CS231n Summary_Lecture3

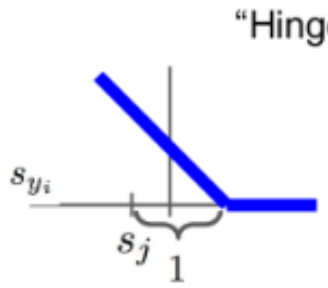
Euron CV팀 임연우

- **손실함수(loss function)란?**

W를 어떻게 정할지 생각해보면, loss func을 이용해 W를 평가해 W를 갱신해나간다.

Loss func은 가중치W가 얼마나 나쁜지를 정량적으로 보여주는 함수.

- **대표적인 손실함수 hinge loss**



$$L_i = \sum_{j \neq y_i} \begin{cases} 0 & \text{if } s_{y_i} \geq s_j + 1 \\ s_j - s_{y_i} + 1 & \text{otherwise} \end{cases}$$

$$= \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$

$$L = \frac{1}{N} \sum_i L_i(f(x_i, W), y_i)$$

ground truth인 Yi를 제외한 모든 y에 대해서,

Syi가 해당 점수보다 1이상 크면 0을, 그렇지 않으면 Sj-Syi+1을 더한 값을 loss func의 값으로 한다.

Q1. 이미 1이상 크다면 더 커지든 말든 loss func 값 동일

Q2. Loss의 min은 0, max는 무한.

Q3. 맨 처음에 W는 아주 작은 값으로 초기화한다. 이때 모든 S는 0에 가깝게 된다. 이런 상황에서의 loss값은 어떻게 될까?

두 score값이 비슷하기 때문에 1 값이 나오고, 그것을 (#class – 1)번 반복하기 때문에, loss 값은 (#class – 1)번이 된다.

Q4. Loss 구할때 (#class – 1)번말고 (#class)번 더한다면, 즉 loss 구하는 식이 ground truth

인 Yi를 포함한다면 loss는 어떻게 될까?

+1이 된다. 성능에는 전혀 문제가 없으나, min 값을 0으로 맞추려면 그러지 않는 것이 좋다.

Q5. 합 대신에 평균을 사용한다면?

전체적으로 같은 값으로 나뉠 뿐 영향을 받지 않는다.

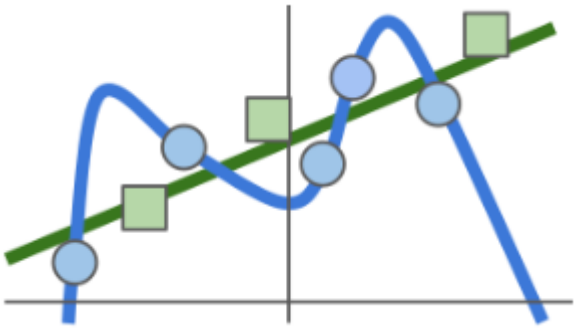
Q6. (Sj-Syi+1)의 제곱을 더한다면 어떻게 될까?

Loss의 값이 완전히 달라진다. 이 loss func을 squared loss func이라 부른다.

- **Regularization이 필요한 이유**

Loss가 0인 W가 unique할까? => 아니다. 2W도 loss가 0.

우리는 training data에 맞추기보다 그렇게 학습된 classifier로 test data의 label을 맞춰야함.
즉, training data에 대한 performance는 별로 신경 안씀.



파란색 training data에 너무 꼭 맞는 (과적합 over-fitting) 선을 그리게 되면,

초록색 test data가 들어왔을 때 잘 구분해내지 못함.

$$L(W) = \underbrace{\frac{1}{N}\sum_{i=1}^{N} L_i(f(x_i, W), y_i)}_{\text{Data loss: Model predictions should match training data}} + \underbrace{\lambda R(W)}_{\text{Regularization: Model should be “simple”, so it works on test data}}$$

이러한 과적합을 해결하기 위해 loss func식에 Regularization term을 붙여서 이용함.

Regularization penalty를 주는 것.

여기서 람다는 hyperparameter.

