Lec3. loss functions

Example Problem

Regularization

Linear Classifier

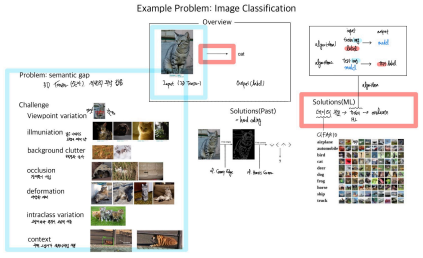
Classifier

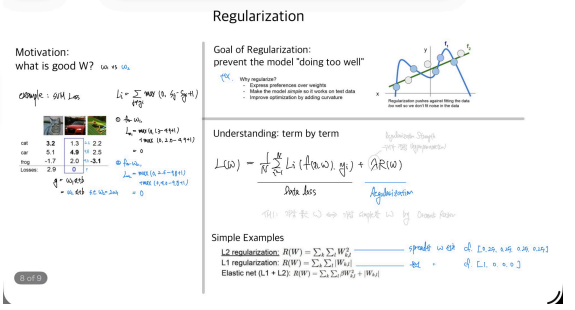
Regression

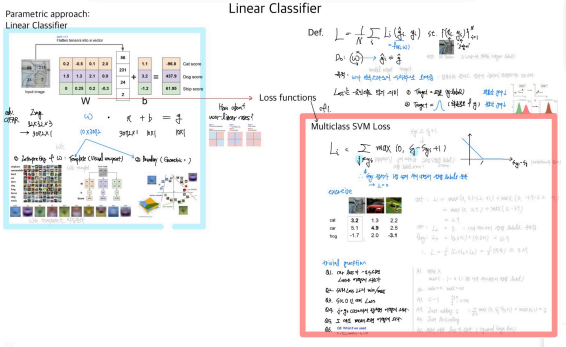
More on Loss functions

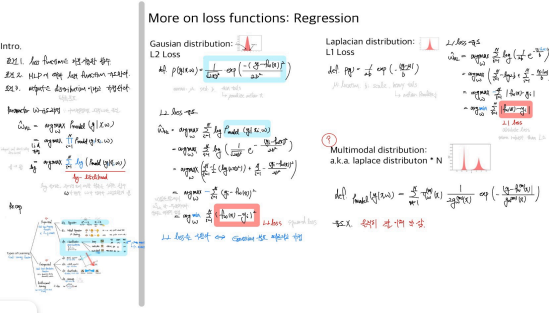
Softmax Classifier

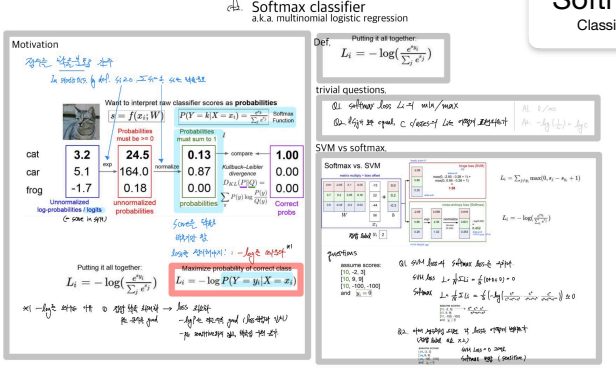
Classification

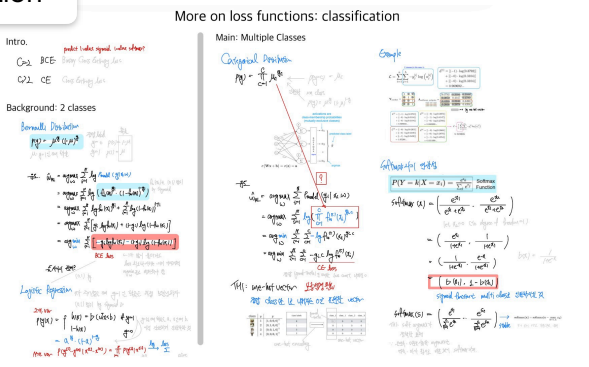






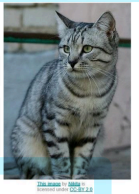






Example Problem: Image Classification

Overview



cat

Input (3D Tensor)

