안 녕 하 세 요 !

CS231n 딥러닝 스터디

1 주 차 _ 1 차 시

| 201711703 김지우 |

INDEX

Computer Vision

Visual Data

+

Algorithm

History of Computer Vision

- Block World
- Stages of Visual Representation
- Generalized Cylinder
- Pictorial Structure

The object is composed of "simple geometric primitives"

History of Computer Vision

◎ Normalized Cut

◎ Face Detection

"Object Segmentation"

History of Computer Vision

- ◎ SIFT
- ◎ Spatial Pyramid Matching
- ◎ Histogram of Gradients
- ◎ Deformable Part Model

“Feature Based Object Recognition”

History of Computer Vision

◎ PASCAL Visual Object Challenge
◎ IMAGENET

"Real Object Recognition"

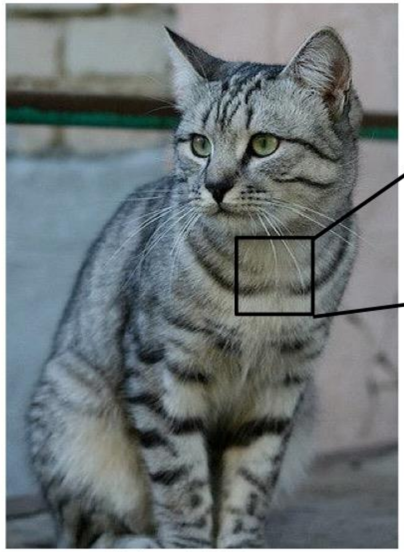
어떤걸 배울까요?

"Convolutional Neural Networks(CNN)"

"Visual Recognition",
그중에서도 "Image Classification"

Semantic Gap

The Problem: Semantic Gap




```
[[105 112 108 111 104  99 106  99  96 103 112 119 104  97  93  87]
 [ 91  98 102 106 104  79  98 103  99 105 123 136 110 105  94  85]
 [ 76  85  90 105 128 105  87  96  95  99 115 112 106 103  99  85]
 [ 99  81  81  93 120 131 127 100  95  98 102  99  96  93 101  94]
 [106  91  61  64  69  91  88  85 101 107 109  98  75  84  96  95]
 [114 108  85  55  55  69  64  54  64  87 112 129  98  74  84  91]
 [133 137 147 103  65  81  80  65  52  54  74  84 102  93  85  82]
 [128 137 144 140 109  95  86  70  62  65  63  63  60  73  86 101]
 [125 133 148 137 119 121 117  94  65  79  80  65  54  64  72  98]
 [127 125 131 147 133 127 126 131 111  96  89  75  61  64  72  84]
 [115 114 109 123 150 148 131 118 113 109 100  92  74  65  72  78]
 [ 89  93  90  97 108 147 131 118 113 114 113 109 106  95  77  80]
 [ 63  77  86  81  77  79 102 123 117 115 117 125 125 130 115  87]
 [ 62  65  82  89  78  71  80 101 124 126 119 101 107 114 131 119]
 [ 63  65  75  88  89  71  62  81 120 138 135 105  81  98 110 118]
 [ 87  65  71  87 106  95  69  45  76 130 126 107  92  94 105 112]
 [118  97  82  86 117 123 116  66  41  51  95  93  89  95 102 107]
 [164 146 112  80  82 120 124 104  76  48  45  66  88 101 102 109]
 [157 170 157 120  93  86 114 132 112  97  69  55  70  82  99  94]
 [130 128 134 161 139 100 109 118 121 134 114  87  65  53  69  86]
 [128 112  96 117 150 144 120 115 104 107 102  93  87  81  72  79]
 [123 107  96  86  83 112 153 149 122 109 104  75  80 107 112  99]
 [122 121 102  80  82  86  94 117 145 148 153 102  58  78  92 107]
 [122 164 148 103  71  56  78  83  93 103 119 139 102  61  69  84]]
```

What the computer sees

An image is just a big grid of numbers between [0, 255]:

e.g. 800 x 600 x 3
(3 channels RGB)

Challenges

◎ Viewpoint Variation
◎ Illumination
◎ Deformation
◎ Occlusion
◎ Background Clutter
◎ Intraclass Variation

Data-Driven Approach

1. Collect a dataset of images and labels
2. Use machine learning to train a classifier
3. Evaluate the classifier on new images

"Classifier"가 필요하다!

