


ML_05. Regularization - 규제

🕒 생성일	@2022년 6월 8일 오전 1:10
🏷️ 유형	머신러닝/딥러닝
👤 작성자	 동훈 오

정형화 (regularization) ?

과대적합이나 과소 적합과 같은 상황에서, 모델이 학습을 진행하며 정답을 잘 찾아가지 못할 때, 모델에 추가적인 정보를 더하는 등 프로세스를 통해 개선시키는 전략이다.

정형화의 종류에는 다음의 것들이 있다.

- 제한을 거는 파라미터 값을 넣는 방법
- 학습 데이터를 설명하는 여러 개의 가설 모델들을 혼합하는 방법
- 최적화 프로세스를 수정하는 방법
- 데이터 분포의 이전 정보를 이용하여 추가적인 학습데이터를 더하는 방법.

Norm 기반 정형화

머신러닝은 학습 가능한 파라미터를 가진 목적함수(= 비용함수, 손실함수) 를 튜닝하는 것이므로 목적함수에 정형화 함수를 추가하여 사용할 수 있다.

정형화 함수에는 여러 종류가 있지만 가장 단순한 형태는 파라미터 θ 의 크기(norm) 을 구하는 것이다. norm 의 크기를 구하는 것이므로 θ 에 절댓값이 붙고, 이것을 그냥 사용할 수도 있지만, 형태를 약간 변형 시켜서 사용할 수도 있다.

혼공머신 : 3장 회귀 알고리즘과 모델 규제

(혼공머신 부분에서는 규제 라이브러리가 어떻게 적용되는지, 코드 상 구현만 확인하고 넘어가면 좋을 것 같다.)

모델이 훈련 세트에 과대적합되지 않도록 만드는 것을 규제라고 한다. 선형 회귀 모델의 경우 특성에 곱해지는 계수의 크기를 작게 만드는 일이다.

일반적으로 선형 회귀 모델에 규제를 적용할 때 계수 값의 크기가 서로 많이 다르면 공정하게 제어되지 않을 것이다. 그래서 규제를 적용하기 전에 먼저 정규화를 해야한다. (특성의 스케일 고려.)

다음은 'ML_02. 회귀 모델' 혼공머신 부분 - 다항 회귀 코드의 연장선상에서, 규제를 적용해보겠다. 우선은 사이킷런에서 제공하는 표준 전처리 클래스인 StandardScaler 를 사용하여 객체 ss 를 초기화한 후 PolynomialFeatures 클래스로 만든 train_poly를 사용해 이 객체를 훈련한다.

```
from sklearn.preprocessing import StandardScaler
ss = StandardScaler()
ss.fit(train_poly)
train_scaled = ss.transform(train_poly)
test_scaled = ss.transform(test_poly)
```

릿지(Ridge)

릿지는 계수를 제공한 값을 기준으로 규제를 적용한다. 사이킷런에서는 sklearn.linear_model 의 하위 클래스로 기능을 제공한다.

```
from sklearn.linear_model import Ridge
ridge = Ridge()
ridge.fit(train_scaled, train_target)

print(ridge.score(train_scaled, train_target))
>>> 0.9896101671037343

print(ridge.score(test_scaled, test_target))
0.9790693977615397
```

라쏘(Lasso)

라쏘는 계수의 절댓값을 기준으로 규제를 적용한다. 사이킷런에서 적용법은 릿지와 동일하다.

```
from sklearn.linear_model import Lasso
lasso = Lasso()
lasso.fit(train_scaled, train_target)

print(lasso.score(train_scaled, train_target))
>>> 0.989789897208096

print(lasso.score(test_scaled, test_target))
0.9800593698421883
```

핸즈온 : 4장 모델 훈련 - 규제가 있는 선형 모델

과대적합을 감소시키는 좋은 방법은 모델을 규제하는 것이다. 자유도를 줄이면 데이터에 과대적합되기 더 어려워진다.

선형 회귀 모델에서는 보통 모델의 가중치를 제한함으로써 규제를 가한다.

