Loss Functions

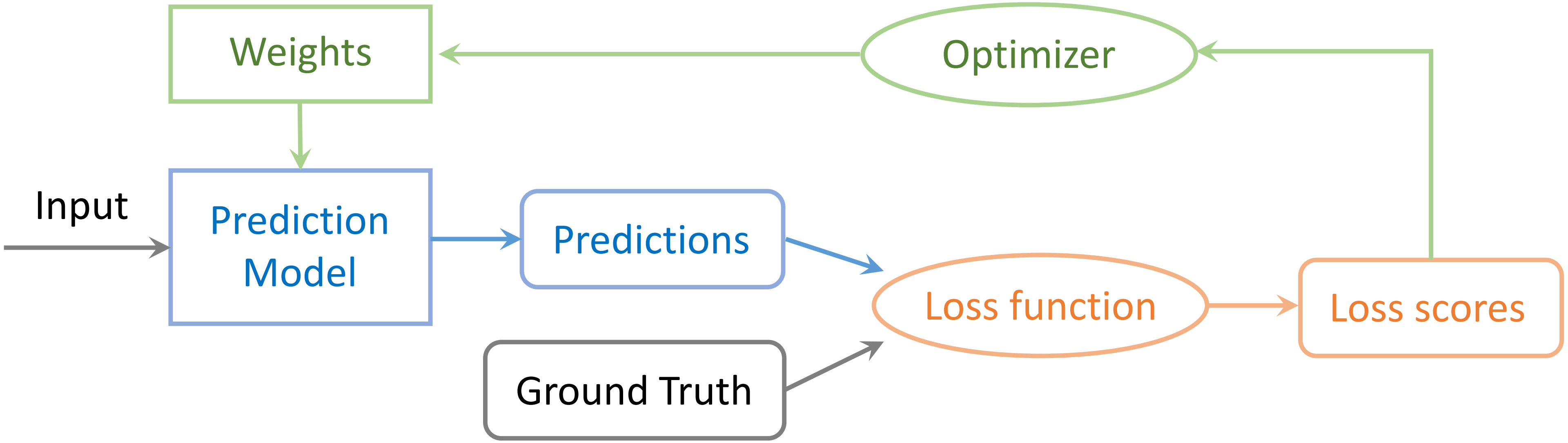
Chia-Po Wei

Department of Electrical Engineering
National Sun Yat-sen University

Outline

- Training Pipeline
- Linear Model
- Loss Functions
    - Multiclass SVM Loss
    - softmax Cross-Entropy Loss
    - Mean Squared Error

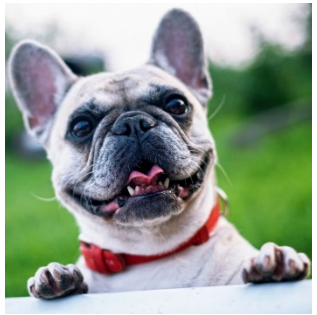
Training Pipeline



- The training pipeline consists of choosing the prediction model, the loss function, and the optimizer.
- Once these choices are made, we can feed the input data and labels to start the training process.

Linear Models

input



$$\underbrace{\begin{bmatrix}0.1 & 1.2 & 2 & 0.8\\ -0.5 & 0.6 & 1.5 & -0.2\\ -0.2 & 0.4 & 0.8 & 0.3\end{bmatrix}}_{\mathbf{W} \in \mathbb{R}^{K \times d}} \underbrace{\begin{bmatrix}24\\ 15\\ 331\\ -10\end{bmatrix}}_{\mathbf{x} \in \mathbb{R}^{d}} + \underbrace{\begin{bmatrix}0.7\\ -1.3\\ 3.6\end{bmatrix}}_{\mathbf{b} \in \mathbb{R}^{K}} \longrightarrow \begin{array}{c} \text{score} \\ \begin{bmatrix}675.1\\ 494.2\\ 266.6\end{bmatrix} \end{array} \begin{array}{l} \\ \text{dog}\\ \text{car}\\ \text{bird}\end{array}$$

