ФИНАНСОВЫЙ УНИВЕРСИТЕТ
ПРИ ПРАВИТЕЛЬСТВЕ РОССИЙСКОЙ ФЕДЕРАЦИИ

Современные нейросетевые технологии

Лекция 2. Линейная классификация изображений

Учебные вопросы

1) Линейный классификатор изображений
2) Преобразование softmax
3) Функция потерь Cross Enthropy Loss

Материалы курса:
github.com/balezz/modern_dl
Срок сдачи A1 – 10.09.2022 г.

Источники:

- dlcourse.ai
- cs231n.stanford.edu
- cs230.stanford.edu

Линейный классификатор

Image Classification: A core task in Computer Vision

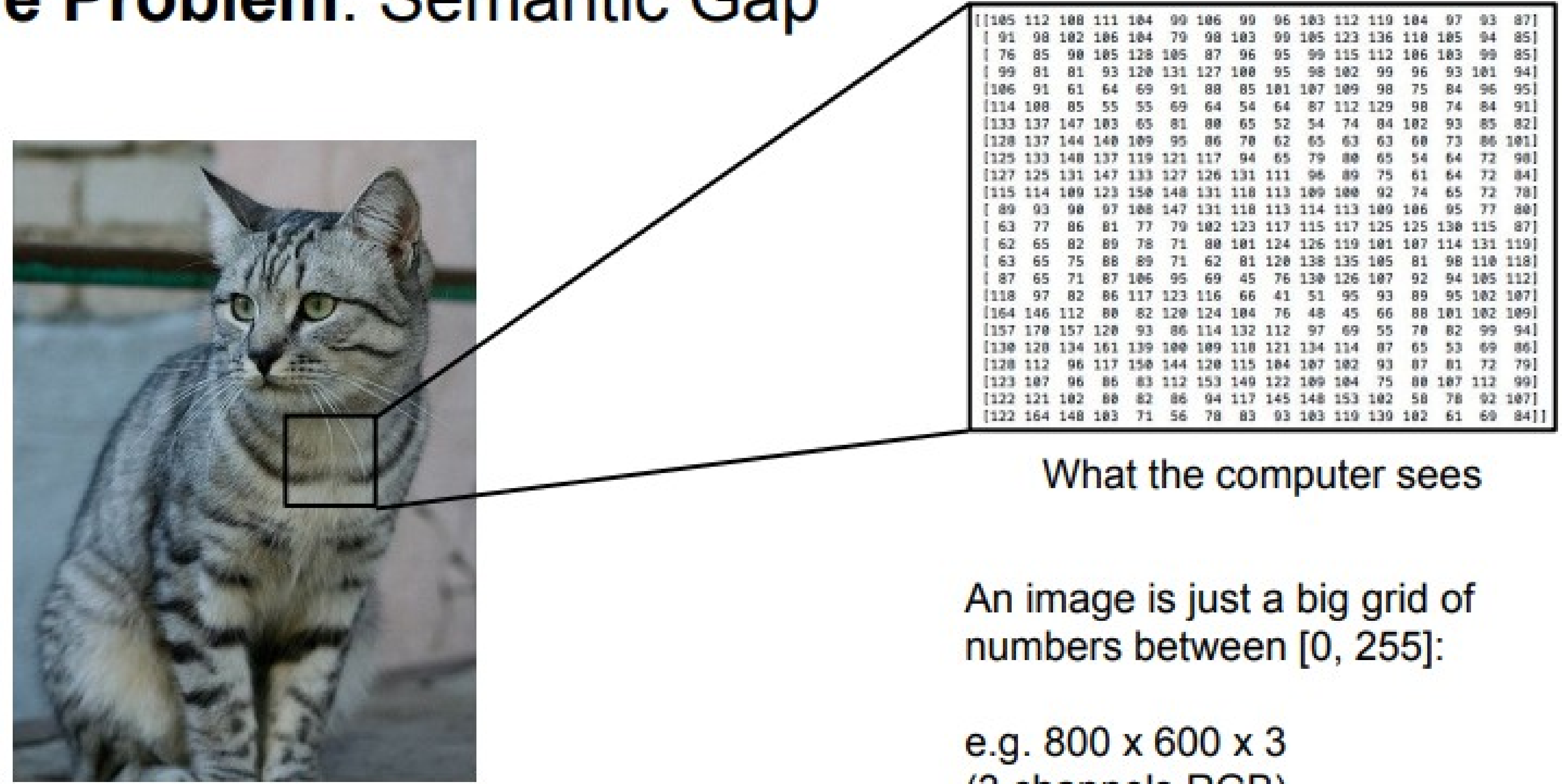
This image by Nikita is licensed under CC-BY 2.0

