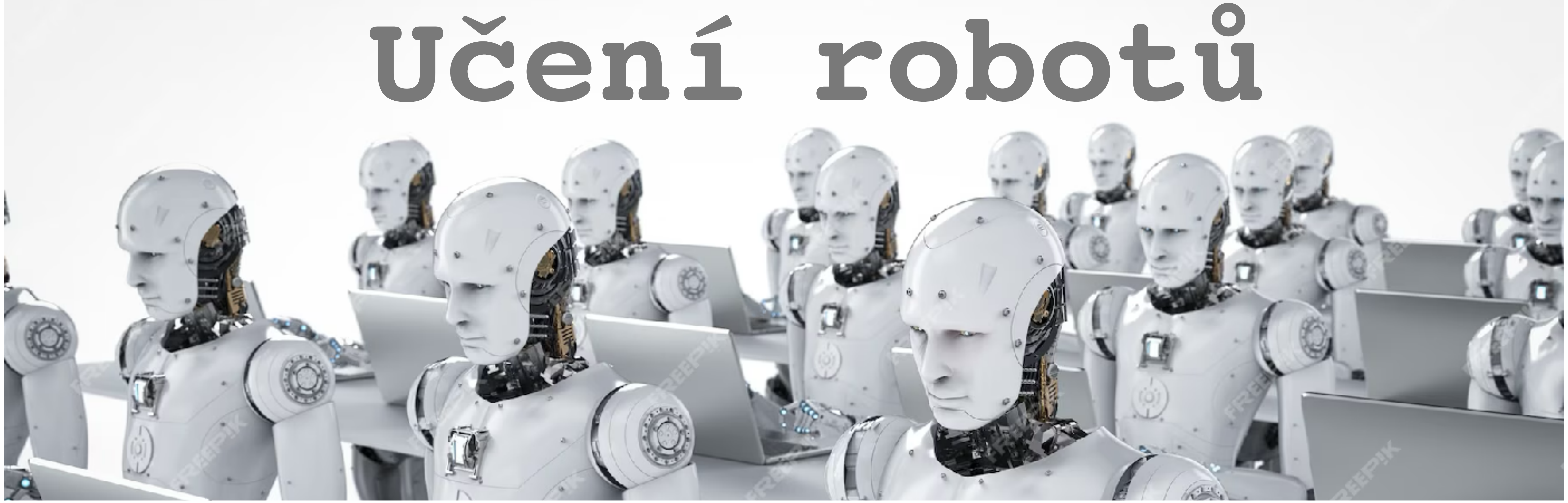


Under the hood of linear classifier

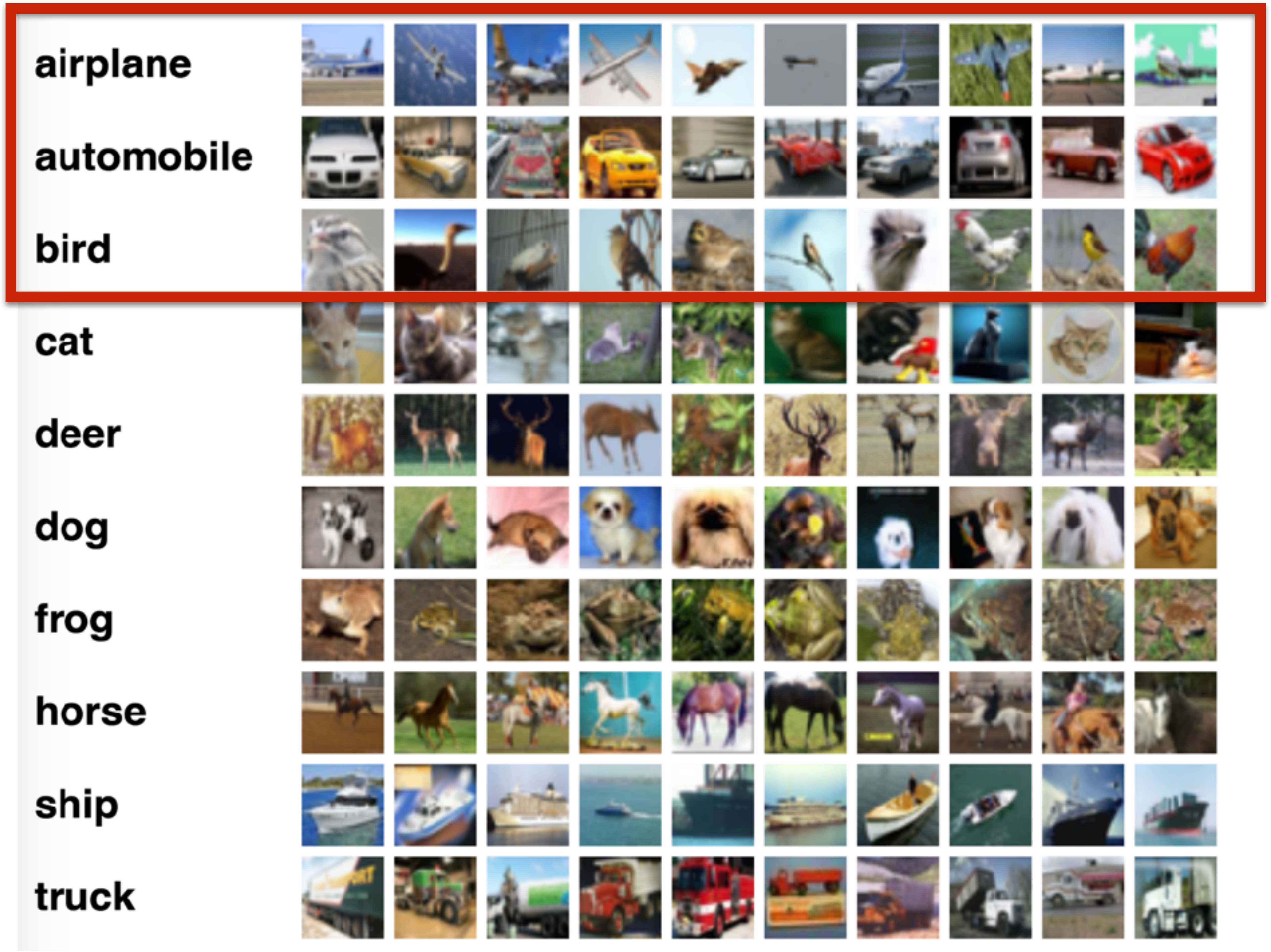
Linear classifier on RGB images

Pre-requisites: linear algebra,

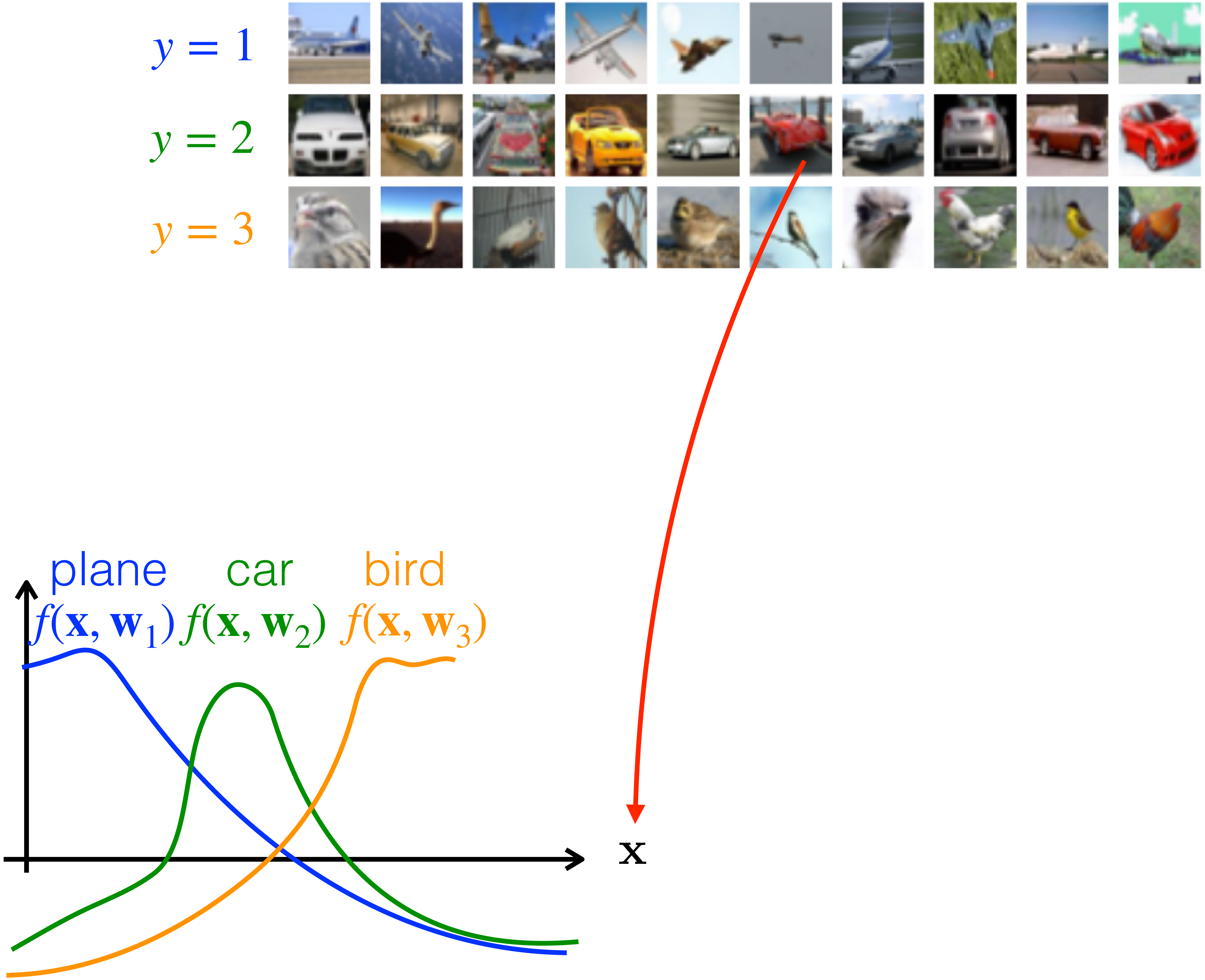
Karel Zimmermann
Czech Technical University in Prague
Faculty of Electrical Engineering, Department of Cybernetics

Financováno Evropskou unií NextGenerationEU
Národní plán obnovy
MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY

Recognition problem



CIFAR-10: classify 32x32 RGB images into 10 categories
https://www.cs.toronto.edu/~kriz/cifar.html

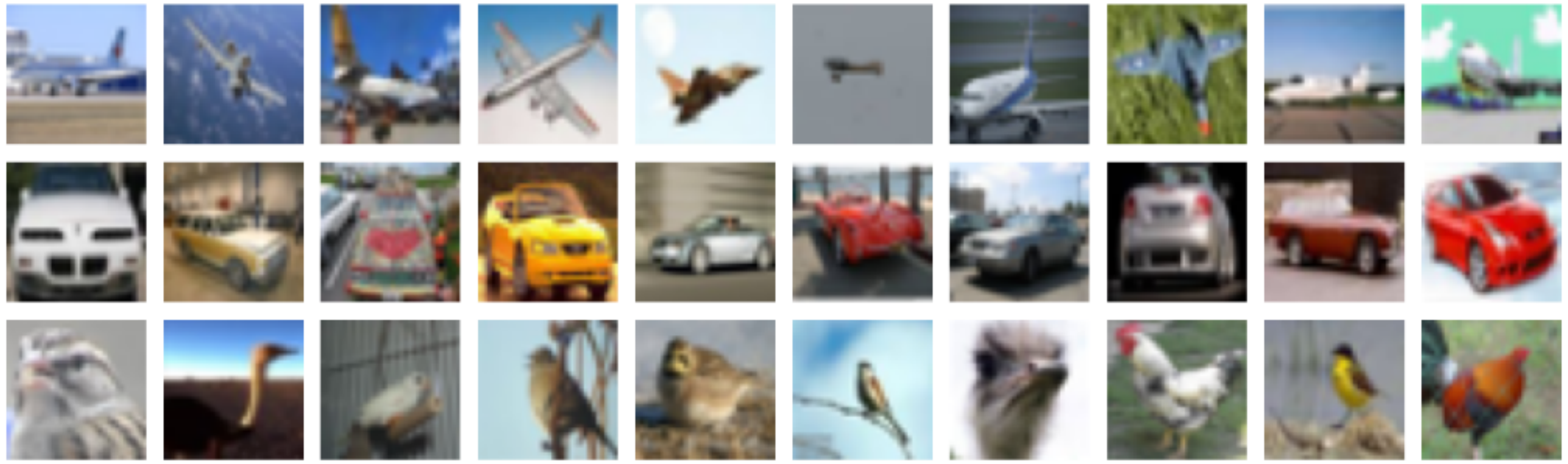


$y = 1$
$y = 2$
$y = 3$
plane
car
bird
$f(\mathbf{x}, \mathbf{w}_1)$
$f(\mathbf{x}, \mathbf{w}_2)$
$f(\mathbf{x}, \mathbf{w}_3)$
$\mathbf{x}$

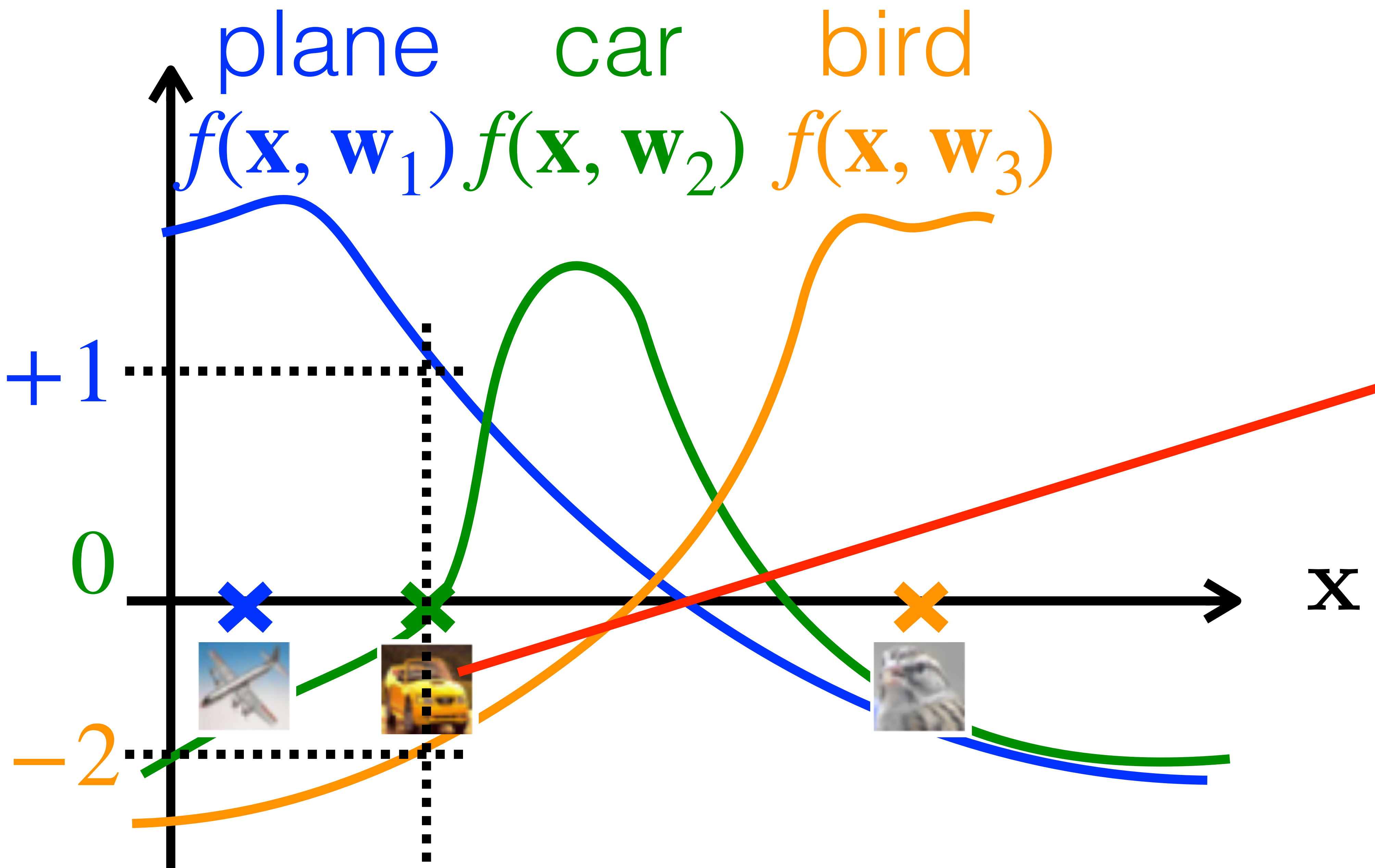
$y = 1$

$y = 2$

$y = 3$



$$p(y \mid \mathbf{x}, \mathbf{W}) = \text{???}$$



Linear classifier

$\mathbf{x} = \text{vec}($  $)$

logits

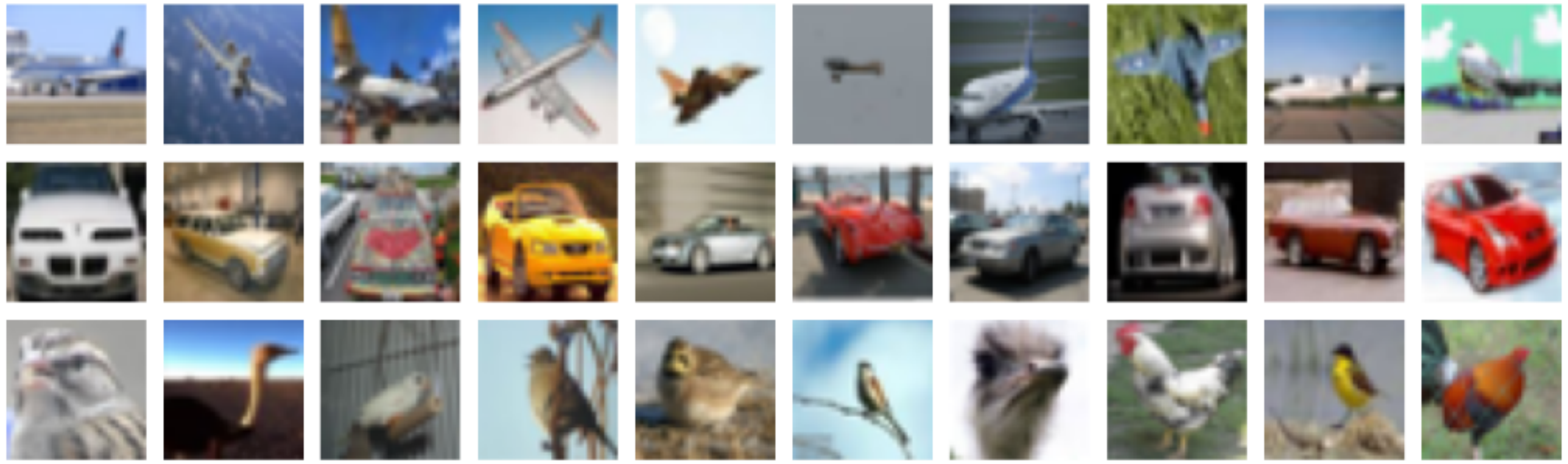
$$\mathbf{f}(\mathbf{x}, \mathbf{W}) = \boxed{\quad \mathbf{W} \quad} \; \boxed{\bar{\mathbf{x}}} = \begin{bmatrix} +1 \\ 0 \\ -2 \end{bmatrix}$$

How does the linear classifier work?