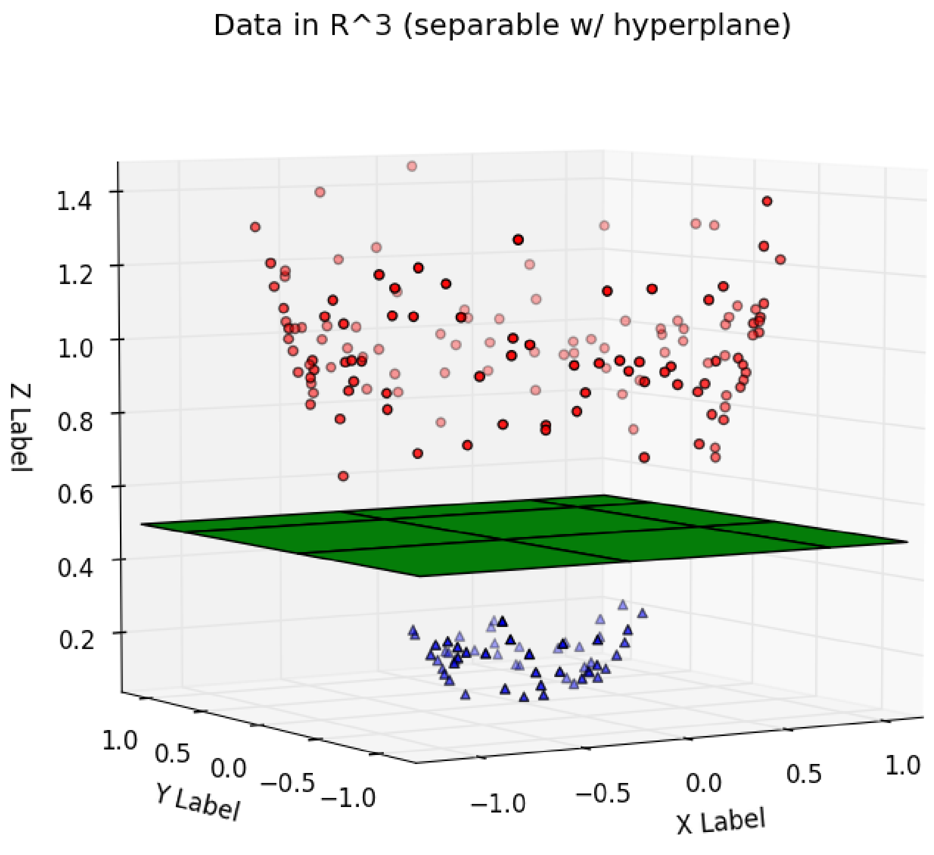
$$f(\mathbf{x}) = \mathbf{W}\mathbf{x} + \mathbf{b} \in \mathbb{R}^{K}$$