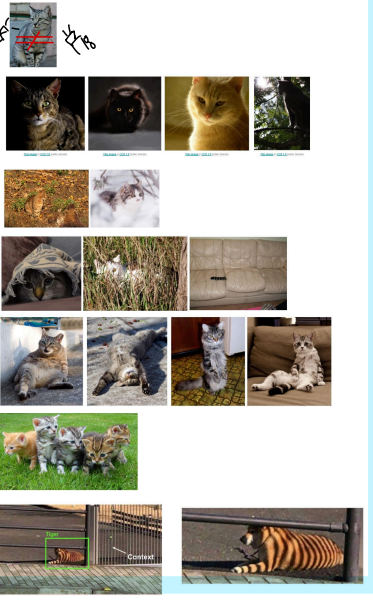
Output (label)

Problem: semantic gap

3D Tensor (숫자). 정보적 의미 없음

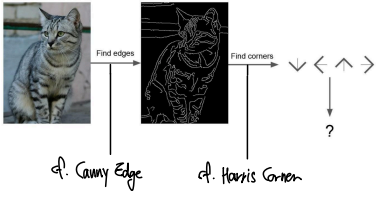
Challenge

- Viewpoint variation
- illmuniation — 같은 이미지도 조명에 따라 다
- background clutter — 배경과 유사
- occlusion — 장애물이 가림
- deformation — 다양한 자세
- intraclass variation — 고양이마다 특징이 조금씩 다름
- context — 주변 그림자가 초록다리처럼 작용



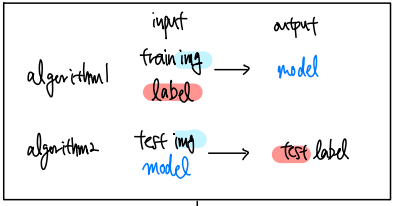
Solutions(Past)

= hard coding



cf. Canny Edge

cf. Harris Corner



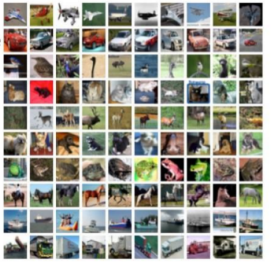
algorithm

Solutions(ML)

데이터 처리 → train → evaluate

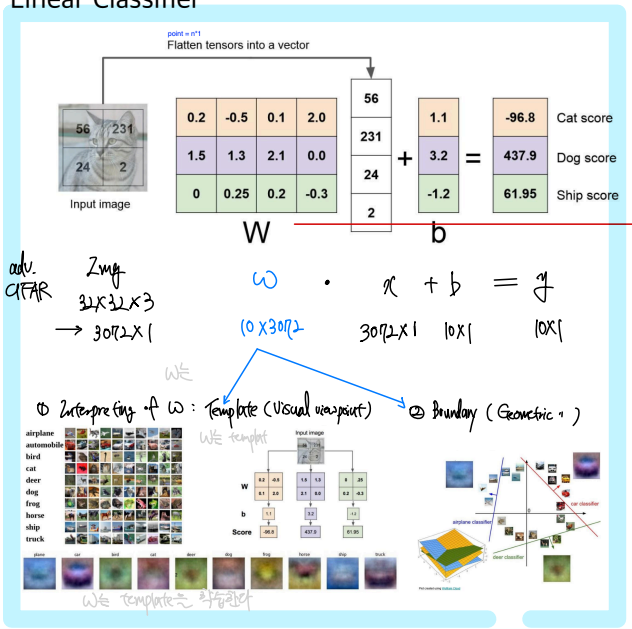
ML

CIFAR10



Linear Classifier

Parametric approach:
Linear Classifier



Loss functions

Def.