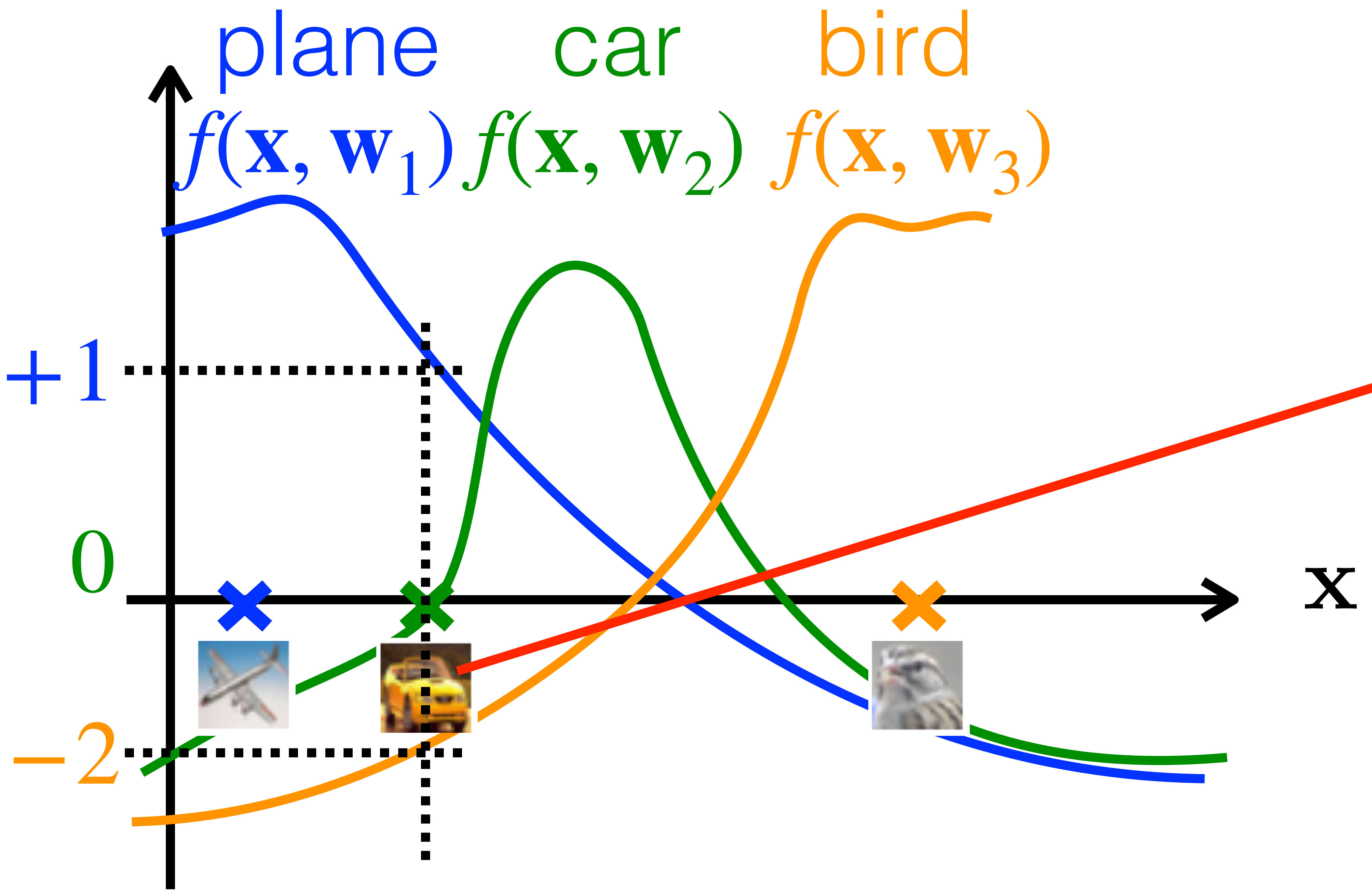
$y = 1$

$y = 2$

$y = 3$



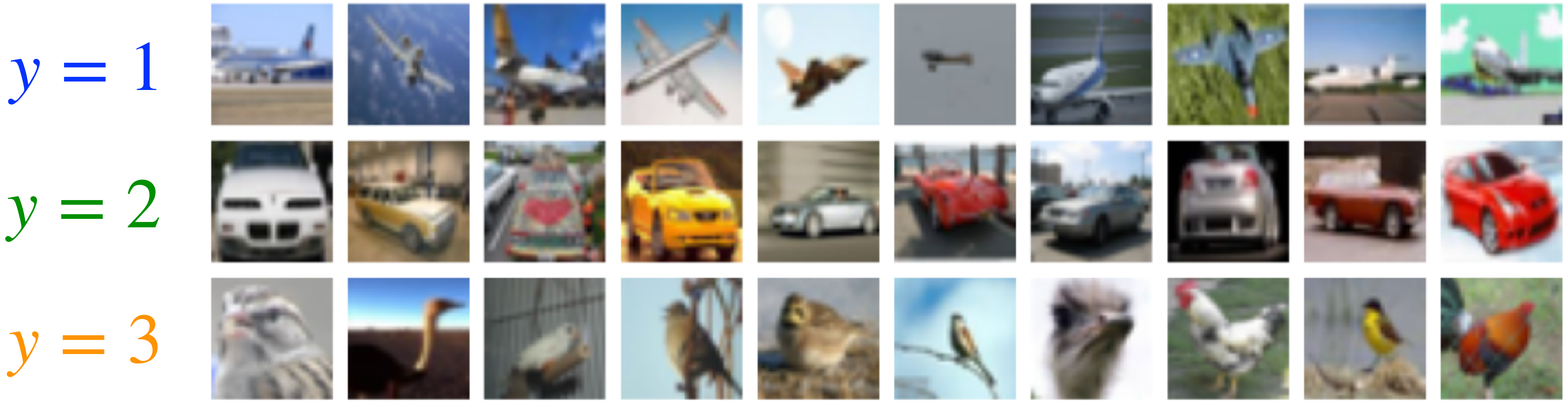
$$p(y\,|\,\mathbf{x}, \mathbf{W}) = \begin{bmatrix} \exp(f(\mathbf{x}, \mathbf{w}_1)) \\ \exp(f(\mathbf{x}, \mathbf{w}_2)) \\ \exp(f(\mathbf{x}, \mathbf{w}_3)) \end{bmatrix} \;/\; \sum_k \exp(f(\mathbf{x}, \mathbf{w}_k)) = \mathbf{s}(\mathbf{f}(\mathbf{x}, \mathbf{W})) = \mathbf{s}\left(\begin{bmatrix} +1 \\ 0 \\ -2 \end{bmatrix} \right) = \begin{bmatrix} 0.71 \\ 0.26 \\ 0.03 \end{bmatrix}$$



Linear classifier

$\mathbf{x} = \mathrm{vec}($  $)$

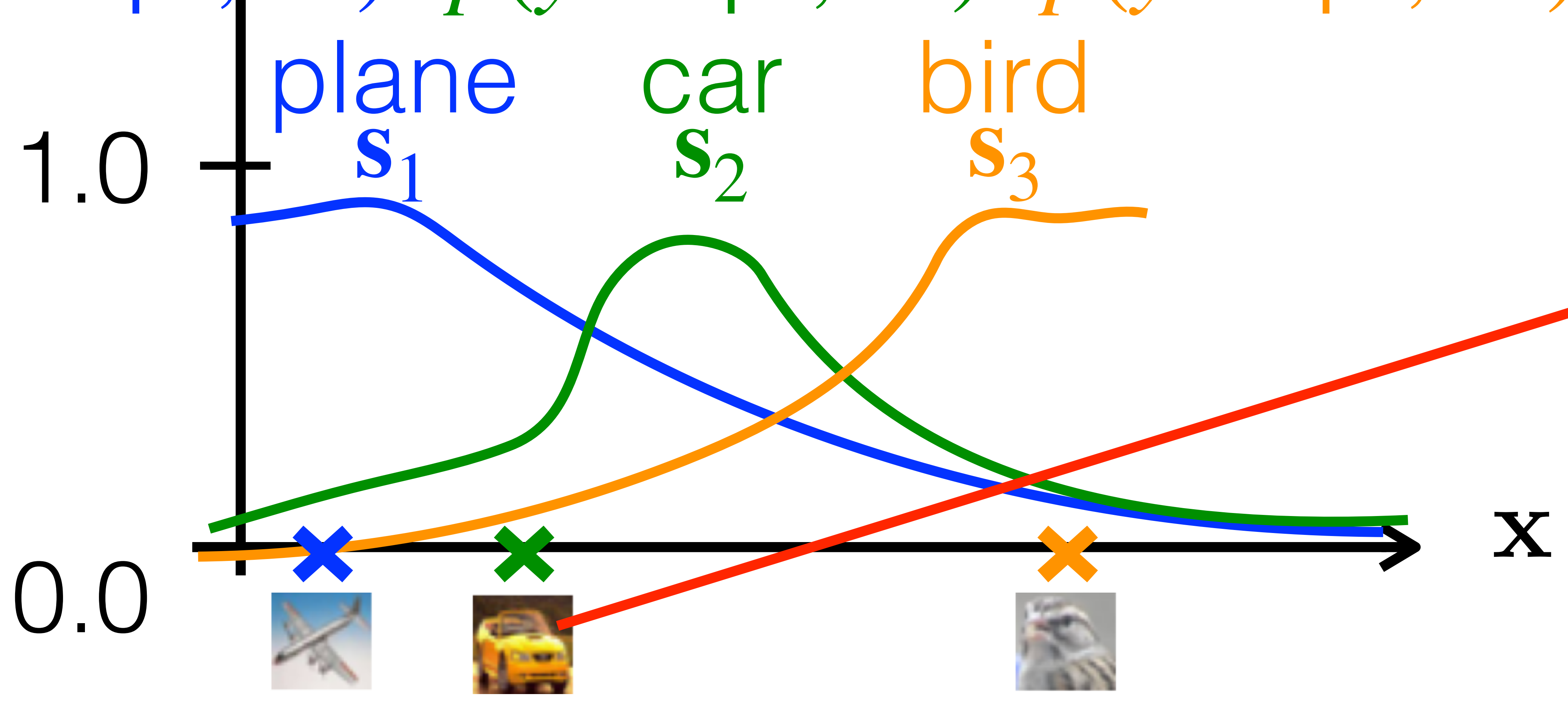
$$\mathbf{f}(\mathbf{x}, \mathbf{W}) = \boxed{\mathbf{W}}\;\boxed{\bar{\mathbf{x}}} = \begin{bmatrix} +1 \\ 0 \\ -2 \end{bmatrix}$$



Model probability distribution over classes by softmax function

$$\mathbf{p}(y \,|\, \mathbf{x}, \mathbf{W}) = \begin{bmatrix} \exp(f(\mathbf{x}, \mathbf{w}_1)) \\ \exp(f(\mathbf{x}, \mathbf{w}_2)) \\ \exp(f(\mathbf{x}, \mathbf{w}_3)) \end{bmatrix} \Big/ \sum_k \exp\big(f\big(\mathbf{x}, \mathbf{w}_k\big)\big) = \mathbf{s}\big(\mathbf{f}\big(\mathbf{x}, \mathbf{W}\big)\big) = \mathbf{s}\left(\begin{bmatrix} +1 \\ 0 \\ -2 \end{bmatrix}\right) = \begin{bmatrix} 0.71 \\ 0.26 \\ 0.03 \end{bmatrix}$$

$p(y{=}1 \,|\, \mathbf{x}, \mathbf{W})$ $p(y{=}2 \,|\, \mathbf{x}, \mathbf{W})$ $p(y{=}3 \,|\, \mathbf{x}, \mathbf{W})$



Linear classifier

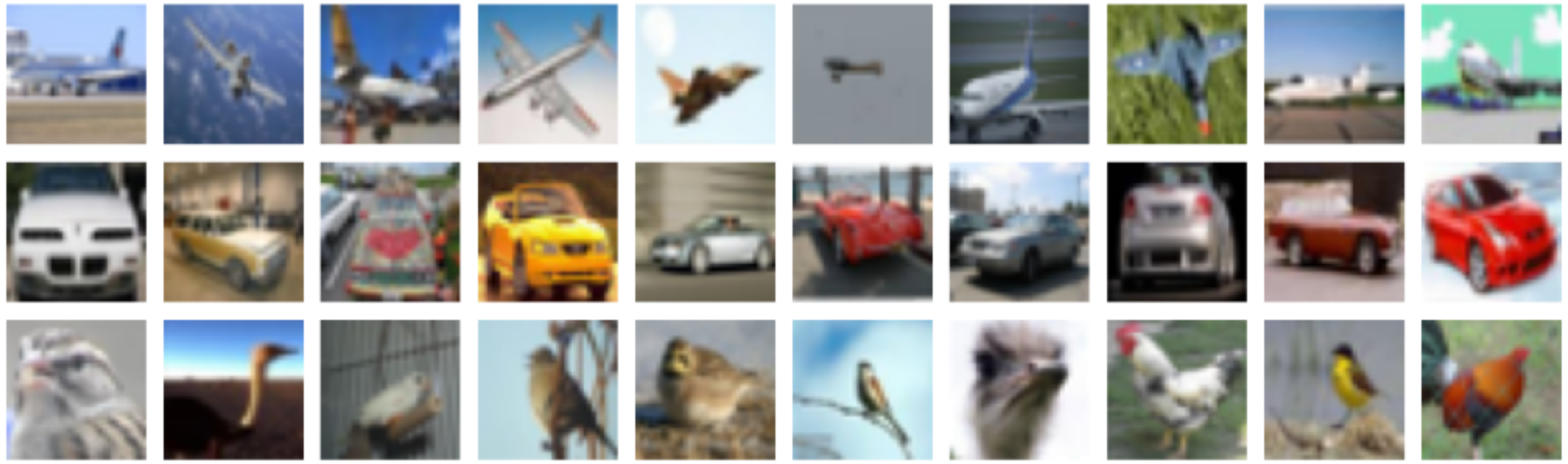
$\mathbf{x} = \mathrm{vec}($  $)$

$$\mathbf{f}(\mathbf{x}, \mathbf{W}) = \boxed{\quad \mathbf{W} \quad} \, \boxed{\bar{\mathbf{x}}} = \begin{bmatrix} +1 \\ 0 \\ -2 \end{bmatrix}$$

$y = 1$

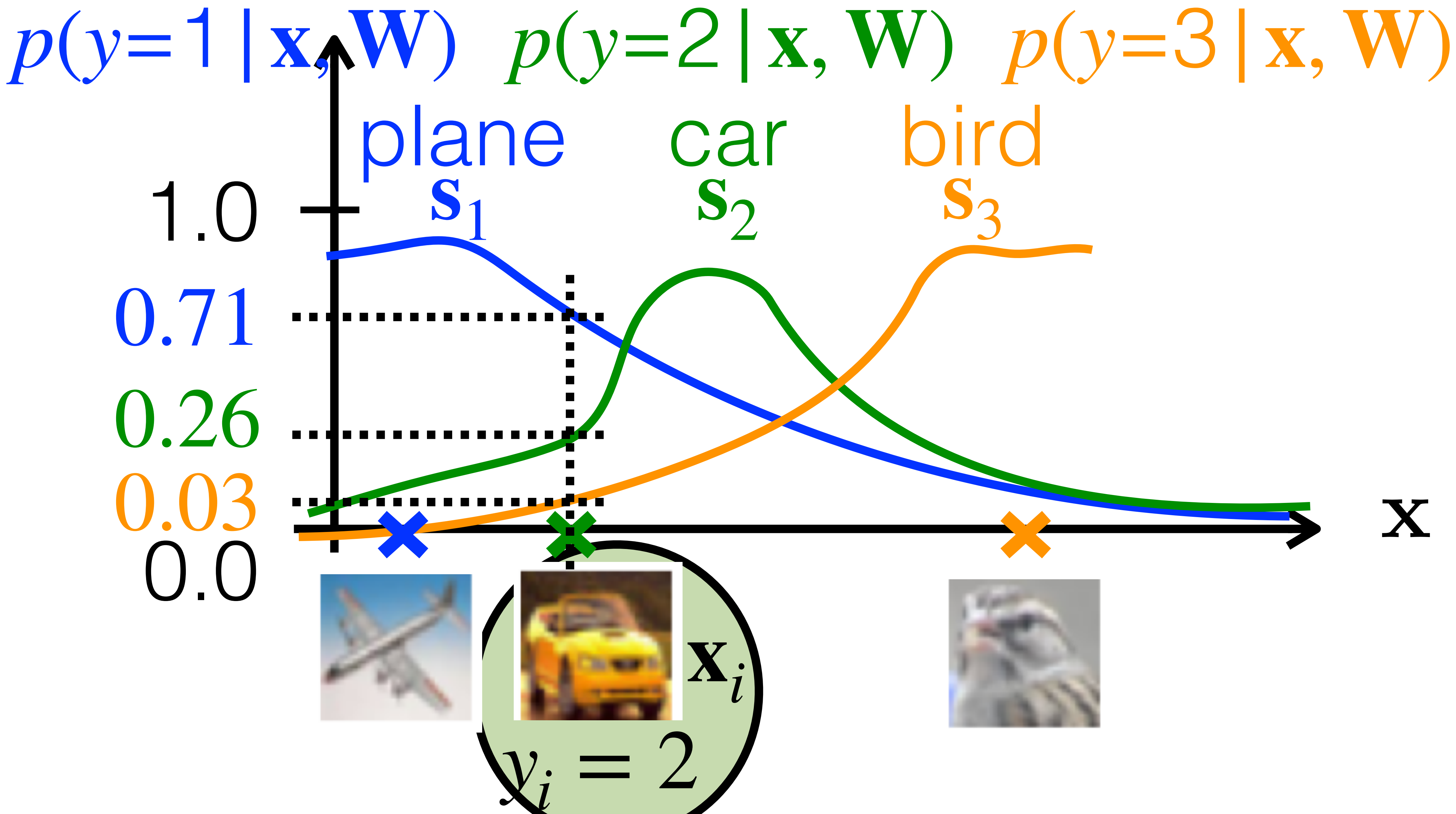
$y = 2$

$y = 3$



Model probability distribution over classes by softmax function

$$\mathbf{p}(y \mid \mathbf{x}, \mathbf{W}) = \begin{bmatrix} \exp(f(\mathbf{x}, \mathbf{w}_1)) \\ \exp(f(\mathbf{x}, \mathbf{w}_2)) \\ \exp(f(\mathbf{x}, \mathbf{w}_3)) \end{bmatrix} / \sum_k \exp(f(\mathbf{x}, \mathbf{w}_k)) = \mathbf{s}(\mathbf{f}(\mathbf{x}, \mathbf{W})) = \mathbf{s}\left(\begin{bmatrix} +1 \\ 0 \\ -2 \end{bmatrix} \right) = \begin{bmatrix} 0.71 \\ 0.26 \\ 0.03 \end{bmatrix}$$

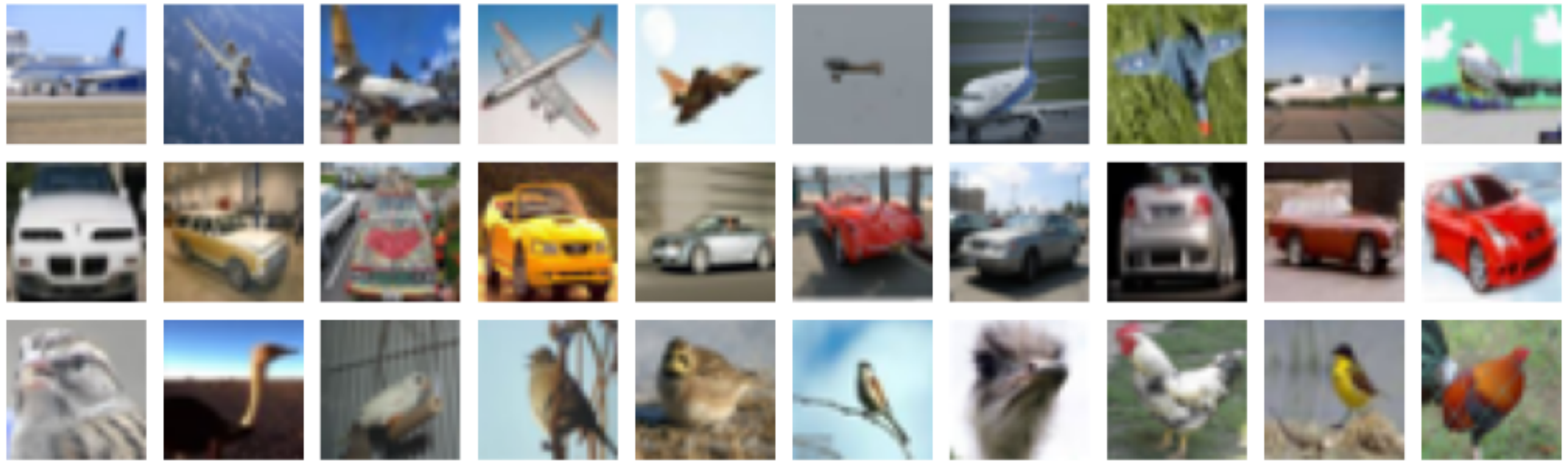


Which image is incorrectly classified?

$y = 1$

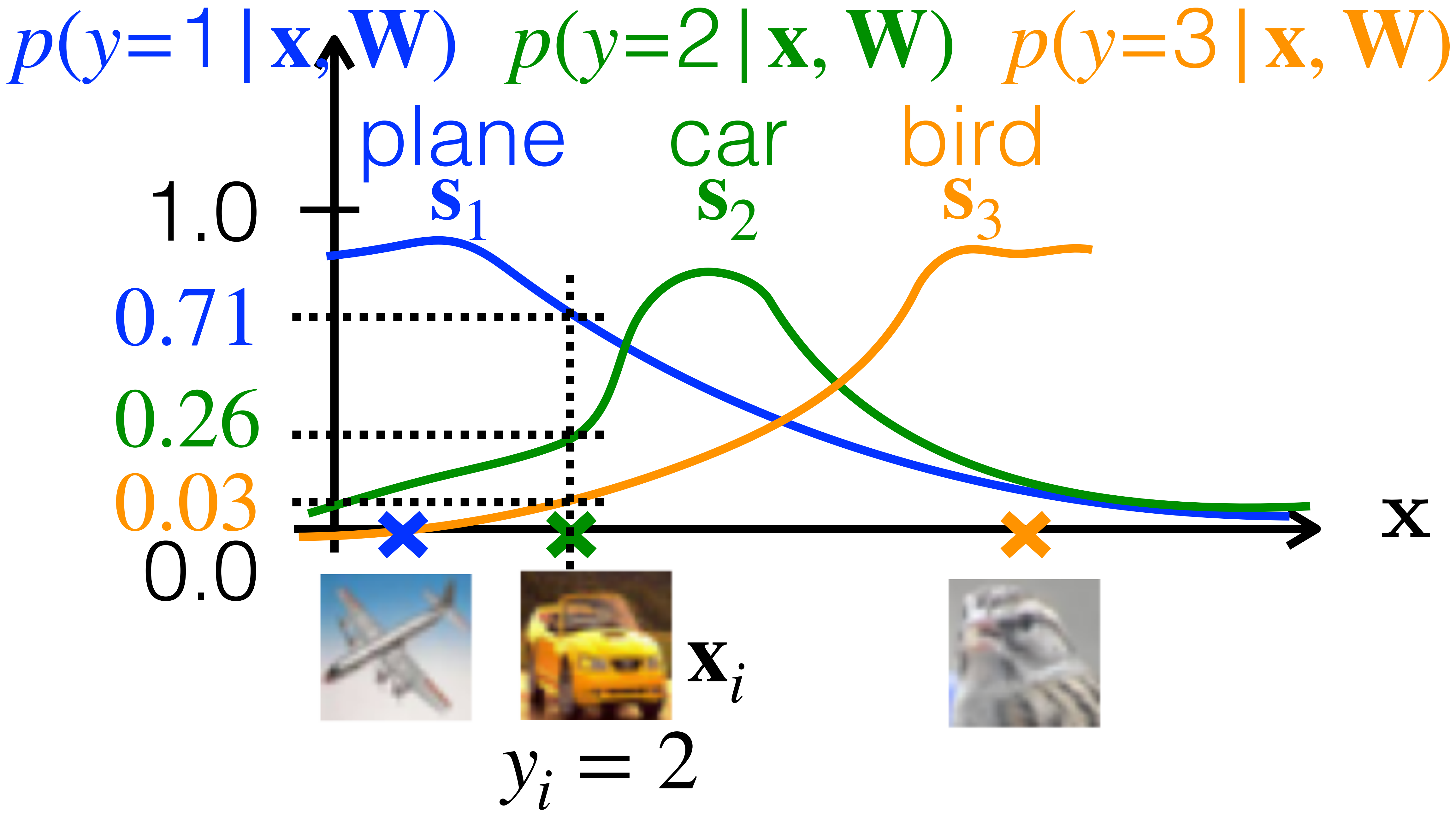
$y = 2$

$y = 3$



Model probability distribution over classes by softmax function

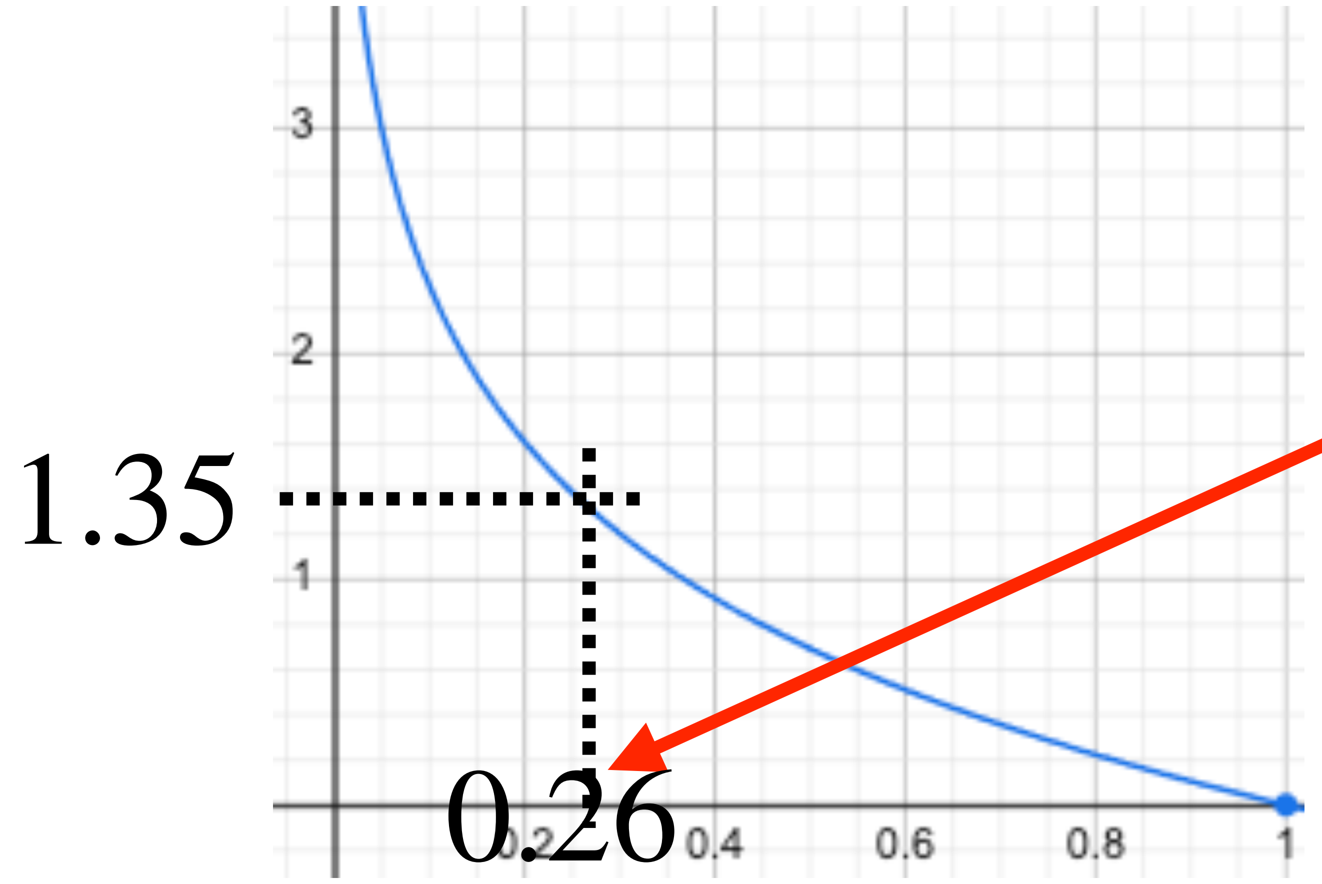
$$\mathbf{p}(y \mid \mathbf{x}, \mathbf{W}) = \begin{bmatrix} \exp(f(\mathbf{x}, \mathbf{w}_1)) \\ \exp(f(\mathbf{x}, \mathbf{w}_2)) \\ \exp(f(\mathbf{x}, \mathbf{w}_3)) \end{bmatrix} / \sum_k \exp(f(\mathbf{x}, \mathbf{w}_k)) = \mathbf{s}(\mathbf{f}(\mathbf{x}, \mathbf{W})) = \mathbf{s}\left(\begin{bmatrix} +1 \\ 0 \\ -2 \end{bmatrix}\right) = \begin{bmatrix} 0.71 \\ \boxed{0.26} \\ 0.03 \end{bmatrix}$$