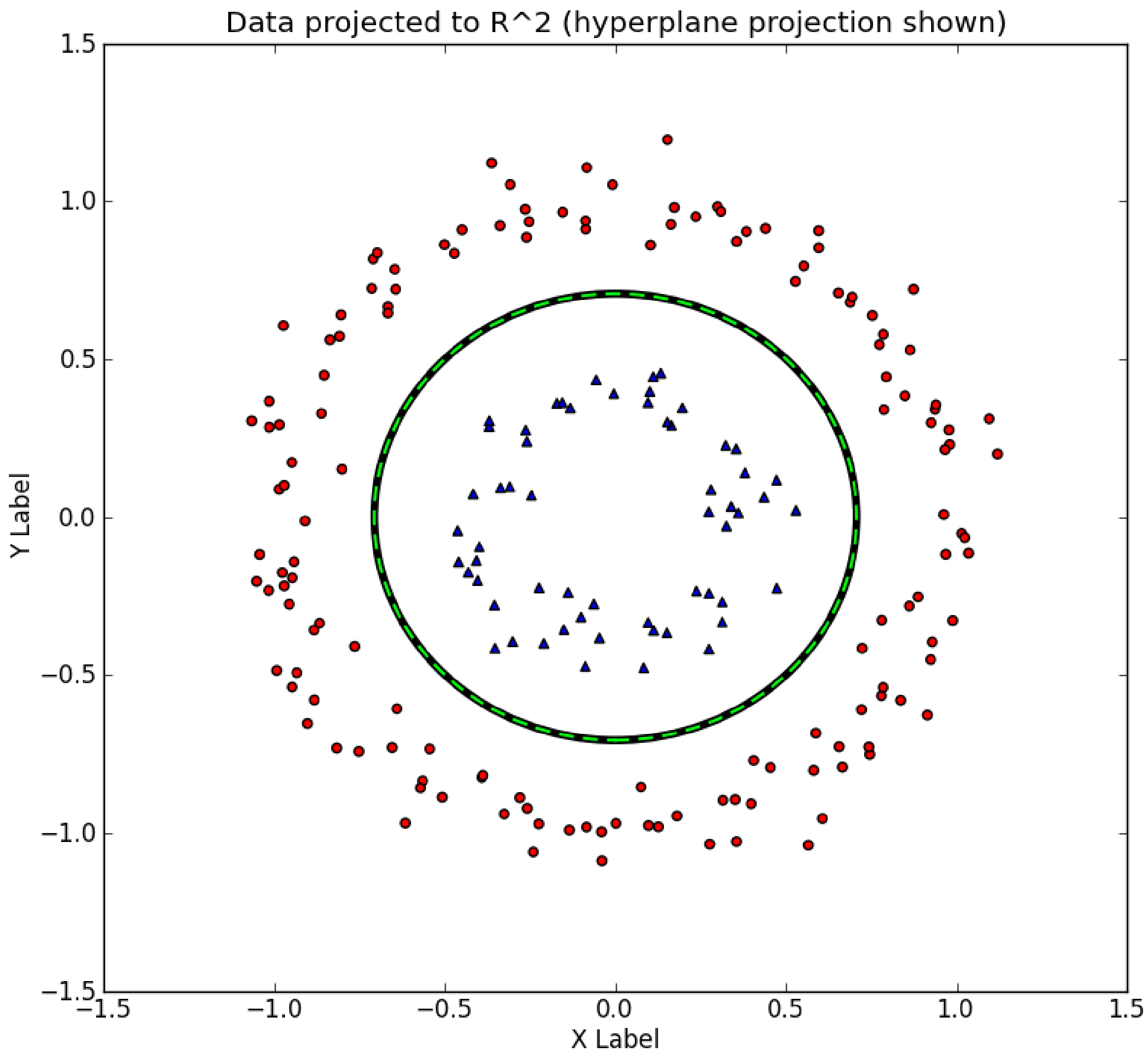
Linear model

Bias Trick

$$
\underset{\mathbf{W}}{\begin{bmatrix} 0.1 & 1.2 & 2 & 0.8 \\ -0.5 & 0.6 & 1.5 & -0.2 \\ -0.2 & 0.4 & 0.8 & 0.3 \end{bmatrix}}
\underset{\mathbf{x}}{\begin{bmatrix} 24 \\ 15 \\ 331 \\ -10 \end{bmatrix}}
+
\underset{\mathbf{b}}{\begin{bmatrix} 0.7 \\ -1.3 \\ 3.6 \end{bmatrix}}
\Leftrightarrow
\underset{[\mathbf{W}\ \ \mathbf{b}]}{\begin{bmatrix} 0.1 & 1.2 & 2 & 0.8 & 0.7 \\ -0.5 & 0.6 & 1.5 & -0.2 & -1.3 \\ -0.2 & 0.4 & 0.8 & 0.3 & 3.6 \end{bmatrix}}
\underset{[\mathbf{x};1]}{\begin{bmatrix} 24 \\ 15 \\ 331 \\ -10 \\ \color{red}{1} \end{bmatrix}}
$$

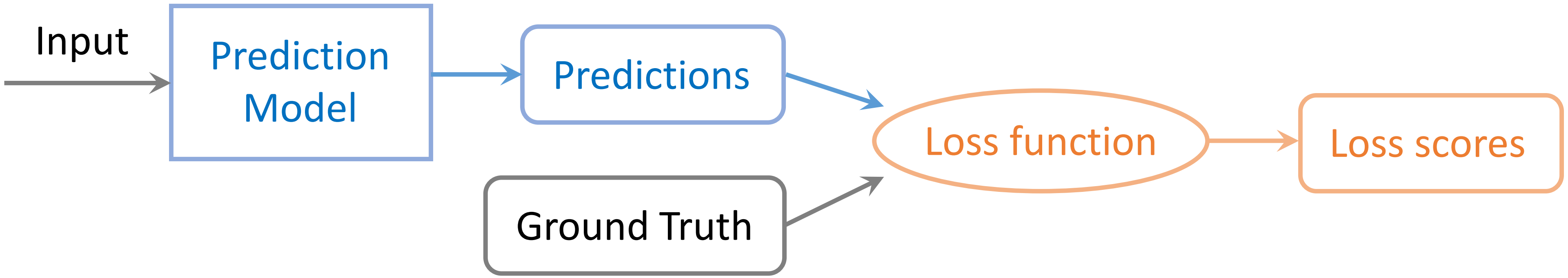
Limitation of Linear Models





Loss Function

- A loss function measures the quality of the scores of the prediction model.
- The prediction model can be a **linear model** or **neural networks**.