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

→ cat

Линейный классификатор

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]]
```

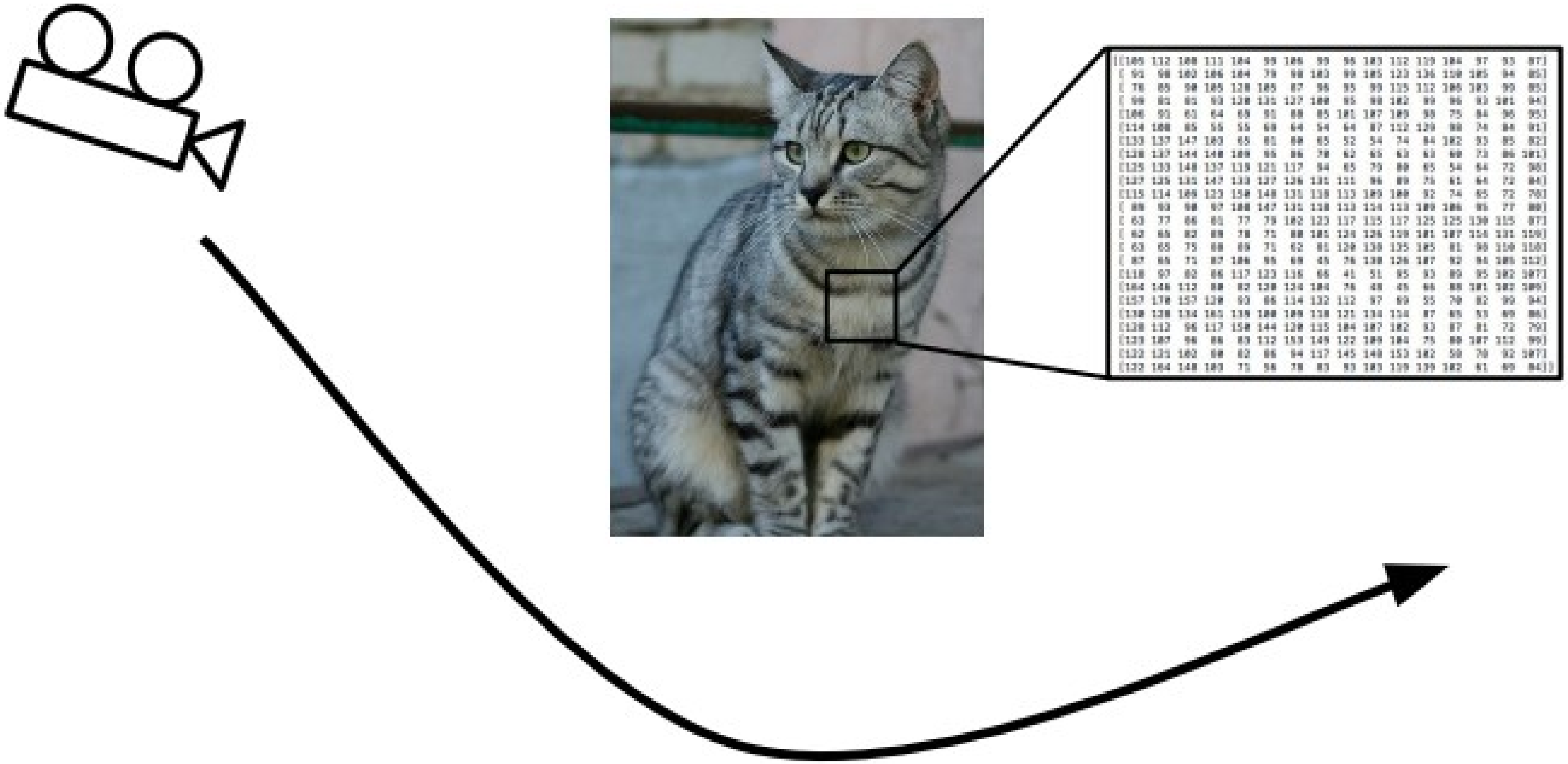
What the computer sees

This image by Nikita is
licensed under CC-BY 2.0

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