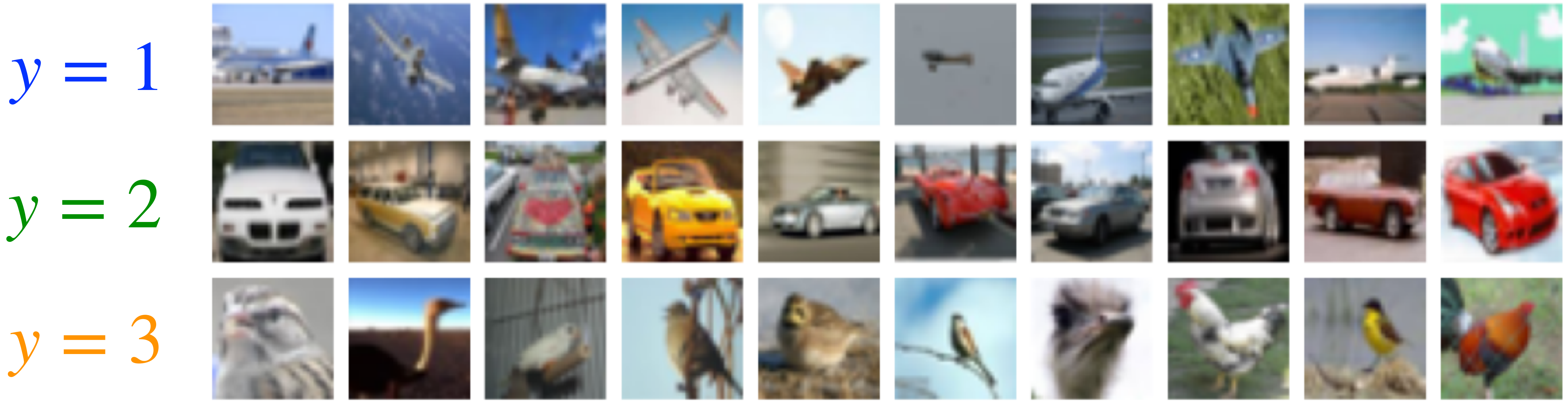


Loss function:

$$\mathcal{L}(\mathbf{W}) = -\log p(y = y_i \mid \mathbf{x}_i, \mathbf{W}) = -\log \mathbf{s}_{y_i}(\mathbf{W}\,\bar{\mathbf{x}}_i)$$

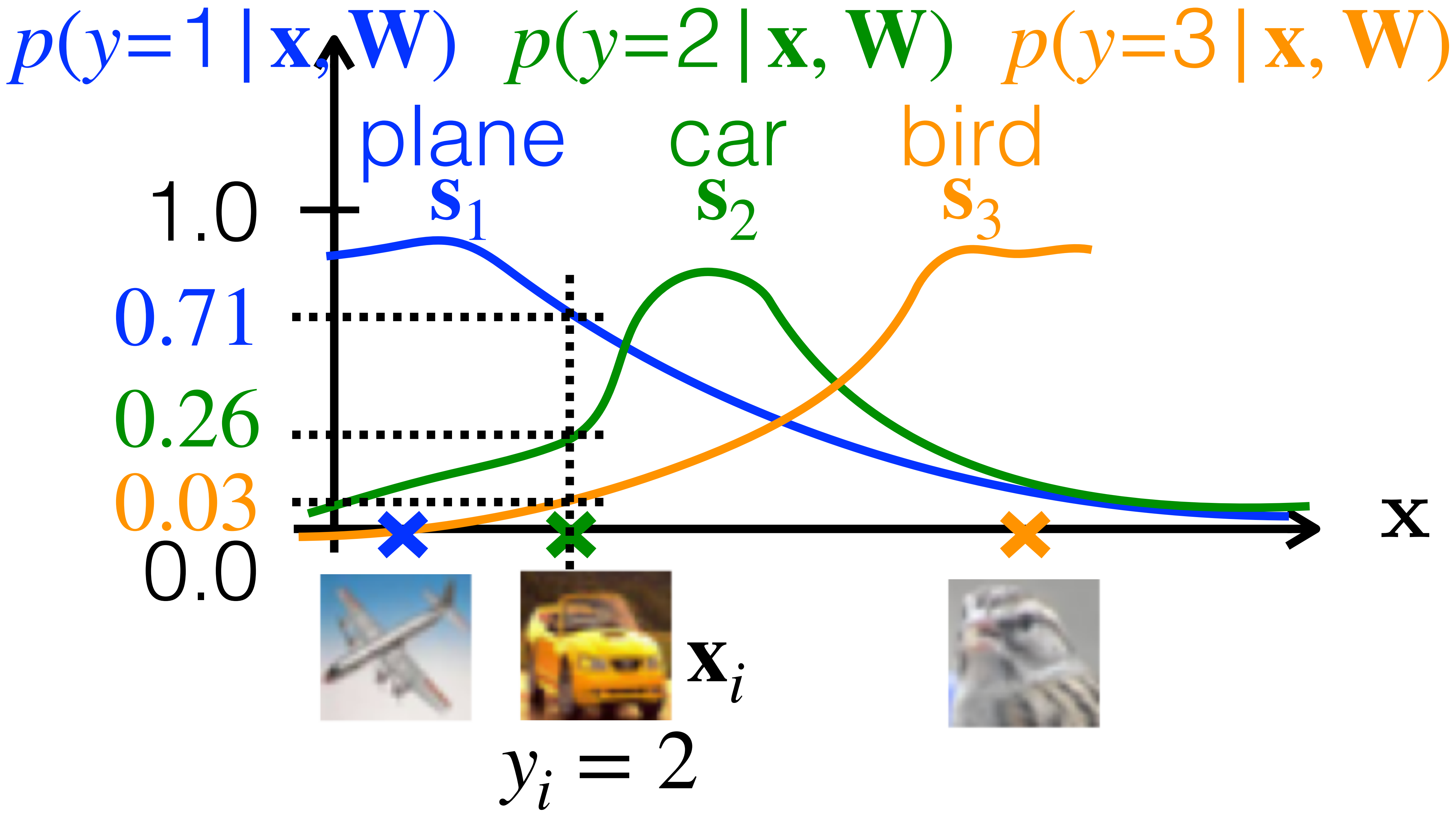
$$= -\log(\boxed{0.26}) = 1.35$$





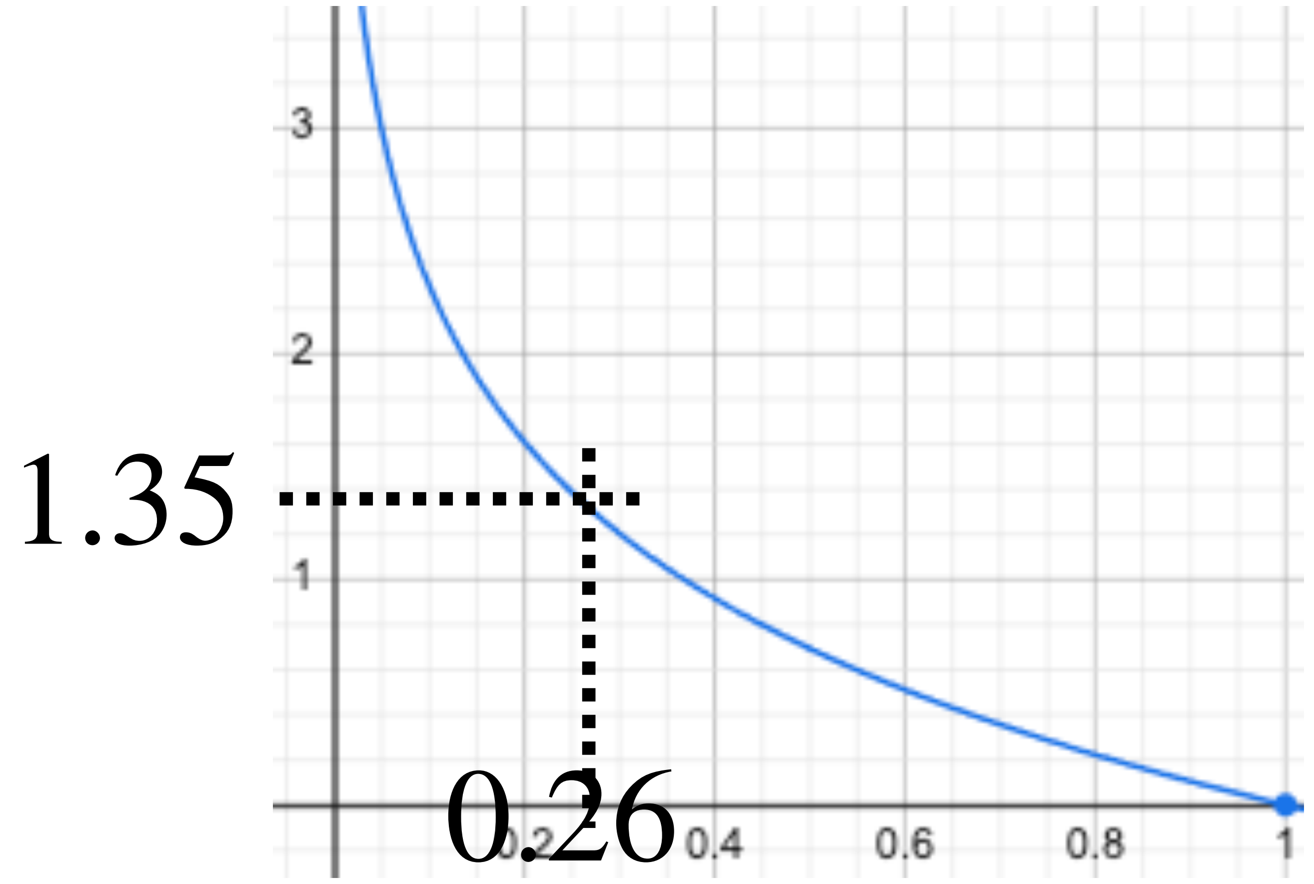
Model probability distribution over classes by softmax function

$$\mathbf{p}(y \mid \mathbf{x}, \mathbf{W}) = \begin{bmatrix} \exp(f(\mathbf{x}, \mathbf{w}_1)) \\ \exp(f(\mathbf{x}, \mathbf{w}_2)) \\ \exp(f(\mathbf{x}, \mathbf{w}_3)) \end{bmatrix} / \sum_k \exp(f\left(\mathbf{x}, \mathbf{w}_k\right) = \mathbf{s}(\mathbf{f}\left(\mathbf{x}, \mathbf{W}\right)) = \mathbf{s}\left(\begin{bmatrix} +1 \\ 0 \\ -2 \end{bmatrix} \right) = \begin{bmatrix} 0.71 \\ 0.26 \\ 0.03 \end{bmatrix}$$



Loss function:

$$\mathscr{L}(\mathbf{W}) = -\log p(y = y_i \mid \mathbf{x}_i, \mathbf{W}) = -\log \mathbf{s}_{y_i}(\mathbf{W}\,\bar{\mathbf{x}}_i)$$



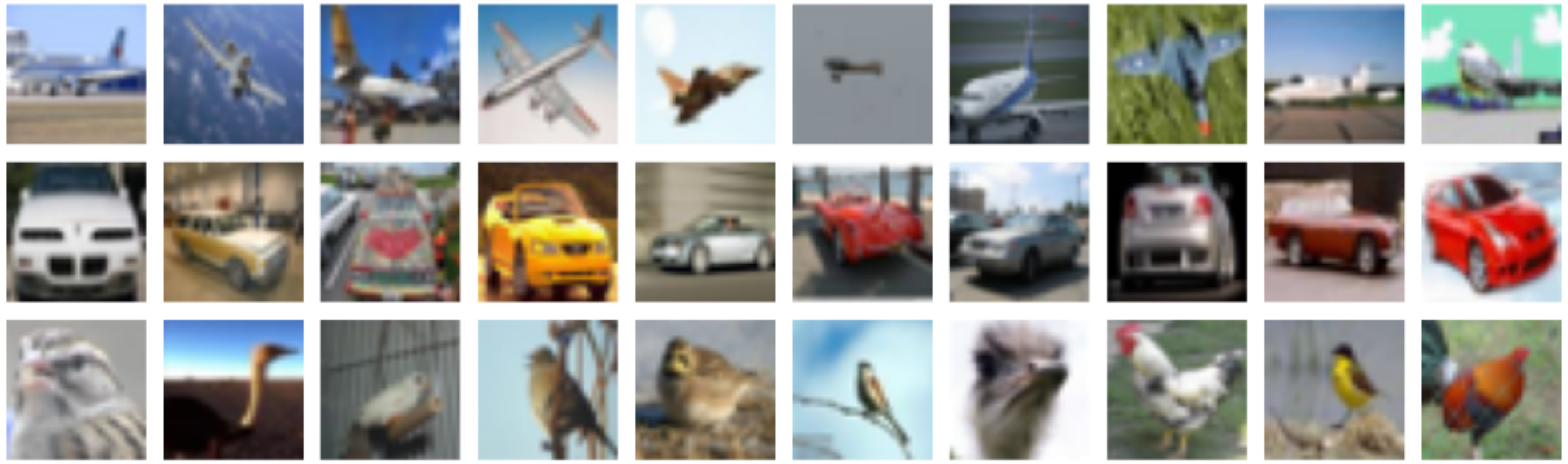
Learning:

$$\arg\min_{\mathbf{W}} \mathscr{L}(\mathbf{W})$$

$y = 1$

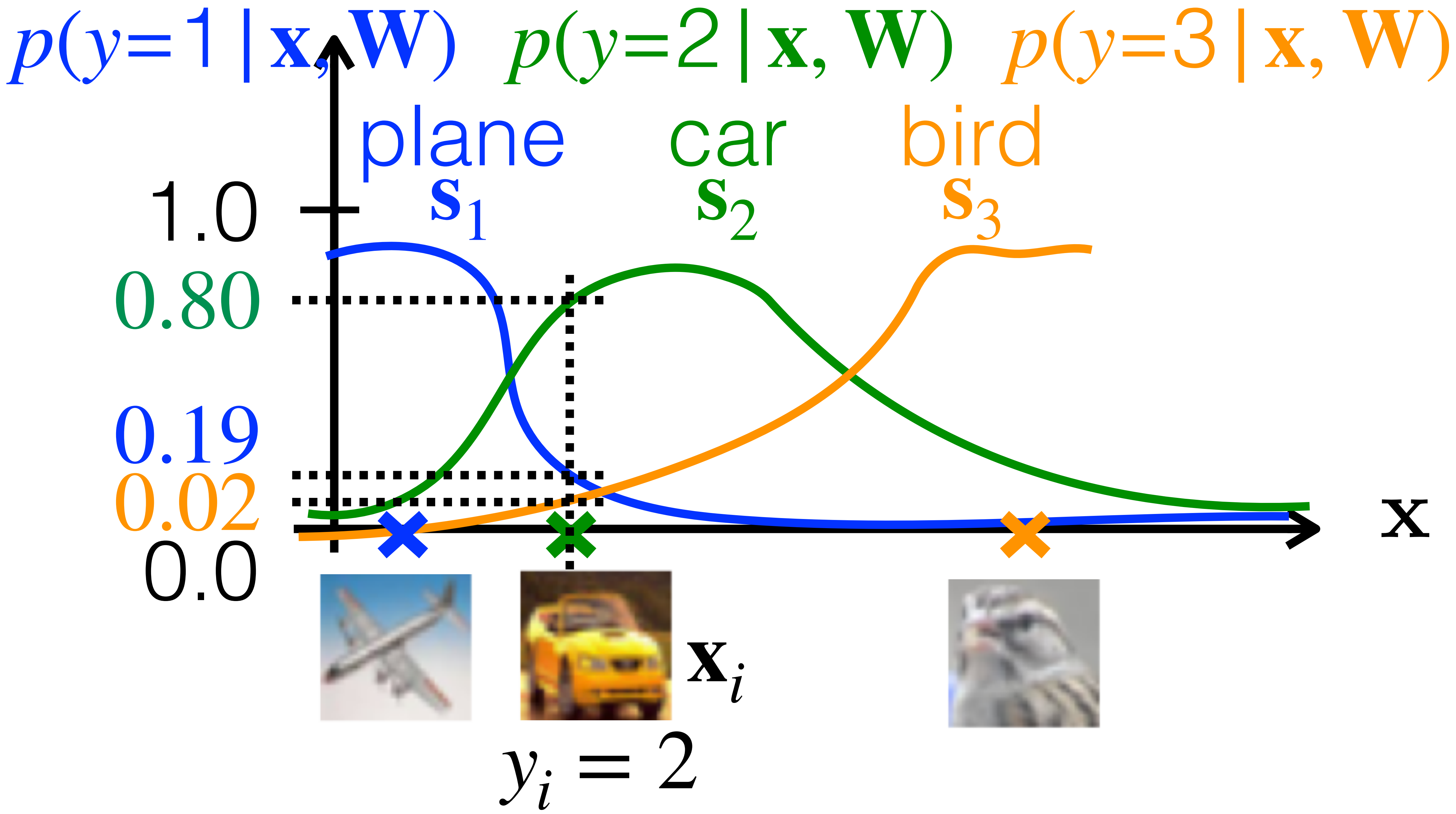
$y = 2$

$y = 3$



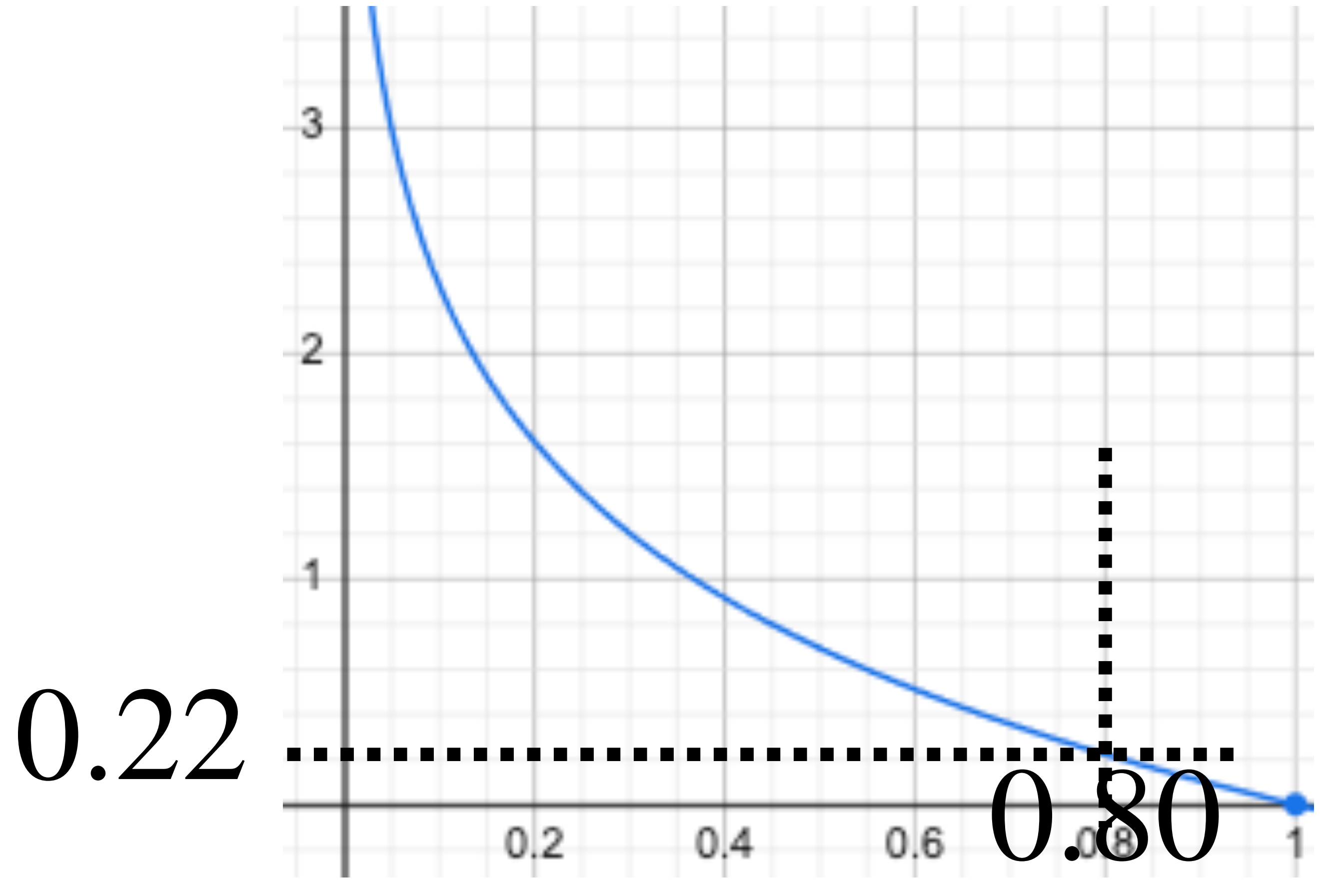
Model probability distribution over classes by softmax function

$$\mathbf{p}(y \,|\, \mathbf{x}, \mathbf{W}) = \begin{bmatrix} \exp(f(\mathbf{x}, \mathbf{w}_1)) \\ \exp(f(\mathbf{x}, \mathbf{w}_2)) \\ \exp(f(\mathbf{x}, \mathbf{w}_3)) \end{bmatrix} \Big/ \sum_k \exp\big(f\big(\mathbf{x}, \mathbf{w}_k\big) = \mathbf{s}\big(\mathbf{f}\big(\mathbf{x}, \mathbf{W}\big)\big) = \mathbf{s}\left(\begin{bmatrix} +1 \\ 0 \\ -2 \end{bmatrix}\right) = \begin{bmatrix} 0.71 \\ 0.26 \\ 0.03 \end{bmatrix}$$



