2018100674 기계공학과 강하늘

[ML_08] Multiclass Classification

One vs Rest Method

- 이진분류기 C개를 독립적으로 사용하여 class k와 나머지 C-1개 class를 분류
- Class k에 대한 이진분류기를 h_k라 하면, $h_k(x)$가 가장 큰 값을 갖는 k로 분류

→ 각 이진분류기에 대해 훈련집합의 불균형을 고려함

One vs One Method

- 이진분류기 $C(C-1)/2$개를 독립적으로 사용하여 class k와 class l을 분류
- 가장 많은 이진분류기가 선택한 class를 최종 결과로 결정

→ 훈련집합의 불균형을 고려하지 않음,
사용되는 이진분류기의 개수가 C^2에 비례함: 높은 복잡도를 가짐.

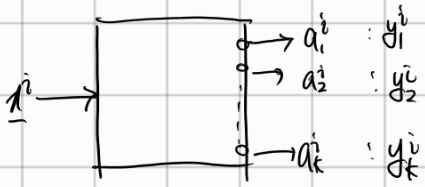
Softmax

- Softmax function

$\sigma : \mathbb{R}^K \to (0,1)^K$

$\sigma(\underline{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$ → $\underline{z} = \begin{bmatrix} 4 \\ 2 \\ -1 \end{bmatrix}$ $\sigma(\underline{z}) = \begin{bmatrix} e^4 \\ e^2 \\ e^{-1} \end{bmatrix} / (e^4+e^2+e^{-1})$

- Cross Entropy Loss



$\Rightarrow \frac{1}{N} \sum_{n=1}^{N} \sum_{k=1}^{K} -y_k^n \log(a_k^n)$

CELoss at output layer

Cross-Entropy Loss

$J(w) = \frac{1}{N} \sum_{n=1}^{N} \sum_{i=1}^{K} -y_i^n \log a_i^n$

$= \frac{1}{N} \sum_{n=1}^{N} \sum_{i=1}^{K} -y_i^n \log \frac{\exp(\underline{w}_i^T \underline{x}^n)}{\sum_{j=1}^{K} \exp(\underline{w}_j^T \underline{x}^n)}$, N=샘플개수, K=출력층노드개수

Gradient of CELoss

$\frac{\partial J(w)}{\partial \underline{w}_k} = \frac{1}{N} \sum_{n=1}^{N} \sum_{i=1}^{K} -y_i^n \frac{\partial}{\partial \underline{w}_k} \log a_i^n$

↳ k번째 노드로 향하는 가중치들의 벡터

→ $\frac{\partial}{\partial \underline{w}_k} \log a_i^n = \frac{\partial \log a_i^n}{\partial z_k^n} \times \frac{\partial z_k^n}{\partial \underline{w}_k}$, $z_k^n = \sum_{i=1}^{K} w_{ki} x_i^n$

$= \frac{\partial \log a_i^n}{\partial z_k^n} \times \underline{x}^n$ $= \underline{w}_k^T \underline{x}^n$

→ $\frac{\partial \log a_i^n}{\partial z_k^n} = \frac{\partial}{\partial z_k^n} \log \frac{\exp(\underline{w}_i^T \underline{x}^n)}{\sum_{j=1}^{K} \exp(\underline{w}_j^T \underline{x}^n)}$

$= \frac{\partial}{\partial z_k^n} \log \frac{\exp(z_i^n)}{\sum_{j=1}^{K} \exp(z_j^n)}$

$= \frac{\partial}{\partial z_k^n} \left(\log \exp(z_i^n) - \log \sum_{j=1}^{K} \exp(z_j^n) \right)$

$= 1\{i=k\} - \frac{\exp(z_k^n)}{\sum_{j=1}^{K} \exp(z_j^n)} = 1\{i=k\} - a_k^n$

$\therefore \frac{\partial J(w)}{\partial \underline{w}_k} = \frac{1}{N} \sum_{n=1}^{N} \sum_{i=1}^{K} -y_i^n \frac{\partial}{\partial \underline{w}_k} \log a_i^n$

$= \frac{1}{N} \sum_{n=1}^{N} \sum_{i=1}^{K} -y_i^n (1\{i=k\} - a_k^n) \underline{x}^n$

$= \frac{1}{N} \sum_{n=1}^{N} \left(-y_k^n + \sum_{i=1}^{K} y_i^n a_k^n \right) \underline{x}^n$

$= \frac{1}{N} \sum_{n=1}^{N} (a_k^n - y_k^n) \underline{x}^n$

$\sum_{i=1}^{K} y_i^n a_k^n = a_k^n$

[ML_09] Overfitting & Regularization

Overfitting & Underfitting

Underfitting (과소적합)

- 모델의 용량이 너무 적어 오차가 클 수밖에 없는 상황

Underfitting 방지하는법

- 비선형모델 등과 같이 용량이 더 큰모델을 사용한다.