```
[[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]]
```

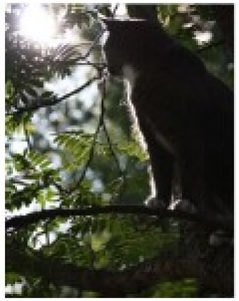
All pixels change when the camera moves!

Проблемы классификации

This image is CC0 1.0 public domain

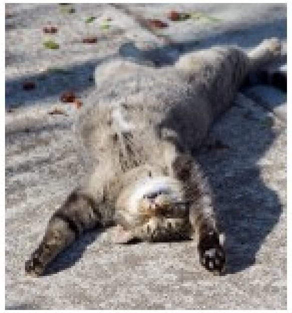
This image is CC0 1.0 public domain

This image is CC0 1.0 public domain



This image is CC0 1.0 public domain

This image by Umberto Salvagnin
is licensed under CC-BY 2.0



This image by Umberto Salvagnin
is licensed under CC-BY 2.0

This image by sare bear is
licensed under CC-BY 2.0

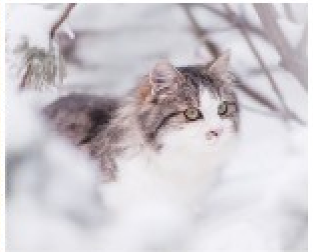
This image by Tom Thai is
licensed under CC-BY 2.0

This image is CC0 1.0 public domain

This image is CC0 1.0 public domain

This image by jonsson is licensed
under CC-BY 2.0

This image is CC0 1.0 public domain

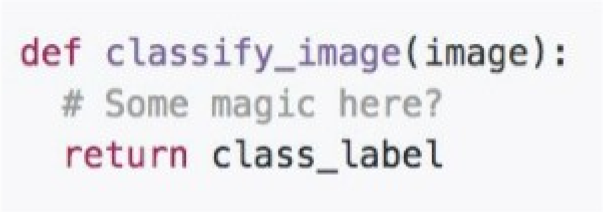


This image is CC0 1.0 public domain

This image is CC0 1.0 public domain

