# CS 4803 / 7643: Deep Learning

Topics:

- Linear Classifiers
- Loss Functions

Dhruv Batra

Georgia Tech

# Administrativia

- Notes and readings on class webpage
  - https://www.cc.gatech.edu/classes/AY2020/cs7643_fall/

- HW0 solutions and grades released

- Issues from PS0 submission
  - Instructions not followed = not graded

1. We will be using Gradescope to collect your assignments. Please read the following instructions for submitting to Gradescope carefully! Failure to follow these instructions may result in parts of your assignment not being graded. We will not entertain regrading requests for failure to follow instructions.

   - For Section 1: Multiple Choice Questions, it is mandatory to use the LaTeX template provided on the class webpage (https://www.cc.gatech.edu/classes/AY2020/cs7643_fall/assets/ps0.zip). For every question, there is only one correct answer. To mark the correct answer, change `\choice` to `\CorrectChoice`
   - For Section 2: Proofs, each problem/sub-problem is in its own page. This section has 5 total problems/sub-problems, so you should have 5 pages corresponding to this section. Your answer to each sub-problem should fit in its corresponding page.
   - For Section 2, LaTeX'd solutions are strongly encouraged (solution template available at https://www.cc.gatech.edu/classes/AY2020/cs7643_fall/assets/ps0.zip), but scanned handwritten copies are acceptable. If you scan handwritten copies, please make sure to append them to the pdf generated by LaTeX for Section 1.

# Recap from last time

# **Image Classification**: A core task in Computer Vision

This image by Nikita is licensed under CC-BY 2.0

$x$

(assume given set of discrete labels)
{dog, cat, truck, plane, ...}

→ cat

$y$

# An image classifier

```
def classify_image(image):
  # Some magic here?
  return class_label
```

Unlike e.g. sorting a list of numbers,

**no obvious way** to hard-code the algorithm for recognizing a cat, or other classes.

# Supervised Learning

- Input: x (images, text, emails…)
- Output: y (spam or non-spam…)

- (Unknown) Target Function
  - f: X ✉ Y (the "true" mapping / reality)

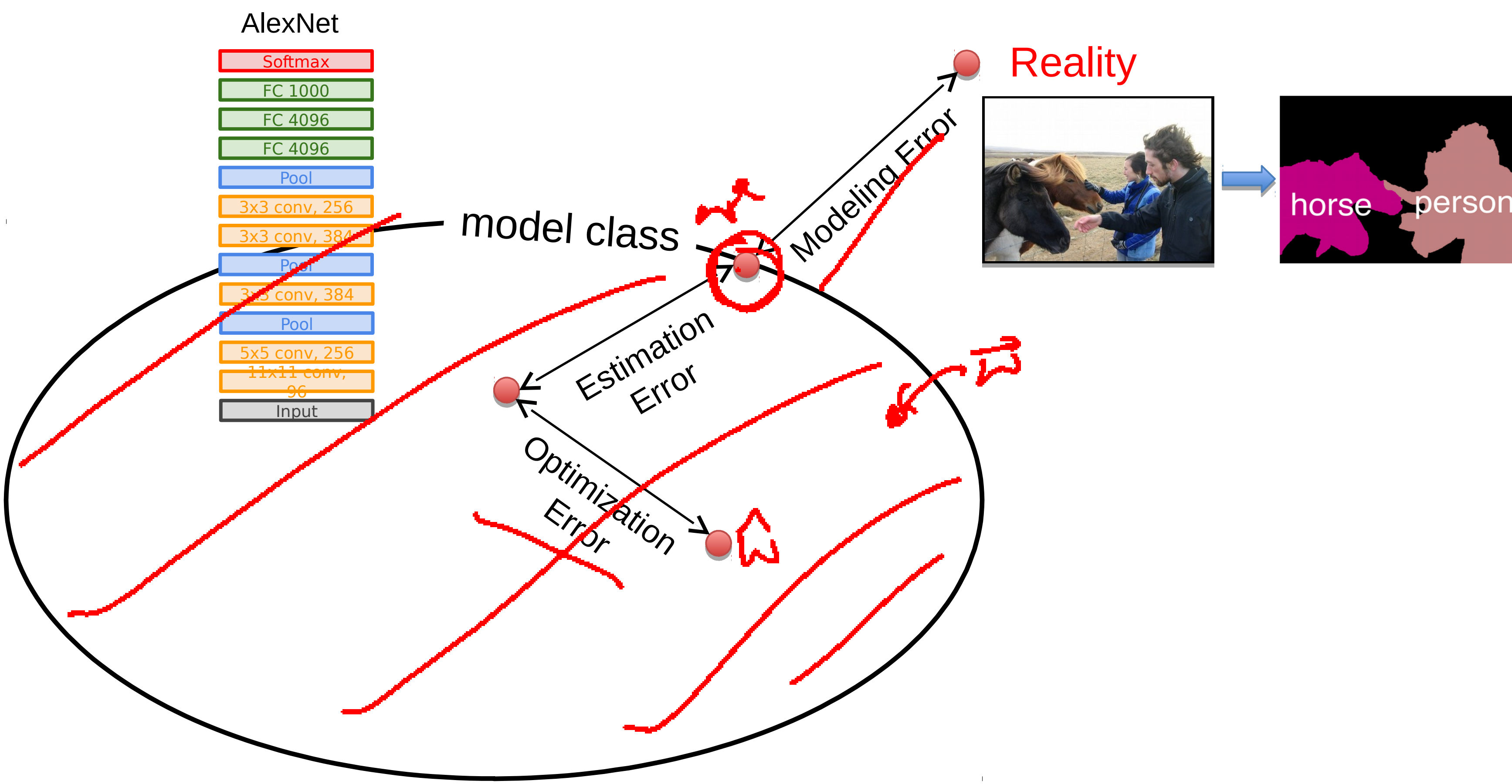
- Data
  - { $(x_1,y_1)$, $(x_2,y_2)$, …, $(x_N,y_N)$ }

- Model / Hypothesis Class
  - H = {h: X ✉ Y}
  - e.g. $y = h(x) = \text{sign}(w^T x)$

- Loss Function
  - How good is a model wrt my data D?

- Learning = Search in hypothesis space
  - Find best h in model class.

# Error Decomposition



# First classifier: **Nearest Neighbor**

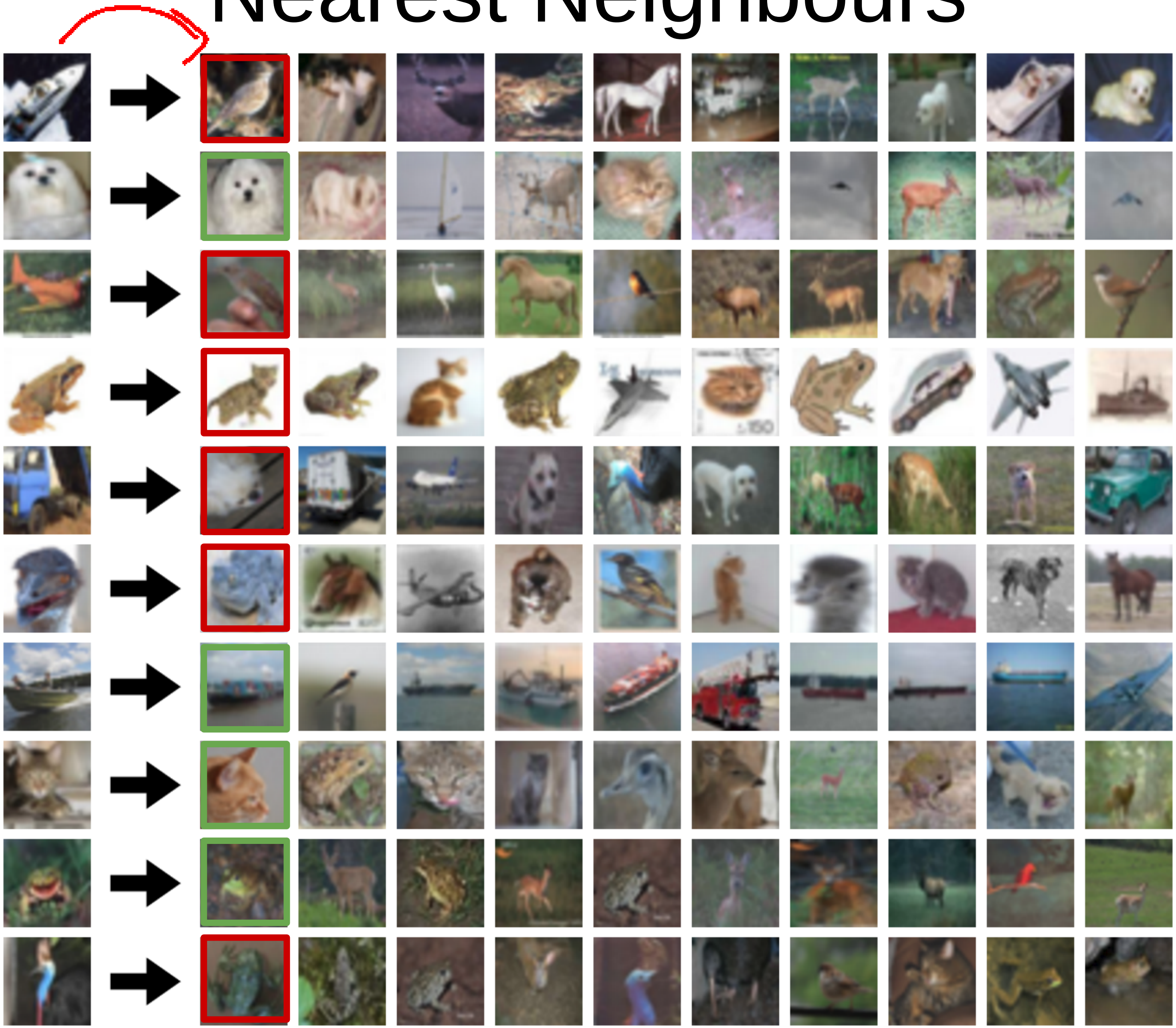
```
def train(images, labels):
  # Machine learning!
  return model
```