Multiclass SVM Loss

exercise

cat	3.2	1.3	2.2
car	5.1	4.9	2.5
frog	-1.7	2.0	-3.1

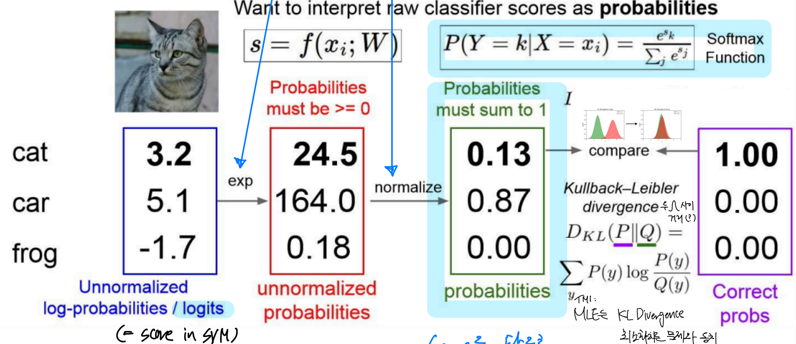
trivial question

Softmax classifier
a.k.a. multinomial logistic regression

Motivation

점수를 확률분포로 간주

In statistics, by def. $s_i \ge 0$, $\sum s_i = 1$, s_i는 확률분포



Score를 확률로 바꿔야 함.

loss를 정의해야지! : −log를 씌우다 *1

Putting it all together:

$$L_i = -\log\left(\frac{e^{s_{y_i}}}{\sum_j e^{s_j}}\right)$$

Maximize probability of correct class

$$L_i = -\log P(Y = y_i | X = x_i)$$

*1 −log를 씌우는 이유 ① 정답 확률 최대화 → loss 최소화
P는 클수록 good
−log P는 작을수록 good (loss 최소화와 일치)
−P는 sensitive하지 않고, 확률값 구하기 쉽다.

Def.

Putting it all together:

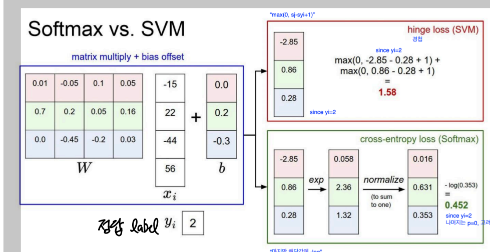
$$L_i = -\log\left(\frac{e^{s_{y_i}}}{\sum_j e^{s_j}}\right)$$

trivial questions.

Q1. softmax loss L_i의 min/max — A1. 0 / ∞

Q2. if s_j가 모두 equal, C classes의 L_i는 어떻게 표현되는가 — A2. $-\log(\frac{1}{C}) = \log C$

SVM vs softmax.



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

$$L_i = -\log\left(\frac{e^{s_{y_i}}}{\sum_j e^{s_j}}\right)$$

questions

assume scores:
[10, -2, 3]
[10, 9, 9]
[10, -100, -100]
and $y_i = 0$

Q1. SVM loss와 softmax loss를 구하라.

SVM loss $L = \frac{1}{N}\sum L_i = \frac{1}{3}(0+0+0) = 0$

Softmax $L = \frac{1}{N}\sum L_i = \frac{1}{3}\left(-\log\left(\frac{e^{10}}{e^{10}+e^{-2}+e^{3}} \cdot \frac{e^{10}}{e^{10}+e^{9}+e^{9}} \cdot \frac{e^{10}}{e^{10}+e^{-100}+e^{-100}}\right)\right) \simeq 0$

Q2. 아래 정답처럼 되면 각 loss는 어떻게 변하는가 (정답 label 값 ×2)

assume scores:
[20, -2, 3]
[20, 9, 9]
[20, -100, -100]
and $y_i = 0$

SVM Loss = 0 그대로

Softmax 변함 (sensitive) : 거의 항상 학습

More on loss functions: Regression

Intro.

전제 1. loss function은 미분 가능한 함수.

전제 2. MLP에 대해 loss function 정의한다.

전제 3. output은 distribution이라고 가정한다. (확률분포)

Parameter W 추정 과정 :: 어떤 변형해도 최적 W는 동일.

$$\hat{\omega}_{ML} = \arg\max_{\omega} P_{model}(y|X,\omega)$$

$$= \arg\max_{\omega} \prod_{i=1}^{N} P_{model}(y_i|x_i,\omega)$$

i.i.d 가정 (independent and identically distributed)

$$\overset{\log}{=} \arg\max_{\omega} \sum_{i=1}^{N} \log\left(P_{model}(y_i|x_i,\omega)\right)$$