릿지 회귀

릿지 회귀는 규제가 추가된 선형 회귀 버전이다. 규제항이 비용함수에 추가된다. 이는 학습 알고리즘을 데이터에 맞추는 것뿐만 아니라 모델의 가중치가 가능한 한 작게 유지되도록 노력한다. 규제항은 훈련하는 동안에만 비용 함수에 추가된다. 모델의 훈련이 끝나면 모델의 성능을 규제가 없는 성능 지표로 평가된다. (테스트 세트에서 성능 평가시, 모델에 규제가 적용되지 않는다.) 다음 식의 릿지 회귀의 비용함수이다.

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

위 식에서 보드시피 규제항은 회귀 모델의 각 파라미터를 제곱한 값에 알파값이 곱해진 형태이다. 하이퍼파라미터 α 는 규제 정도를 조절한다. α 가 아주 크면 모든 가중치가 거의 0에 가까워지고 결국 데이터의 평균을 지나는 수평선이 된다.

아래 그래프는 α 에 따른 모델의 전개가 다르게 됨을 보인다. 특히, 오른쪽 다항 회귀에서는 α 를 증가시킬수록 직선에 가까워지는 것을 볼 수 있으며 모델의 분산은 감소하고, 편향은 커지게 된다.

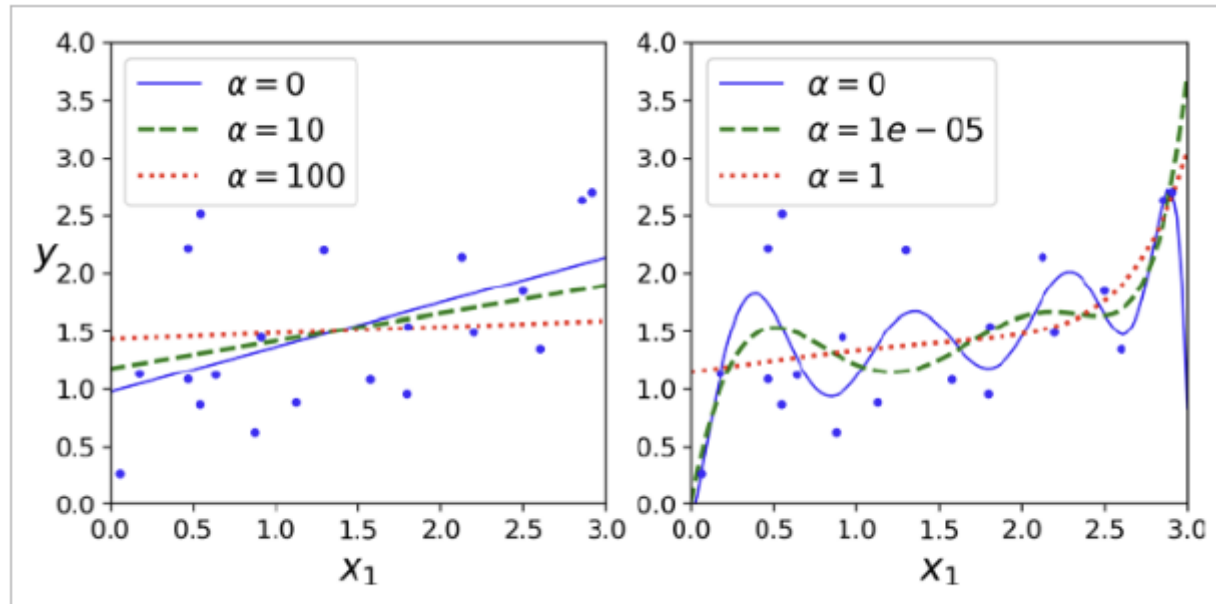


그림 4-17 다양한 수준의 릿지 규제를 사용한 선형 회귀(왼쪽)와 다항 회귀(오른쪽).

라쏘 회귀

라쏘 회귀는 릿지 회귀처럼 비용 함수에 규제항을 더하지만 가중치 벡터의 $l_1 norm$ 을 사용한다.

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

라쏘 회귀의 중요한 특징은 덜 중요한 특성의 가중치를 제거하려고 한다는 점이다. (일부 특성의 가중치를 0으로 만든다.) 다른 표현으로, 라쏘 회귀는 자동으로 특성 선택을 하고 희소 모델을 만든다.(0이 아닌 특성의 가중치가 적게 된다.)