Multiclass SVM

- Given image $\mathbf{x}_i$ and its label y_i. Suppose there are K distinct classes.
- Let $\mathbf{s} \in \mathbb{R}^K$ be the predicted score for image $\mathbf{x}_i$, and s_j be the jth element of $\mathbf{s}$.
- The multiclass SVM loss function for the ith training image is defined as

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

- For example, let $\mathbf{s} = [15, -3, 7] = [s_0, s_1, s_2]$
- If the label $y_i = 0$, then $L_i = \max(0, s_1 - s_0 + 1) + \max(0, s_2 - s_0 + 1) = 0 + 0 = 0$
- If the label $y_i = 1$, then $L_i = \max(0, s_0 - s_1 + 1) + \max(0, s_2 - s_1 + 1) = 19 + 11 = 30$

Multiclass SVM (cont.)

	Scores ($\mathbf{s}$)	Label (y_i)	
Dog	15	1	don't care
Car	-3	0	max(0, -3 -15 +1) = 0
Bird	7	0	max(0, 7 -15 +1) = 0

→ correct prediction

Loss = 0
(Loss is small)

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

Multiclass SVM

	Scores ($\mathbf{s}$)	Label (y_i)	
Dog	15	0	max(0, 15 -(-3) +1) = 19
Car	-3	1	don't care
Bird	7	0	max(0, 7 -(-3) +1) = 11

incorrect prediction

Loss = 30
(Loss is large)

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

The Margin of Multiclass SVM

- The multiclass SVM loss function is defined as (Δ is the margin)

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

- For example, let $\mathbf{s} = [15, -3, 7] = [\mathrm{s}_0, \mathrm{s}_1, \mathrm{s}_2]$ and $\Delta = 20$
- If the label $y_i = 0$, then

$$L_i = \max(0, s_1 - s_0 + 20) + \max(0, s_2 - s_0 + 20) = 2 + 12 = 14$$

- If the label $y_i = 1$, then

$$L_i = \max(0, s_0 - s_1 + 20) + \max(0, s_2 - s_1 + 20) = 38 + 10 = 48$$

The Margin of Multiclass SVM (cont.)

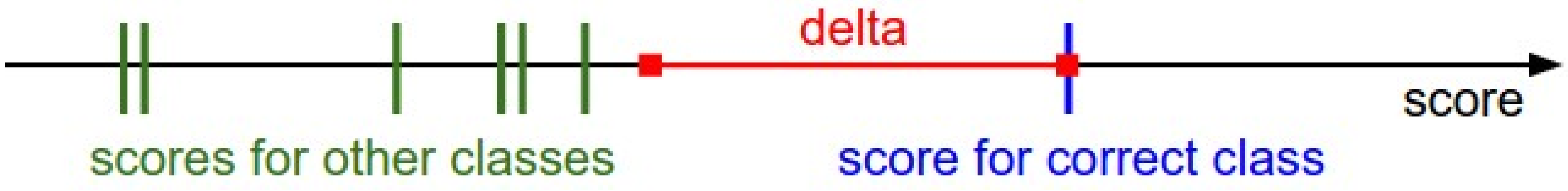
	Scores ($\mathbf{s}$)	Label (y_i)	→ correct prediction → Loss = ?
Dog	15	1	→ don't care
Car	-3	0	→ max(0, -3 -15 + Δ) = ?
Bird	7	0	→ max(0, 7 -15 + Δ) = ?

Loss = ?

$L_i(\mathbf{s}, y_i) = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + \Delta)$, where Δ is called the margin

Rationale for Multiclass SVM

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



- The score of the correct class must be higher than other scores by at least a margin of Δ.
- Otherwise, there will be accumulated loss.

Relation to Binary SVM

- The multiclass SVM can be written as

$$L_i(\mathbf{s}, y_i) = \sum_{j \neq y_i} \max\left(0, s_j - s_{y_i} + 1\right)$$
$$= \sum_{j \neq y_i} \max\left(0, w_j^T \mathbf{x}_i - w_{y_i}^T \mathbf{x}_i + 1\right)$$

$$f(\mathbf{x}_i) = \mathbf{W}\mathbf{x}_i = \begin{bmatrix} w_1^T \\ \vdots \\ w_K^T \end{bmatrix} \mathbf{x}_i = \begin{bmatrix} w_1^T \mathbf{x}_i \\ \vdots \\ w_K^T \mathbf{x}_i \end{bmatrix} = \begin{bmatrix} s_1 \\ \vdots \\ s_K \end{bmatrix}$$

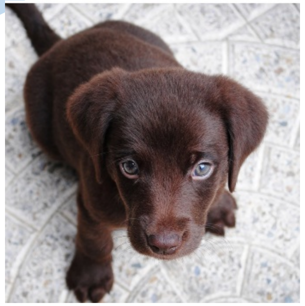
- Recall that the binary SVM (soft-margin) is to minimize

$$\left[\frac{1}{n} \sum_{i=1}^{n} L_i\right] + \lambda \|\mathbf{w}\|_2^2,$$

where $L_i = \max(0, 1 - y_i(\mathbf{w}^T \mathbf{x}_i - b))$ with $y_i = \pm 1$

- Can multiclass SVM reduce to binary SVM when K=2?