→ Memorize all data and labels

```
def predict(model, test_images):
  # Use model to predict labels
  return test_labels
```

→ Predict the label of the most similar training image

# Nearest Neighbours

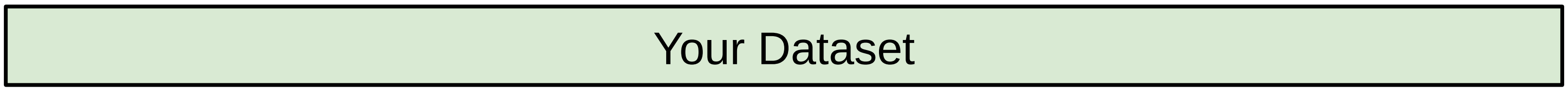


# Instance/Memory-based Learning

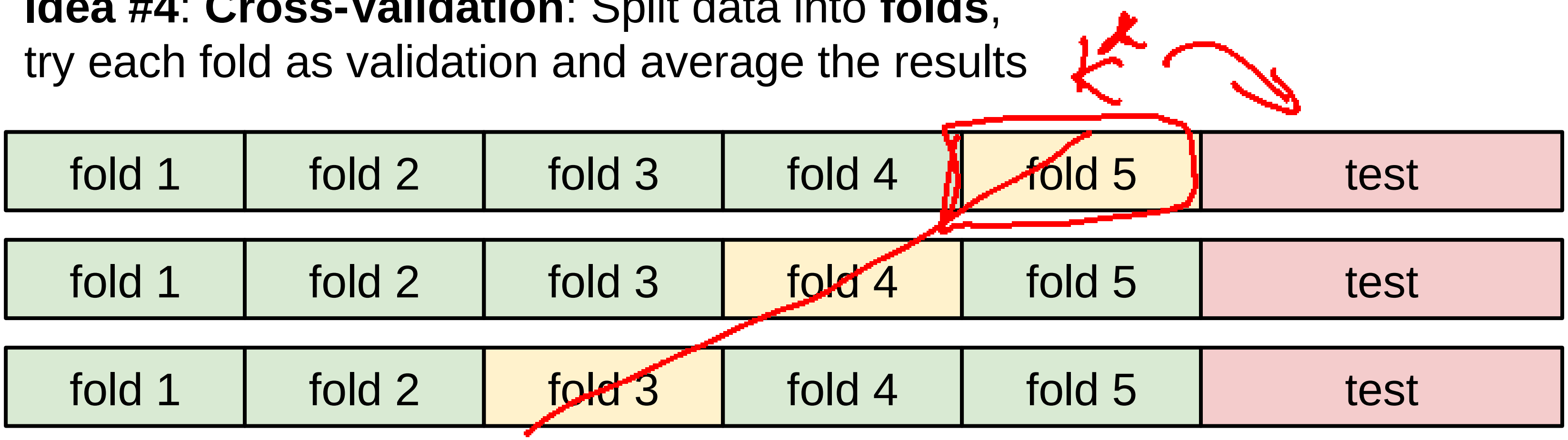
Four things make a memory based learner:

- *A distance metric*
- *How many nearby neighbors to look at?*
- *A weighting function (optional)*
- *How to fit with the local points?*

# Hyperparameters




**Idea #4**: **Cross-Validation**: Split data into **folds**, try each fold as validation and average the results




Useful for small datasets, but not used too frequently in deep learning

# Problems with Instance-Based Learning

- Expensive
  - No Learning: most real work done during testing
  - For every test sample, must search through all dataset – very slow!
  - Must use tricks like approximate nearest neighbour search

- Doesn't work well when large number of irrelevant features
  - Distances overwhelmed by noisy features

- Curse of Dimensionality
  - Distances become meaningless in high dimensions
  - (See proof in next lecture)

# Plan for Today

- Linear Classifiers
  - Linear scoring functions
- Loss Functions
  - Multi-class hinge loss
  - Softmax cross-entropy loss

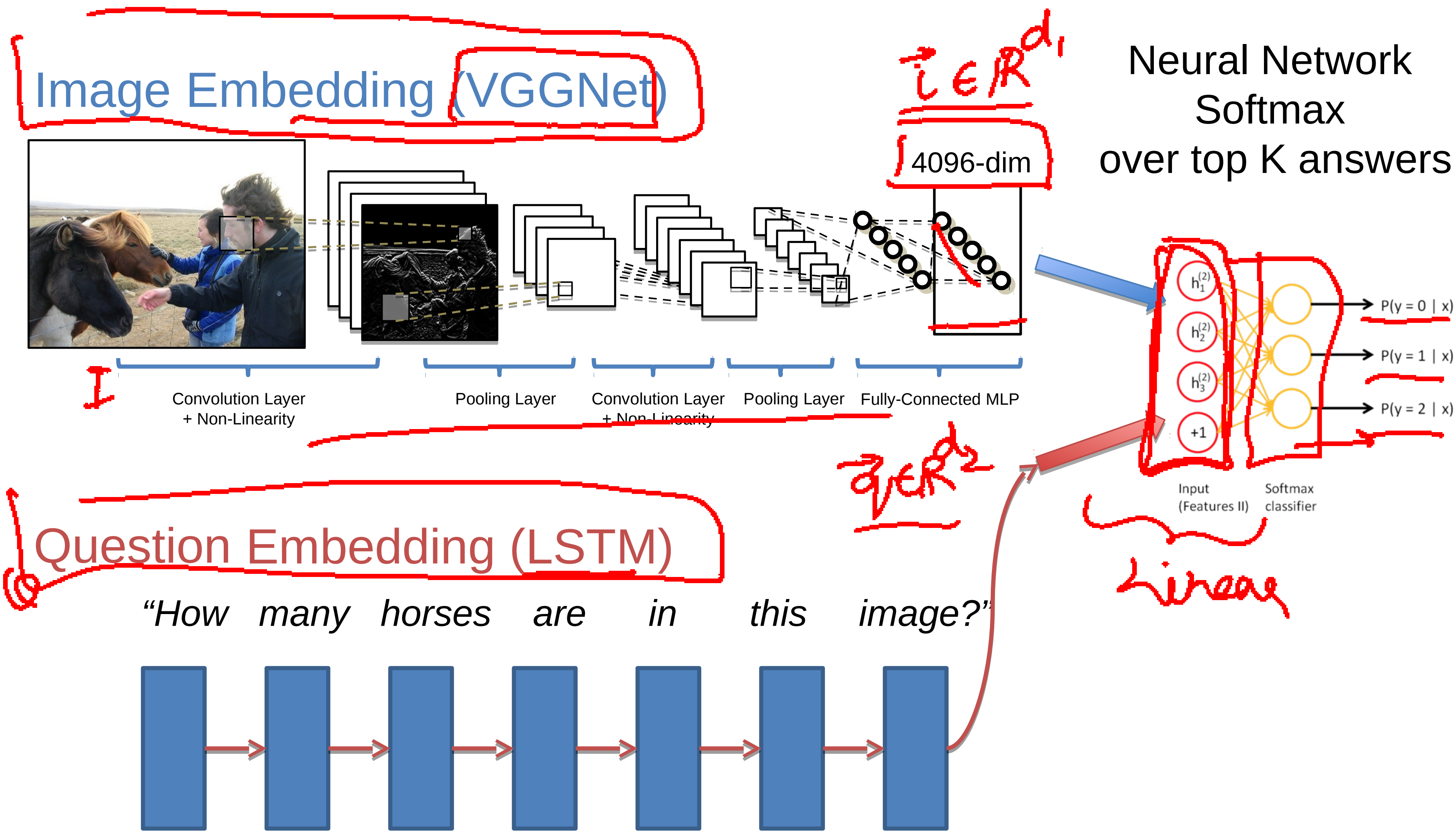
# Linear Classification

Neural Network

Linear classifiers

This image is CC0 1.0 public domain

# Visual Question Answering



# Recall CIFAR10


**50,000** training images
each image is **32x32x3**

**10,000** test images.

# Parametric Approach

$\mathcal{H} = \{h : X \to Y\}_{w}$

Image

Array of **32x32x3** numbers
(3072 numbers total)

f(x,W)

**10** numbers giving class scores

$\vec{s} = \begin{bmatrix} s_1 \\ s_2 \\ \vdots \\ s_{10} \end{bmatrix}$

W
parameters
or weights

# Parametric Approach: Linear Classifier

$\mathcal{H} = \{h : X \to Y\}$

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

Image

Array of **32x32x3** numbers (3072 numbers total)