Линейный классификатор

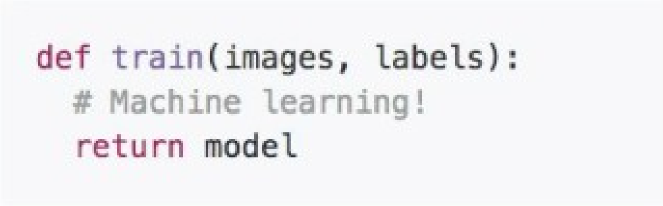
Naïve image classifier



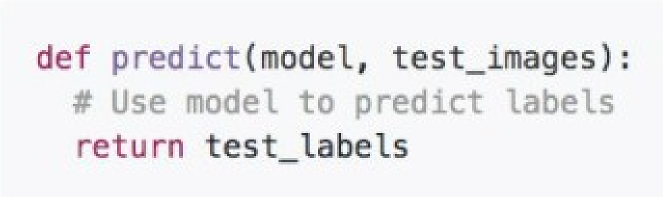
```
def classify_image(image):
  # Some magic here?
  return class_label
```

Unlike sorting, no obvious way to hard-cod the algorithm for recognizing a cat or other classes

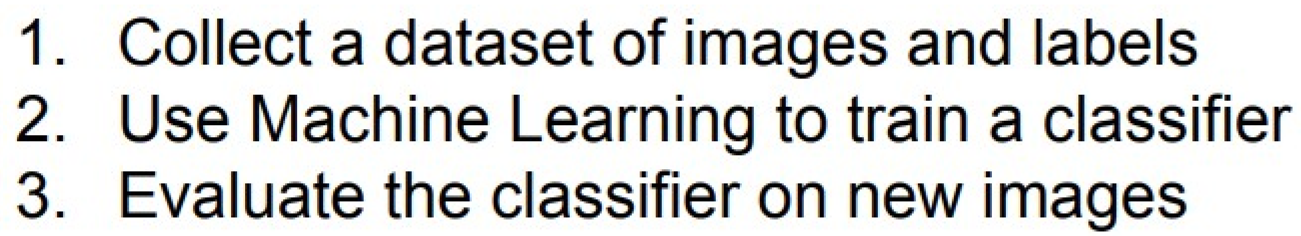
Data driven approach



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



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

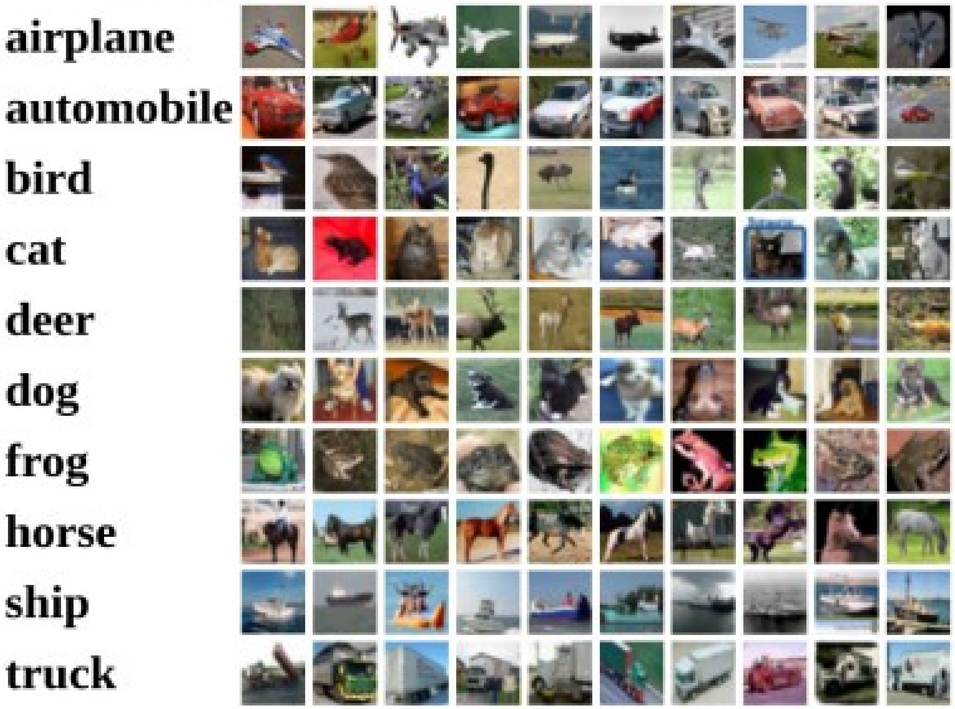


1. Collect a dataset of images and labels
2. Use Machine Learning to train a classifier
3. Evaluate the classifier on new images

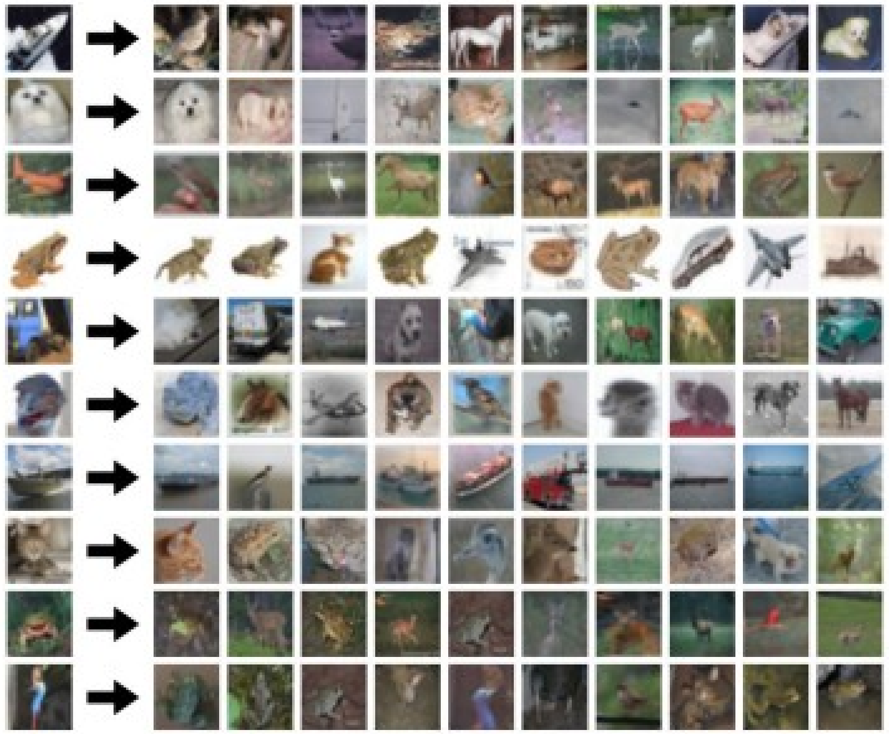
Линейный классификатор

Example Dataset: **CIFAR10**

10 classes
50,000 training images
10,000 testing images

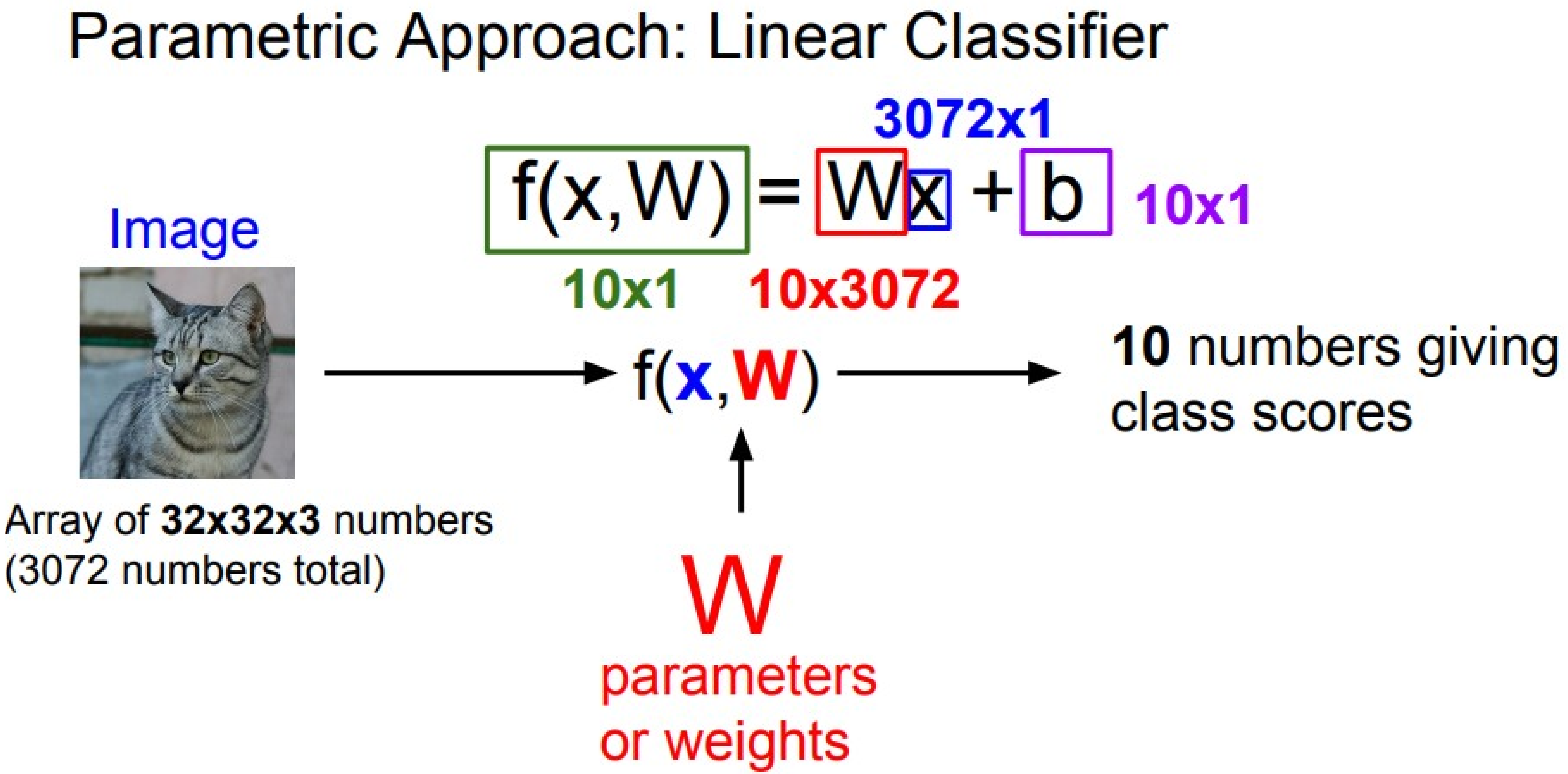


Test images and nearest neighbors