Example for Multiclass SVM

Dog	**3.0**	0.9	1.6
Bird	6.2	**5.1**	2.3
Truck	-2.1	3.4	**-4.5**
Loss	4.2	0	14.9

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

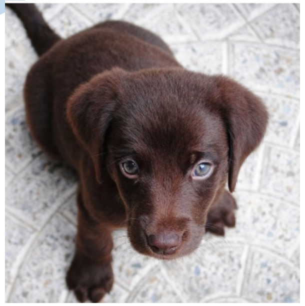
For the dog image

$L_1(\mathbf{s}, y_1) =$ max(0, 6.2-3.0+1) + max(0, -2.1-3.0+1) = 4.2 + 0 = 4.2

For the bird image

$L_2(\mathbf{s}, y_2) =$ max(0, 0.9-5.1+1) + max(0, 3.4-5.1+1) = 0 + 0 = 0

Example for Multiclass SVM (cont.)

Dog	**3.0**	0.9	1.6
Bird	6.2	**5.1**	2.3
Truck	-2.1	3.4	**-4.5**
Loss	4.2	0	14.9

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

The loss over this batch (three images):

$$L = \frac{1}{N} \sum_{i=1}^{N} L_i$$

That is, $L = (L_1 + L_2 + L_3)/3 = (4.2 + 0 + 14.9)/3$

Quiz 1

$$L_i(\mathbf{s}, y_i) = \sum\nolimits_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$

1. What is the min/max possible value of L_i?
2. Suppose the initial value of $\mathbf{s}$ is $[0,0, \dots, 0] \in \mathbb{R}^K$. What is the loss?
3. What if the sum is over all classes (including $j = y_i$)?

Regularization

L2 regularization: $R(\mathbf{W}) = \sum_k \sum_l \mathbf{W}_{k,l}^2$
L1 regularization: $R(\mathbf{W}) = \sum_k \sum_l |\mathbf{W}_{k,l}|$

- $L = \underbrace{\frac{1}{N}\sum_i L_i(f(\mathbf{x}_i; \mathbf{W}), y_i)}_{\text{Data loss}} + \underbrace{\lambda R(\mathbf{W})}_{\text{Regularization loss}}$

- The data loss controls that model predictions should match the training data.
- The regularization prevents that the model f overfits the training data.
- Note that the regularization loss is not a function of the training data.
- The hyperparameter λ controls the strength of regularization.

Regularization (cont.)