$x \in \mathbb{R}^{3072}$

f(**x**,**W**) → **10** numbers giving class scores

**W**

parameters or weights

# Parametric Approach: Linear Classifier

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

3072x1

10x1

10x1 10x3072

Image

Array of **32x32x3** numbers (3072 numbers total)

$f(\mathbf{x},\mathbf{W})$

**10** numbers giving class scores

W

parameters or weights

Example with an image with 4 pixels, and 3 classes (cat/dog/ship)

Stretch pixels into column


Input image

# Example with an image with 4 pixels, and 3 classes (cat/dog/ship)

Stretch pixels into column

Input image

W ($W_{cat}^T$ = first row; 3x4)

| | | | |
|---|---|---|---|
| 0.2 | -0.5 | 0.1 | 2.0 |
| 1.5 | 1.3 | 2.1 | 0.0 |
| 0 | 0.25 | 0.2 | -0.3 |

$\vec{x}$

| |
|---|
| 56 |
| 231 |
| 24 |
| 2 |

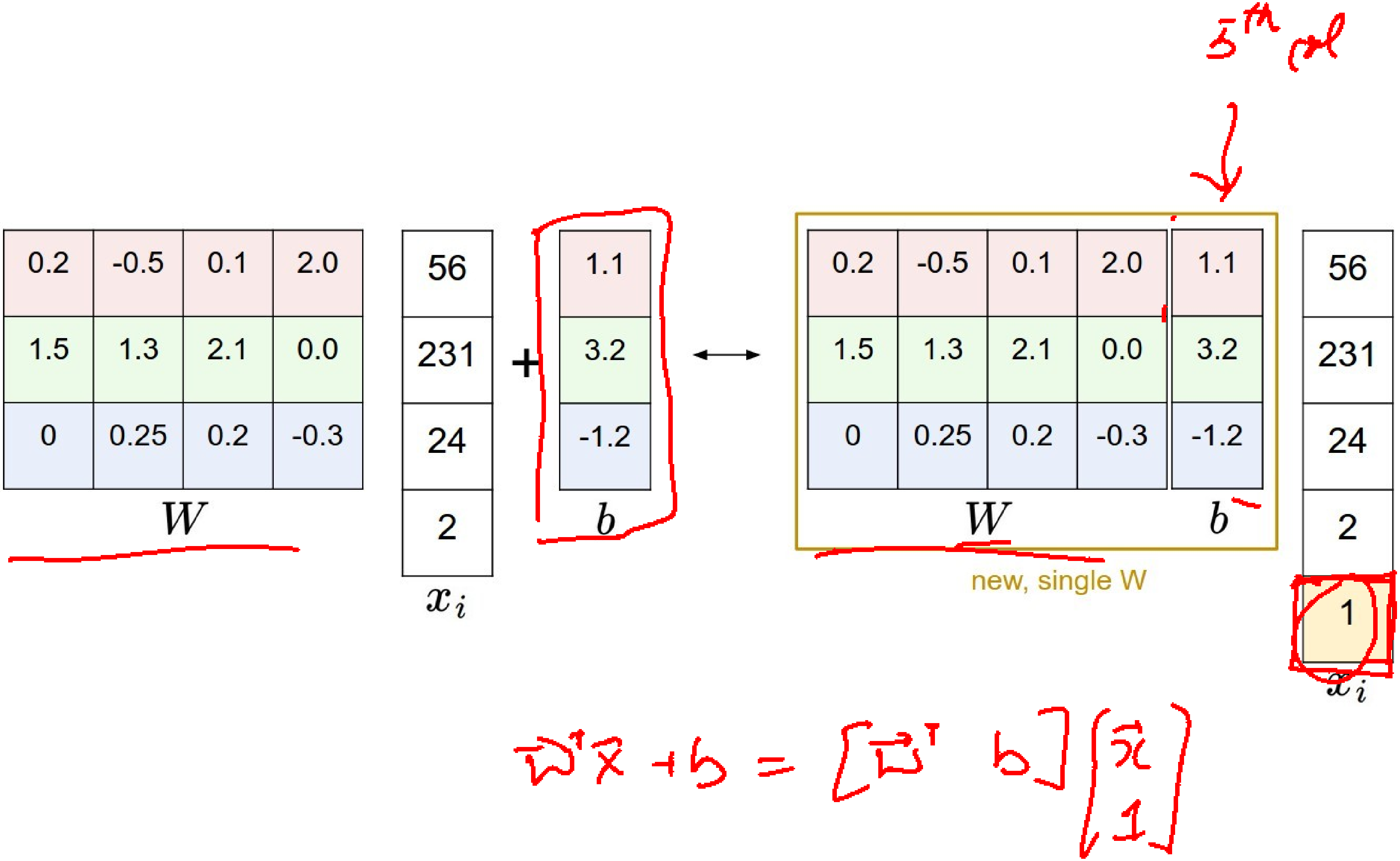
\+ b (3x1)

| | |
|---|---|
| 1.1 | $b_{cat}$ |
| 3.2 | |
| -1.2 | |

= s (3x1)

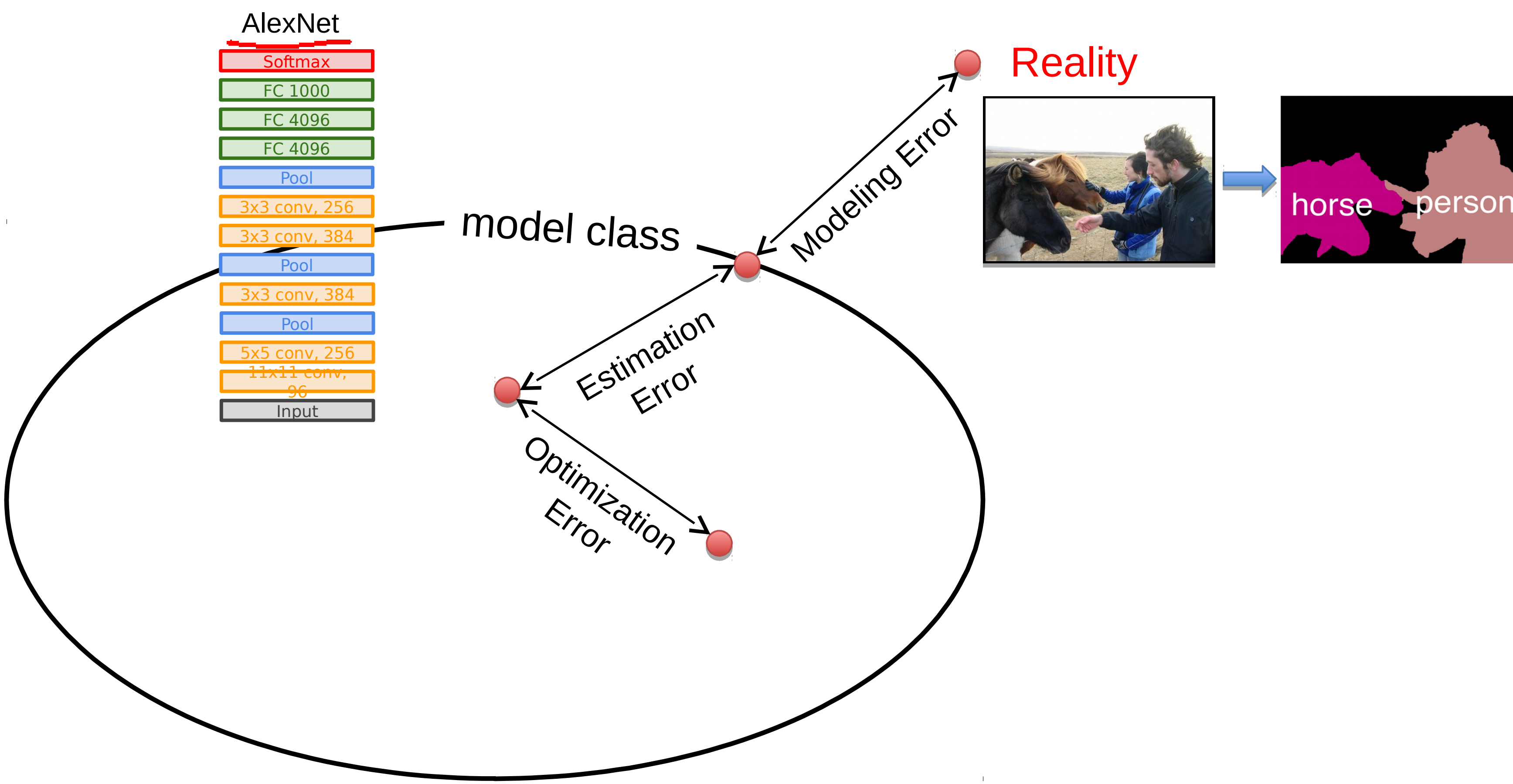
| | |
|---|---|
| -96.8 | Cat score |
| 437.9 | Dog score |
| 61.95 | Ship score |