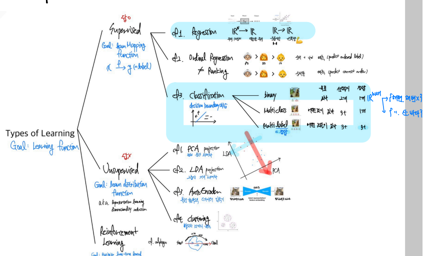
곱 → 합

log-likelihood

log 둥이면, 주어진 x에 대한 확률을 구하는 함수

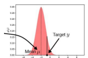
ω가 변수. ω가 얼마나 그럴듯한지 봄

Recap



Gausian distribution:

L2 Loss



def. $p(y|x,\omega) = \frac{1}{\sqrt{2\pi b^2}} \cdot \exp\left(\frac{-(y - f_\omega(x))^2}{2b^2}\right)$

mean: μ. std: b. thin tails ↳ penalize outlier ↑

L2 loss 유도

$$\hat{\omega}_{ML} = \arg\max_{\omega} \sum_{i=1}^{N} \log P_{model}(y_i|x_i,\omega)$$

$$= \arg\max_{\omega} \sum_{i=1}^{N} \log\left(\frac{1}{\sqrt{2\pi b^2}} e^{-\frac{(y_i - f_\omega(x))^2}{2b^2}}\right)$$

$$= \arg\max_{\omega} \left(\sum_{i=1}^{N} -\frac{1}{2}(\log(2\pi b^2)) + \sum_{i=1}^{N} -\frac{(y_i - f_\omega(x))^2}{2b^2}\right)$$

$$= \arg\max_{\omega} -\sum_{i=1}^{N} (y_i - f_\omega(x))^2$$

ω와 무관한 항 제거. $\hat{\omega}_{ML}$ 나 우리가 구하는 ω는 동일

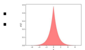
$$= \arg\min_{\omega} \sum_{i=1}^{N} (f_\omega(x) - y_i)^2$$

L2 loss squared loss

L2 loss를 구한다 ⟺ Gaussian 분포를 따른다고 가정

Laplacian distribution:

L1 Loss



def. $p(y) = \frac{1}{2b} \exp\left(-\frac{|y-\mu|}{b}\right)$

μ: location. b: scale. heavy tails ↳ outlier penalize↓

L1 loss 유도

$$\hat{\omega}_{ML} = \arg\max_{\omega} \sum_{i=1}^{N} \log\left(\frac{1}{2b} e^{-\frac{|y_i - f_\omega(x)|}{b}}\right)$$

$$= \arg\max_{\omega} \left(\sum_{i=1}^{N} -\log 2b + \sum_{i=1}^{N} -\frac{|y_i - f_\omega(x)|}{b}\right)$$

$$= \arg\max_{\omega} -\sum_{i=1}^{N} |f_\omega(x) - y_i|$$

$$= \arg\min_{\omega} \sum_{i=1}^{N} |f_\omega(x) - y_i|$$

L1 loss

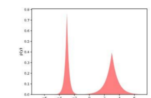
absolute loss

more robust than L2

?

Multimodal distribution:

a.k.a. laplace distributon * N



def. $P_{model}(y|x,\omega) = \sum_{m=1}^{N} \pi_\omega^{(m)}(x) \frac{1}{2g_\omega^{(m)}(x)} \exp\left(-\frac{|y - f_\omega^{(m)}(x)|}{g_\omega^{(m)}(x)}\right)$

유도X. 온전히 잘 이해 안 감.

More on loss functions: classification

Intro.

predict 1value sigmoid. 1value softmax?

$C=2$ BCE Binary Cross Entropy loss

$C>2$ CE Cross Entropy loss

Background: 2 classes

Bernoulli Distribution

$p(y) = \mu^{y}(1-\mu)^{1-y}$

μ: $y=1$인 때 확률

정답 label 확률
$y=0$ $p(0) = 1-\mu$
$y=1$ $p(1) = \mu$

유도.

$$\hat{w}_{ML} = \arg\max_w \sum_{i=1}^{N} \log P_{model}(y_i \mid x_i, w)$$

$$= \arg\max_w \sum_{i=1}^{N} \log \left(f_w(x_i)^{y_i} \cdot (1-f_w(x_i))^{1-y_i} \right)$$