Alex Krizhevsky, "Learning Multiple Layers of Features from Tiny Images", Technical Report, 2009.

Линейный классификатор



Parametric Approach: Linear Classifier

$$\underbrace{\boxed{f(x,W)}}_{10x1} = \underbrace{\boxed{W}}_{10x3072}\overset{3072x1}{\boxed{x}} + \boxed{b}\quad 10x1$$

Image

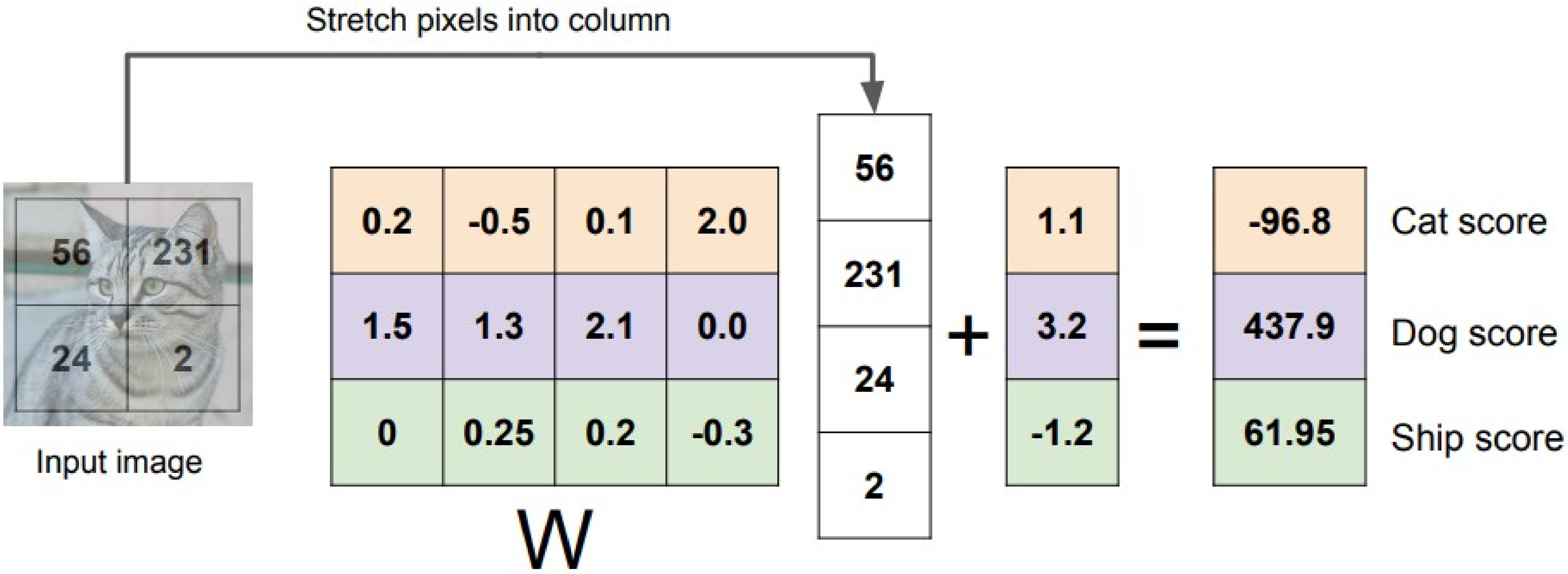
Array of **32x32x3** numbers
(3072 numbers total)

$\xrightarrow{\quad} f(x,W) \xrightarrow{\quad}$

10 numbers giving
class scores

W
parameters
or weights

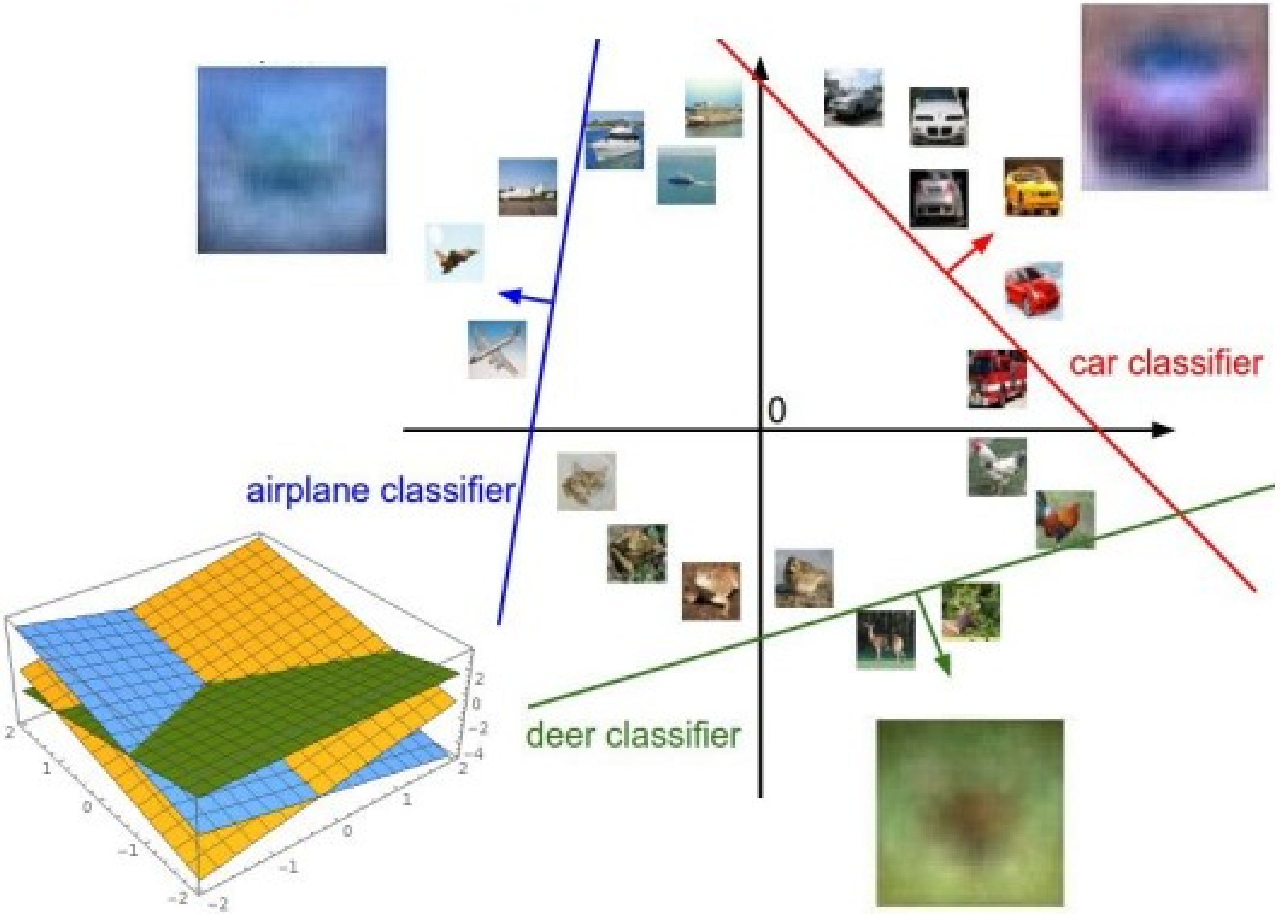
Линейный классификатор



$$\begin{bmatrix} 0.2 & -0.5 & 0.1 & 2.0 \\ 1.5 & 1.3 & 2.1 & 0.0 \\ 0 & 0.25 & 0.2 & -0.3 \end{bmatrix} \begin{bmatrix} 56 \\ 231 \\ 24 \\ 2 \end{bmatrix} + \begin{bmatrix} 1.1 \\ 3.2 \\ -1.2 \end{bmatrix} = \begin{bmatrix} -96.8 \\ 437.9 \\ 61.95 \end{bmatrix}$$

Линейный классификатор

Interpreting a Linear Classifier



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