$$\langle W_{cat}, \vec{x} \rangle + b_{cat} = s_{cat}$$

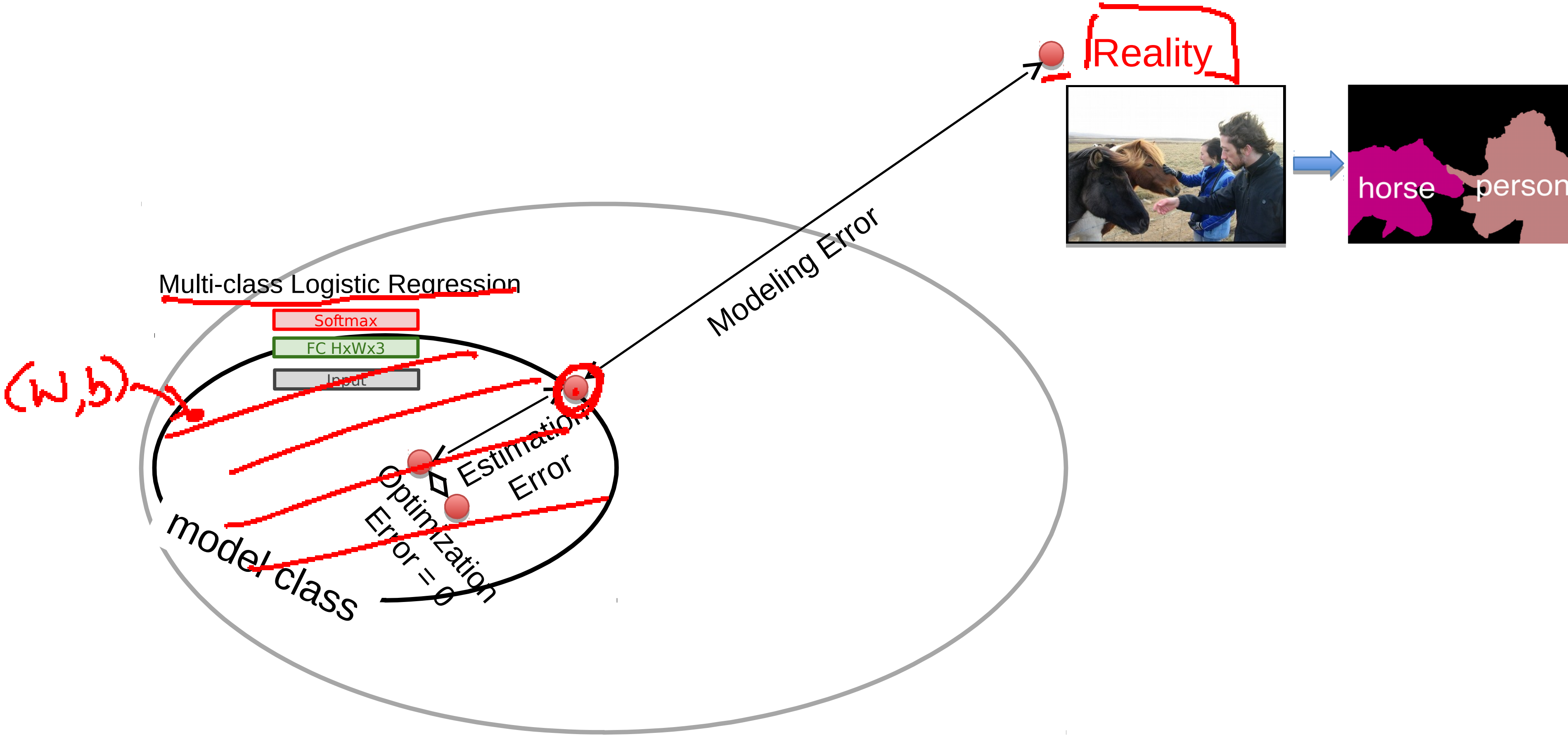


0.2 -0.5 0.1 2.0
1.5 1.3 2.1 0.0
0 0.25 0.2 -0.3
W
56
231
24
2
$x_i$
+
1.1
3.2
-1.2
b
0.2 -0.5 0.1 2.0 1.1
1.5 1.3 2.1 0.0 3.2
0 0.25 0.2 -0.3 -1.2
W
b
new, single W
56
231
24
2
1
$x_i$