Nearest Neighbor

How to?

1. Memorize all data and labels
2. Predict the label of the most similar training image

가장 비슷한 이미지를 어떻게 찾을까?

"Distance Metric"을 이용하고,
"L1 distance"를 사용한다.

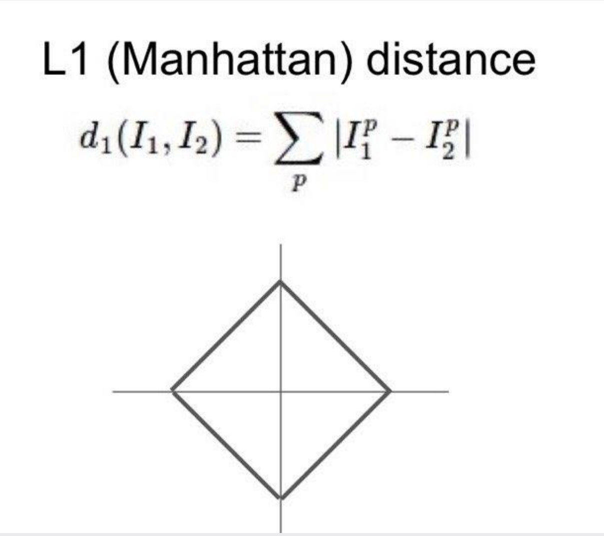
그러나, 이 방법은 Classifier를 Training하는데 오래 걸리지 않지만,
Prediction하는데 오래 걸리기 때문에 좋은 방법은 아니다.

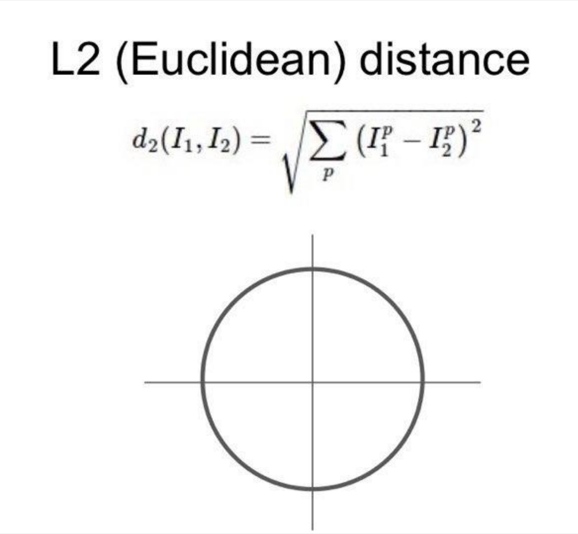
K-Nearest Neighbors

How to?
Instead of copying label from nearest neighbor, take majority vote from K closet points

장점?
K-Nearest Neighbors를 이용하면, decision boundary를 부드럽게 만들어주고 더 나은 결과로 이끌어준다.