Loss function:

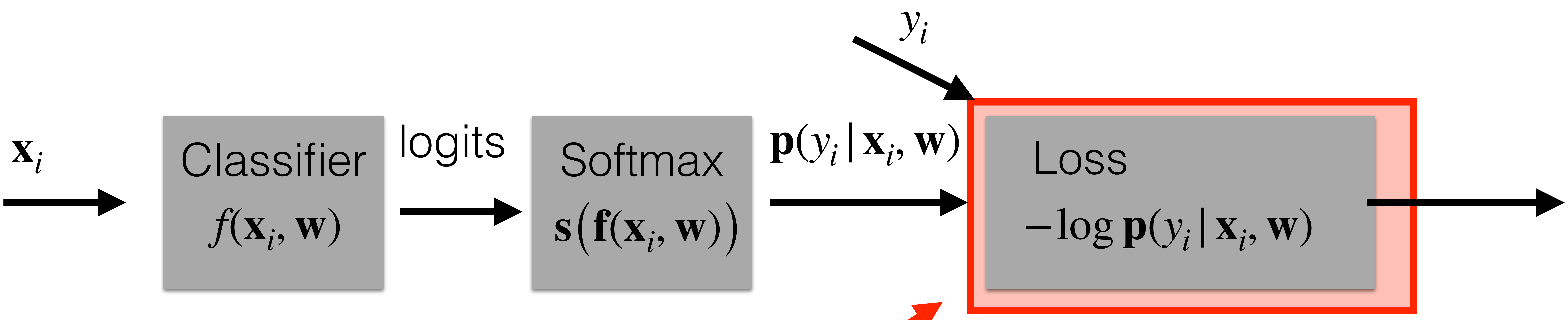
$$\mathscr{L}(\mathbf{W}) = -\log p(y = y_i \,|\, \mathbf{x}_i, \mathbf{W}) = -\log \mathbf{s}_{y_i}(\mathbf{W}\,\bar{\mathbf{x}}_i)$$



Learning:

$$\arg\min_{\mathbf{W}} \mathscr{L}(\mathbf{W})$$

Various implementation and names



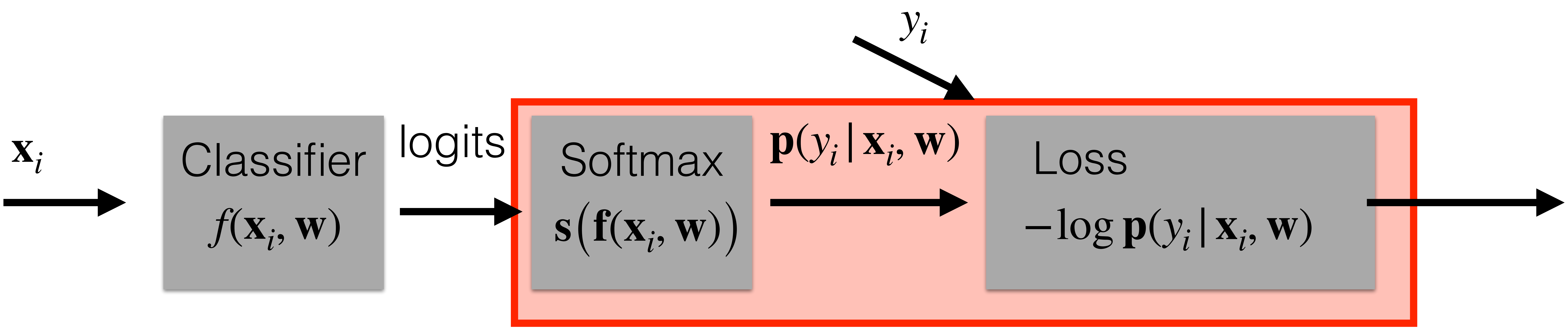
Input: probabilities

(Multinomial) Logistic loss,
Cross-entropy loss, …

$$\mathscr{L}(\mathbf{w}) = CE\left(y_i, \mathbf{p}(y_i \mid \mathbf{x}_i, \mathbf{w})\right)$$

Pytorch: BCELoss
TensorFlow: log_loss
Caffe: Multinomial Logistic Loss Layer

Various implementation and names



Input: probabilities

(Multinomial) Logistic loss,
Cross-entropy loss, …

$$\mathscr{L}(\mathbf{w}) = CE\left(y_i, \mathbf{p}(y_i \mid \mathbf{x}_i, \mathbf{w})\right)$$

Pytorch: NLLLoss
TensorFlow: log_loss
Caffe: Multinomial Logistic Loss Layer

Input: logits

Softmax loss,
Categorical Cross-entropy loss, …

$$\mathscr{L}(\mathbf{w}) = CE\left(y_i, f(\mathbf{x}_i, \mathbf{w})\right)$$

Pytorch: CrossEntropyLoss
TensorFlow: softmax_cross_entropy
Caffe: SoftmaxWithLoss Layer

def train(

1 1 1 2 2 2 3 3 3):

$\mathbf{x}_i = \mathrm{vec}($  $)$

$$\mathbf{W}^\star = \arg\min_{\mathbf{W}} \mathscr{L}(\mathbf{W}) = \arg\min_{\mathbf{W}} \sum_i -\log \mathbf{s}_{y_i}(\mathbf{W}\,\bar{\mathbf{x}}_i)$$ … gradient optimization

return $\mathbf{W}^\star$

reshape($\mathbf{w}_1$) reshape($\mathbf{w}_2$) reshape($\mathbf{w}_3$) reshape($\mathbf{w}_4$) reshape($\mathbf{w}_5$) reshape($\mathbf{w}_6$) reshape($\mathbf{w}_7$) reshape($\mathbf{w}_8$) reshape($\mathbf{w}_9$) reshape($\mathbf{w}_{10}$)

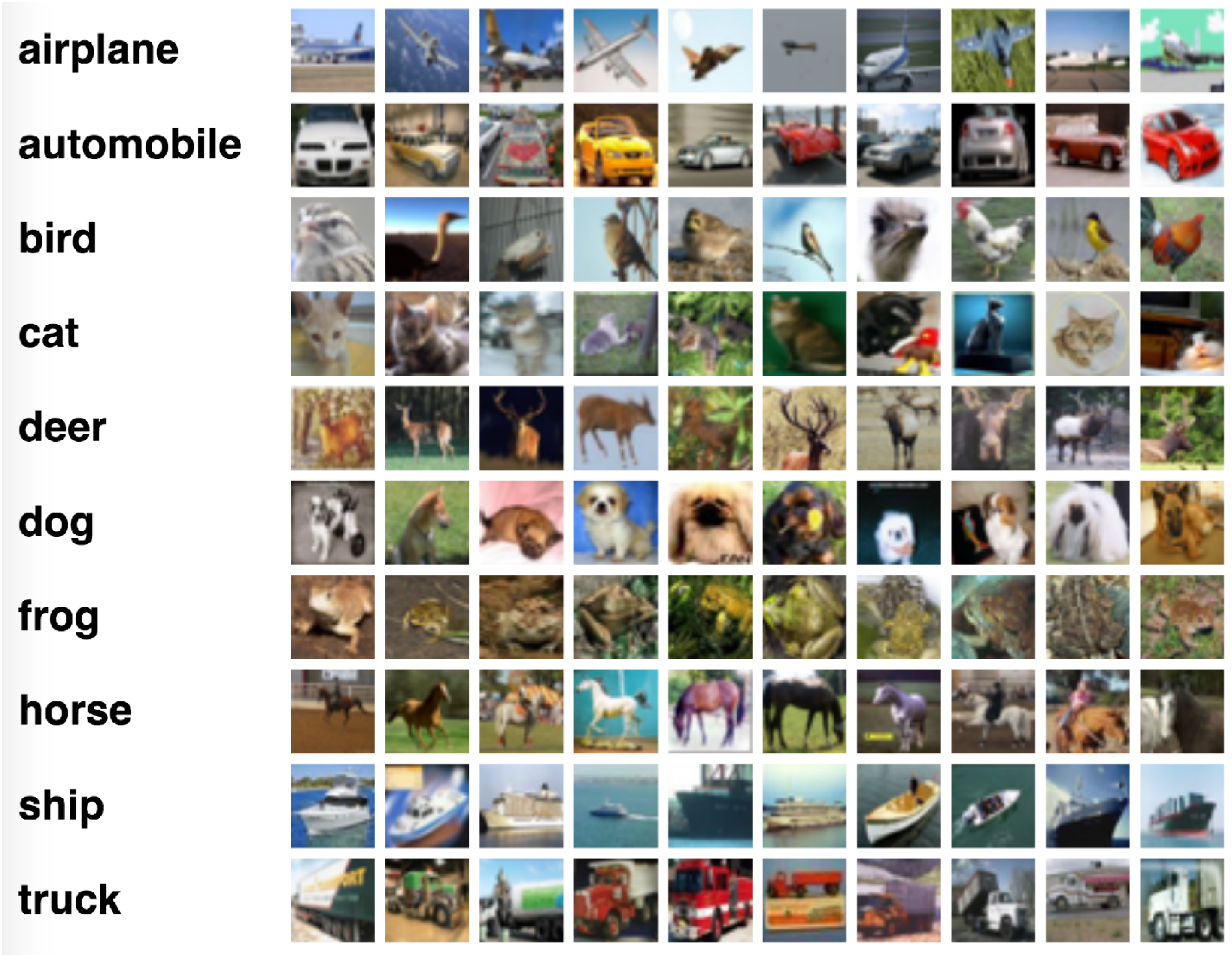
def train(

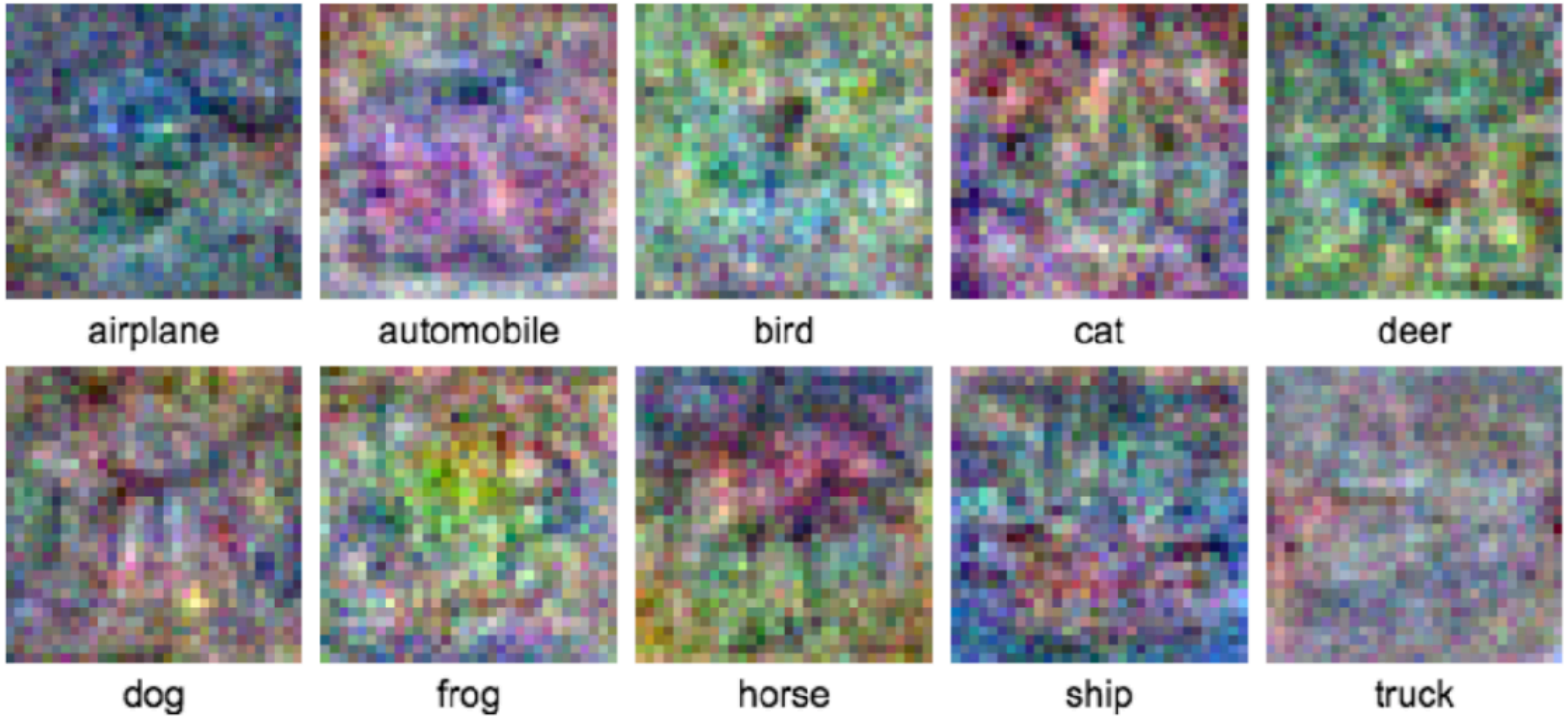
$1 \quad 1 \quad 1 \quad 2 \quad 2 \quad 2 \quad 3 \quad 3 \quad 3$):

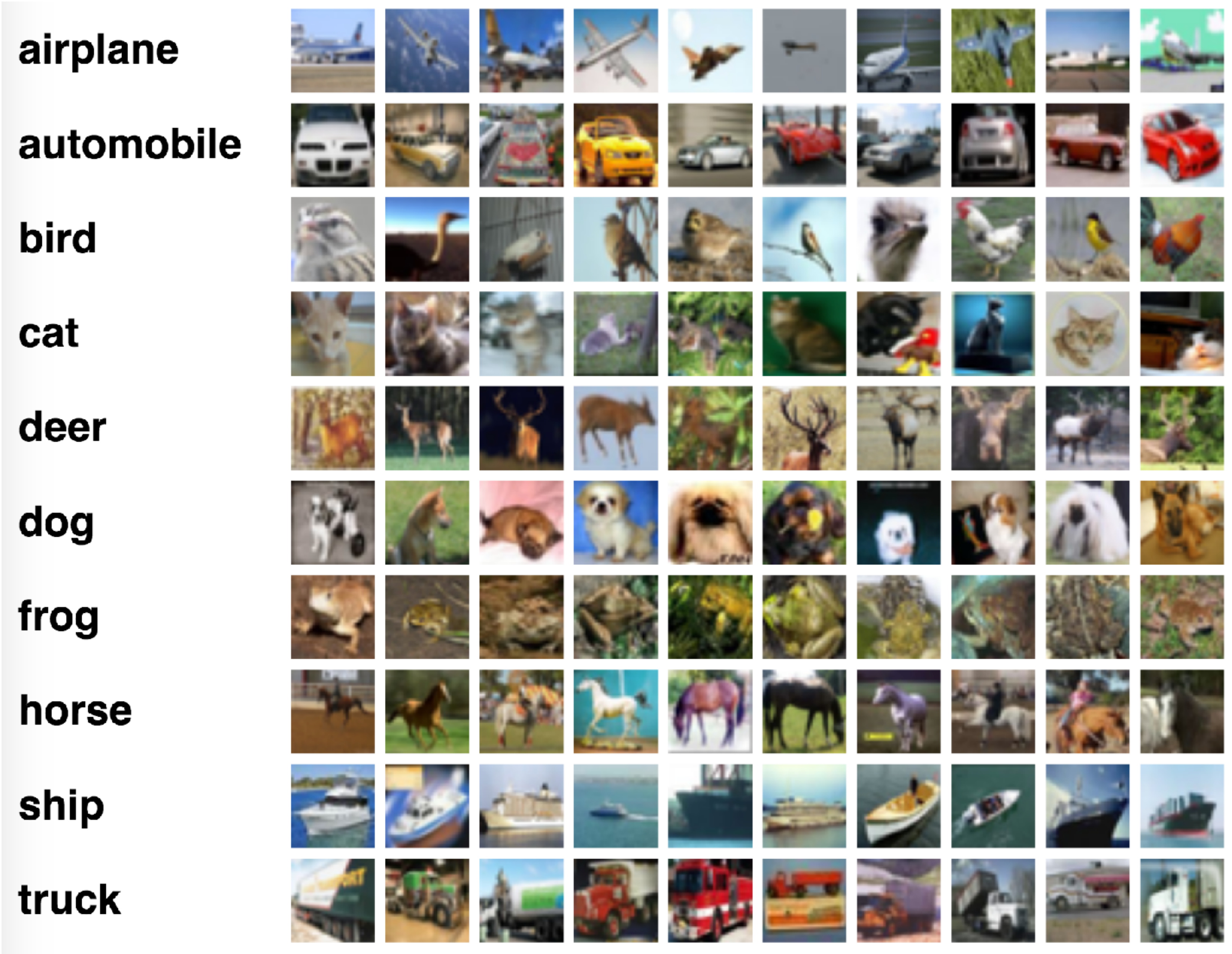
$\mathbf{x}_i = \mathrm{vec}($  $)$

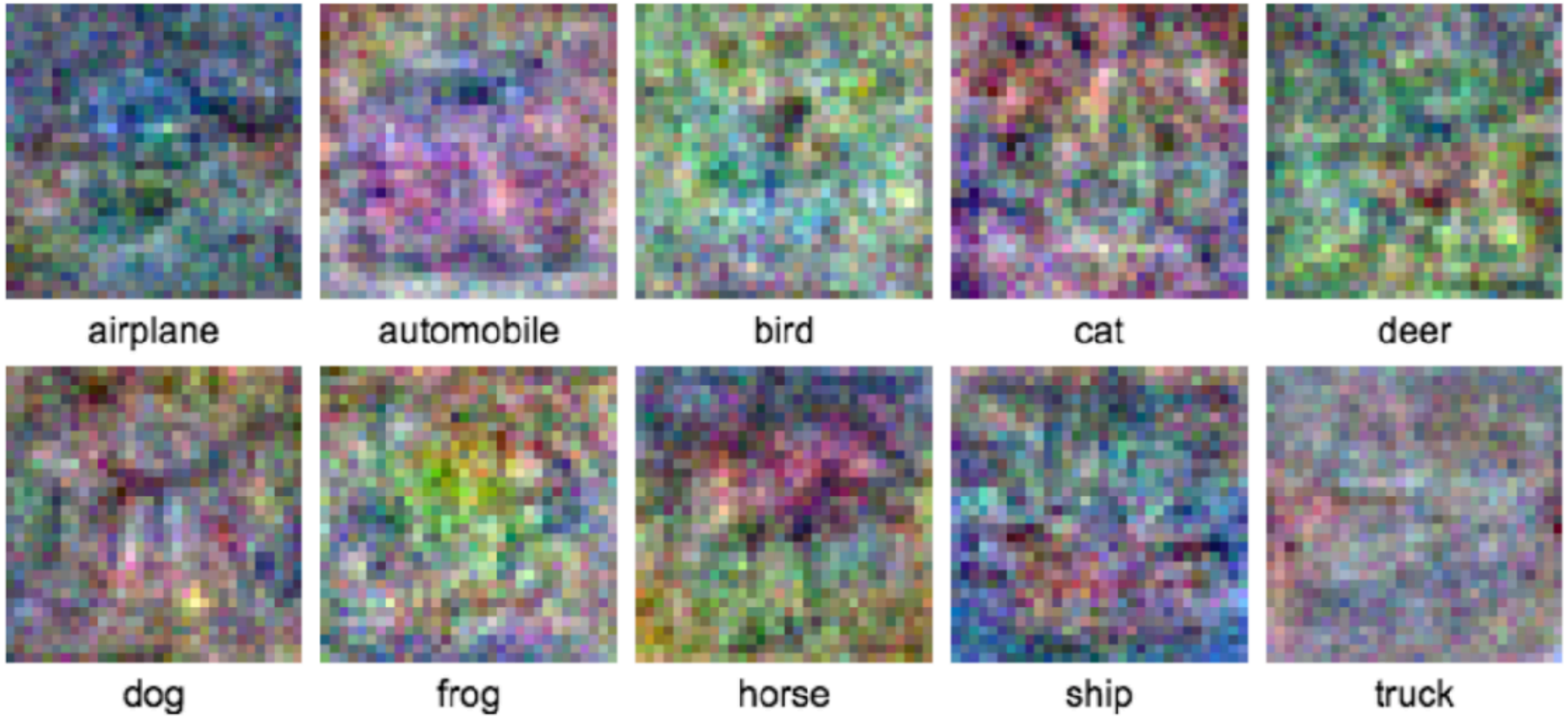
$$\mathbf{W}^\star = \arg\min_{\mathbf{W}} \mathscr{L}(\mathbf{W}) = \arg\min_{\mathbf{W}} \sum_i -\log \mathbf{s}_{y_i}(\mathbf{W}\,\overline{\mathbf{x}}_i)$$ … gradient optimization