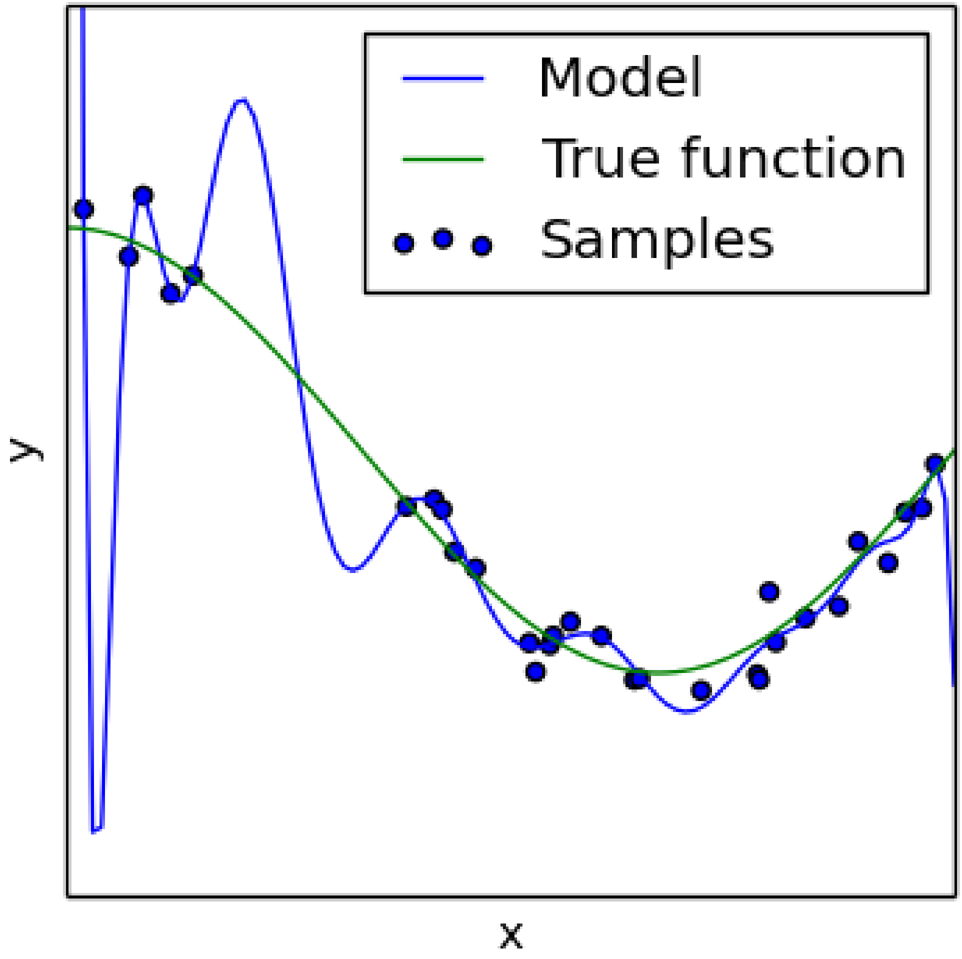
- $L = \frac{1}{N}\sum_i L_i(f(\mathbf{x}_i; \mathbf{W}), y_i) + \lambda R(\mathbf{W})$
- Penalizing large weights tends to improve generalization
- Because no input dimension can have a very large influence on the scores
- For example,

$$\mathbf{x} = [1,1,1,1], \quad \mathbf{w}_1 = [1,0,0,0], \quad \mathbf{w}_2 = [0.25, 0.25, 0.25, 0.25]$$

$$\mathbf{w}_1^T\mathbf{x} = \mathbf{w}_2^T\mathbf{x} = 1, \quad R(\mathbf{w}_1) = 1, \quad R(\mathbf{w}_2) = 0.25$$ (Using L2 penalty.)

- The weight vector $\mathbf{w}_2$ would be preferred because of a lower loss.
- That is, L2 penalty prefers **smaller and more diffuse** weight vectors.

Regularization Prevents Overfitting



- The overfitting occurs when a model begins to memorize training data rather than learning to generalize from a trend.
- In practice, the training data are noisy. Overfitting the noisy data leads to poor generalization.

softmax Classifier

- Given image $\mathbf{x}_i$ and its label y_i. Suppose there are K distinct classes.
- Let $\mathbf{s} \in \mathbb{R}^K$ be the predicted score for image $\mathbf{x}_i$, and s_j be the jth element of $\mathbf{s}$.
- The cross-entropy loss for the ith training image is defined as
- $L_i(\mathbf{s}, y_i) = -\log\left(\frac{e^{s_{y_i}}}{\sum_j e^{s_j}}\right) = -\log\big(h(\mathbf{s}, y_i)\big)$

  where $h(\mathbf{s}, y_i) = \frac{e^{s_{y_i}}}{\sum_j e^{s_j}}$ is called the softmax function
- The softmax classifier is also known as **multinomial logistic regression**.

softmax Classifier (cont.)

	Scores ($\mathbf{s}$)	$\xrightarrow{\text{exp}}$	$\xrightarrow{\text{normalize}}$ Prob.	Label (y_i)
Dog	4.3	73.7	0.11	1
Car	-2.1	0.12	$2e^{-4}$	0
Bird	6.4	601.8	0.89	0

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

-log(0.11) = 0.96

cross-entropy loss

(wrong prediction)

Loss is large

softmax Classifier (cont.)

	Scores ($\mathbf{s}$)	exp		normalize	Prob.	Label (y_i)
Dog	4.3		73.7		0.11	0
Car	-2.1	$\rightarrow$	0.12	$\rightarrow$	$2e^{-4}$	0
Bird	6.4		601.8		0.89	1

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

-log(0.89) = 0.05

cross-entropy loss

(correct prediction)

Loss is small

Quiz 2:

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

1. What is the min/max possible value of L_i?
2. Suppose the initial value of $\mathbf{s}$ is $[1,1,\ldots,1] \in \mathbb{R}^K$. What is the cross-entropy loss?

Why Using softmax for Normalization?