- Regularization의 종류

Regularization

λ = regularization strength (hyperparameter)

$$L = \frac{1}{N}\sum_{i=1}^{N}\sum_{j\neq y_i}\max(0, f(x_i; W)_j - f(x_i; W)_{y_i} + 1) + \lambda R(W)$$

In common use:

L2 regularization $R(W) = \sum_k \sum_l W_{k,l}^2$

L1 regularization $R(W) = \sum_k \sum_l |W_{k,l}|$

Elastic net (L1 + L2) $R(W) = \sum_k \sum_l \beta W_{k,l}^2 + |W_{k,l}|$

Max norm regularization (might see later)

Dropout (will see later)

Fancier: Batch normalization, stochastic depth

- L1, L2 regularization 비교

L2 Regularization (Weight Decay)

$x = [1, 1, 1, 1]$

$R(W) = \sum_k \sum_l W_{k,l}^2$

$w_1 = [1, 0, 0, 0]$

$w_2 = [0.25, 0.25, 0.25, 0.25]$

(If you are a Bayesian: L2 regularization also corresponds MAP inference using a Gaussian prior on W)

$$w_1^T x = w_2^T x = 1$$

W1과 W2 모두, xW로 내적하면 1이 나온다.

W1은 첫번째 원소 외에 다른 원소들은 무시, W2는 모든 원소들을 골고루 반영.

L1

- W1을 선호. Sparse solution.
- 0의 개수에 따라 모델 복잡도가 달라짐. W의 원소 대부분을 0이 되게함.

(복잡도 = W의 0이 아닌 개수)

L2

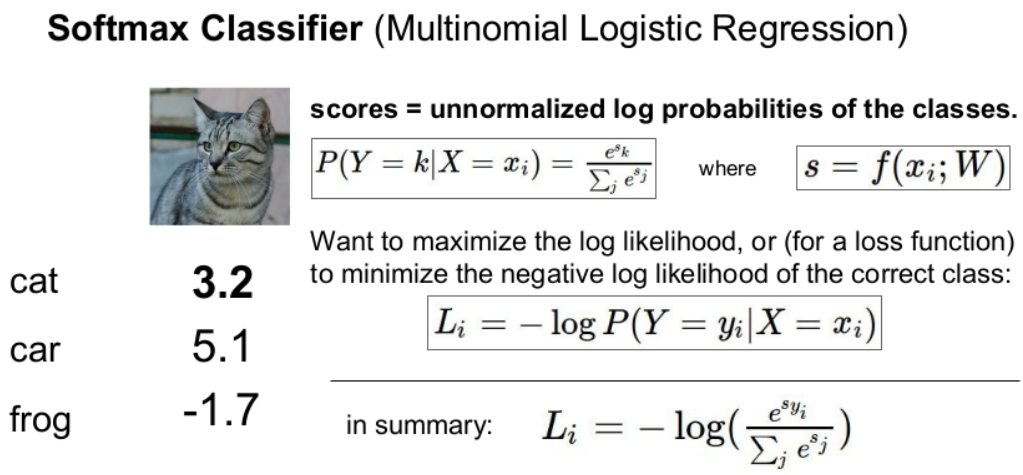
- W2를 선호. Coarse solution.
- 모든 원소가 골고루 영향을 미치게함.
- W의 feature를 최대한 고려하며, 동일한 점수라면 더 넓게 퍼져있는 것을 선호함. 숫자가 더 넓게 퍼질 때 모델 복잡도는 덜 복잡해짐.

- Softmax Classifier

또 다른 classifier로, 이항의 logistic regession을 다차원으로 일반화 시킨 것.

Multiclass SVM loss는 예측 점수 자체는 고려x, 정답클래스의 예측점수가 오답클래스의 예측점수보다 크기만을 바람.

그러나 softmax는 클래스별 확률 분호를 사용해 예측점수 자체에 추가적인 의미를 부여.



여기서 사용하는 loss를 cross entropy loss라고 하며 정답이 나올 확률의 최소값을 의미함.

Loss = -log(정답클래스가 정답일 확률).

- cross entropy loss

Q1. Cross entropy loss의 최소, 최대값은?

0 부터 무한대까지

최솟값 = -log(1) = 0 = 정답일 때

최댓값 = -log(0) = 무한 (그러나 유한정밀도를 가지고 무한을 얻을 수는 없음. 이런 일은 발생하지 않음.)

Q2. W 를 처음 학습할 때 W 를 아주 작은 값으로 초기화한다. 이때 s 는 0 에 가까워진다. 이런 상황에서 loss 값은?

Debugging Strategy (sanity check) : 첫번째 iteration 에서 Loss 가 log(C)가 아니라면 버그가 걸린 것

Q3. 예측 점수를 조금 변화시킨다면 각각의 함수에서 loss 는 어떻게 될까?

SVM: 점수자체보다 정답클래스의 예측점수 vs 오답클래스의 예측점수의 차이에 집중하기에, loss 변하지 않음. (예측 점수가 많이 변하면 hinge loss 도 변하긴함)

Softmax: 정답클래스의 예측점수를 확률로 만들기에 loss 가 변함.