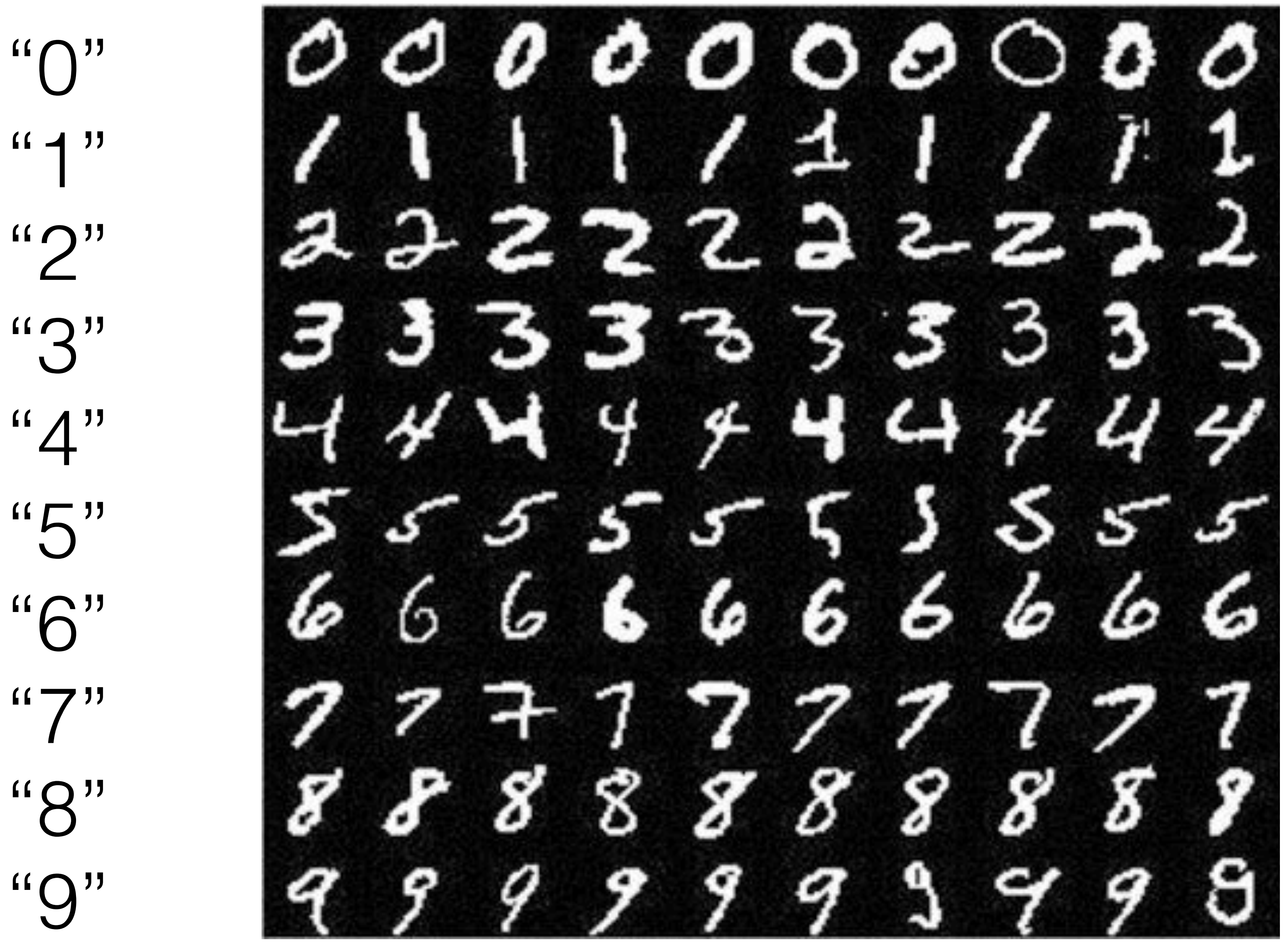
return $\mathbf{W}^\star$

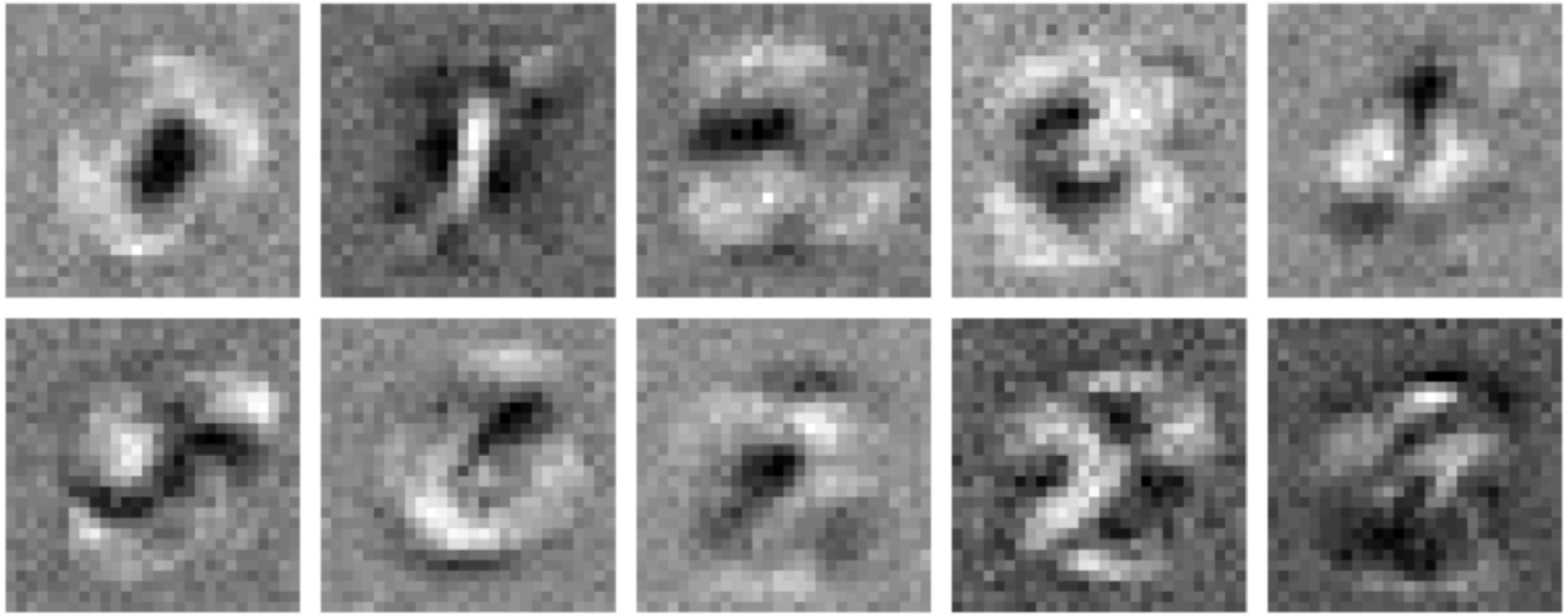

airplane
automobile
bird
cat
deer
dog
frog
horse
ship
truck

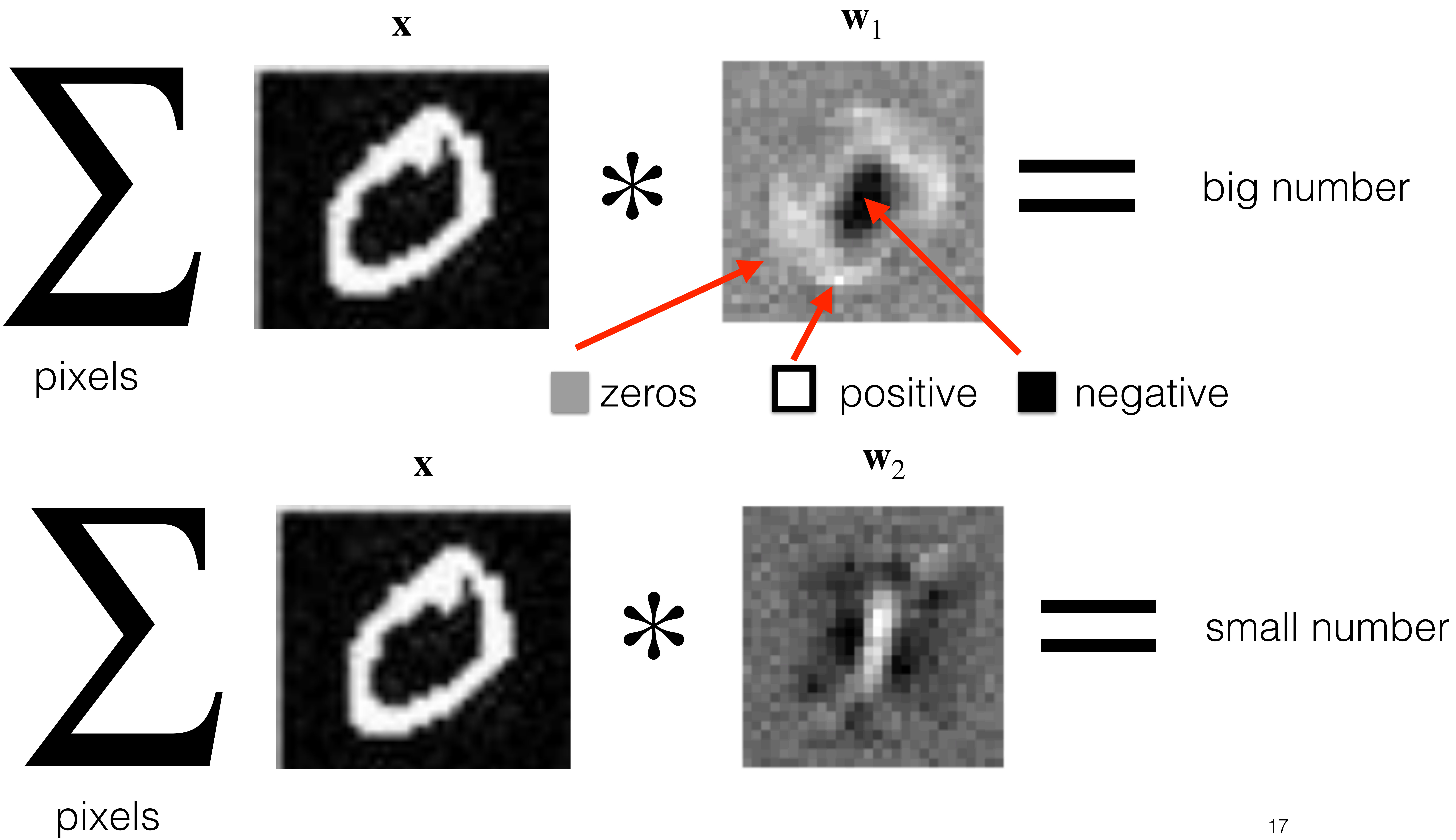

airplane
automobile
bird
cat
deer
dog
frog
horse
ship
truck


airplane
automobile
bird
cat
deer
dog
frog
horse
ship
truck


airplane
automobile
bird
cat
deer
dog
frog
horse
ship
truck


“0”
“1”
“2”
“3”
“4”
“5”
“6”
“7”
“8”
“9”



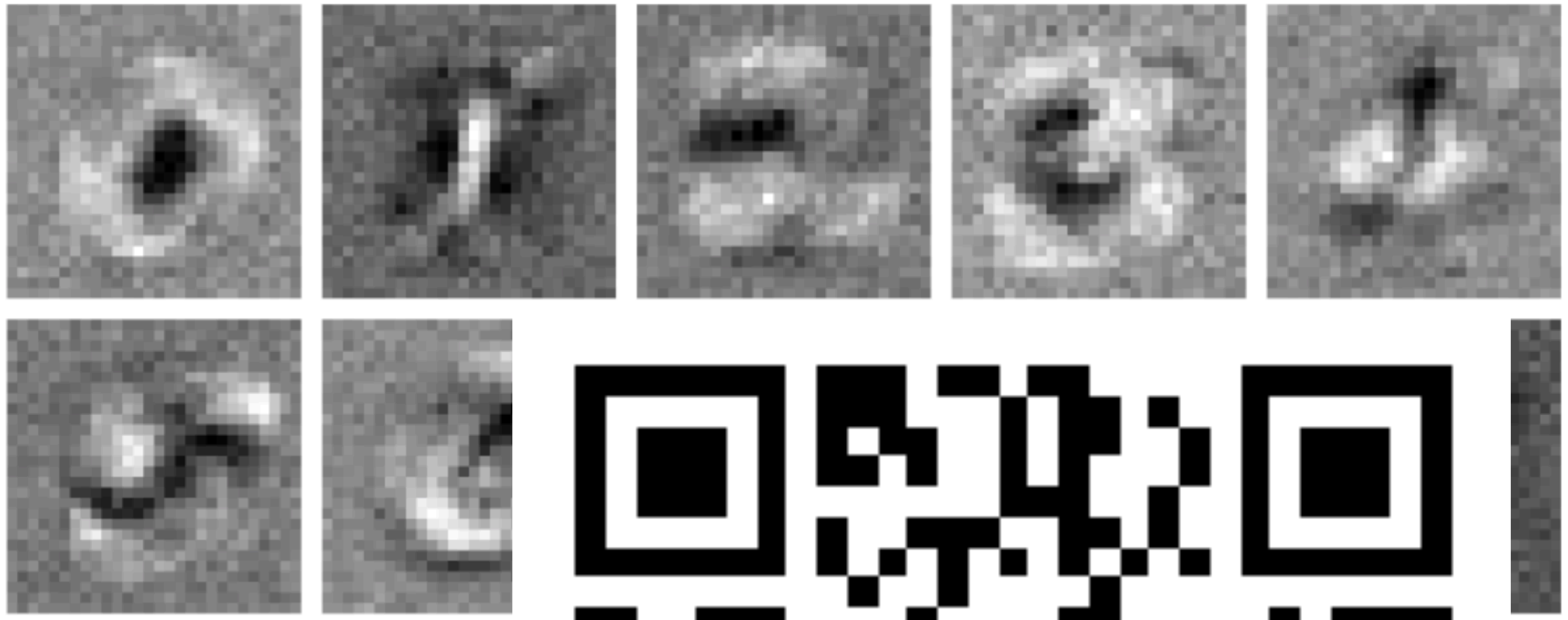


x
w1
pixels
big number
zeros
positive
negative
x
w2
pixels
small number

Dataset Label Images Learned weights of linear classifier Error

MNIST

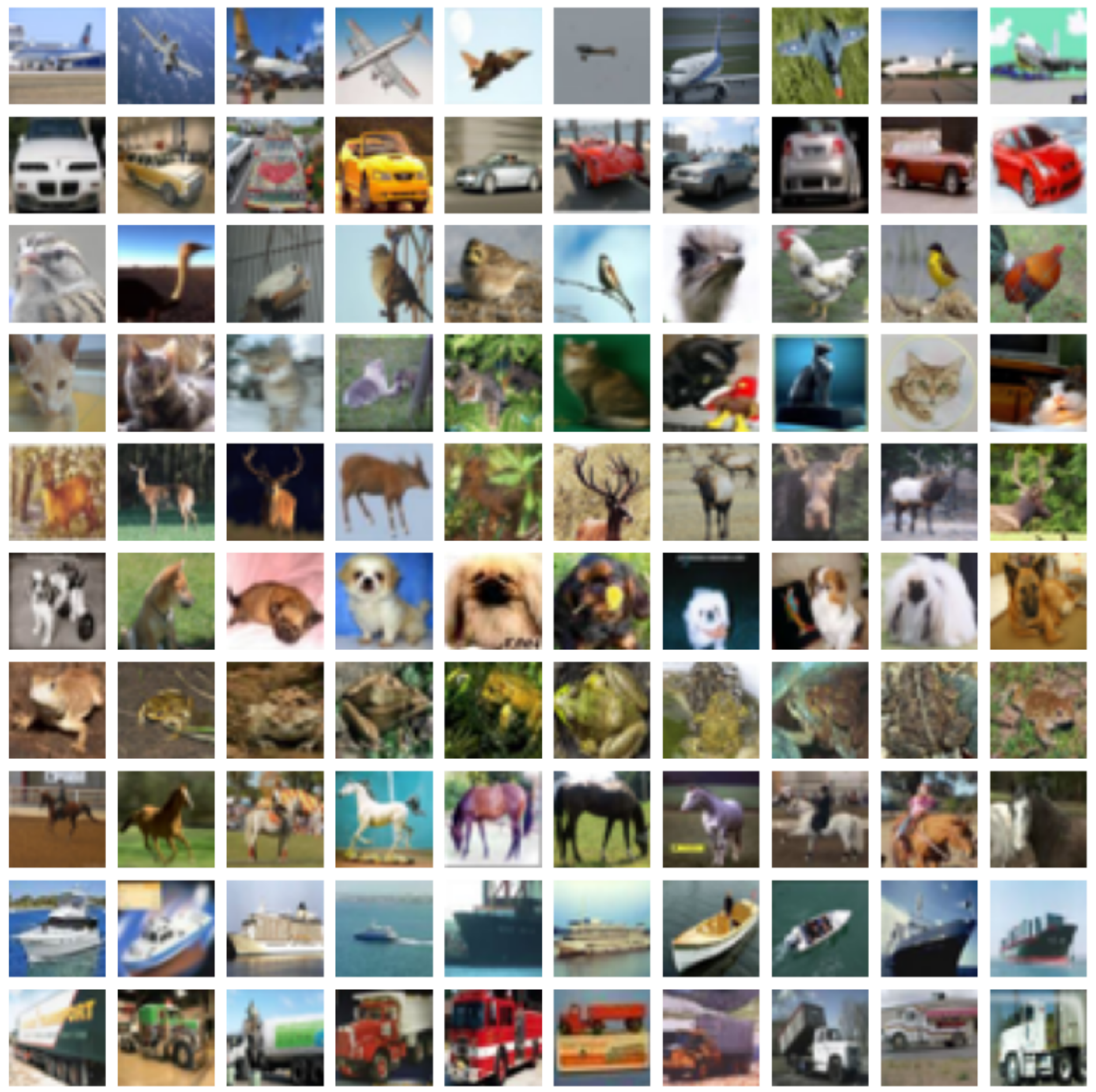
“0”
“1”
“2”
“3”
“4”
“5”
“6”
“7”
“8”
“9”



???

CIFAR-10

airplane
automobile
bird
cat
deer
dog
frog
horse
ship
truck

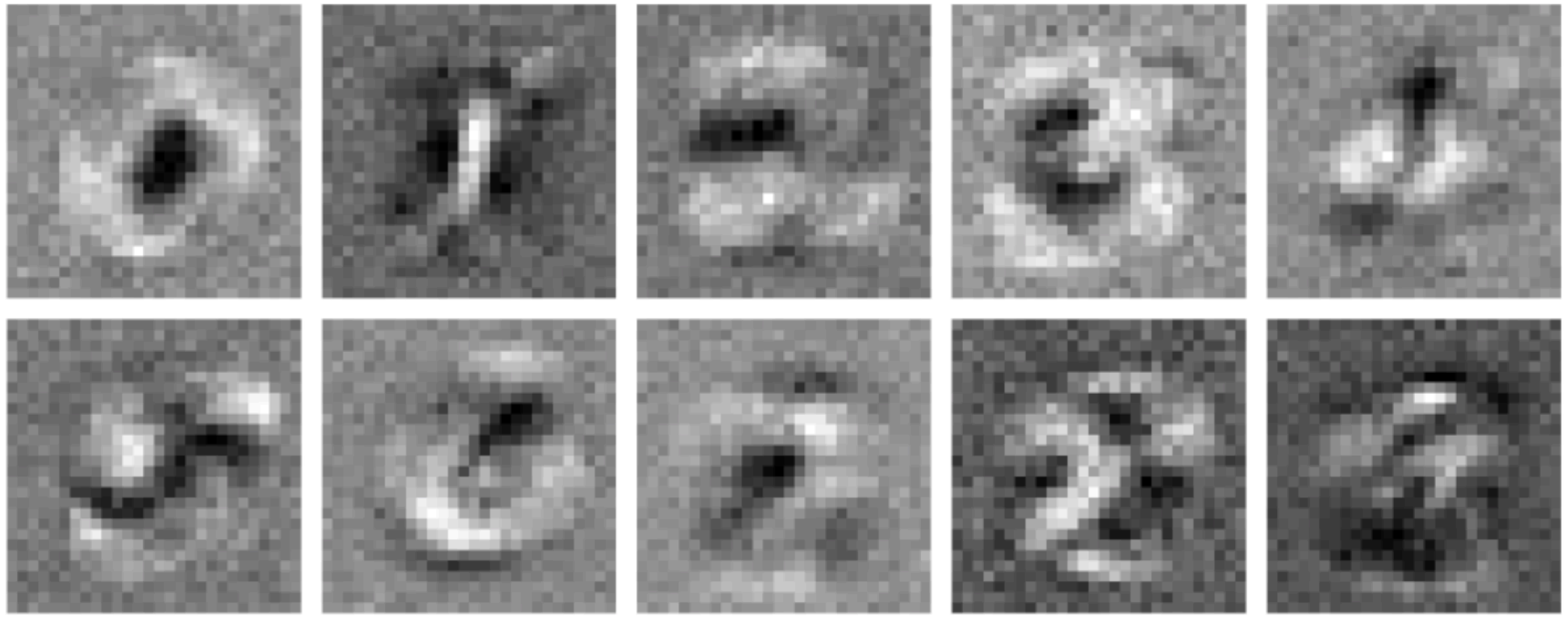


???