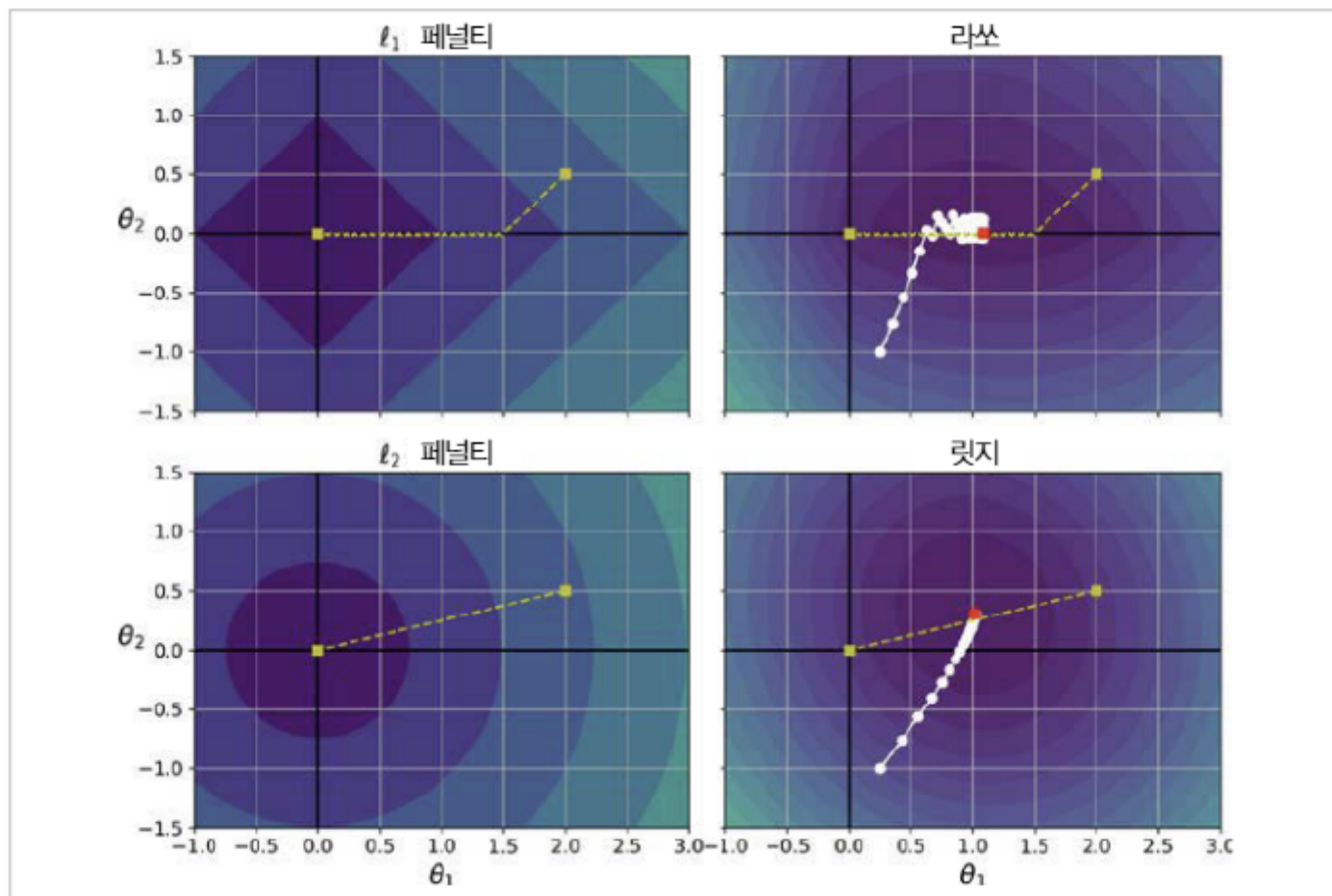


그림 4-19 라쏘 대 릿지 규제

위 이미지에서 두 축은 모델 파라미터 두 개를 나타내고 배경의 등고선은 각기 다른 손실 함수를 나타낸다.