# Error Decomposition



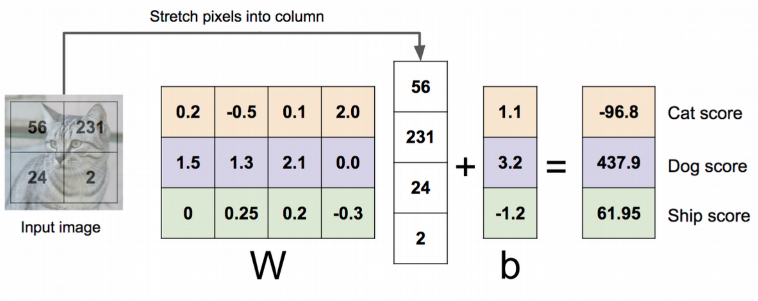
# Error Decomposition



# Example with an image with 4 pixels, and 3 classes (cat/dog/ship)

Algebraic Viewpoint

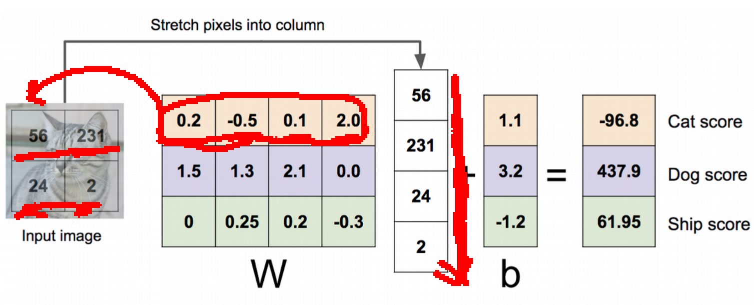
$$f(x,W) = Wx + b$$

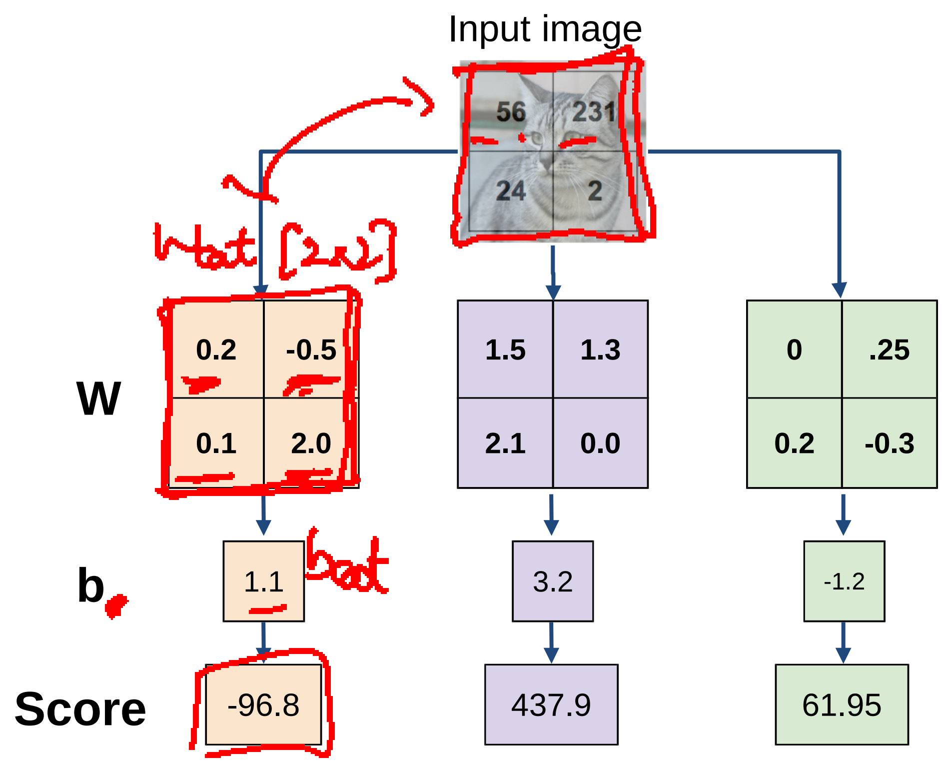


# Example with an image with 4 pixels, and 3 classes (cat/dog/ship)

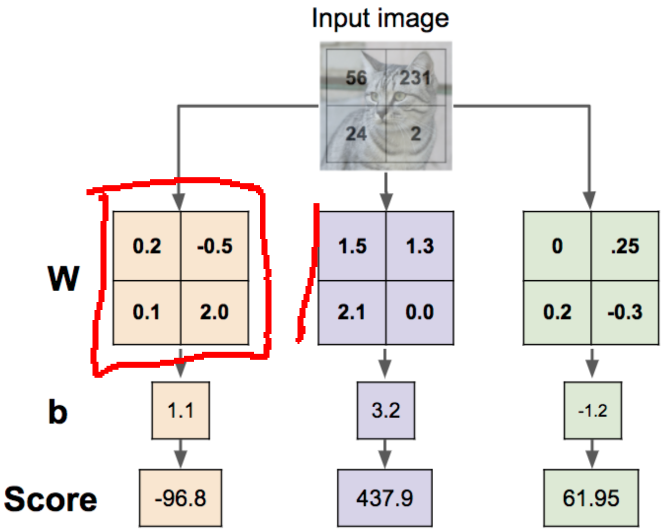
## Algebraic Viewpoint

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

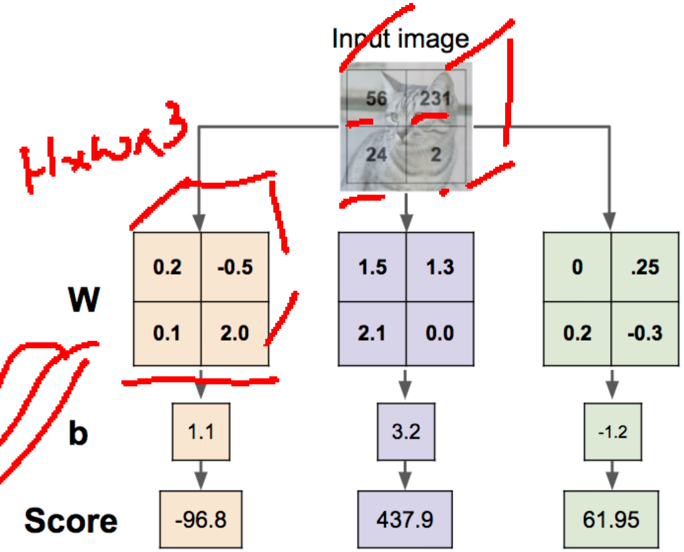


# Interpreting a Linear Classifier

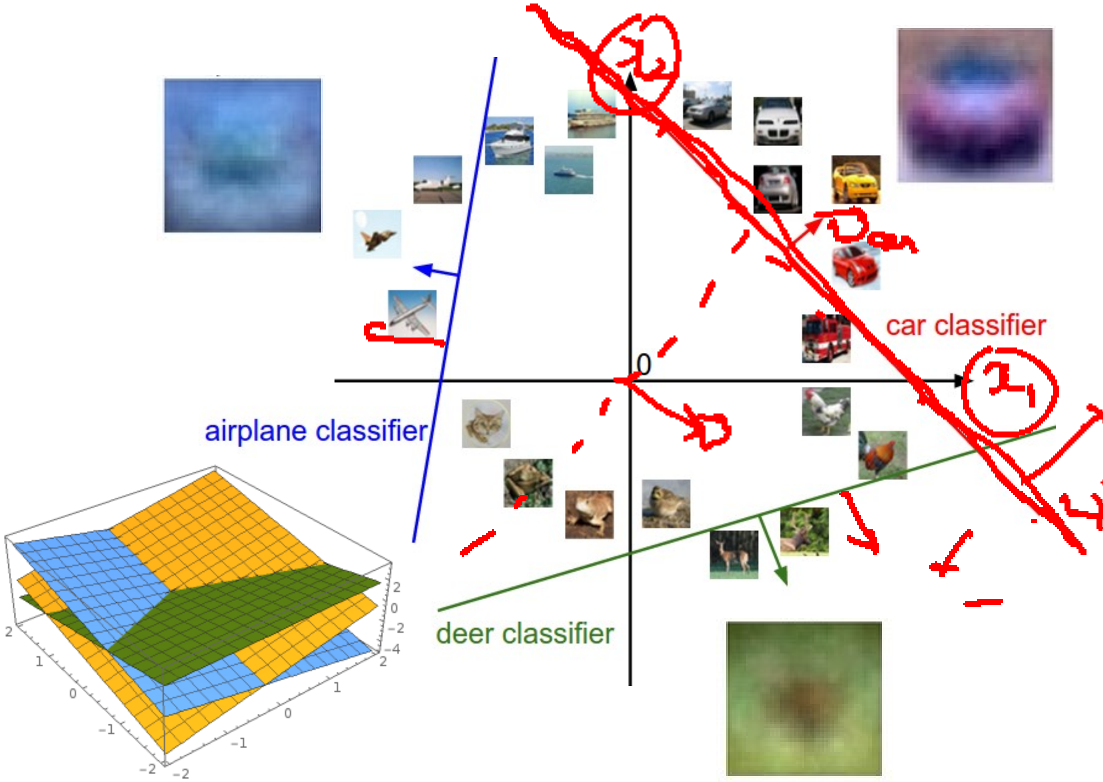
# Interpreting a Linear Classifier:
## Visual Viewpoint



Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

# Interpreting a Linear Classifier: Geometric Viewpoint




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

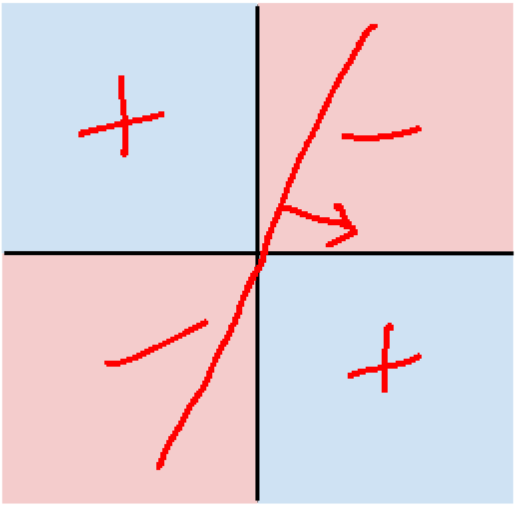
Array of **32x32x3** numbers
(3072 numbers total)

Plot created using Wolfram Cloud

# Hard cases for a linear classifier

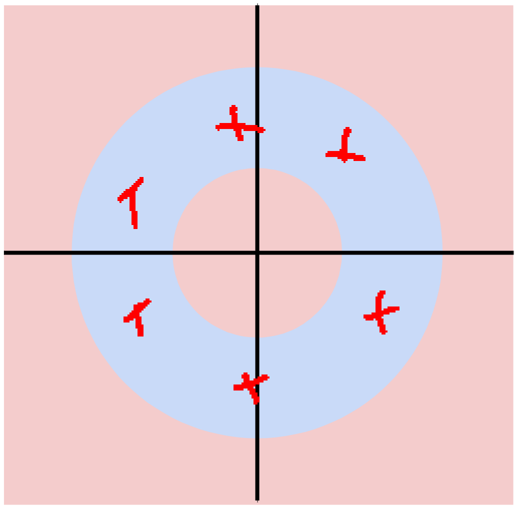
**Class 1**:
First and third quadrants

**Class 2**:
Second and fourth quadrants



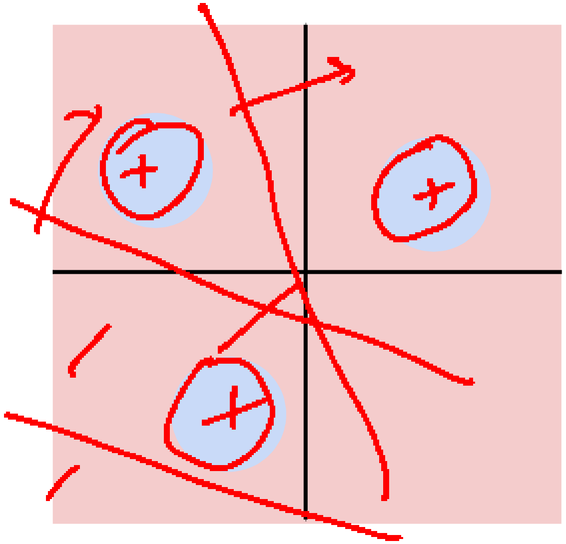
**Class 1**:
1 <= L2 norm <= 2

**Class 2**:
Everything else



**Class 1**:
Three modes

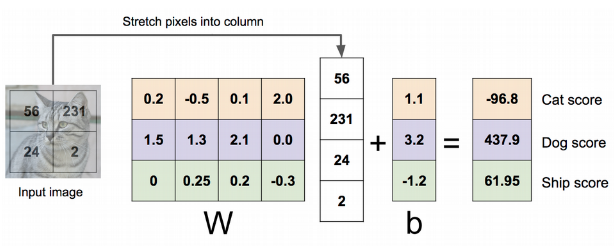
**Class 2**:
Everything else



# Linear Classifier: Three Viewpoints

<u>Algebraic Viewpoint</u>

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

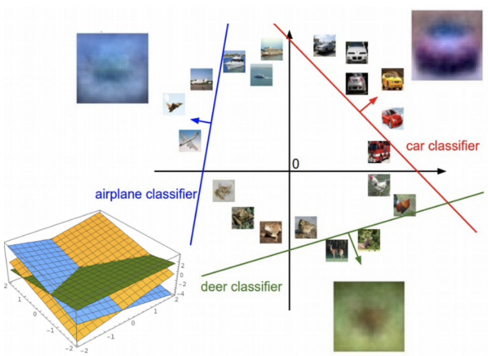



<u>Visual Viewpoint</u>

One template
per class

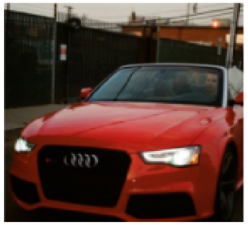
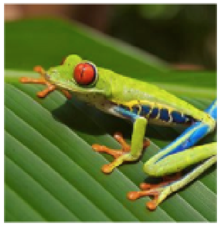

<u>Geometric Viewpoint</u>

Hyperplanes
cutting up space



# **So far**: Defined a (linear) score function

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

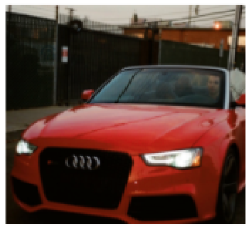
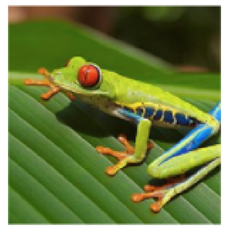
| | | | |
|---|---|---|---|
| **airplane** | -3.45 | -0.51 | 3.42 |
| **automobile** | -8.87 | **6.04** | 4.64 |
| **bird** | 0.09 | 5.31 | 2.65 |
| **cat** | **2.9** | -4.22 | 5.1 |
| **deer** | 4.48 | -4.19 | 2.64 |
| **dog** | 8.02 | 3.58 | 5.55 |
| **frog** | 3.78 | 4.49 | **-4.34** |
| **horse** | 1.06 | -4.37 | -1.5 |
| **ship** | -0.36 | -2.09 | -4.79 |
| **truck** | -0.72 | -2.93 | 6.14 |



Example class scores for 3 images for some W:

How can we tell whether this W is good or bad?