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

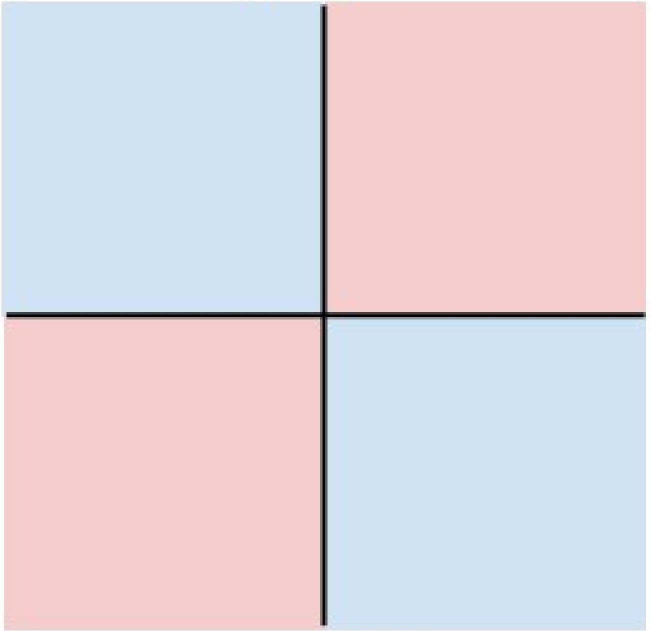
Plot created using Wolfram Cloud

Cat image by Nikita is licensed under CC-BY 2.0

Hard cases for a linear classifier

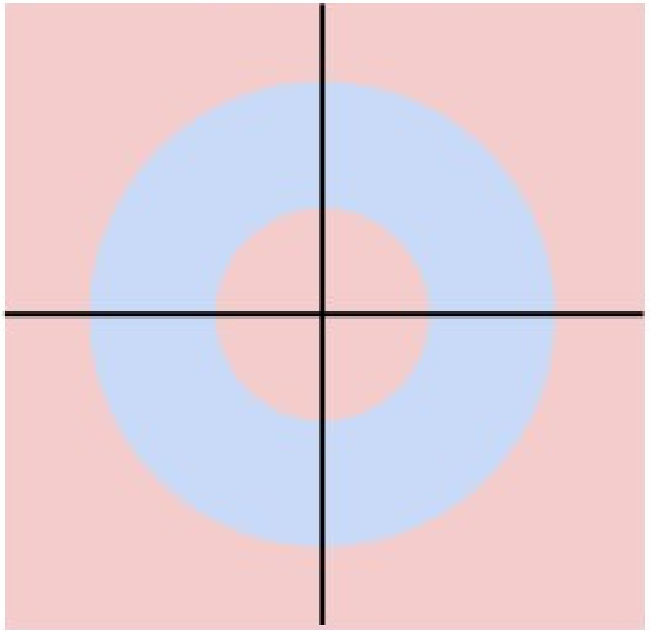
Class 1:
number of pixels > 0 odd

Class 2:
number of pixels > 0 even



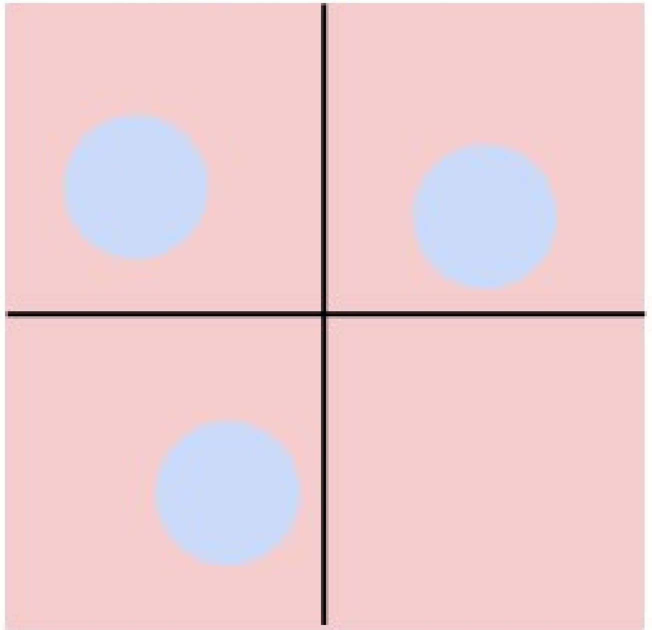
Class 1:
1 <= L2 norm <= 2

Class 2:
Everything else

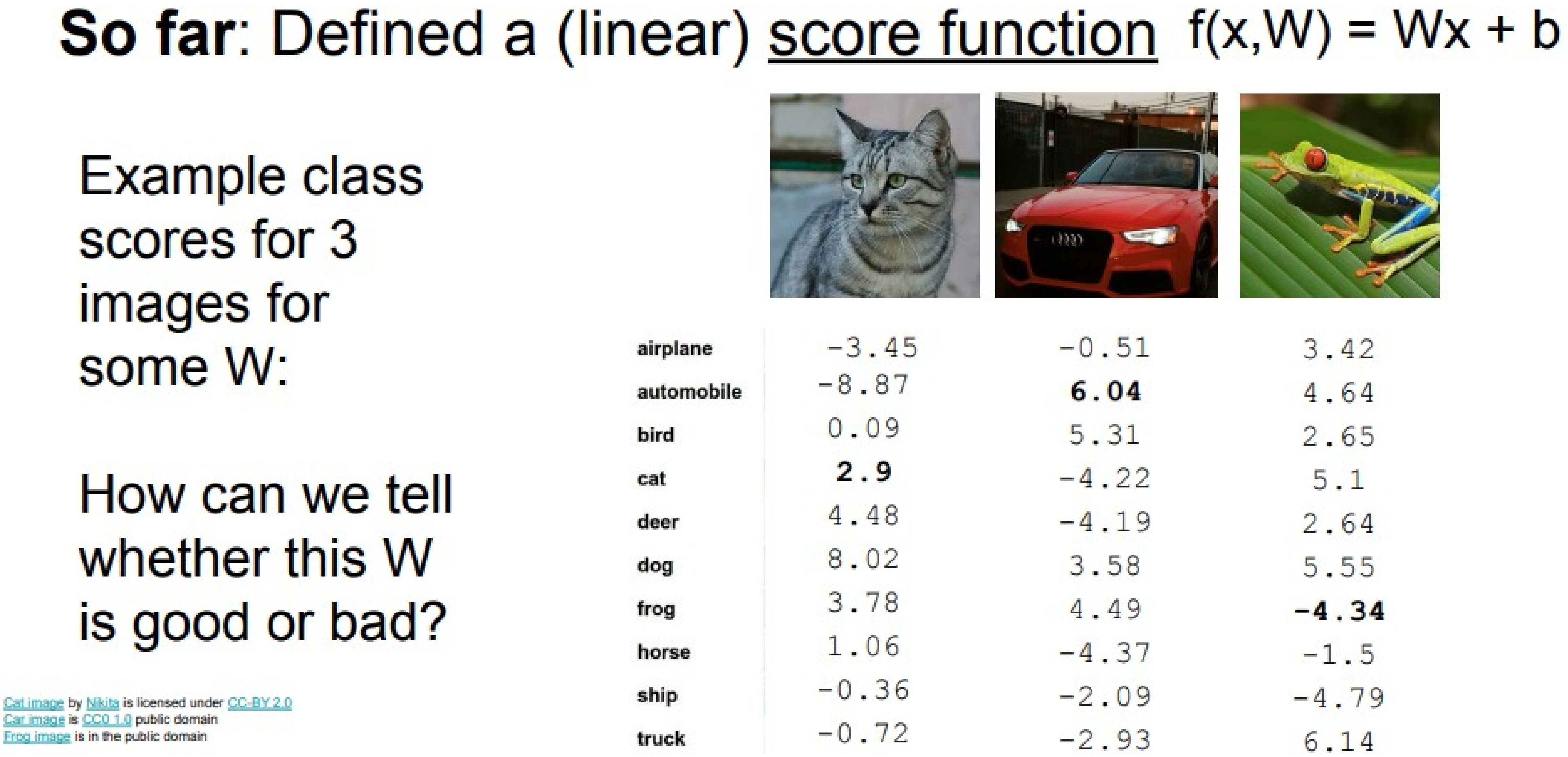


Class 1:
Three modes

Class 2:
Everything else



Функция потерь



So far: Defined a (linear) score function $f(x,W) = Wx + b$

Example class scores for 3 images for some W:

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

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

Cat image by Nikita is licensed under CC-BY 2.0
Car image is CC0 1.0 public domain
Frog image is in the public domain

Функция потерь

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**