Dataset | Label | Images | Learned weights of linear classifier | Error

MNIST

"0"
"1"
"2"
"3"
"4"
"5"
"6"
"7"
"8"
"9"

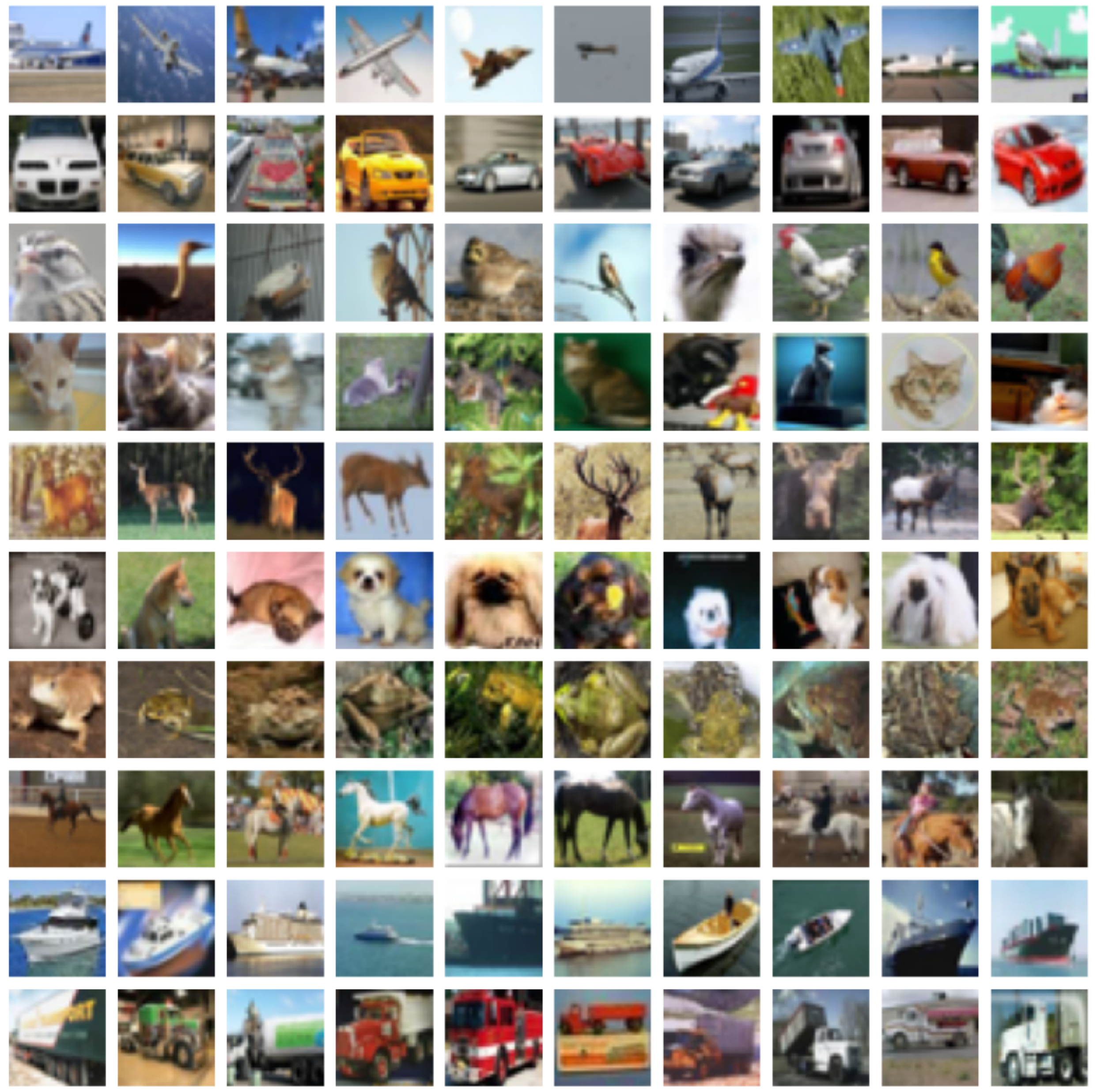


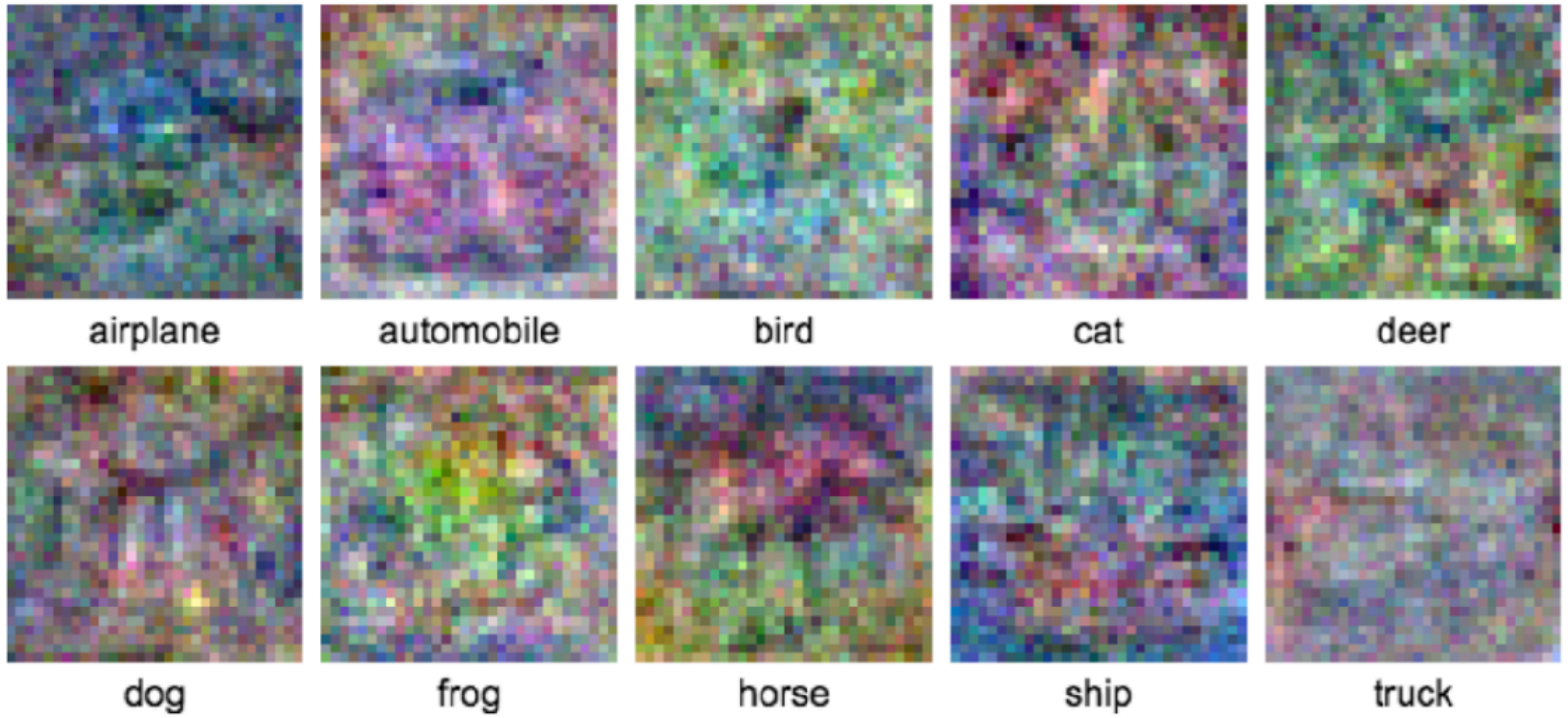
8%

Why is it simple?

CIFAR-10

airplane
automobile
bird
cat
deer
dog
frog
horse
ship
truck



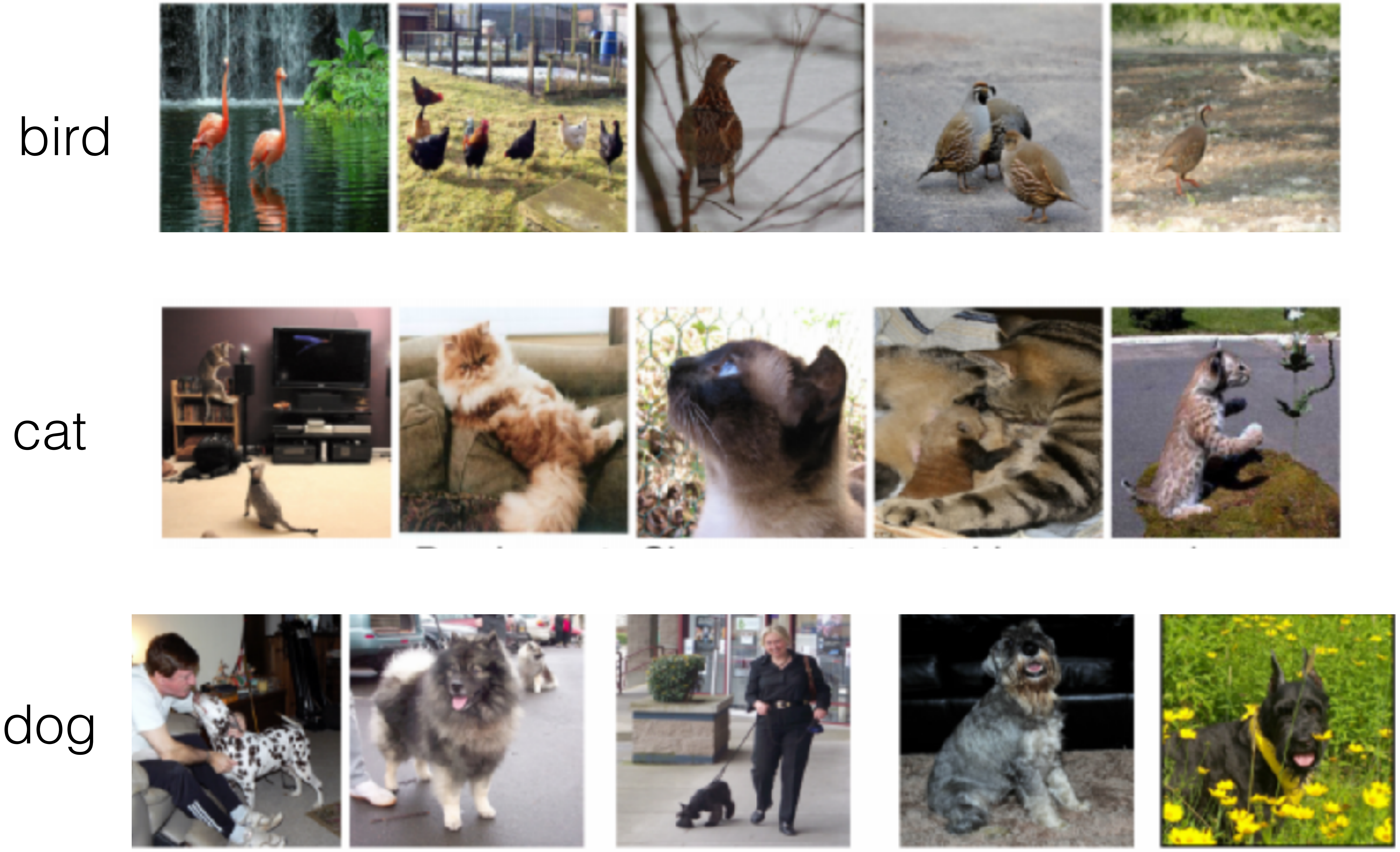


63%

Why is it hard?

Why is it hard?

Huge within-class variability!



Due to:

???

- Viewpoint
- Occlusion
- Illumination
- Pose
- Type
- Context

Why is it hard?

Huge within-class variability!

Due to:

- Viewpoint
- Occlusion
- Illumination
- Pose
- Type
- Context

Why is it hard?

Huge among-class similarity!

Why is it hard?

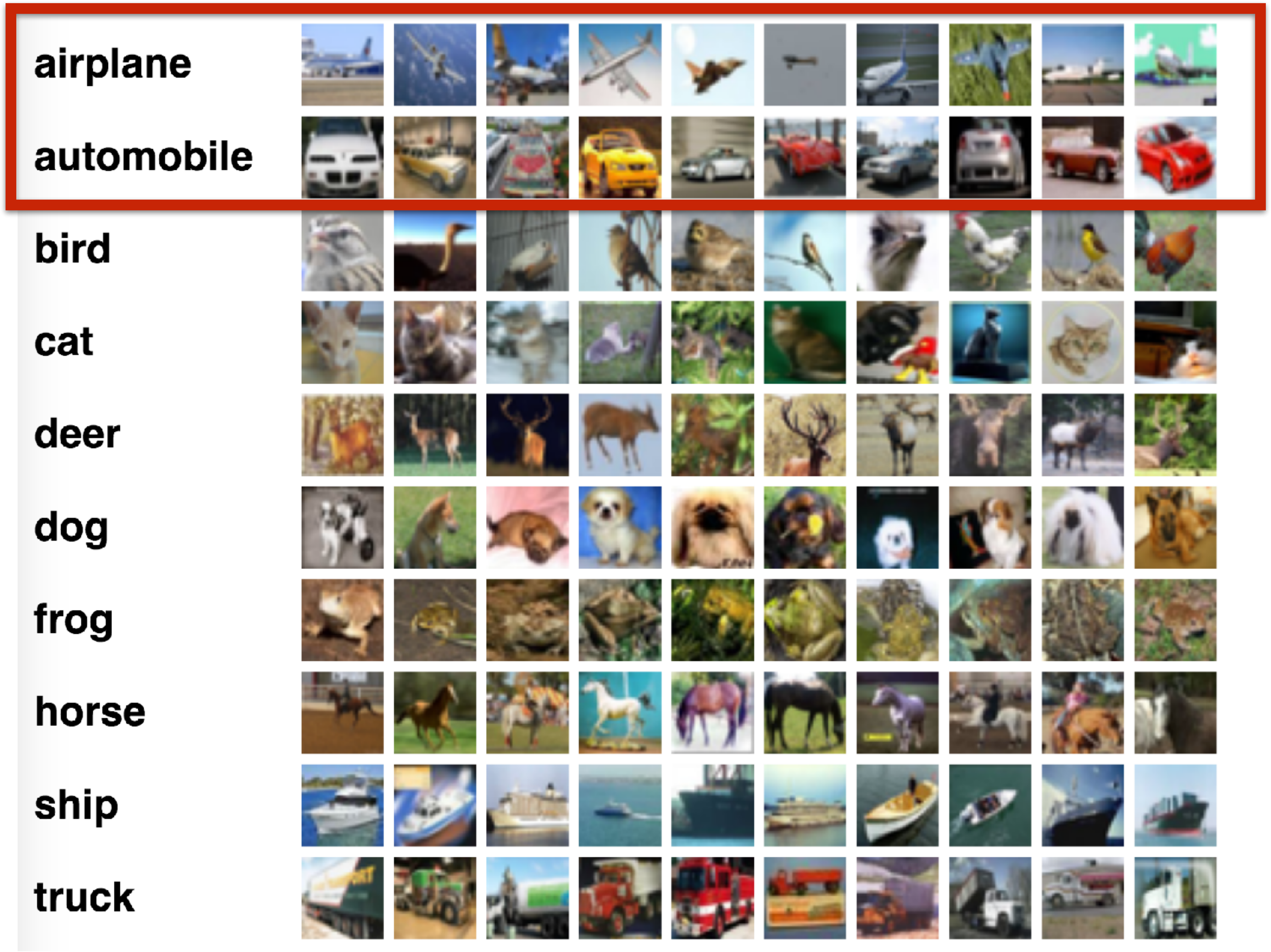
Huge among-class similarity!

Why is it hard?

Huge among-class similarity!

Recognition problem

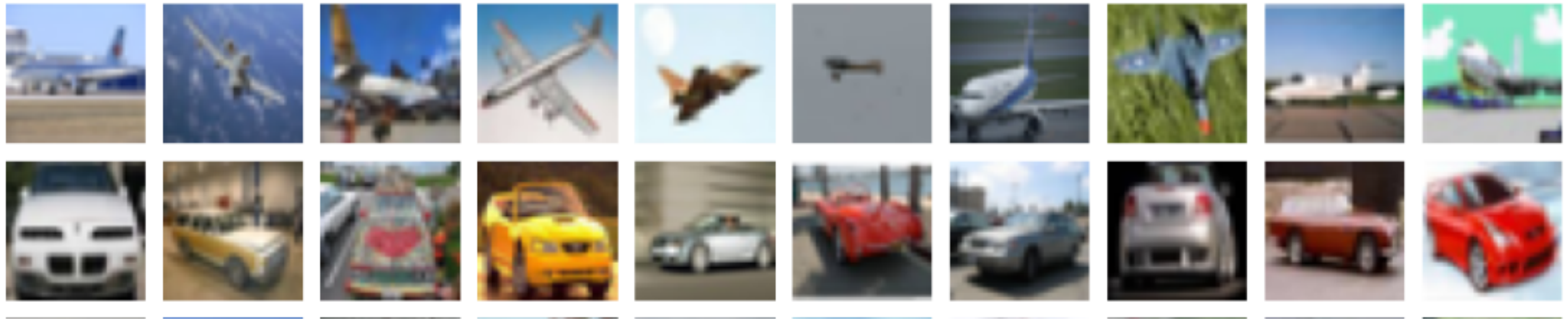


CIFAR-10: classify 32x32 RGB images into 10 categories

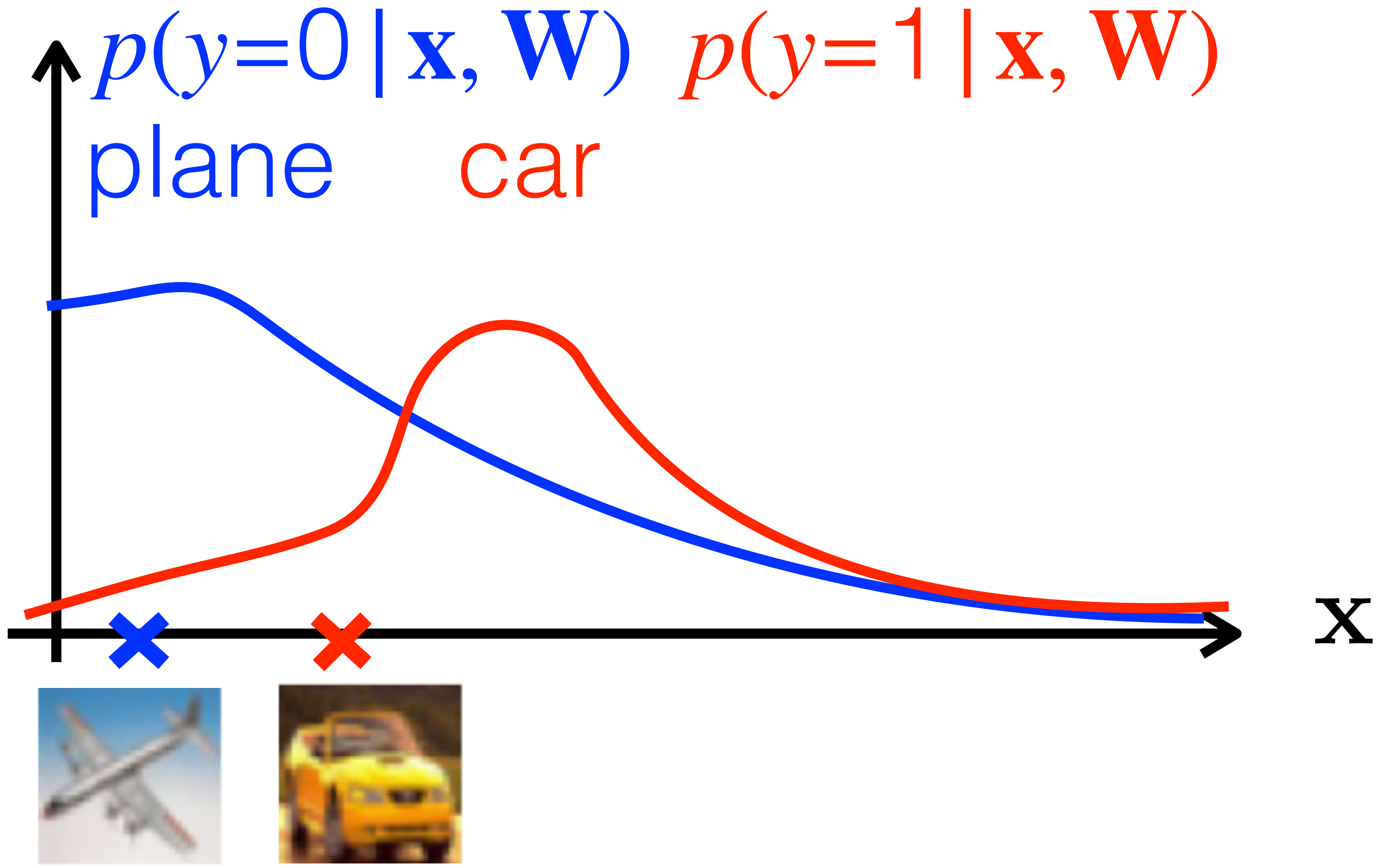
https://www.cs.toronto.edu/~kriz/cifar.html

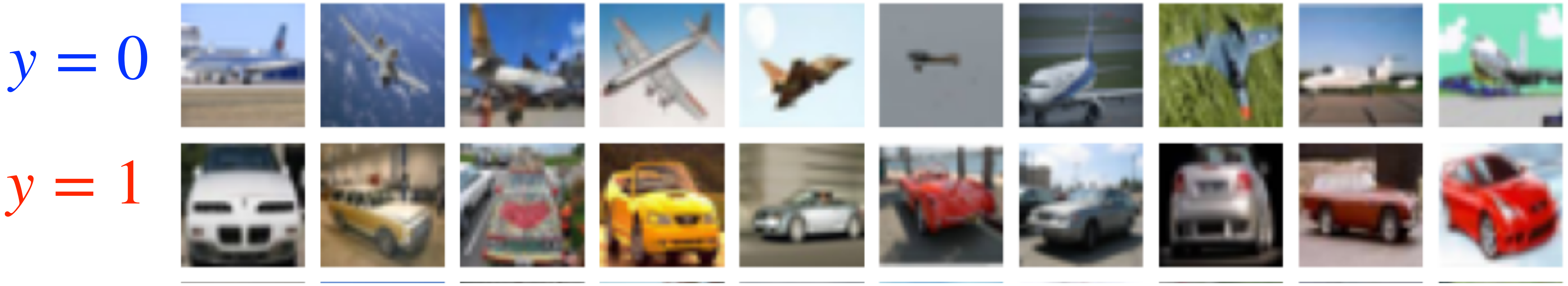
$y = 0$

$y = 1$



$$\mathbf{p}(y \,|\, \mathbf{x}, \mathbf{W}) = \begin{bmatrix} \exp(f(\mathbf{x}, \mathbf{w}_1)) \\ \exp(f(\mathbf{x}, \mathbf{w}_2)) \end{bmatrix} \Big/ \sum_k \exp\big(f(\mathbf{x}, \mathbf{w}_k\big) = \mathbf{s}(\mathbf{f}\big(\mathbf{x}, \mathbf{W})\big) \in \mathbb{R}^2$$