- softmax function: $h(\mathbf{s}, y_i) = \frac{e^{s_{y_i}}}{\sum_j e^{s_j}}$

- softmax([1, 2]) = [0.27, 0.73]
- softmax([10, 20]) = [0, 1]

- std_normalization([1, 2]) = [1/3, 2/3]
- std_normalization([10, 20]) = [1/3, 2/3]

- Standard normalization outputs the same vector as long as the proportions are the same.
- Softmax reacts to low stimulation (think blurry image) with rather uniform distribution.
- Softmax reacts to high stimulation (large numbers, think crisp image) with probabilities close to 0 and 1.

Implementations of the softmax function

- $h(\mathbf{s}, y_i) = \frac{e^{s_{y_i}}}{\sum_j e^{s_j}}$

```
def softmax1(s, y):
    p = np.exp(s[y]) / np.sum(np.exp(s))
    return p
```

```
def softmax2(s, y):
    s -= np.max(s)
    p = np.exp(s[y]) / np.sum(np.exp(s))
    return p
```

- The above are two implementations for softmax.
- Which version would you use?

Proof for the Correctness of Version 2

- Suppose the scores are $[s_1, s_2, s_3]$.
- Assume s_1 is the score for the correct class.
- The output of the softmax function is

$$\frac{e^{s_1}}{e^{s_1} + e^{s_2} + e^{s_3}}$$

- Suppose s_2 is the **largest score**.
- Subtract s_2 from the scores, we obtain $[s_1 - s_2,\ s_2 - s_2,\ s_3 - s_2]$
- then the output of the softmax function becomes

$$\frac{e^{s_1 - s_2}}{e^{s_1 - s_2} + e^{s_2 - s_2} + e^{s_3 - s_2}} = \frac{e^{s_1}/e^{s_2}}{e^{s_1}/e^{s_2} + e^{s_2}/e^{s_2} + e^{s_3}/e^{s_2}} = \frac{e^{s_1}}{e^{s_1} + e^{s_2} + e^{s_3}}$$

Relation to Logistic Regression

- softmax function

$$\Pr(Y_i = k) = \frac{e^{\boldsymbol\beta_k \cdot \mathbf{X}_i}}{\sum_{0 \le c \le K} e^{\boldsymbol\beta_c \cdot \mathbf{X}_i}}$$

- When K = 2 (two-class logitstic regression)

$$\Pr(Y_i = 0) = \frac{e^{\boldsymbol\beta_0 \cdot \mathbf{X}_i}}{\sum_{0 \le c \le K} e^{\boldsymbol\beta_c \cdot \mathbf{X}_i}} = \frac{e^{\boldsymbol\beta_0 \cdot \mathbf{X}_i}}{e^{\boldsymbol\beta_0 \cdot \mathbf{X}_i} + e^{\boldsymbol\beta_1 \cdot \mathbf{X}_i}} = \frac{e^{(\boldsymbol\beta_0 - \boldsymbol\beta_1) \cdot \mathbf{X}_i}}{e^{(\boldsymbol\beta_0 - \boldsymbol\beta_1) \cdot \mathbf{X}_i} + 1} = \frac{e^{-\boldsymbol\beta . \mathbf{X}_i}}{1 + e^{-\boldsymbol\beta \cdot \mathbf{X}_i}}$$

$$\Pr(Y_i = 1) = \frac{e^{\boldsymbol\beta_1 \cdot \mathbf{X}_i}}{\sum_{0 \le c \le K} e^{\boldsymbol\beta_c \cdot \mathbf{X}_i}} = \frac{e^{\boldsymbol\beta_1 \cdot \mathbf{X}_i}}{e^{\boldsymbol\beta_0 \cdot \mathbf{X}_i} + e^{\boldsymbol\beta_1 \cdot \mathbf{X}_i}} = \frac{1}{e^{(\boldsymbol\beta_0 - \boldsymbol\beta_1) \cdot \mathbf{X}_i} + 1} = \frac{1}{1 + e^{-\boldsymbol\beta . \mathbf{X}_i}}$$

with $\boldsymbol\beta = -(\boldsymbol\beta_0 - \boldsymbol\beta_1)$

softmax Classifier vs. SVM

softmax Classifier

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

SVM

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

Scores	Softmax Classifier	SVM
[10, 7, 5]	-log(0.95) = 0.05	0
[10, 9, 9]	-log(0.58) = 0.55	0
[10, -10, -10]	-log(1.00) = $4e^{-9}$	0

- Suppose 10 is the correct prediction.

softmax Classifier vs. SVM (cont.)