Distance Metric

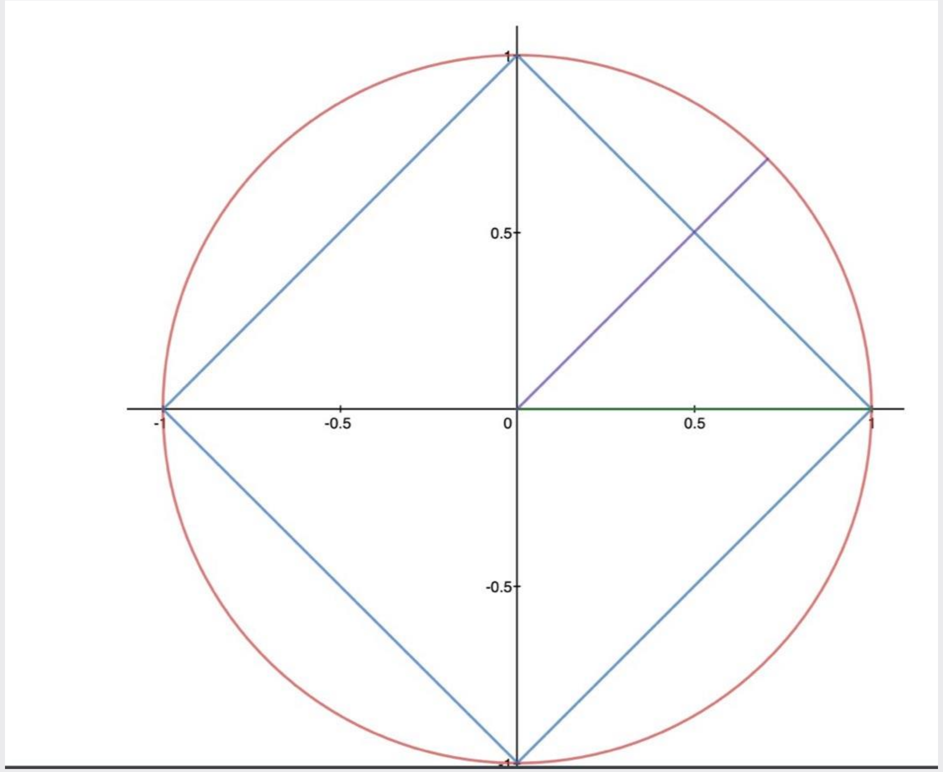




1. 좌표계 선택에 따라 달려있다.
2. 좌표축을 회전시키면, 점들 사이의 거리가 달라진다.
3. 벡터의 각각의 entry들이 중요한 의미를 가질 때 사용한다.

좌표축을 바꿔도 L2 distance는 달라지지 않는다.

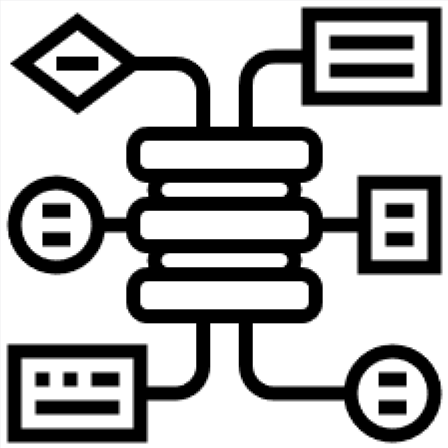
Distance Metric



Setting Hyperparameters

◎ Data를 **Train, Validation, Test** 데이터로 나눠
Training Data를 통해 알고리즘을 학습시키고,
Validation Data를 통해 Hyperparameter를 선택하고,
Test Data를 통해 알고리즘이 새로운 데이터에서 얼마나
잘 작동하는 지 확인한다.

◎ **Cross-Validation**은 Training Data를 n개의 부분으로 나누어 n-1개를 Training Data로 활용하고, 나머지 1개를 Validation Data로 활용하여 n번의 Validation을 할 수 있는 방법을 말한다.
그러나 주로 데이터셋의 크기가 작을 때 유용하기 때문에,
딥러닝에는 잘 쓰이지 않는다.



Never Used

1. Very slow at test time
2. Distance metrics on pixels are not informative
3. Curse of dimensionality

“K-Nearest Neighbor on images never used”

Parametric Approach

Data-Driven Approach

1. Collect a dataset of images and labels
2. Use machine learning to train a classifier
3. Evaluate the classifier on new images

Parametric Approach

1. Summarize our knowledge of the training data and stick all that knowledge into these parameters
2. At test time, we only need these parameters(=weights), W

“테스트 할 때, 더 이상 실제 트레이닝 데이터가 필요하지 않다.”