$f_w(x_i)$는 (0,1) 범위 by sigmoid

$$= \arg\max_w \sum_{i=1}^{N} \log f_w(x_i)^{y_i} + \sum_{i=1}^{N} \log (1-f_w(x_i))^{1-y_i}$$

$$= \arg\max_w \sum_{i=1}^{N} \left[y_i \log f_w(x_i) + (1-y_i) \log (1-f_w(x_i)) \right]$$

$$= \arg\min_w \sum_{i=1}^{N} \left[-y_i \log f_w(x_i) - (1-y_i) \log (1-f_w(x_i)) \right]$$

BCE loss

(-1)가 앞에 붙더라도 loss 최소화시키는 w의 가치여서 argmin으로 바꿔줘야 함

두 사이 관계?

(0,1) by

Logistic Regression

x가 주어졌을 때 y=1 인 확률을 직접 모델링하자
(0,1) 범위 by sigmoid σ

2개. ver

$$p(y \mid x) = \begin{cases} h(x) = \sigma(w^T x + b) & \text{if } y=1 \\ 1-h(x) & y=0 \end{cases}$$

y=1일 때 확률은 a, 0인 때 b. 이걸 간단하게 일반화하게 한 것

$= a^{y} \cdot (1-a)^{1-y}$

N개. ver $P(y^{(1)} \ldots y^{(N)} \mid x^{(1)} \ldots x^{(N)}) = \prod_{i=1}^{N} P(y^{(i)} \mid x^{(i)})$ $\xrightarrow{\log}$ $\xrightarrow{\text{loss}}$ Σ

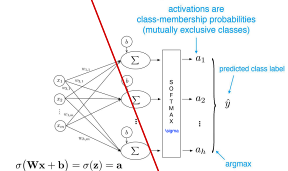
Main: Multiple Classes

Categorical Distribution

$$p(y) = \prod_{c=1}^{C} \mu_c^{y_c}$$

$p(y=c) = \mu_c$

일반화 2개 class $p(y) = \mu^{y}(1-\mu)^{1-y}$



?

유도

$$\hat{w}_{ML} = \arg\max_w \sum_{i=1}^{N} P_{model}(y_i \mid x_i, w)$$

$$= \arg\max_w \sum_{i=1}^{N} \log \left(\prod_{c=1}^{C} f_w^{(c)}(x_i)^{y_{i,c}} \right)$$

$$= \arg\min_w \sum_{i=1}^{N} \sum_{c=1}^{C} -\log f_w^{(c)}(x_i)^{y_{i,c}}$$

$$= \arg\min_w \sum_{i=1}^{N} \sum_{c=1}^{C} -y_{i,c} \log f_w^{(c)}(x_i)$$

CE loss

정답 (ground-truth)인 class only loss count, 나머지 0

TMI: one-hot vector 보충설명 필요

정답 class만 1로 나머지는 0으로 표현한 vector

class	y	$\mathbf{y}$
	1	$(1,0,0,0)^T$
	2	$(0,1,0,0)^T$
	3	$(0,0,1,0)^T$
	4	$(0,0,0,1)^T$

class labels
0
1
3
2

→ one-hot encoding

class_0	class_1	class_2	class_3
1	0	0	0
0	1	0	0
0	0	0	1
0	0	1	0

ground-truth; one-hot vector

Example

C classes (in this case, h)

$$\mathcal{L} = \sum_{i=1}^{n} \sum_{j=1}^{h} -y_j^{[i]} \log\left(a_j^{[i]}\right)$$

$$\mathcal{L}^{[1]} = [(-1)\cdot\log(0.3792)] + [(-0)\cdot\log(0.3104)] + [(-0)\cdot\log(0.3104)] = 0.969692\ldots$$

$$\mathbf{Y}_{\text{onehot}} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{bmatrix} \quad \mathbf{A}_{\text{softmax outputs}} = \begin{bmatrix} 0.3792 & 0.3104 & 0.3104 \\ 0.3072 & 0.4147 & 0.2780 \\ 0.4263 & 0.2248 & 0.3490 \\ 0.2668 & 0.2978 & 0.4354 \end{bmatrix}$$

