현재 그래디언트를 계산하여 감소하는 방향으로 진행합니다.
- $\theta$ 를 임의의 값으로 시작해서 한 번에 조금씩 비용 함수가 감소되는 방향으로 진행합니다.
- 학습률의 크기에 따라 알고리즘의 수렴 속도가 틀리고, 클 경우 발산하거나 작을 경우 지역 최소값에 빠지기도 합니다.



$$\begin{aligned}\frac{\partial}{\partial \theta_j} \text{MSE}(\boldsymbol{\theta}) &= \frac{2}{m} \sum_{i=1}^m (\boldsymbol{\theta}^T \mathbf{x}^{(i)} - y^{(i)}) x_j^{(i)} \\ \nabla_{\boldsymbol{\theta}} \text{MSE}(\boldsymbol{\theta}) &= \frac{2}{m} \mathbf{X}^T (\mathbf{X}\boldsymbol{\theta} - \mathbf{y})\end{aligned}\tag{4}$$

- 매 경사 하강법 스텝에서 전체 훈련 세트  $\mathbf{X}$ 에 대해 계산하는 방법입니다.
- 식 4는 배치 경사 하강법의 비용함수 그래디언트 벡터를 나타내는 것입니다.
- 경사 하강법의 스텝:  $\boldsymbol{\theta}^{(\text{다음 스텝})} = \boldsymbol{\theta} - \eta \nabla_{\boldsymbol{\theta}} \text{MSE}(\boldsymbol{\theta})$

- 매 스텝에서 딱 1개의 샘플을 무작위로 선택하여 그래디언트를 계산하는 방법입니다.
- 매우 적은 데이터를 처리하므로 배치 경사 하강법에 비해 훨씬 빠르고, 메모리를 적게 차지합니다.
- 무작위로 샘플을 선택하므로 전역 최소값 주변을 맴돌면서 수렴하게 됩니다.
- 학습 스케줄을 조절하여 주변을 맴도는 문제를 해결할 수 있습니다.

- 미니배치라고 부르는 임의의 작은 샘플 세트에 대해 그래디언트를 계산하는 방법입니다.
- 미니배치 크기를 적당히 조절하면 확률적 경사 하강법보다 덜 불규칙하게 수렴하게 됩니다.
- GPU를 사용한 병렬 계산에 유리한 방법입니다.

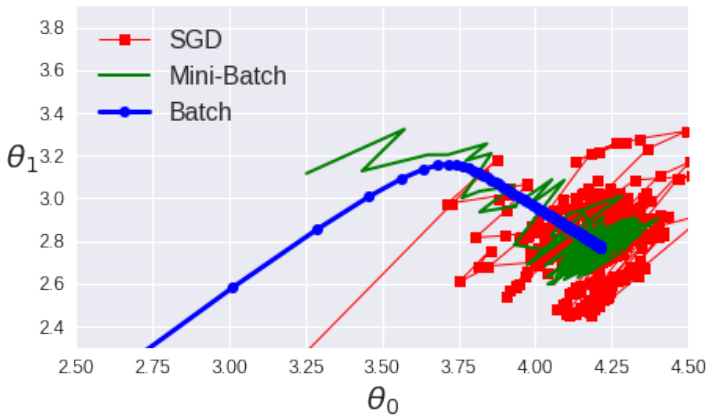


Figure 2: 파라미터 공간에 표시된 경사 하강법의 경로

# 선형 회귀 알고리즘 비교

알고리즘	$m$ 이 클 때	외부 메모리 학습 지원	$n$ 이 클 때	하이퍼파라미터 수	스케일조정 필요
정규방정식	빠름	No	느림	0	No
SVD	빠름	No	느림	0	No
GD	느림	No	빠름	2	Yes
SGD	빠름	Yes	빠름	$\geq 2$	Yes
Mini-batch	빠름	Yes	빠름	$\geq 2$	Yes

Table 1: 선형 회귀를 사용한 알고리즘 비교

- 각 특성의 거듭 제곱을 새로운 특성으로 추가하여 확장된 특성을 포함한 데이터셋에 대해 선형 모델을 훈련 시키는 방법입니다.

```
1 from sklearn.preprocessing import PolynomialFeatures
2 poly_ = PolynomialFeatures(degree=2, include_bias=False)
3 X_poly = poly_.fit_transform(X)
4
5 lin_reg = LinearRegression()
6 lin_reg.fit(X_poly, y)
```