Linear Classifier

$$f(x, W) = Wx + b$$

X: input data(=training data)
W: parameters or weights
b: bias term(=a constant vector of elements that does not interact with training data)

“Giving class scores”

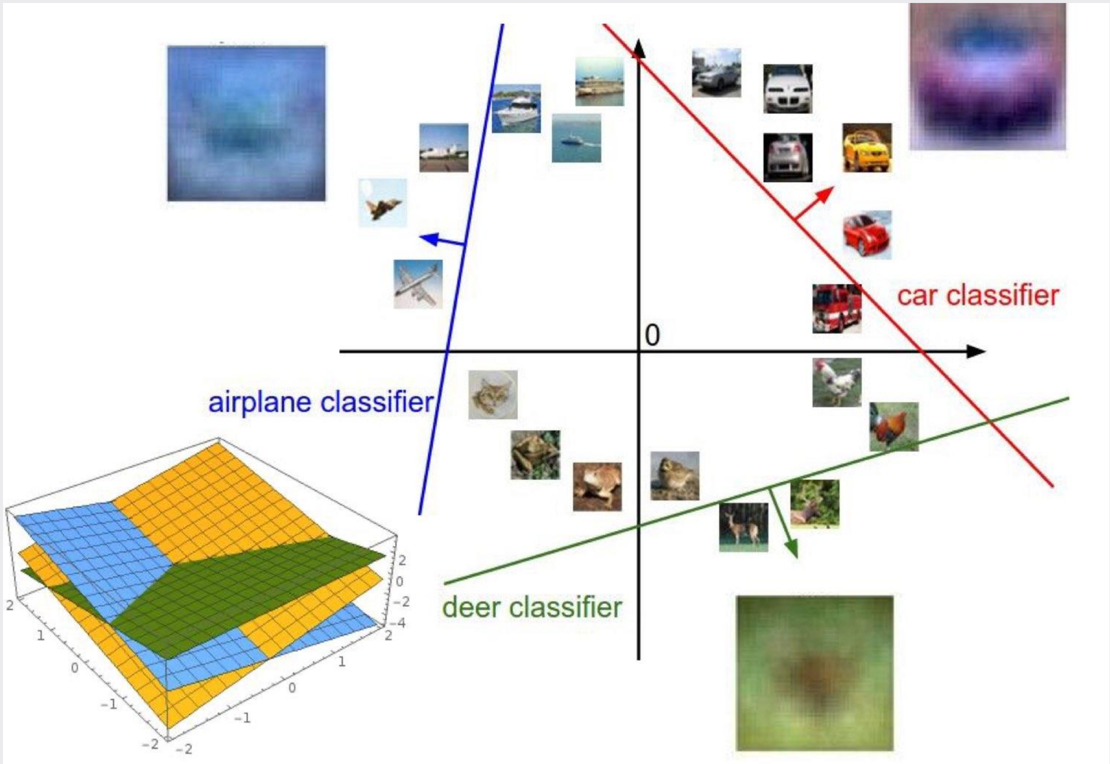
“Data independent preferences for some classes over another”

Linear Classifier

◎ W matrix의 각 행은 각 class의 template에 상응한다.

◎ W와 x를 내적한 것은, class(종류,분류)의 template과 image간의 유사성을 나타낸다.

Linear Classifier



“Draw linear separation between one category and the rest of categories”

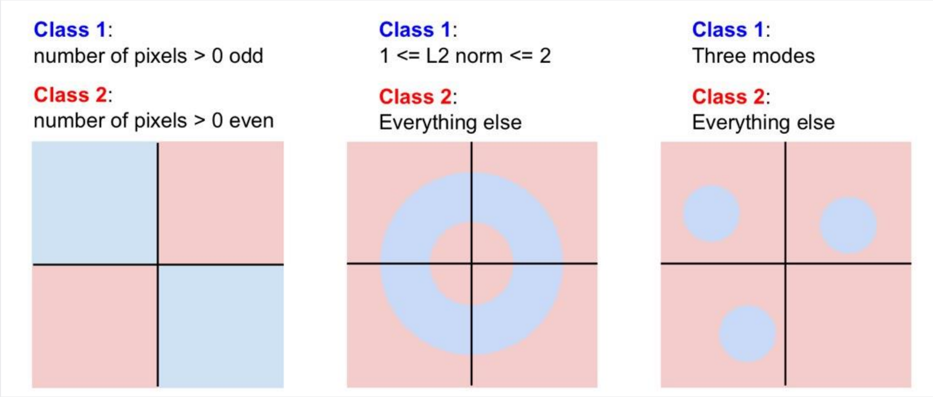
“linear decision boundaries”

Linear Classifier

Problem

The linear classifier is only learning one template for each class.
So if there's sort of variations in how class might appear, it's trying to average out all of those different variations.