(shaded) = 0 by one-hot vector

$$\mathcal{L}^{[1]} = [(-1)\cdot\log(0.3792)] + [(-0)\cdot\log(0.3104)] + [(-0)\cdot\log(0.3104)] = 0.969692\ldots$$

$$\mathcal{L}^{[2]} = [(-0)\cdot\log(0.3072)] + [(-1)\cdot\log(0.4147)] + [(-0)\cdot\log(0.2780)] = 0.880200\ldots$$

$$\mathcal{L}^{[3]} = [(-0)\cdot\log(0.4263)] + [(-0)\cdot\log(0.2248)] + [(-1)\cdot\log(0.3490)] = 1.05268\ldots$$

$$\mathcal{L}^{[4]} = [(-0)\cdot\log(0.2668)] + [(-0)\cdot\log(0.2978)] + [(-1)\cdot\log(0.4354)] = 0.831490\ldots$$

$$\rightarrow \mathcal{L} = \frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{h} -y_j^{[i]} \log\left(a_j^{[i]}\right) \approx 0.9335$$

Softmax 수식 연관성

$$P(Y=k \mid X=x_i) = \frac{e^{s_k}}{\sum_j e^{s_j}}$$ Softmax Function

$$\text{softmax}(x) = \left(\frac{e^{x_1}}{e^{x_1}+e^{x_2}}, \frac{e^{x_2}}{e^{x_1}+e^{x_2}} \right)$$

let $x_2 = 0$ (to degree of freedom = 1)

$$= \left(\frac{e^{x_1}}{1+e^{x_1}}, \frac{1}{1+e^{x_1}} \right)$$

$$= \left(\frac{1}{1+e^{-x_1}}, \frac{e^{-x_1}}{1+e^{-x_1}} \right)$$

$\sigma(x) = \frac{1}{1+e^{-x}}$

$$= \left(\sigma(x_1), 1-\sigma(x_1) \right)$$

sigmoid function을 multi class로 일반화시킨 것

$$\text{softmax}(s) = \left(\frac{e^{s_1}}{\sum_{k=1}^{C} e^{s_k}} \cdots \frac{e^{s_C}}{\sum_{k=1}^{C} e^{s_k}} \right) \rightarrow \text{stable}$$

$\text{softmax}(x) = \text{softmax}(x - \max_{k=1..L} x_k)$

x가 커도 가능, 수치적으로 안정

TMI: soft argmax가 정확한 표현

∵ 불연속·미분불가능한 argmax는 연속·미분 가능한 버전 것이 softmax니까.

Regularization

Motivation:
what is good W? w_1 vs w_2

example : SVM Loss

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

cat	**3.2**	1.3	2.2
car	5.1	**4.9**	2.5
frog	-1.7	2.0	**-3.1**
Losses:	2.9	0	

① for w_1,

$L_{w_1} = \max(0, 1.3 - 4.9 + 1) + \max(0, 2.0 - 4.9 + 1) = 0$

② for w_2,

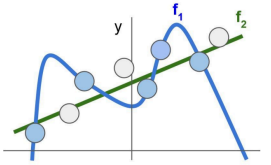
$L_{w_2} = \max(0, 2.6 - 9.8 + 1) + \max(0, 4.0 - 9.8 + 1) = 0$

$s = w_1 x + b$

$= w_2 x + b$ s.t. $w_2 = 2w_1$

Goal of Regularization:
prevent the model "doing too well"

tx. Why regularize?
- Express preferences over weights
- Make the model *simple* so it works on test data
- Improve optimization by adding curvature



Regularization pushes against fitting the data *too* well so we don't fit noise in the data

Understanding: term by term

Regularization Strength
정규화 강도 (Hyperparameter)

$$L(W) = \underbrace{\frac{1}{N}\sum_{i=1}^{N} L_i(f(x_i, W), y_i)}_{\text{Data loss}} + \underbrace{\lambda R(W)}_{\text{Regularization}}$$

cf: 강한 W ⇔ 강한 simple한 W by Occam's Razor

Simple Examples

L2 regularization: $R(W) = \sum_k \sum_l W_{k,l}^2$ — spread한 W 선호 cf. [0.25, 0.25, 0.25, 0.25]

L1 regularization: $R(W) = \sum_k \sum_l |W_{k,l}|$ — 동일 " cf. [1, 0, 0, 0]

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

부록