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

Given a dataset of examples

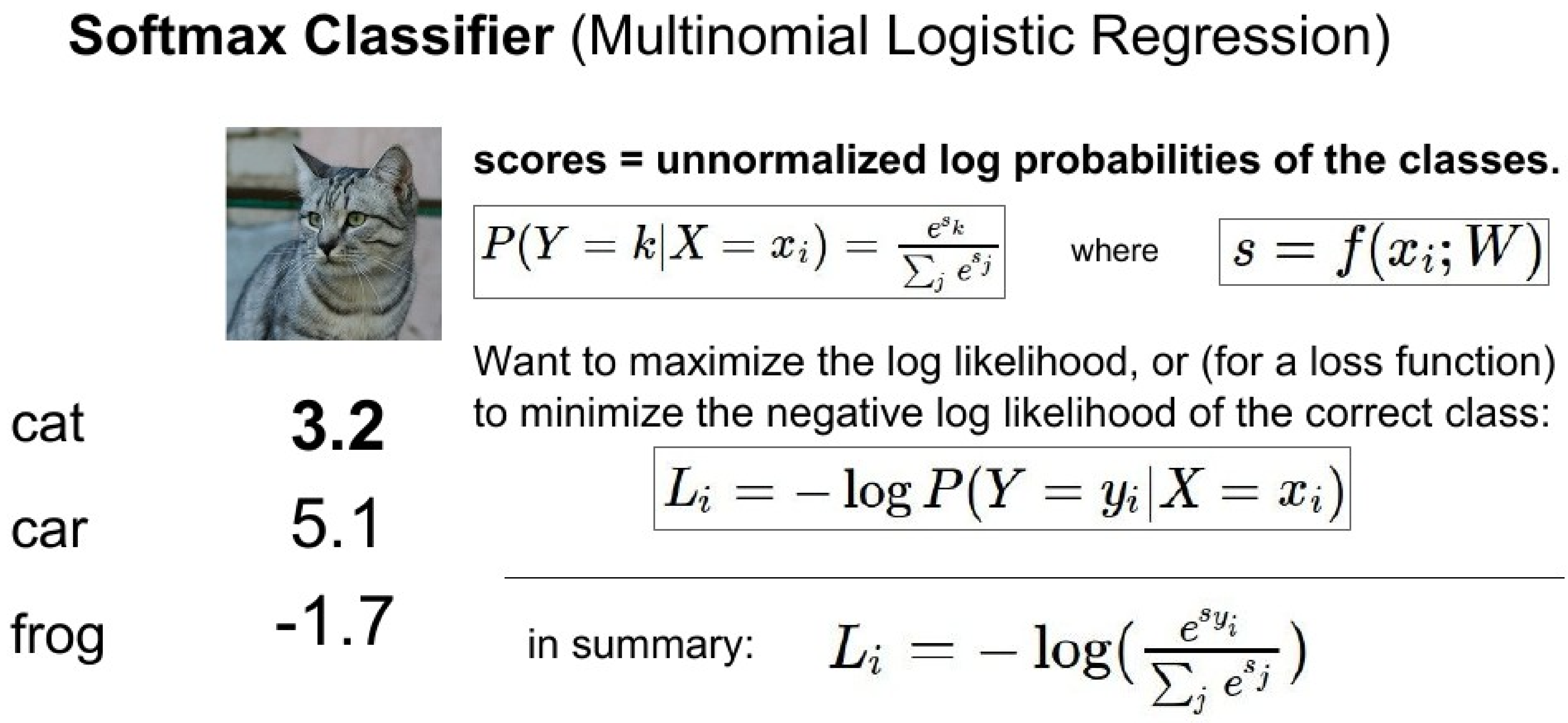
$$\{(x_i, y_i)\}_{i=1}^{N}$$

Where x_i is image and y_i is (integer) label

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

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

Преобразование Softmax



Softmax Classifier (Multinomial Logistic Regression)

cat	**3.2**
car	5.1
frog	-1.7

scores = unnormalized log probabilities of the classes.

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

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

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

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

Функция потерь Cross Enthropy

$$\begin{array}{ccccccccc}
\text{X} & & & & \text{Z-scores} & & & \text{y\_prob} & \text{y\_true} \\