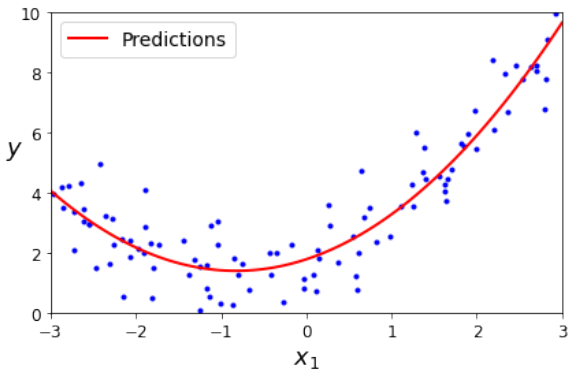


Figure 3: 다항 회귀 모델의 예측

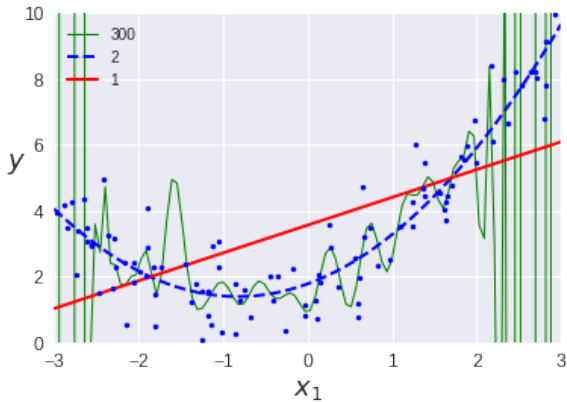


Figure 4: 고차 다항 회귀



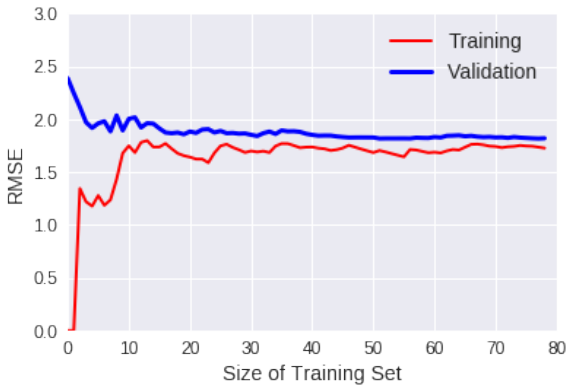


Figure 5: 선형 회귀 학습 곡선

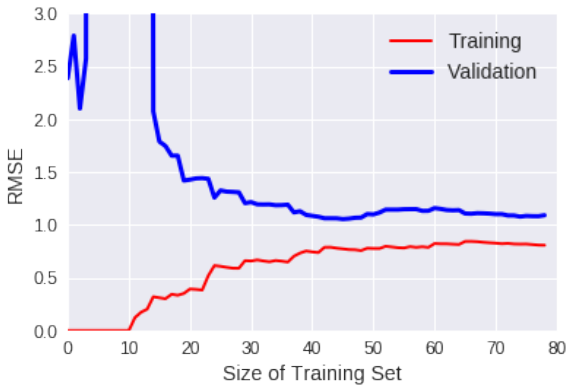


Figure 6: 다항 회귀 학습 곡선

- 편향
  - 잘못된 가정으로 발생합니다.
  - 과소적합이 되기 쉬습니니다.
- 분산
  - 훈련 데이터에 있는 작은 변동에 모델이 과도하게 민감하기 때문에 발생합니다.
  - 자유도가 높은 모델이 높은 분산을 가지기 쉬워 과대 적합 경향이 있습니다.
- 줄일 수 없는 오차
  - 데이터 자체에 있는 노이즈 때문에 발생합니다.
  - 데이터에서 노이즈를 제거하는 방법이 유일합니다.
- 모델의 복잡도가 커지면 분산이 늘어나고 편향은 줄어듭니다.
- 모델의 복잡도가 줄어들면 분산은 작아지고 편향이 커집니다.

$$J(\theta) = \text{MSE}(\theta) + \alpha \frac{1}{2} \sum_{i=1}^n \theta_i^2 \quad (5)$$