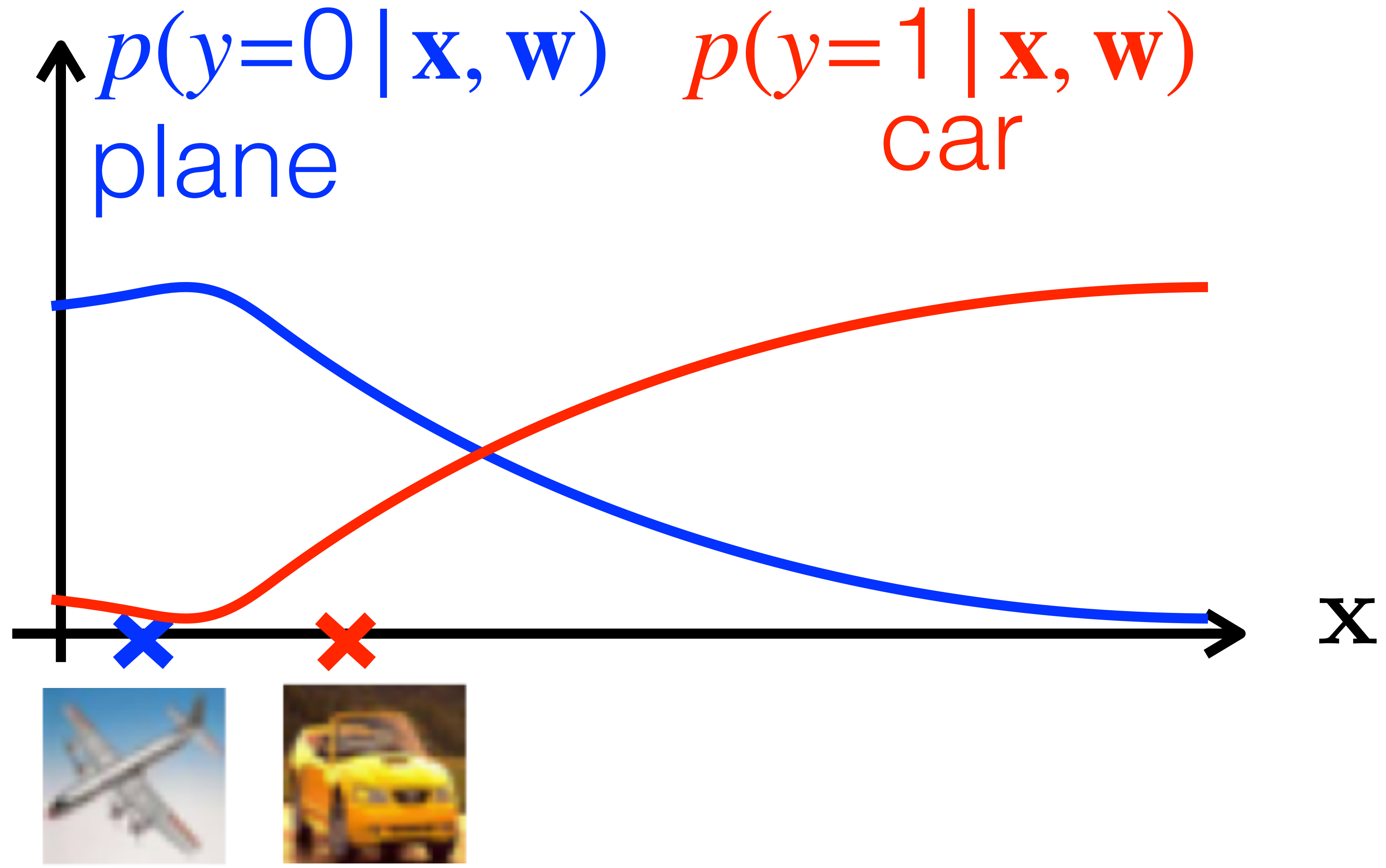




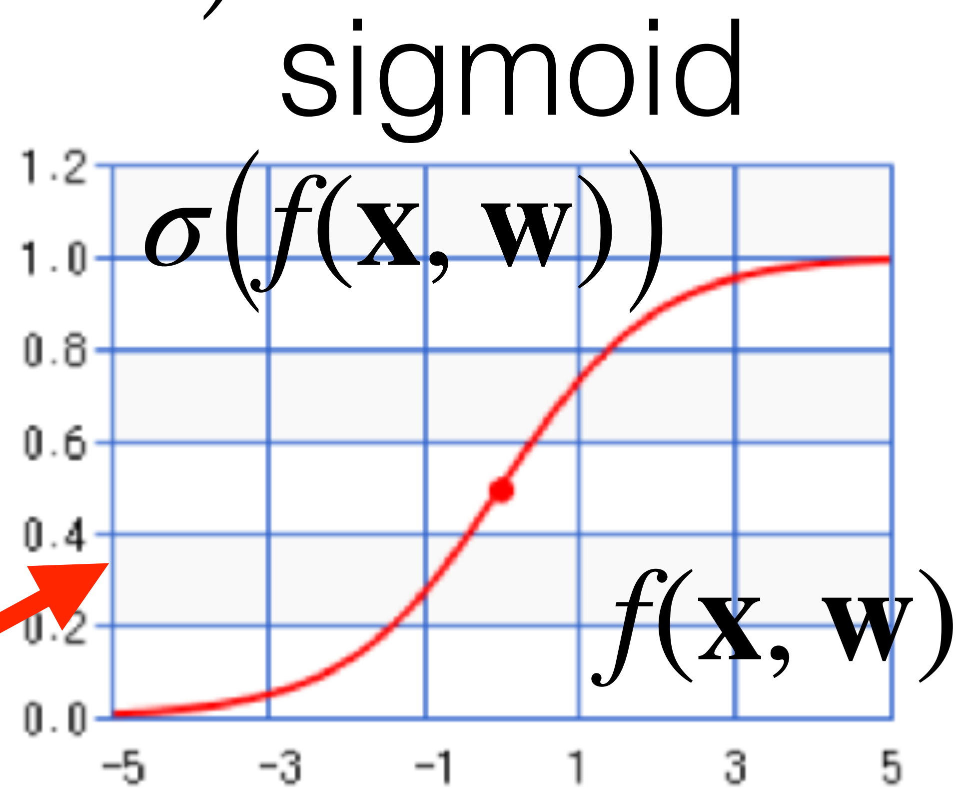
$$\mathbf{p}(y \,|\, \mathbf{x}, \mathbf{W}) = \begin{bmatrix} \exp(f(\mathbf{x}, \mathbf{w}_1)) \\ \exp(f(\mathbf{x}, \mathbf{w}_2)) \end{bmatrix} \Big/ \sum_k \exp\big(f(\mathbf{x}, \mathbf{w}_k\big) = \mathbf{s}\big(\mathbf{f}\big(\mathbf{x}, \mathbf{W}\big)\big) \in \mathbb{R}^2$$

$$\mathbf{p}(y \,|\, \mathbf{x}, \mathbf{w}) = \begin{bmatrix} \exp(f(\mathbf{x}, \mathbf{w})/2) \\ \exp(-f(\mathbf{x}, \mathbf{w})/2) \end{bmatrix} \Big/ \Big(\exp\big(f(\mathbf{x}, \mathbf{w})/2\big) + \exp(-f(\mathbf{x}, \mathbf{w})/2)\big)\Big)$$

$$= \begin{bmatrix} \dfrac{1}{1 + \frac{\exp(-f(\mathbf{x}, \mathbf{w})/2)}{\exp(f(\mathbf{x}, \mathbf{w})/2)}} \\ 1 - \dfrac{1}{1 + \frac{\exp(-f(\mathbf{x}, \mathbf{w})/2)}{\exp(f(\mathbf{x}, \mathbf{w})/2)}} \end{bmatrix}$$

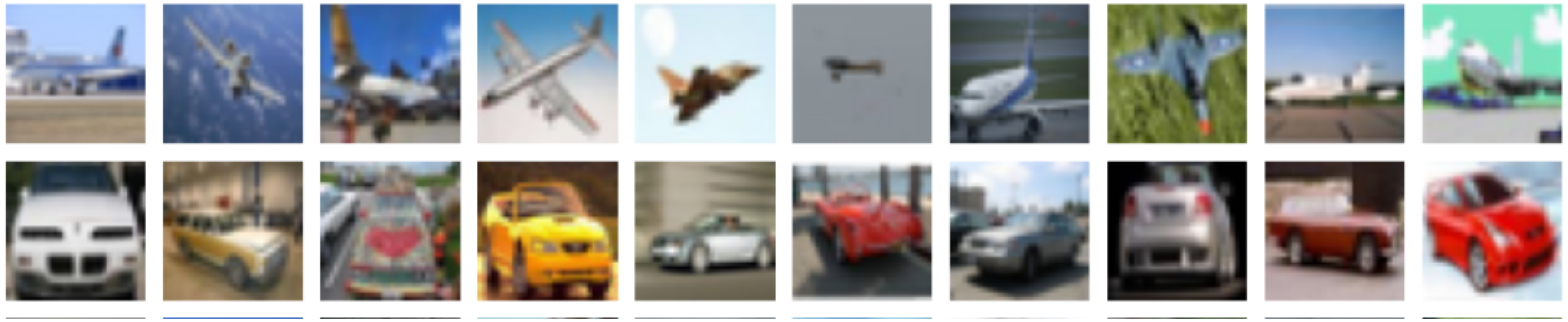


$$= \begin{bmatrix} \boxed{\dfrac{1}{1 + \exp(-f(\mathbf{x}, \mathbf{w}))}} \\ 1 - \dfrac{1}{1 + \exp(-f(\mathbf{x}, \mathbf{w}))} \end{bmatrix} = \begin{bmatrix} \sigma\big(f(\mathbf{x}, \mathbf{w})\big) \\ 1 - \sigma\big(f(\mathbf{x}, \mathbf{w})\big) \end{bmatrix}$$



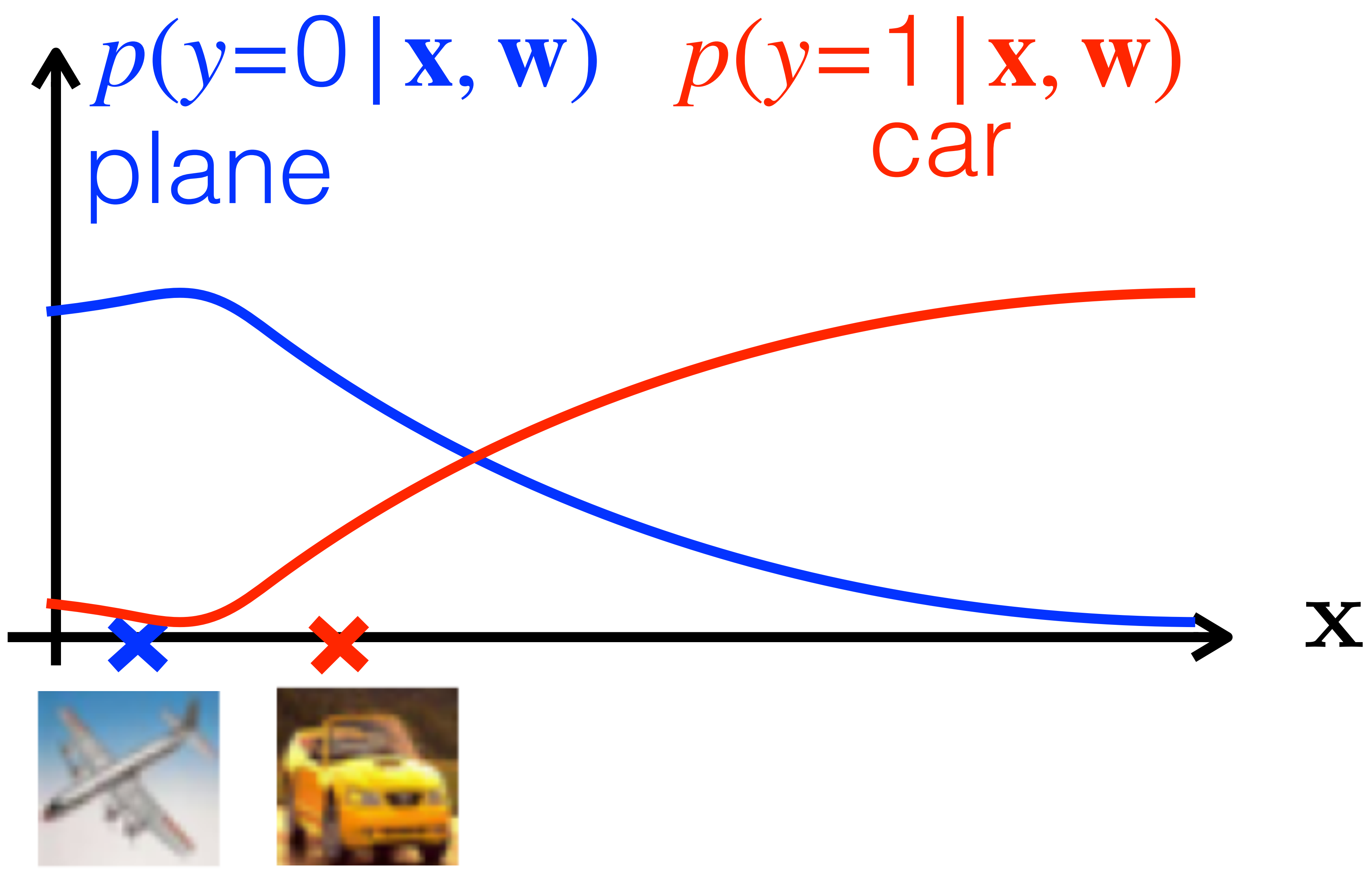
$y = 0$

$y = 1$



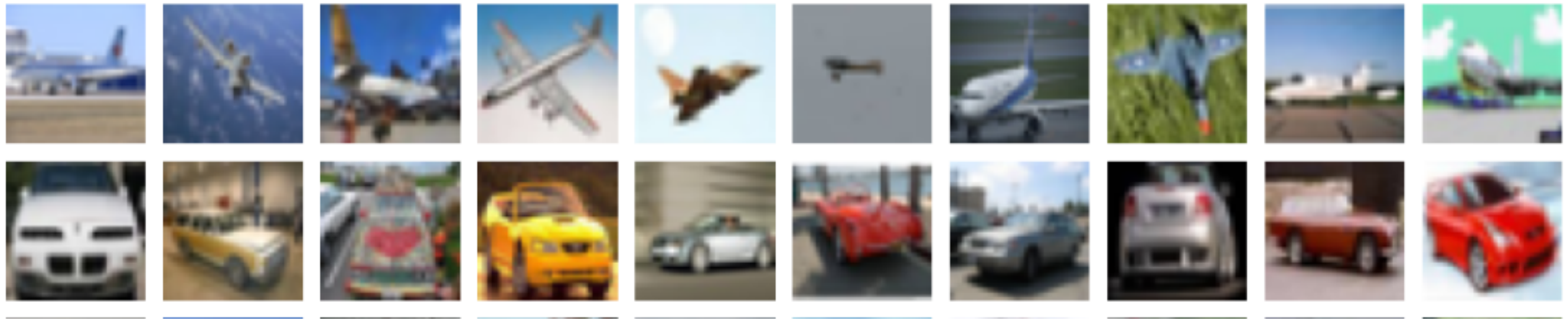
$$\mathbf{p}(y \mid \mathbf{x}, \mathbf{W}) = \begin{bmatrix} \exp(f(\mathbf{x}, \mathbf{w}_1)) \\ \exp(f(\mathbf{x}, \mathbf{w}_2)) \end{bmatrix} \bigg/ \sum_k \exp\left(f\left(\mathbf{x}, \mathbf{w}_k\right)\right) = \mathbf{s}\left(\mathbf{f}\left(\mathbf{x}, \mathbf{W}\right)\right) \in \mathbb{R}^2$$

$$\mathbf{p}(y \mid \mathbf{x}, \mathbf{w}) = \begin{bmatrix} \sigma\left(f(\mathbf{x}, \mathbf{w})\right) \\ 1 - \sigma\left(f(\mathbf{x}, \mathbf{w})\right) \end{bmatrix} \in \mathbb{R}^2$$



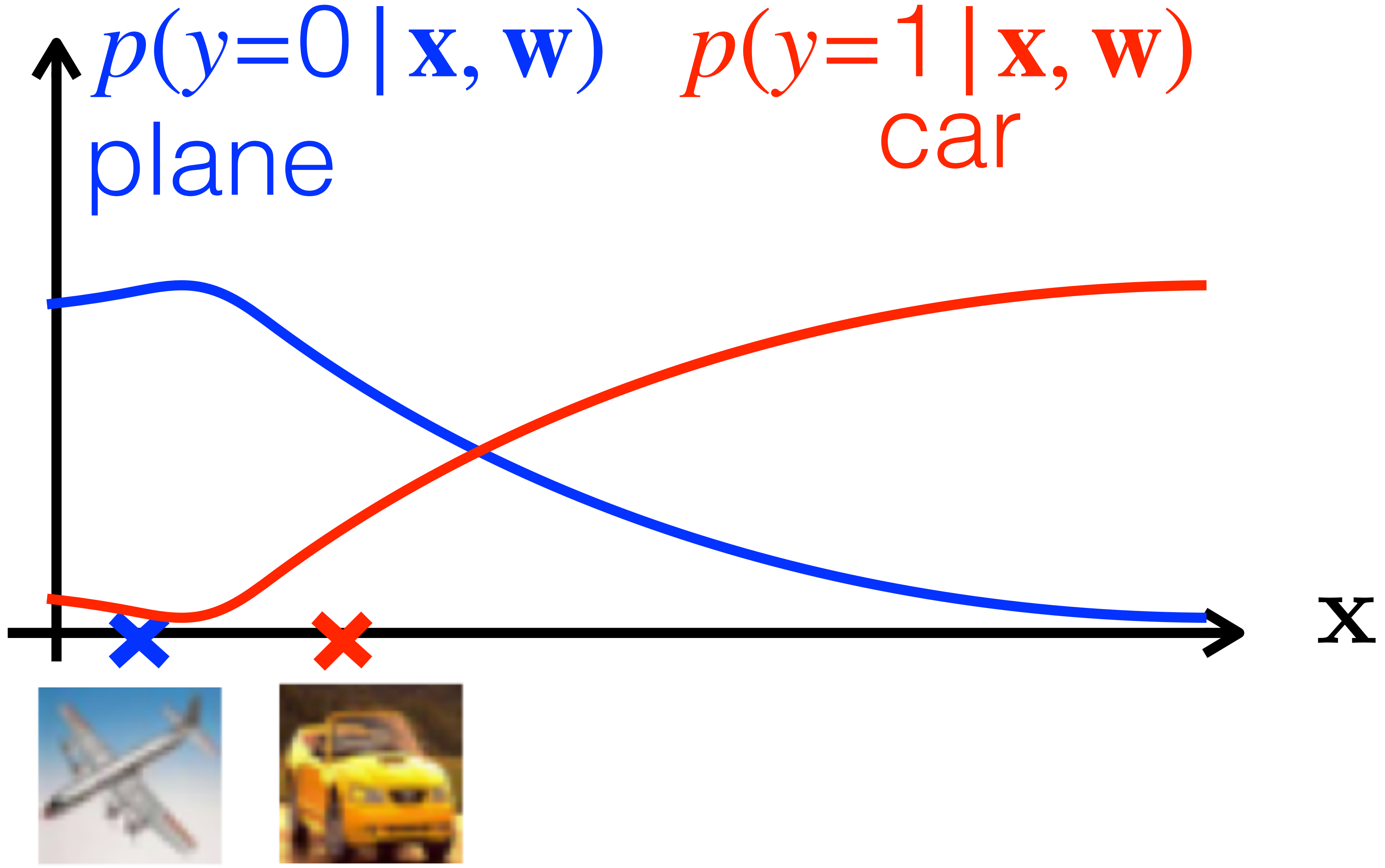
$y = 0$

$y = 1$



$$\mathbf{p}(y \mid \mathbf{x}, \mathbf{W}) = \begin{bmatrix} \exp(f(\mathbf{x}, \mathbf{w}_1)) \\ \exp(f(\mathbf{x}, \mathbf{w}_2)) \end{bmatrix} \Big/ \sum_k \exp\big(f\big(\mathbf{x}, \mathbf{w}_k\big)\big) = \mathbf{s}\big(\mathbf{f}\big(\mathbf{x}, \mathbf{W}\big)\big) \in \mathbb{R}^2$$

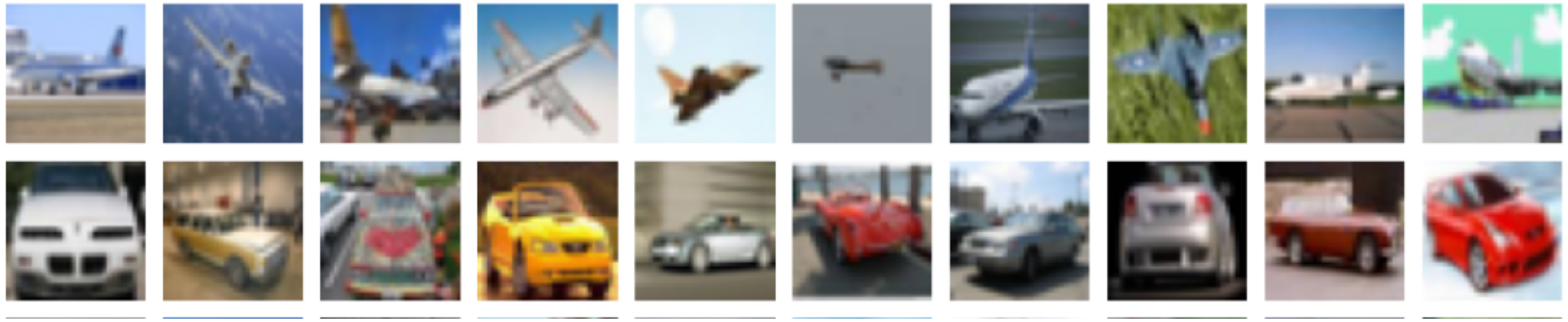
$$p(y \mid \mathbf{x}, \mathbf{w}) = \begin{cases} \sigma(f(\mathbf{x}, \mathbf{w})) & y = 1 \\ 1 - \sigma(f(\mathbf{x}, \mathbf{w})) & y = 0 \end{cases} \in \mathbb{R}^1$$