\begin{bmatrix}1\\56\\125\\4\\33\\42\\69\\167\\93\\66\end{bmatrix} & @ & \boxed{\quad W \quad} & = & \begin{bmatrix}10\\0\\-6\\25\\12\\16\\0\\-6\\-50\\11\end{bmatrix} & \dfrac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} & = & \begin{bmatrix}0.1\\0\\0\\0.5\\0.1\\0.2\\0\\0\\0\\0.1\end{bmatrix} & \begin{bmatrix}0\\0\\0\\1\\0\\0\\0\\0\\0\\0\end{bmatrix}
\end{array}$$

Функция потерь Cross Enthropy

y_prob	y_true
0.1	0
0	0
0	0
0.5	1
0.1	0
0.2	0
0	0
0	0
0	0
0.1	0

$$H(p,q) = -\sum_{x \in \mathcal{X}} p(x)\, \log q(x)$$

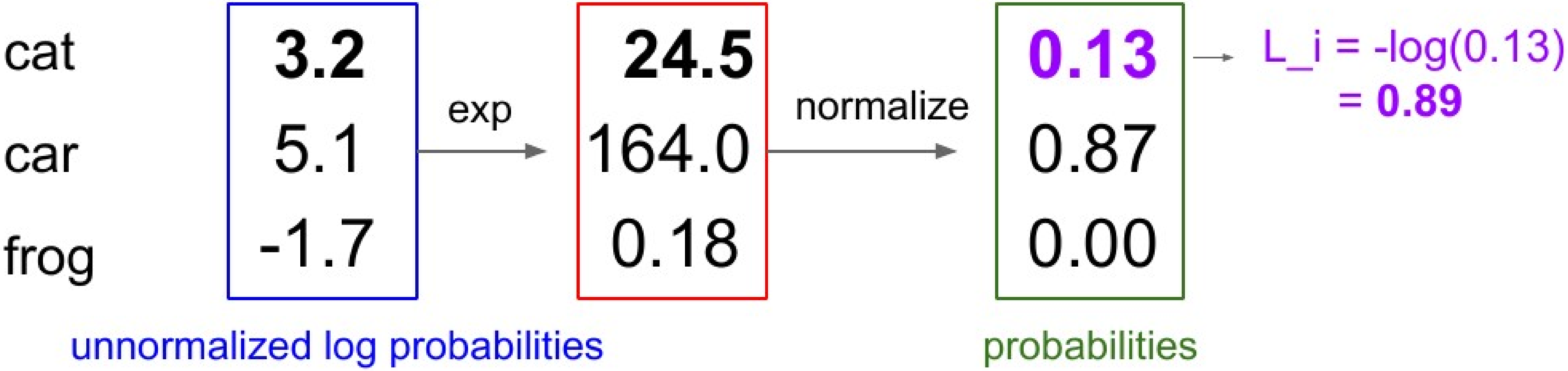
$$L_i = -\log\left(\frac{e^{s_{y_i}}}{\sum_j e^{s_j}}\right)$$

Функция потерь Cross Enthropy

Softmax Classifier (Multinomial Logistic Regression)

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

unnormalized probabilities



cat 3.2 $\xrightarrow{\text{exp}}$ 24.5 $\xrightarrow{\text{normalize}}$ 0.13 → L_i = -log(0.13) = 0.89

car 5.1 → 164.0 → 0.87

frog -1.7 → 0.18 → 0.00

unnormalized log probabilities

probabilities