Hard cases for a linear classifier



"Parity problem"
"Multimodal data"

Loss function

“We need to quantify the badness of any particular W.”

“Loss Function”

Loss function

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

"Loss over the dataset is a average of these losses summed over that entire data set."

- Hinge Loss (SVM Classifier)
- Cross-Entropy Loss (Softmax Classifier)

Hinge Loss

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

" If the score for the correct category is greater than the score of the incorrect category by some safe margin(=1), if that's the case that means the score for the true category is much larger than any of false categories, we'll get a loss of zero."

Hinge Loss

$$f(x, W) = Wx$$

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

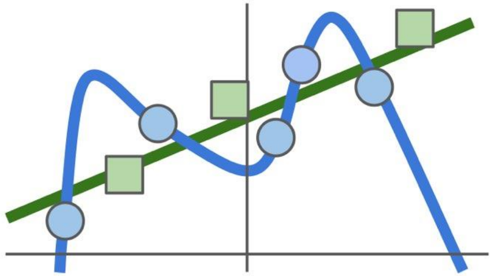
“L=0으로 만드는 W는 unique하지 않다 "

Regularization

$$L(W) = \frac{1}{N} \sum_{i=1}^{N} L_i(f(x_i, W), y_i) + \lambda R(W)$$

Data loss: Model predictions should match training data

Regularization: Model should be "simple", so it works on test data



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

Softmax Classifier

scores = unnormalized log probabilities of the classes.

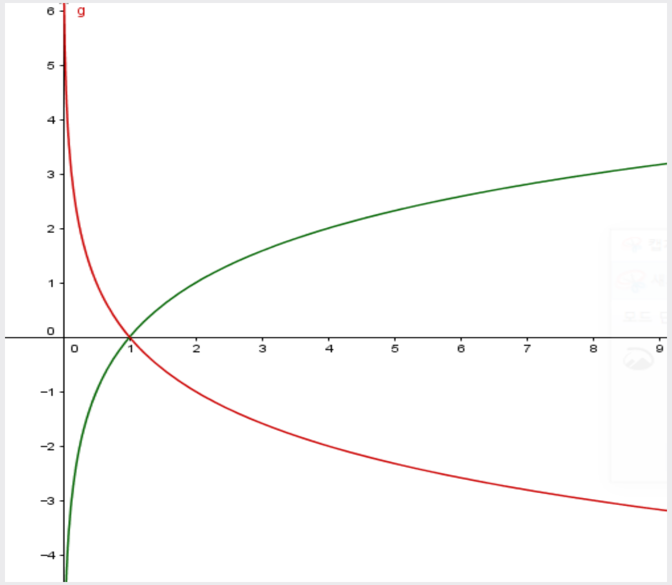
$$\boxed{P(Y=k|X=x_i)=\frac{e^{s_k}}{\sum_j e^{s_j}}} \quad \text{where} \quad \boxed{s=f(x_i;W)}$$

Want to maximize the log likelihood, or (for a loss function) to minimize the negative log likelihood of the correct class:

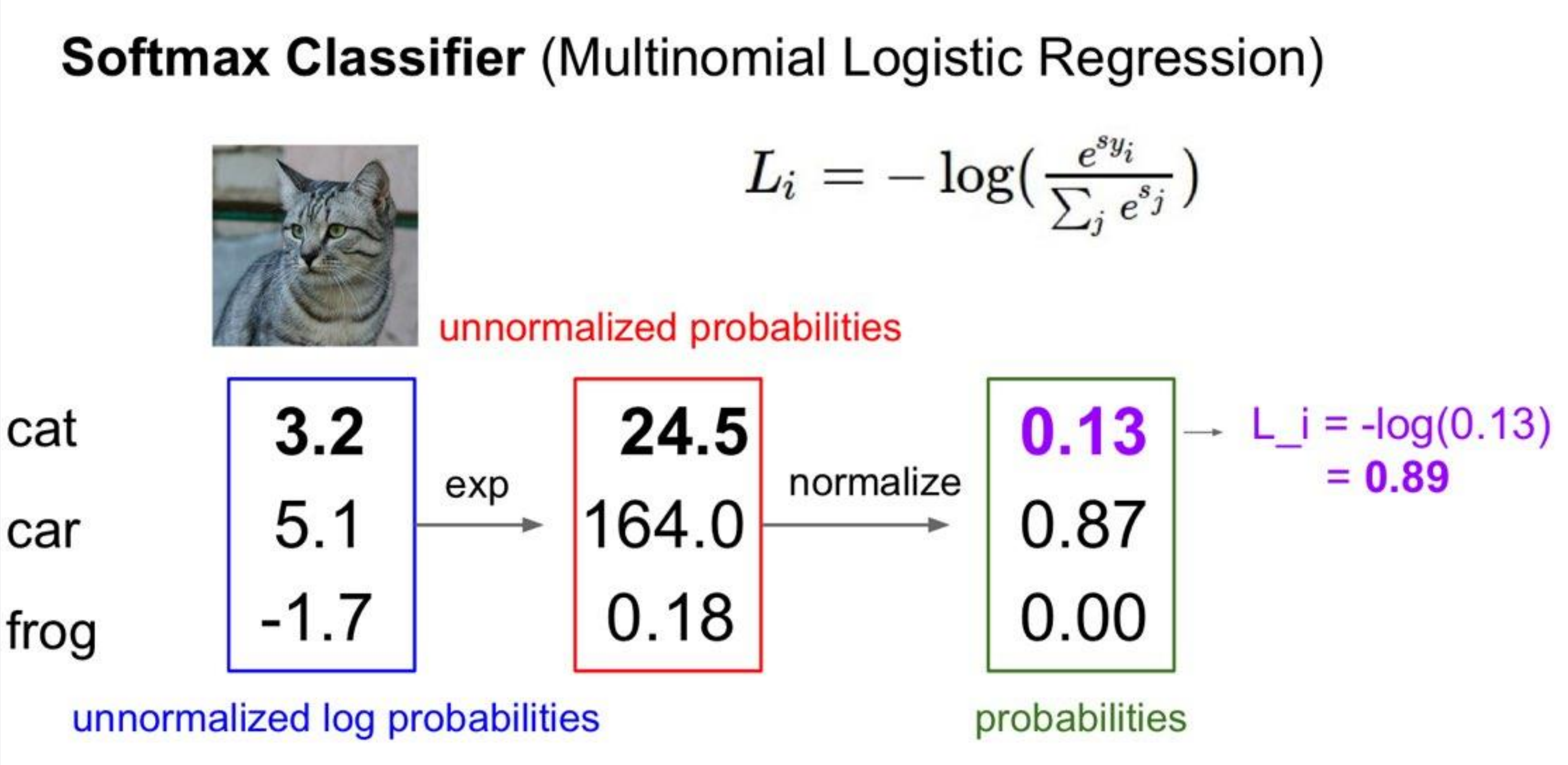
$$\boxed{L_i=-\log P(Y=y_i|X=x_i)}$$

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

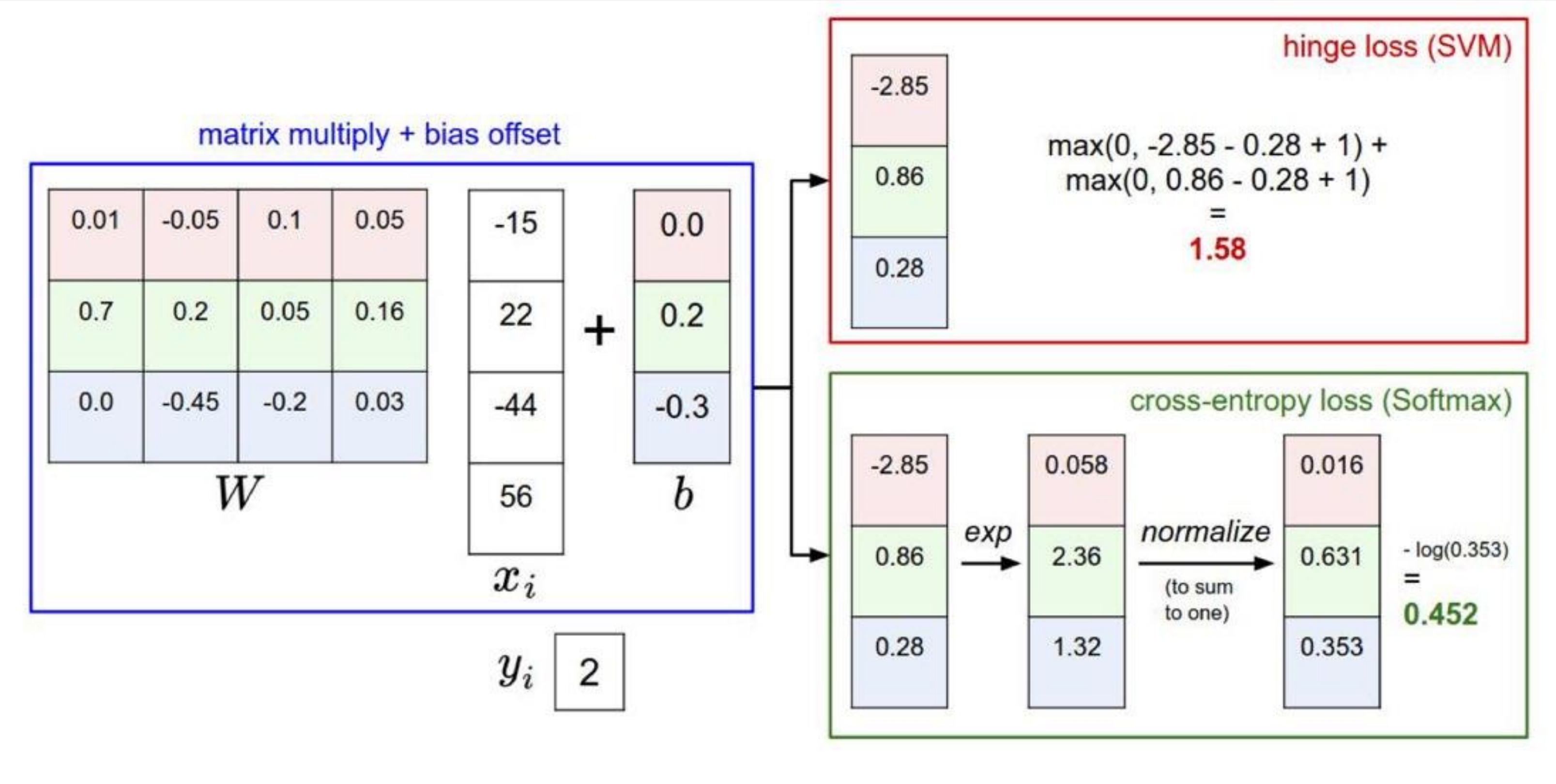
"Encourage our computed probability distribution that's coming out of this softmax function to match this target probability distribution"



Softmax Classifier



Difference between the two loss functions



"The difference between the two loss functions is how we choose to interpret those scores to quantitively measure the badness afterwards."

SVM: 바르게 분류되고 나면 더 이상 신경쓰지 않는다.
Softmax: 맞게 분류되더라도 맞는(올바른) class에 확률 질량이 더 많이 모여지는 방식으로 작동한다.

감 사 합 니 다 !

발표 들어주셔서 감사합니다