# **So far**: Defined a (linear) score function

| | | | |
|---|---|---|---|
| airplane | -3.45 | -0.51 | 3.42 |
| automobile | -8.87 | **6.04** | 4.64 |
| bird | 0.09 | 5.31 | 2.65 |
| cat | **2.9** | -4.22 | 5.1 |
| deer | 4.48 | -4.19 | 2.64 |
| dog | 8.02 | 3.58 | 5.55 |
| frog | 3.78 | 4.49 | **-4.34** |
| horse | 1.06 | -4.37 | -1.5 |
| ship | -0.36 | -2.09 | -4.79 |
| truck | -0.72 | -2.93 | 6.14 |



TODO:

1. Define a **loss function** that quantifies our unhappiness with the scores across the training data.
2. Come up with a way of efficiently finding the parameters that minimize the loss function. **(optimization)**

# Supervised Learning

- Input: x (images, text, emails…)
- Output: y (spam or non-spam…)

- (Unknown) Target Function
  - f: X ✉ Y (the "true" mapping / reality)

- Data
  - $(x_1,y_1), (x_2,y_2), \ldots, (x_N,y_N)$

- Model / Hypothesis Class
  - {h: X ✉ Y}
  - e.g. $y = h(x) = \text{sign}(w^T x)$

- Loss Function
  - How good is a model wrt my data D?

- Learning = Search in hypothesis space
  - Find best h in model class.

# Loss Functions

min Loss

max Reward

Suppose: 3 training examples, 3 classes.
With some W the scores $f(x, W) = Wx$ are:



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

Suppose: 3 training examples, 3 classes.
With some W the scores $f(x, W) = Wx$ are:

| | i=1 | i=2 | i=3 |
|---|---|---|---|
| |  |  |  |
| cat | **3.2** | 1.3 | 2.2 |
| car | 5.1 | **4.9** | 2.5 |
| frog | -1.7 | 2.0 | **-3.1** |

$s_i$

A **loss function** tells how good our current classifier is

Given a dataset of examples

$$\{(x_i, y_i)\}_{i=1}^{N}$$

Where $x_i$ is image and $y_i$ is (integer) label

Loss over the dataset is a sum of loss over examples:

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

$s_i$

Suppose: 3 training examples, 3 classes.
With some W the scores $f(x, W) = Wx$ are:

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

## Multiclass SVM loss:

Given an example $(x_i, y_i)$
where $x_i$ is the image and
where $y_i$ is the (integer) label,

and using the shorthand for the scores vector: $s = f(x_i, W)$

the SVM loss has the form:

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

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

Suppose: 3 training examples, 3 classes.
With some W the scores $f(x, W) = Wx$ are:

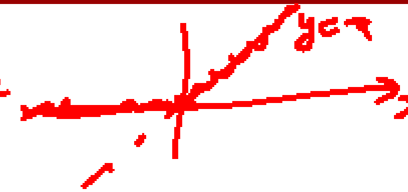
| | | | |
|---|---|---|---|
| cat | **3.2** | | |
| car | 5.1 | | |
| frog | -1.7 | | |

## Multiclass SVM loss:

Given an example $(x_i, y_i)$
where $x_i$ is the image and
where $y_i$ is the (integer) label,

and using the shorthand for the scores vector: $s = f(x_i, W)$

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

max(0, x) = 


Suppose: 3 training examples, 3 classes.
With some W the scores $f(x, W) = Wx$ are:

| | |
|---|---|
| cat | **3.2** |
| car | 5.1 |
| frog | -1.7 |

## Multiclass SVM loss:

Given an example $(x_i, y_i)$
where $x_i$ is the image and
where $y_i$ is the (integer) label,

and using the shorthand for the scores vector: $s = f(x_i, W)$

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

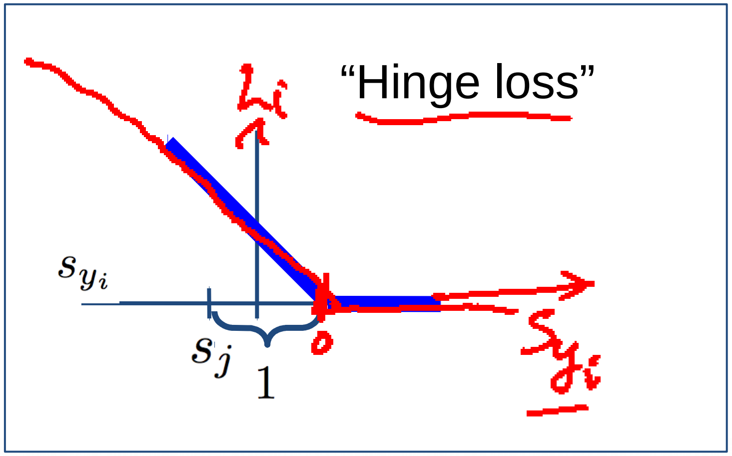
$$= \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$

Suppose: 3 training examples, 3 classes.
With some W the scores $f(x, W) = Wx$ are:



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

**Multiclass SVM loss:**




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

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

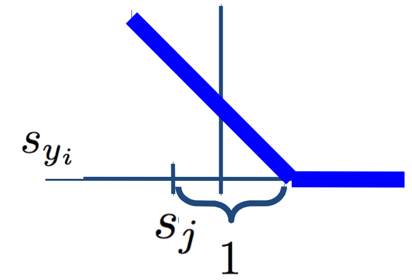
Suppose: 3 training examples, 3 classes.
With some W the scores $f(x, W) = Wx$ are:

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

## Multiclass SVM loss:

"Hinge loss"




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

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


Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

Suppose: 3 training examples, 3 classes.
With some W the scores $f(x,W) = Wx$ are:



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

**Multiclass SVM loss:**

Given an example $(x_i, y_i)$
where $x_i$ is the image and
where $y_i$ is the (integer) label,

and using the shorthand for the scores vector: $s = f(x_i, W)$

the SVM loss has the form:

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

Suppose: 3 training examples, 3 classes.
With some W the scores $f(x, W) = Wx$ are:



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

$L_i$

## Multiclass SVM loss:

Given an example $(x_i, y_i)$
where $x_i$ is the image and
where $y_i$ is the (integer) label,

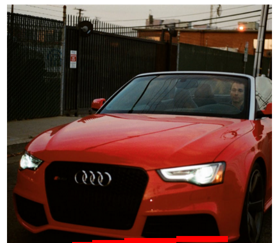
and using the shorthand for the scores vector: $s = f(x_i, W)$

the SVM loss has the form:

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

= max(0, 5.1 - 3.2 + 1)
+max(0, -1.7 - 3.2 + 1)
= max(0, 2.9) + max(0, -3.9)
= 2.9 + 0
= 2.9

Suppose: 3 training examples, 3 classes.
With some W the scores $f(x, W) = Wx$ are:


| | | | |
|---|---|---|---|
| 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 | |

## Multiclass SVM loss:

Given an example $(x_i, y_i)$ where $x_i$ is the image and where $y_i$ is the (integer) label,

and using the shorthand for the scores vector: $s = f(x_i, W)$

the SVM loss has the form:

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

= max(0, 1.3 - 4.9 + 1)
  +max(0, 2.0 - 4.9 + 1)
= max(0, -2.6) + max(0, -1.9)
= 0 + 0
= 0

Suppose: 3 training examples, 3 classes.
With some W the scores $f(x, W) = Wx$ are:



| | | | |
|---|---|---|---|
| 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 | 12.9 |

## Multiclass SVM loss:

Given an example $(x_i, y_i)$
where $x_i$ is the image and
where $y_i$ is the (integer) label,

and using the shorthand for the scores vector: $s = f(x_i, W)$

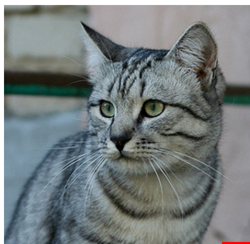
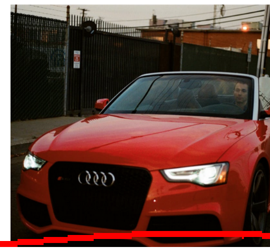
the SVM loss has the form:

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

= max(0, 2.2 - (-3.1) + 1)
+max(0, 2.5 - (-3.1) + 1)
= max(0, 6.3) + max(0, 6.6)
= 6.3 + 6.6
= 12.9

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

Suppose: 3 training examples, 3 classes.
With some W the scores $f(x, W) = Wx$ are:

| | | | |
|---|---|---|---|
| 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 | 12.9 |
| | $L_1$ | $L_2$ | $L_3$ |

## Multiclass SVM loss:

Given an example $(x_i, y_i)$
where $x_i$ is the image and
where $y_i$ is the (integer) label,

and using the shorthand for the scores vector: $s = f(x_i, W)$

the SVM loss has the form:

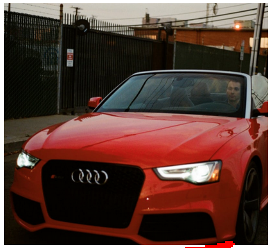
$$L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$

Loss over full dataset is average:

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

L = (2.9 + 0 + 12.9)/3
= **5.27**

Suppose: 3 training examples, 3 classes.
With some W the scores $f(x, W) = Wx$ are:

| | | | |
|---|---|---|---|
| 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 | 12.9 |

**Multiclass SVM loss:**

Given an example $(x_i, y_i)$
where $x_i$ is the image and
where $y_i$ is the (integer) label,

and using the shorthand for the scores vector: $s = f(x_i, W)$

the SVM loss has the form:

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

Q: What happens to loss if car image scores change a bit?

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

Suppose: 3 training examples, 3 classes.
With some W the scores $f(x, W) = Wx$ are:

| | | | |
|---|---|---|---|
| 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 | 12.9 |

**Multiclass SVM loss:**

Given an example $(x_i, y_i)$
where $x_i$ is the image and
where $y_i$ is the (integer) label,

and using the shorthand for the scores vector: $s = f(x_i, W)$

the SVM loss has the form:

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

Q2: what is the min/max possible loss?

Suppose: 3 training examples, 3 classes.
With some W the scores $f(x, W) = Wx$ are:

| | | | |
|---|---|---|---|
| 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 | 12.9 |

## Multiclass SVM loss:

Given an example $(x_i, y_i)$
where $x_i$ is the image and
where $y_i$ is the (integer) label,

and using the shorthand for the scores vector: $s = f(x_i, W)$

the SVM loss has the form:

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

Q3: At initialization W is small so all s ≈ 0. What is the loss?

#classes -1

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

Suppose: 3 training examples, 3 classes.
With some W the scores $f(x, W) = Wx$ are:

| | | | |
|---|---|---|---|
| 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 | 12.9 |

## Multiclass SVM loss:

Given an example $(x_i, y_i)$
where $x_i$ is the image and
where $y_i$ is the (integer) label,

and using the shorthand for the scores vector: $s = f(x_i, W)$

the SVM loss has the form:

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

Q4: What if the sum was over all classes? (including j = y_i)

+1

$L_i \leftarrow L_i + 1$

Suppose: 3 training examples, 3 classes.
With some W the scores $f(x, W) = Wx$ are:



| | | | |
|---|---|---|---|
| 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 | 12.9 |

## Multiclass SVM loss:

Given an example $(x_i, y_i)$ where $x_i$ is the image and where $y_i$ is the (integer) label,

and using the shorthand for the scores vector: $s = f(x_i, W)$

the SVM loss has the form:

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

Q5: What if we used mean instead of sum?

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

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

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

E.g. Suppose that we found a W such that L = 0.

Q7: Is this W unique?

$L(W) = 0$

$\Rightarrow L(2W) = 0$

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

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

E.g. Suppose that we found a W such that L = 0.

Q7: Is this W unique?

**No! 2W is also has L = 0!**

Suppose: 3 training examples, 3 classes.
With some W the scores $f(x, W) = Wx$ are:

| | | | |
|---|---|---|---|
| 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 | |

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

**Before:**

= max(0, 1.3 - 4.9 + 1)
+max(0, 2.0 - 4.9 + 1)
= max(0, -2.6) + max(0, -1.9)
= 0 + 0
= 0

**With W twice as large:**

= max(0, 2.6 - 9.8 + 1)
+max(0, 4.0 - 9.8 + 1)
= max(0, -6.2) + max(0, -4.8)
= 0 + 0
= 0

# Multiclass SVM Loss: Example code

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

```
def L_i_vectorized(x, y, W):
  scores = W.dot(x)
  margins = np.maximum(0, scores - scores[y] + 1)
  margins[y] = 0
  loss_i = np.sum(margins)
  return loss_i
```

## **Softmax Classifier** (Multinomial Logistic Regression)

Want to interpret raw classifier scores as **probabilities**

| | |
|---|---|
| cat | **3.2** |
| car | 5.1 |
| frog | -1.7 |

**Softmax Classifier** (Multinomial Logistic Regression)

Want to interpret raw classifier scores as **probabilities**

$$s = f(x_i; W)$$

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

Softmax function

| | |
|---|---|
| cat | **3.2** |
| car | 5.1 |
| frog | -1.7 |

## **Softmax Classifier** (Multinomial Logistic Regression)

Want to interpret raw classifier scores as **probabilities**

$$s = f(x_i; W)$$

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

Softmax Function

Probabilities must be >= 0

| | | | |
|---|---|---|---|
| cat | **3.2** | exp → | **24.5** |
| car | 5.1 | | 164.0 |
| frog | -1.7 | | 0.18 |

unnormalized probabilities

## **Softmax Classifier** (Multinomial Logistic Regression)

Want to interpret raw classifier scores as **probabilities**

$$s = f(x_i; W)$$

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

Softmax Function

Probabilities must be >= 0

Probabilities must sum to 1

| | scores | exp → | unnormalized probabilities | normalize → | probabilities |
|---|---|---|---|---|---|
| cat | **3.2** | | **24.5** | | **0.13** |
| car | 5.1 | | 164.0 | | 0.87 |
| frog | -1.7 | | 0.18 | | 0.00 |

unnormalized probabilities

probabilities

## **Softmax Classifier** (Multinomial Logistic Regression)

Want to interpret raw classifier scores as **probabilities**

$$s = f(x_i; W)$$

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

Softmax Function

Probabilities must be >= 0

Probabilities must sum to 1

| | | exp | | normalize | |
|---|---|---|---|---|---|
| cat | **3.2** | | **24.5** | | **0.13** |
| car | 5.1 | | 164.0 | | 0.87 |
| frog | -1.7 | | 0.18 | | 0.00 |

Unnormalized log-probabilities / logits

unnormalized probabilities

probabilities

## **Softmax Classifier** (Multinomial Logistic Regression)

Want to interpret raw classifier scores as **probabilities**

$$s = f(x_i; W)$$

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

| | |
|---|---|
| cat | **3.2** |
| car | 5.1 |
| frog | -1.7 |

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

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

Maximize log-prob of the correct class =
Maximize the log likelihood =
Minimize the negative log likelihood

## **Softmax Classifier** (Multinomial Logistic Regression)

Want to interpret raw classifier scores as **probabilities**

$$s = f(x_i; W)$$

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

Softmax Function

Probabilities must be >= 0

Probabilities must sum to 1

| | Unnormalized log-probabilities / logits | | unnormalized probabilities | | probabilities |
|---|---|---|---|---|---|
| cat | **3.2** | exp → | **24.5** | normalize → | **0.13** |
| car | 5.1 | | 164.0 | | 0.87 |
| frog | -1.7 | | 0.18 | | 0.00 |

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

# **Softmax Classifier** (Multinomial Logistic Regression)

Want to interpret raw classifier scores as **probabilities**

$$s = f(x_i; W)$$

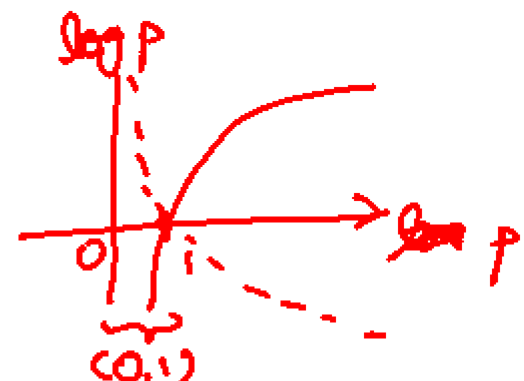
$$P(Y = k | X = x_i) = \frac{e^{s_k}}{\sum_j e^{s_j}}$$ Softmax Function

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

$y_i$=cat

| | Unnormalized log-probabilities / logits | exp → | unnormalized probabilities (Probabilities must be >= 0) | normalize → | probabilities (Probabilities must sum to 1) |
|---|---|---|---|---|---|
| cat | **3.2** | | **24.5** | | **0.13** |
| car | 5.1 | | 164.0 | | 0.87 |
| frog | -1.7 | | 0.18 | | 0.00 |