- 규제항  $\alpha \sum_{i=1}^n \theta_i^2$ 이 비용 함수에 추가된 것입니다.
- 규제항은 훈련하는 동안에만 비용 함수에 추가 되고, 테스트 시엔 사용하지 않습니다.
- $\alpha = 0$ 이면 선형 회귀,  $\alpha$ 가 아주 크면 모든 가중치가 거의 0에 가까워지고 데이터의 평균을 지나는 수평선이 됩니다.

$$J(\boldsymbol{\theta}) = \text{MSE}(\boldsymbol{\theta}) + \alpha \sum_{i=1}^n |\theta_i| \quad (6)$$

- Least Absolute Shrinkage and Selection Operator
- 규제항  $\alpha \sum_{i=1}^n |\theta_i|$ 이 비용 함수에 추가된 것입니다.
- 덜 중요한 특성의 가중치를 완전히 제거하는 특징이 있으므로 희소 모델을 만듭니다.

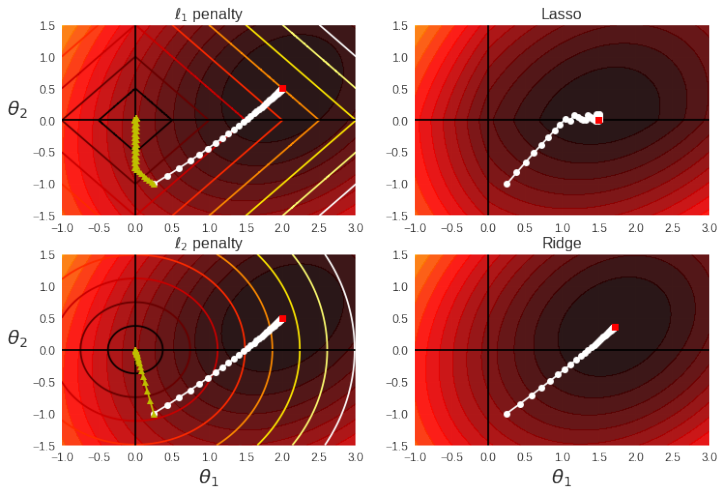


Figure 7: 라쏘 대 릿지 규제

$$J(\theta) = \text{MSE}(\theta) + r\alpha \sum_{i=1}^n |\theta_i| + \frac{1-r}{2}\alpha \sum_{i=1}^n \theta_i^2, \quad 0 \leq r \leq 1 \quad (7)$$

- 릿지 회귀와 라쏘 회귀를 절충한 모델입니다.
- 혼합 비율  $r$ 을 사용해 조절하여  $r = 0$ 이면 릿지 회귀이고  $r = 1$ 이면 라쏘 회귀입니다.
- 릿지 회귀가 기본이 되지만 실제 쓰이는 특성이 몇 개뿐이라면 라쏘 회귀나 엘라스틱넷을 씁니다.
- 특성 수가 훈련 샘플 수보다 많거나 특성 몇개가 강하게 연관되어 있다면 엘라스틱넷을 씁니다.

- 검증 에러가 최소값에 도달하면 바로 훈련을 중지하는 방법입니다.
- 검증 에러가 증가하기 시작하면 과대 적합이 시작하는 것을 의미합니다.
- 검증 에러가 일정 시간 동안 최소값보다 크다는 확신이 들 때 학습을 멈추고 최소였을 때의 모델 파라미터로 되돌립니다.
- Beautiful free lunch, *Geoffrey Hinton*



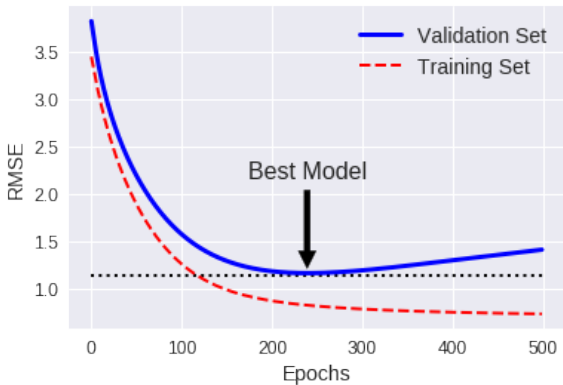


Figure 8: 라쏘 대 릿지 규제

- 실습: <https://github.com/rickiepark/handson-ml2>
- 핸드즈온 머신러닝 1판: <https://bit.ly/2Tgfd2w>