- The performance of softmax classifiers and SVM are usually comparable.
- SVM does not care about the details of individual scores: [10, -5, -5] or [10, 9, 9] (Suppose 10 is the correct prediction.) This can be a feature or a bug.
- softmax classifiers are never fully satisfied with the prediction scores. The correct class could always have a higher probability.

car

truck

- How should a car classifier score a truck image?

Mean Squared Error

$$L_i(\mathbf{s}, \mathbf{y}_i) = \frac{1}{K}\|\mathbf{s} - \mathbf{y}_i\|_2^2 = \frac{1}{K}\sum_j \left(\mathbf{s}(j) - \mathbf{y}_i(j)\right)^2$$

	Scores ($\mathbf{s}$)	Label (y_i)	Label (y_i)
Dog	1.5	1	0
Car	-2	0	1
Bird	1.0	0	0
		Case 1	Case 2

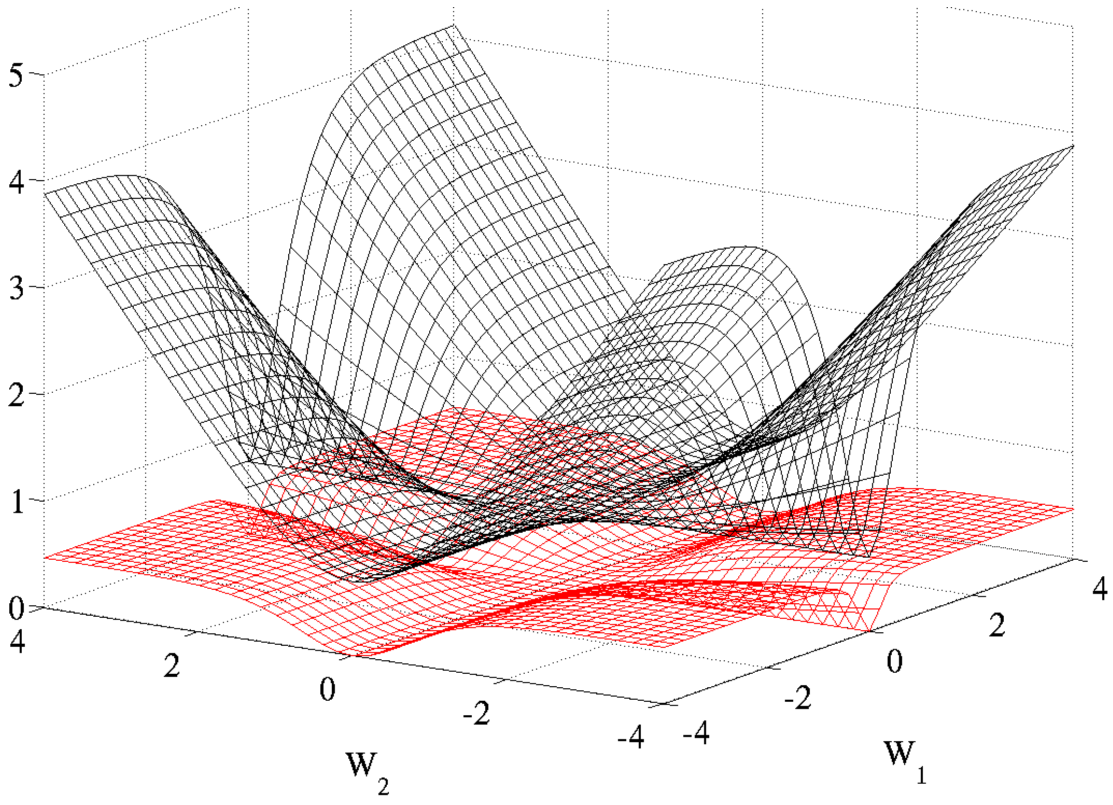
Case 1: Correct prediction

$L_i = ((1.5 - 1)^2 + (-2 - 0)^2 + (1.0 - 0)^2) / 3$
$= (0.25 + 4 + 1) / 3 = 1.75$

Case 2: Incorrect prediction

$L_i = ((1.5 - 0)^2 + (-2 - 1)^2 + (1.0 - 0)^2) / 3$
$= (2.25 + 9 + 1) / 3 = 4.1$

Cross-Entropy Loss vs. Mean Squared Error



- The black surface denotes the cross-entropy loss.
- The red surface denotes the mean squared error.
- w1 and w2 are weights of a network with two layers.

[1] X Glorot & Y Bengio, Understanding the difficulty of training deep feedforward neural networks