Overfitting (과잉적합)

- 용량이 너무커서 학습과정에서 잡음까지 수용

→ 즉, 훈련집합에만 적합하게 되어 훈련집합에는 높은 성능을 보여주지만, 다른 새로운 데이터가 들어오게 되면 성능이 떨어짐.

Overfitting의 원인

1. 충분하지 않은 training dataset.
2. 너무 용량이 큰 Parameter

Overfitting 해결방

1. 충분히 많은 training data를 사용한다.

→ 만약 training data가 적다면 데이터 증대 기법을 통해 적은의 training data 증대 가능

2. 검증집합을 이용한 모델선택

- 훈련집합과 테스트집합과 다른 별도의 검증집합을 준비한다.
- 검증집합에 대해 최고의 성능을 보인 모델을 선택한다.

3. 규제 (Regularization)

- 용량이 충분히 큰모델 + 다양한 feature 고려 + 다양한 규제 기법을 적용

Regularization (규제)

$$J_{reg}(\theta) = \underbrace{J(\theta)}_{\text{목적함수}} + \lambda \underbrace{R(\theta)}_{\text{규제항}}$$

→ 규제항 $R(\theta)$로 무엇을 사용할 것인가? ⇒ 가중치 감쇠

L_2Norm 사용 : $R(\theta) = \|\theta\|_2^2$

L_1norm 사용 : $R(\theta) = \|\theta\|_1$

⇒ 최종해를 원점 가까이 당기는 효과

L_2Norm Regularization (Ridge)

$$J_{reg}(\underline{\theta}; \underline{x}, \underline{y}) = J(\underline{\theta}; \underline{x}, \underline{y}) + \lambda\|\theta\|_2^2$$

$$\nabla J_{reg}(\underline{\theta}; \underline{x}, \underline{y}) = \nabla J(\underline{\theta}; \underline{x}, \underline{y}) + 2\lambda\underline{\theta}$$

- Parameter Update

$$\underline{\theta} = \underline{\theta} - \rho\nabla J_{reg}(\underline{\theta}; \underline{x}, \underline{y})$$
$$= \underline{\theta} - \rho(\nabla J(\underline{\theta}; \underline{x}, \underline{y}) + 2\lambda\underline{\theta})$$
$$= (1-2\rho\lambda)\underline{\theta} - \rho\nabla J(\underline{\theta}; \underline{x}, \underline{y})$$

→ L_2규제는 θ를 $2\rho\lambda$의 비율로 줄인후 업데이트 하는 셈

L_1 norm Regularization (Lasso)

$$J_{reg}(\underline{\theta}; \underline{x}, \underline{y}) = J(\underline{\theta}; \underline{x}, \underline{y}) + \lambda\|\theta\|_1$$

$$\nabla J_{reg}(\underline{\theta}; \underline{x}, \underline{y}) = \nabla J(\underline{\theta}; \underline{x}, \underline{y}) + \lambda Sgn(\theta)$$

- Parameter Update

$$\underline{\theta} = \underline{\theta} - \rho\nabla J_{reg}(\underline{\theta}; \underline{x}, \underline{y})$$
$$= \underline{\theta} - \rho(\nabla J(\underline{\theta}; \underline{x}, \underline{y}) + \lambda Sgn(\theta))$$
$$= \underline{\theta} - \rho\nabla J(\underline{\theta}; \underline{x}, \underline{y}) - \underbrace{\rho\lambda Sgn(\theta)}$$

→ θ의 크기와 상관이 없음 즉 가중치와 상관없이 고정값으로 $\rho\lambda$만큼 penalty가 적용됨

⇒ L_1 규제의 희소효과 : 0이 되는 가중치가 많이 발생

→ 즉 학습에 별로 도움이 되지 않는 가중치의 경우 0이 됨.

→ 특징 선택효과가 있음