→ $L_i$ = -log(0.13)
= **2.04**



# Log-Likelihood / KL-Divergence / Cross-Entropy

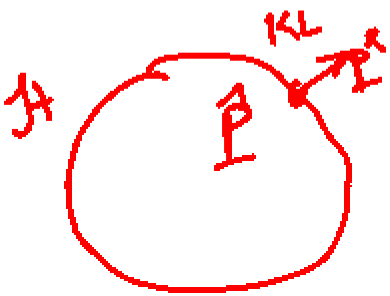
$$p^* = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} \leftarrow y_i \qquad \hat{p} = \begin{bmatrix} P_\theta(Y=1 \mid x_i, w) \\ P_\theta(Y=2 \mid x_i, w) \\ \vdots \\ P_\theta(Y=k \mid x_i, w) \end{bmatrix}$$

$$\min_w KL(p^* \,\|\, \hat{p}) = \sum_y p^*(y) \log \frac{p^*(y)}{\hat{p}(y)}$$

$$= \sum_y p^*(y) \log p^*(y) - \sum_y p^*(y) \log \hat{p}(y)$$

$$-H(p^*) \qquad \min_w H(p^*, \hat{p})$$

$$\min_w -\log \hat{p}(y_i)$$



# Log-Likelihood / KL-Divergence / Cross-Entropy

# **Softmax Classifier** (Multinomial Logistic Regression)

Want to interpret raw classifier scores as **probabilities**

$$s = f(x_i; W)$$

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

Softmax Function

Maximize probability of correct class

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

Putting it all together:

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

| | |
|---|---|
| cat | **3.2** |
| car | 5.1 |
| frog | -1.7 |

# **Softmax Classifier** (Multinomial Logistic Regression)

Want to interpret raw classifier scores as **probabilities**

$$s = f(x_i; W)$$

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

Softmax Function

| | |
|---|---|
| cat | **3.2** |
| car | 5.1 |
| frog | -1.7 |

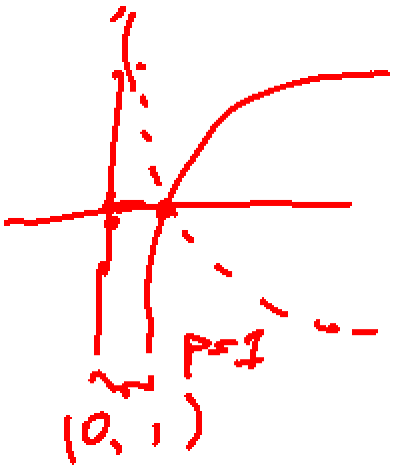
Maximize probability of correct class

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

Putting it all together:

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

Q: What is the min/max possible loss L_i?



Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

# **Softmax Classifier** (Multinomial Logistic Regression)

| | |
|---|---|
| cat | **3.2** |
| car | 5.1 |
| frog | -1.7 |

Want to interpret raw classifier scores as **probabilities**

$$s = f(x_i; W)$$

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

Softmax Function

Maximize probability of correct class

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

Putting it all together:

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

Q: What is the min/max possible loss L_i?

A: ~~min~~ inf 0, max infinity

## **Softmax Classifier** (Multinomial Logistic Regression)

Want to interpret raw classifier scores as **probabilities**

$$s = f(x_i; W)$$

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

Maximize probability of correct class

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

Putting it all together:

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

$W \sim 0$

| | | |
|---|---|---|
| cat | **3.2** | $\approx 0$ |
| car | 5.1 | $\approx 0$ |
| frog | -1.7 | $\approx 0$ |

Q2: At initialization all s will be approximately equal; what is the loss?

$\hat{P} = \begin{bmatrix} 1/k \\ 1/k \\ 1/k \end{bmatrix}$ $\quad -\log\left(\frac{1}{k}\right)$

$\log(k)$

## **Softmax Classifier** (Multinomial Logistic Regression)

| | |
|---|---|
| cat | **3.2** |
| car | 5.1 |
| frog | -1.7 |

Want to interpret raw classifier scores as **probabilities**

$$s = f(x_i; W)$$

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

Softmax Function

Maximize probability of correct class

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

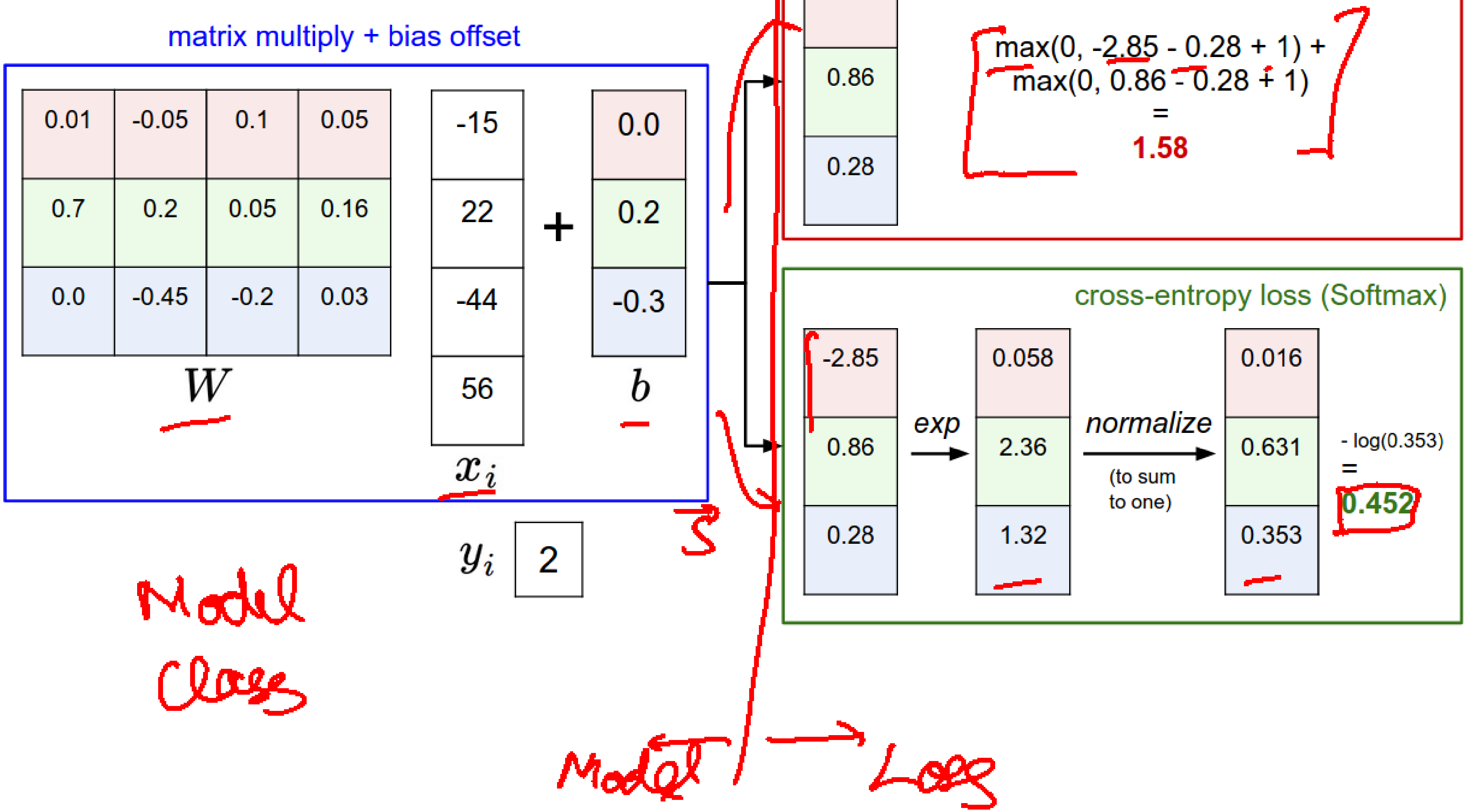
Putting it all together:

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

Q2: At initialization all s will be approximately equal; what is the loss?
A: log(C), eg log(10) ≈ 2.3

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

# Softmax vs. SVM



# Softmax vs. SVM

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

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

# Recap

- We have some dataset of (x,y)
- We have a **score function:** $s = f(x; W) \stackrel{\text{e.g.}}{=} Wx$
- We have a **loss function**:

Softmax

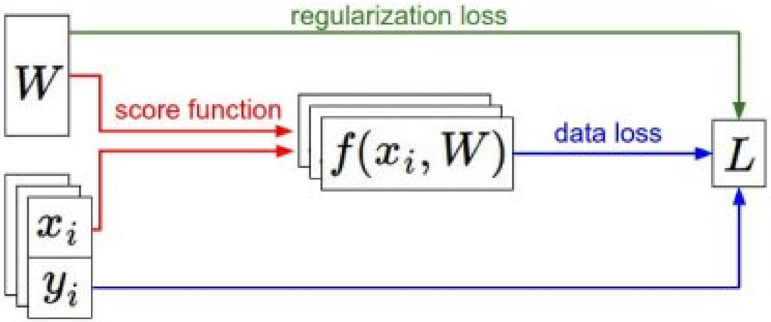
$$L_i = -\log(\frac{e^{s_{y_i}}}{\sum_j e^{s_j}})$$

SVM

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

$$L = \frac{1}{N} \sum_{i=1}^{N} L_i + R(W)$$

Full loss




Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

# Recap

**How do we find the best W?**

$W^*$

- We have some dataset of (x,y)
- We have a **score function:** $s = f(x; W) \overset{\text{e.g.}}{=} Wx$
- We have a **loss function**:

Softmax

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

SVM

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

$$L = \frac{1}{N} \sum_{i=1}^{N} L_i + R(W)$$ Full loss