1. 왼쪽 위 그래프의 등고선은 l_1 손실($=|\theta_1| + |\theta_2|$) 을 나타낸다. 축에 가까워지면서 선형적으로 줄어든다.
2. 오른쪽 위 그래프의 등고선은 라쏘 손실 함수를 나타낸다. (l_1 손실 + MSE) 하얀 작은 원들의 집합이 모델 파라미터를 최적화하는 과정을 보여준다.
3. 왼쪽 아래 그래프에서는 l_2 손실을 나타낸다. 원점에 선형적 접근하는 것을 확인할 수 있다.
4. 오른쪽 아래 그래프의 등고선은 릿지 회귀의 비용 함수를 나타낸다. (l_2 + MSE) 라쏘와는 다르게, 전역 최적점에 가까워질수록 그래디언트가 작아진다. 경사하강법이 자동으로 느려지고 수렴에 도움이 되는 경향이 강하다.

엘라스틱 넷 (elastic net)

릿지 회귀와 라쏘 회귀를 절충한 모델이다. 규제항은 릿지와 라쏘의 규제항을 단순히 더해서 사용하며 혼합 정도는 혼합 비율 r 을 사용해서 조절한다. 다음 식은 엘라스틱넷의 비용함수이다.

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

조기 종료(early stopping)

경사 하강법과 같은 반복적인 학습 알고리즘을 규제하는 아주 색다른 방식은 검증 에러가 최솟값에 도달하면 바로 훈련을 중지시키는 것이다. 아래 이미지에서 검증 세트 그래프를 따라가면, 감소하던 검증 에러(RMSE)가 어느 순간 상승하는 것을 확인할 수 있다. 이 구간부터 과대적합이 시작되는 것이고, 조기 종료는 이 시점에서 훈련을 멈추는 것이다.

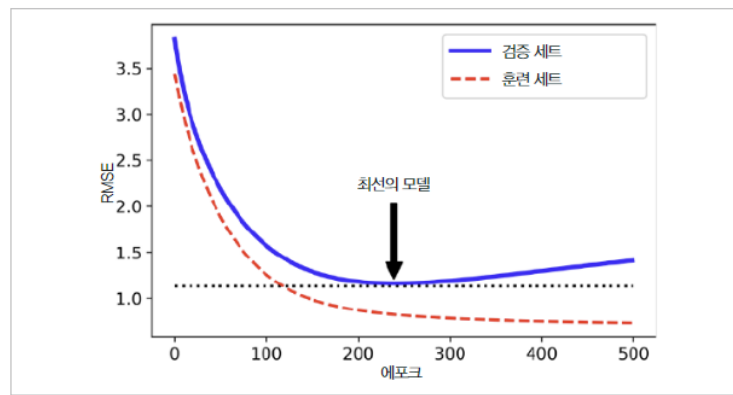


그림 4-20 조기 종료 규제

텐서플로우 케라스에서는 callbacks 라이브러리에 EarlyStopping 클래스를 구현해놓았다. 아직 딥러닝 모델을 설명하지 않았지만, 딥러닝 모델을 훈련하기 직전(=모델 컴파일할 때)에 EarlyStopping을 첨가해 준다고 생각하면 된다.

```
tf.keras.callbacks.EarlyStopping(
    # 다음의 매개변수는 디폴트로 지정된 상태를 보여준다.
    # 원하는 값으로 조정할 수 있다.
    monitor='val_loss',
    min_delta=0,
    patience=0,      # 중요한 파라미터, 손실함수가 몇 번이나 연속으로 감소변화가 없을 때까지 참을 것인지 결정.
    verbose=0,
    mode='auto',
    baseline=None,
    restore_best_weights=False
)
```

ISLR : 6. Linear model selection and Regularization

파라미터를 규제하는 것은 비용함수에서의 분산을 상당히 줄이고 예측 정확도를 높인다.

(규제에 대해 원문에서는 shrinkage methods 라고 표현하며 regularization 영역에서의 활용할 수 있는 아이디어 중 하나라고 설명한다.) 이해를 위해 규제라는 표현을 shrinking 과 같은 의미로 사용하겠다.

회귀 모델에서 파라미터를 규제하는 (예측에 영향을 안미치는 일부 파라미터를 0으로 만드느) 잘 알려진 전략은 릿지와 라쏘 회귀이다.