선형회귀와 선형분류 모델을 이용해서 성능을 높이기 위해서 사용했던 팁

1. Learning rate를 어떤 방향으로 조절해야 하는지
2. Data preprocessing 데이터 정규화가 필요한 이유
3. Overfitting을 피하기 위한 방법
4. 성능을 높일 수 있는 방법

# Learning rate

Gradient descent를 진행하면서 다음 step을 위한 보폭= learning rate

Learning rate값이 크다=보폭이 크다.

1. Leaning rate가 너무 클 경우

Overshooting이 발생한다.

NaN: 발산했다는 의미

1. Leaning rate가 너무 작을 경우

너무 시간이 오래 걸리는 단점이 있다.

그렇다고 빠른 수렴을 위해서 learning rate를 크게 하는 것은 올바르지 않다.

때론 Global minimum를 찾지 못할 수 있기 때문이다.

그래서 학습 중간중간에 literation에 따라 learning rate를 감소시킨다. 0.1 -> 0.01 -> 0.001

이를 위해 pytorch함수 내에서도 Learning rate를 위한 스케줄러를 설정할 수 있다. 학습을 용이하게 할 수 있다.

🡪 적절한 learning rate를 찾는 것이 중요하다. 특별한 방법은 없고, 실험적으로 찾아야만 한다.

# Data preprocessing. 데이터 정규화가 필요한 이유

1D의 cost함수는 weight가 x축, cost가 y축인 2차함수형태가 나타난다.

하지만 2D의 경우 3차원으로 볼륨이 있게 나타난다.

7p 그림은 이를 위에서 바라본 것이다.

W1과 w2에 대해서 정방원이 그려지고 있다. W1과 w2가 비슷한 상태에서 Gradient descent가 이루어지고 있는 것이다.

Sgb와 같은 최적화 알고리즘을 사용 시 영역을 탐색하는 데 있어서 적절한 알고리즘이다.

8p

X1 x2가 scale이 다르다. Y는 클래스

X1 특성과 x2 특성의 표현하는 값의 range가 너무 차이가 나기 때문에

이를 맞추기 위해서 w1과 w2의 scale이 차이가 나게 된다. W1의 range가 크고, w2의 range가 작게 된다.

그래서 그림을 그려보면 비정방원으로 구성된 탐색영역이 만들어진다.

단점: 최적화 문제를 풀기위한 gradient descent문제를 적용했을 때, 진동하여 조밀조밀한 영역을 탐색할 수 없게 된다. Learning rate가 모든 가중치(w1, w2)에 동일하게 적용이 되어서 w1은 곱해지는 값에 대해 많은 영향력을 받게 되기 때문이다.

🡪 그래서 데이터 정규화를 통해 변형시켜주어야 한다.

가장 보편적인 방법은 zero-centeroid normalized 방법

Original data가 center가 아닌 살짝 오른쪽에 분포 되어있다.

이를 평균값이 좌표의 원점에 오도록 이동시킨다. (zero-centered data)

그리고 좌표계의 각 축의 방향에 미치는 영향력의 scale이 서로 다른 것을 동일한 크기로 맞추는 과정을 거친다.(normalized data)

🡪 최종적인 데이터는 평균값이 원점에 있고, 좌표계의 scale이 동일 해진다.

우리에게 주어지는 일반적인 데이터는 정규화 되어있지 않다. 그래서 데이터 정규화를 통해 좀더 안정적으로 탐색하면서 수렴하는 값을 찾도록 도와야한다.

# Overfitting

: 주어진 데이터에 대해서만 잘 표현하려고 하는 방향성.

캐글에서 모델 학습 시, train데이터를 이용하고,

정답을 알 수 없는 test데이터를 통해 실제 모델이 얼마나 test데이터에 대한 성능을 가지고 있는지 평가했다.

학습데이터의 정확도만 보았을 때에는 정확히 분류하고 있기때문에 100%이지만,

성능이 80%로 낮아질 수 있다.

하지만 왼쪽이 good model이다.

우리의 모델은 train과 test데이터 둘다에 있어서 성능이 좋아야 한다.

우리는 보통 데이터를 랜덤하게 train, test로 나누기 때문에 두 데이터의 경향성은 매우 유사하다.

실제로 overfitting모델의 경우 과도하게 학습데이터만 잘 분류하려고 하다보니

전체적인 경향성을 잃어버리고, test데이터에 대해서는 오분류한다.

train데이터에 대해서만 성능이 좋고, test데이터에 대해서는 성능이 좋지 않은 모델.

Overfitting이 되었음을 알 수 있나요?

학습 데이터에 대한 정확도는 높은데, test데이터에 대한 성능이 낮다면 overfitting이다.

모델이 분류를 잘하고, 데이터의 경향성을 잘 표현한 데이터이다. 학습 데이터와 test데이터의 성능이 거의 유사해

\* 학습 과정에서 Overfitting을 줄이는 방법

1. 학습데이터를 많이 모으기

: 모델은 데이터의 경향성을 잘 표현하기 위해 노력하게 된다.

1. Feature의 수를 줄이기

feature를 분석해서 의미 없는 feature를 줄이는 것이 좋다.

Ex) 상관관계를 따져 보기

1. Regularization

= 정규화. w가 너무 커지지 않는 방향으로 설계하고자 한다.

w가 너무 크면 overfitting가능성이 높기 때문이다.

W가 크면 w를 미세조정해서 loss를 낮추는 방향이 아닌, 학습방향이 w를 줄이는 방향이 될 수 있다. 그래서 w자체가 커지지 않도록 제어하는 식을 포함한 것이다. (=regularization)

이를 통해 overfitting을 피한다. 감마는 정규화 반영정도를 나타내는 파라미터이다.

# Performance evaluation

학습데이터를 통해 학습된 모델에 테스트 데이터를 넣어서 값을 분류하거나 추정했다.

그리고 나서 test의 추정된 값이 정답과 얼마나 유사한지를 평가했다.

평가의 핵심은 학습된 파라미터 w와 b이다. (모델은 w와 b를 구하는 과정이므로)

평가하는 과정에서는 반드시 학습데이터로 사용된 적 없는 unseen data를 사용해야 한다.

Train과 test데이터가 서로 겹치지 않게 만드는 것이 핵심이다. (disjointed하게 랜덤으로)

모델이 데이터를 외우는 것이 아니라,

데이터를 이해하고 분석해서 유사한 다른 상황에도 모델이 올바르게 예측하는 것을 목표로 한다.

17p validation

학습하는 과정 중간중간에 모델이 잘 학습되고있는지 validation데이터로 평가할 수 있다.

train데이터를 training과 validation으로 나눈다.

Radom 샘플링해서 disjointed하게 나누면 된다. 비율은 보통 7:3, 8:2이다.