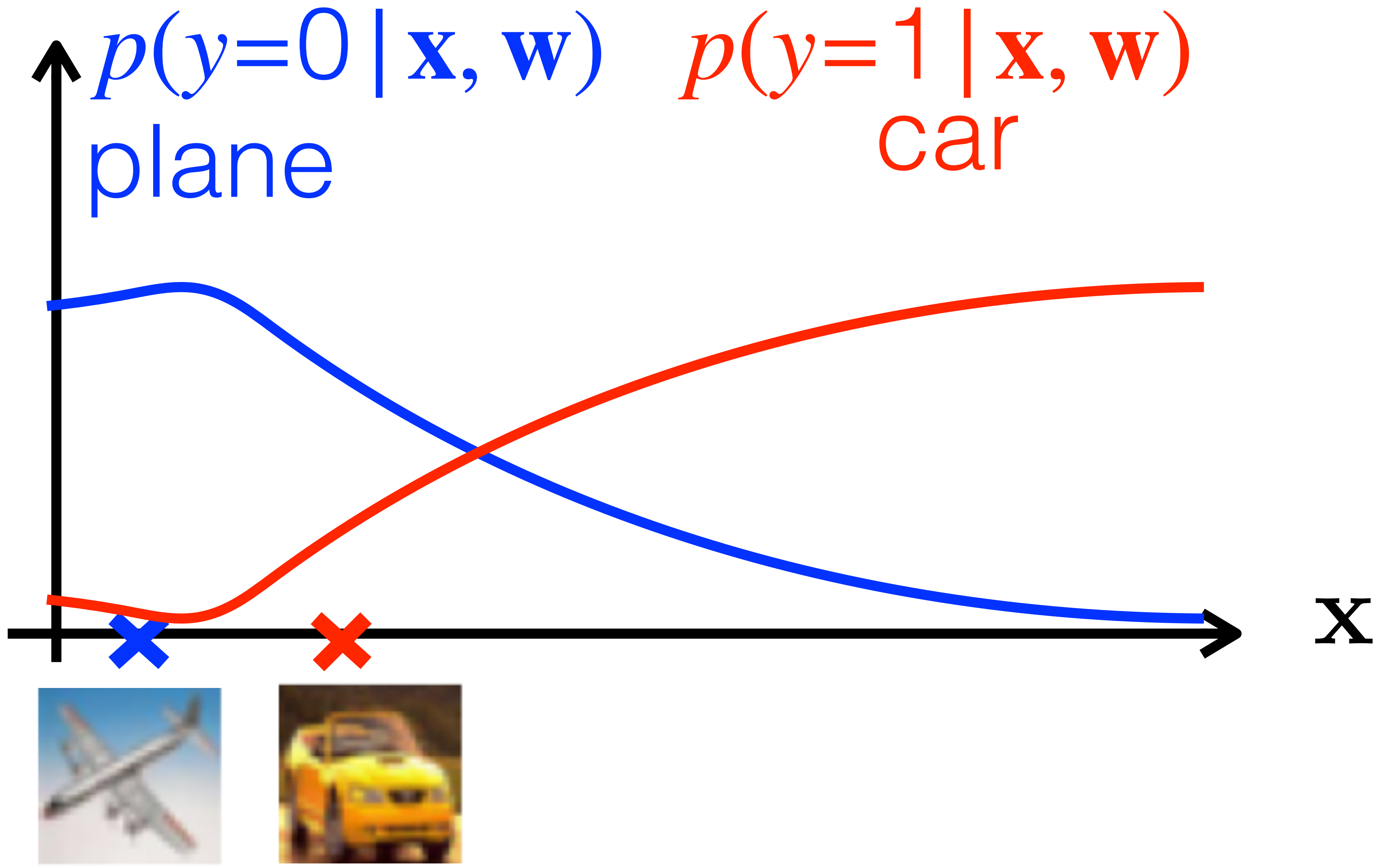
$$\cancel{\mathbf{p}(y \mid \mathbf{x}, \mathbf{w}) = \begin{bmatrix} \sigma\big(f(\mathbf{x}, \mathbf{w})\big) \\ 1 - \sigma\big(f(\mathbf{x}, \mathbf{w})\big) \end{bmatrix}} \in \mathbb{R}^2$$

$y = 0$

$y = 1$

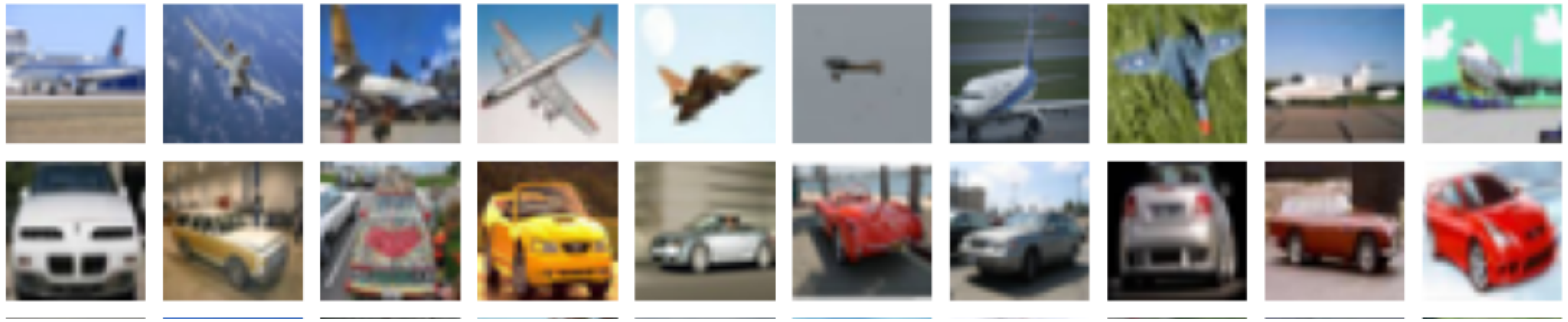


$$p(y\,|\,\mathbf{x}, \mathbf{w}) = \begin{cases} \sigma(f(\mathbf{x}, \mathbf{w})) & y = 1 \\ 1 - \sigma(f(\mathbf{x}, \mathbf{w})) & y = 0 \end{cases} = y \cdot \sigma(f(\mathbf{x}, \mathbf{w})) + (1 - y) \cdot (1 - \sigma(f(\mathbf{x}, \mathbf{w})))$$



$y = 0$

$y = 1$



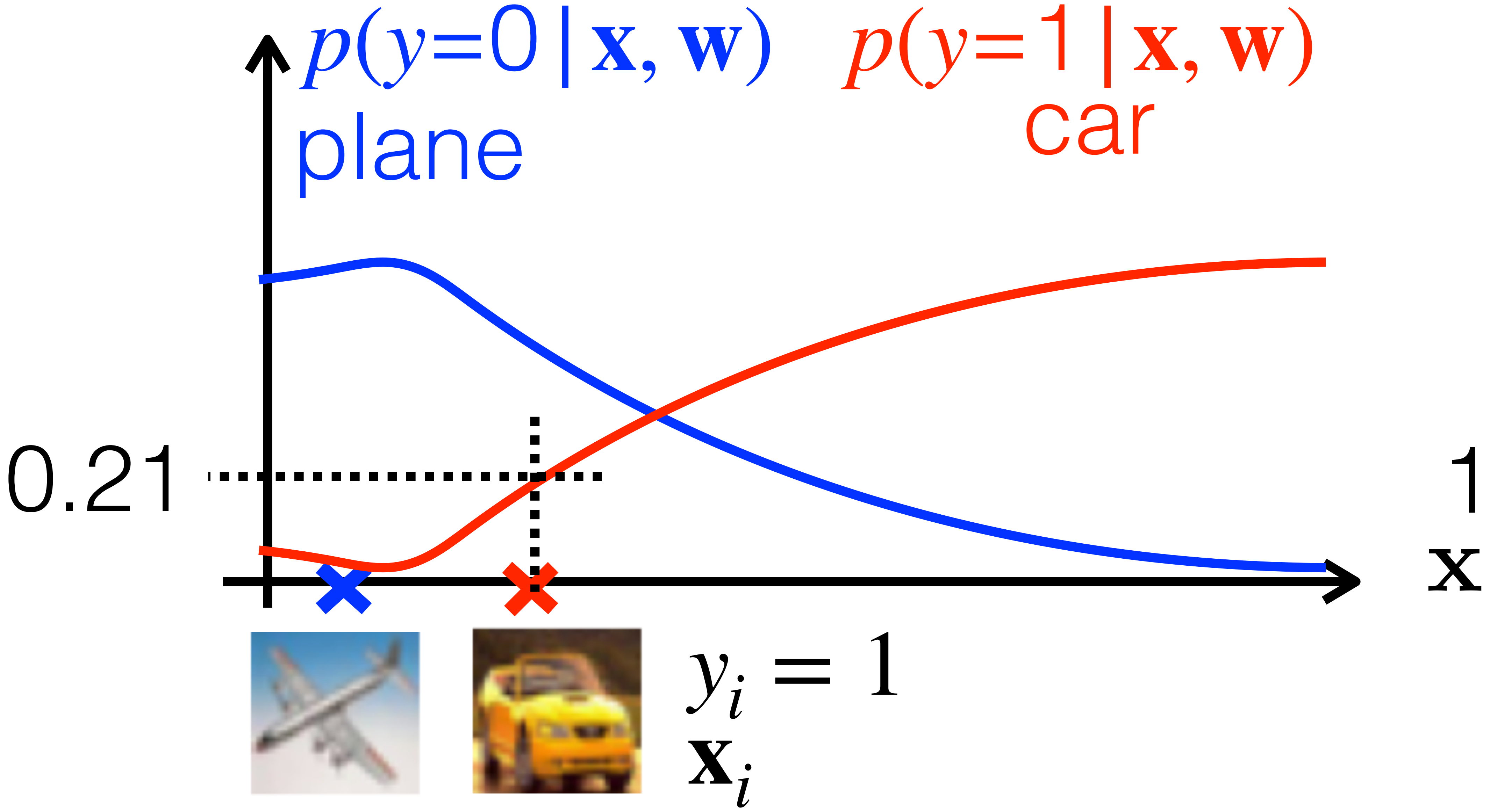
$$p(y\,|\,\mathbf{x}, \mathbf{w}) = \begin{cases} \sigma(f(\mathbf{x}, \mathbf{w})) & y = 1 \\ 1 - \sigma(f(\mathbf{x}, \mathbf{w})) & y = 0 \end{cases} \quad = y_i \cdot \sigma(f(\mathbf{x}, \mathbf{w})) + (1 - y_i) \cdot (1 - \sigma(f(\mathbf{x}, \mathbf{w})))$$

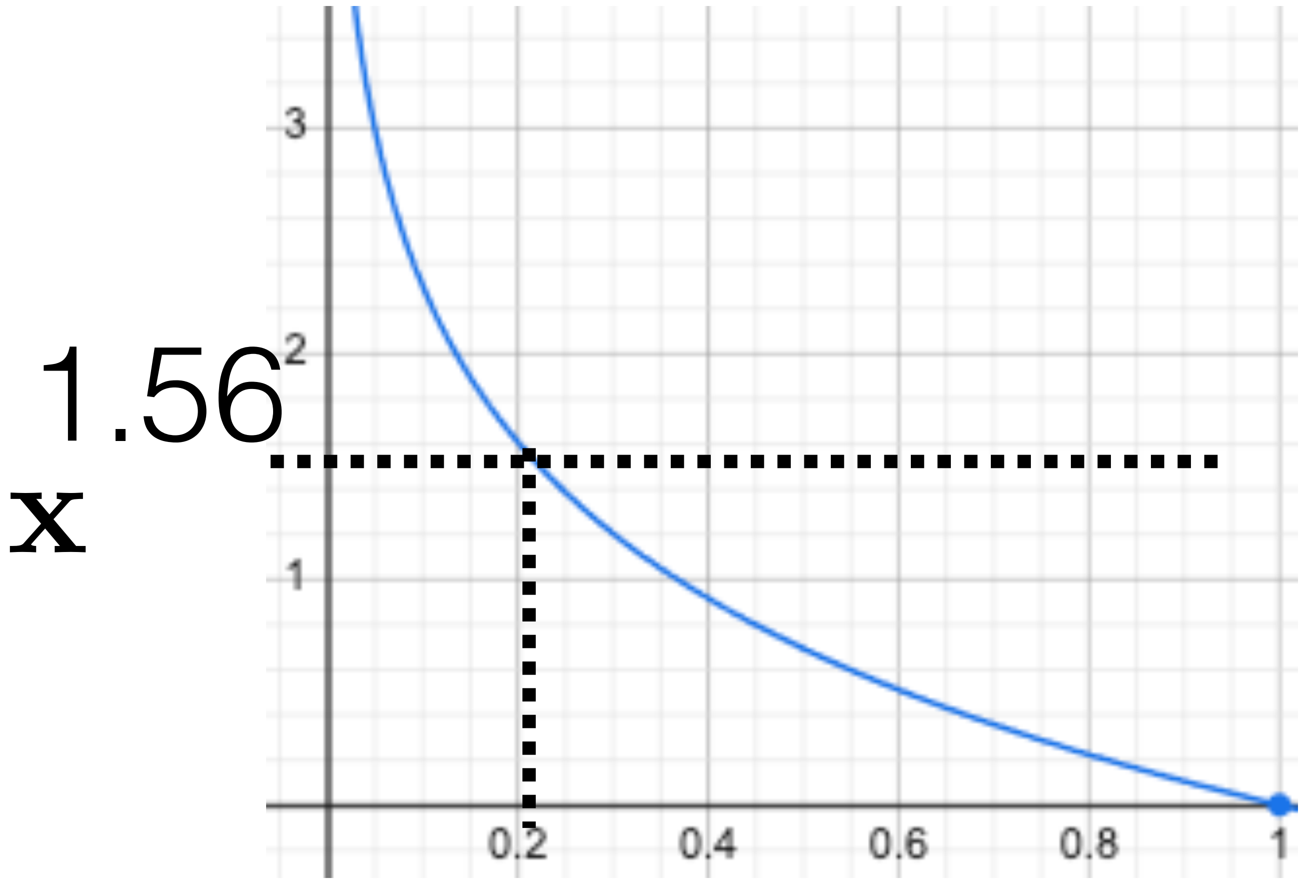
Loss function (subst. car ✖):

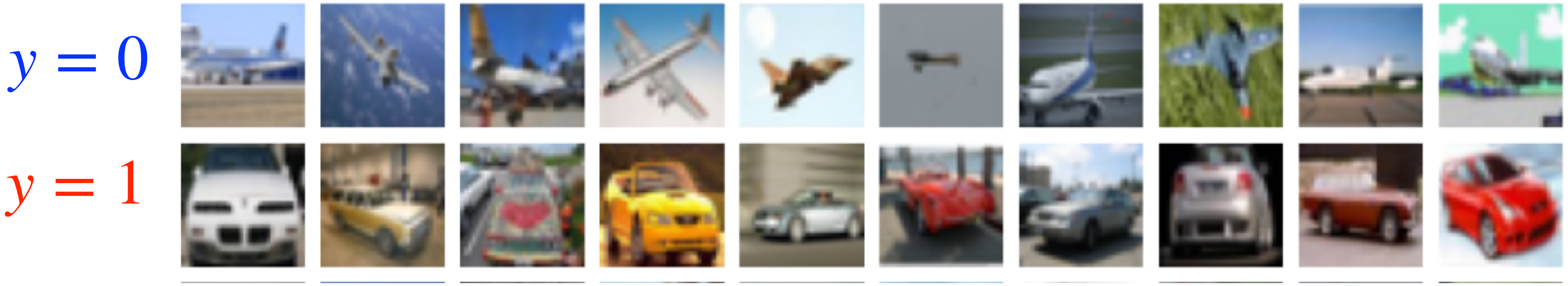
$$\mathscr{L}(\mathbf{w}) = \; -\log p(y_i\,|\,\mathbf{x}_i, \mathbf{w})$$

$$= -\log\big(y_i \cdot \sigma(f(\mathbf{x}, \mathbf{w})) + (1 - y_i) \cdot \sigma(f(\mathbf{x}, \mathbf{w}))\big)$$

$$= -\log\big(1 \cdot 0.21 + (1 - 1) \cdot 0.21\big) = 1.56$$







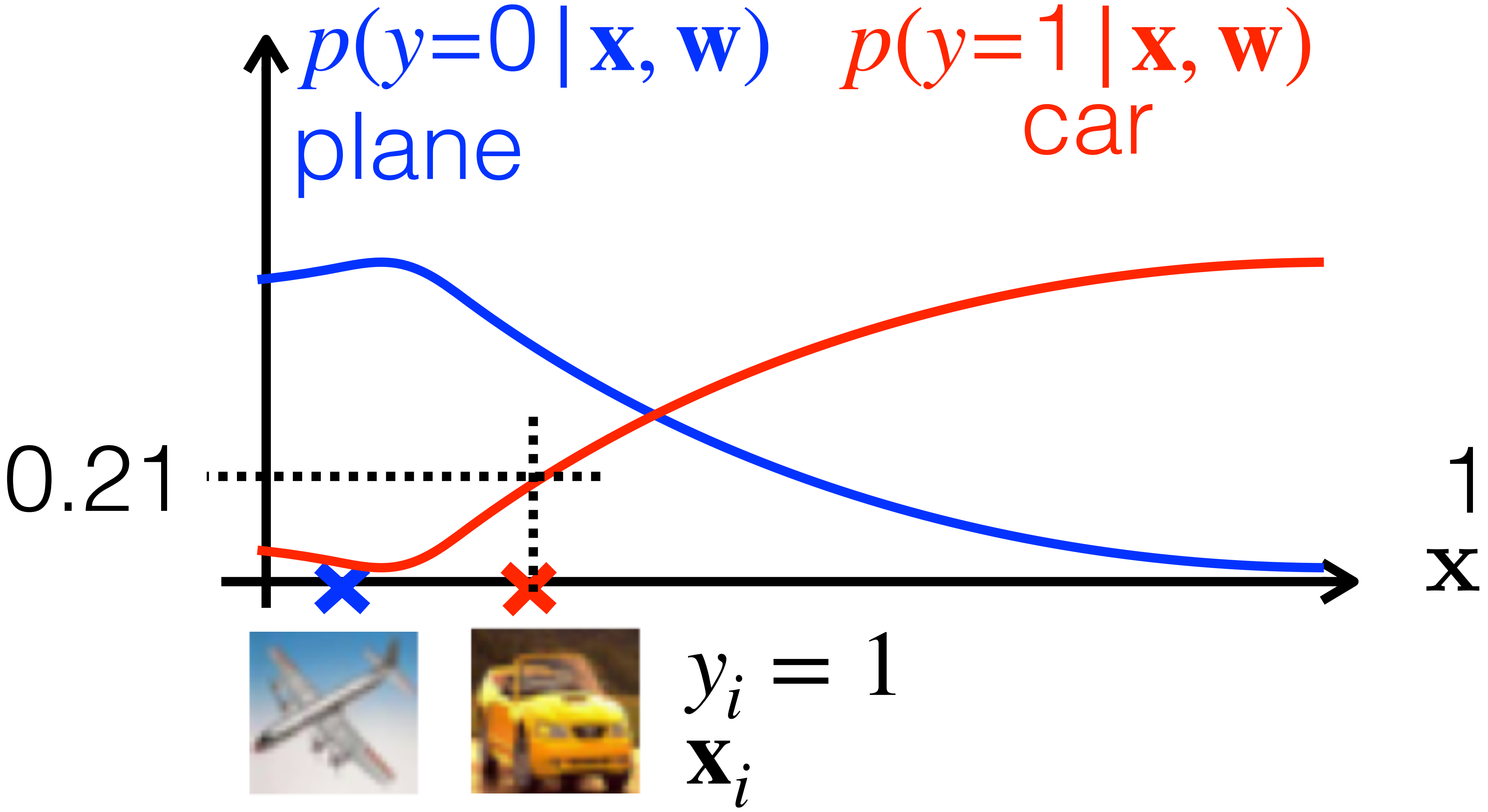
$$p(y\,|\,\mathbf{x}, \mathbf{w}) = \begin{cases} \sigma(f(\mathbf{x}, \mathbf{w})) & y = 1 \\ 1 - \sigma(f(\mathbf{x}, \mathbf{w})) & y = 0 \end{cases} = y \cdot \sigma(f(\mathbf{x}, \mathbf{w})) + (1 - y) \cdot (1 - \sigma(f(\mathbf{x}, \mathbf{w})))$$

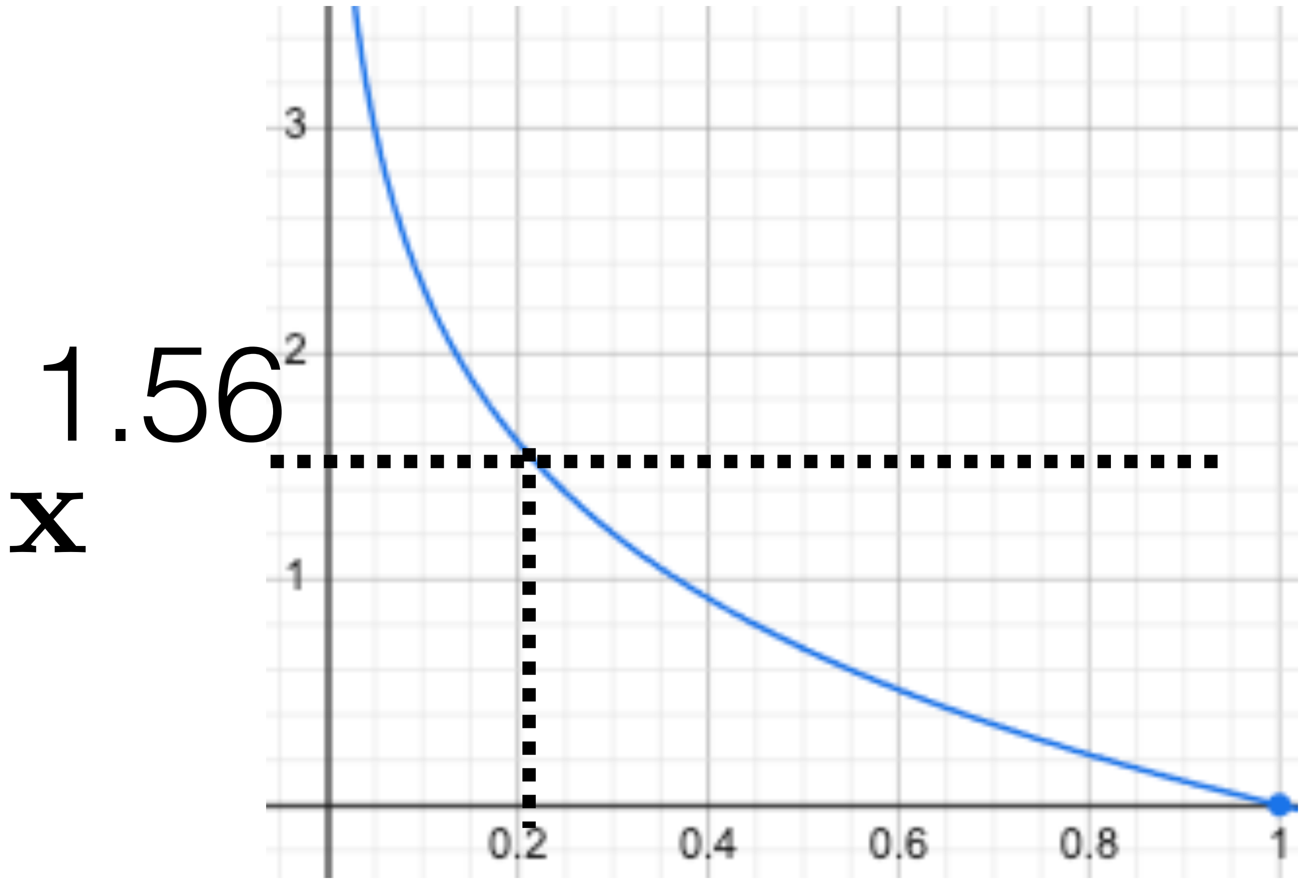
Loss function (subst. car ✖):

$$\mathscr{L}(\mathbf{w}) = \; -\log p(y_i\,|\,\mathbf{x}_i, \mathbf{w})$$

$$= -\log\big(y_i \cdot \sigma(f(\mathbf{x}, \mathbf{w})) + (1 - y_i) \cdot \sigma(f(\mathbf{x}, \mathbf{w}))\big)$$

$$= -\log\big(1 \cdot 0.21 + (1 - 1) \cdot 0.21\big) = 1.56$$



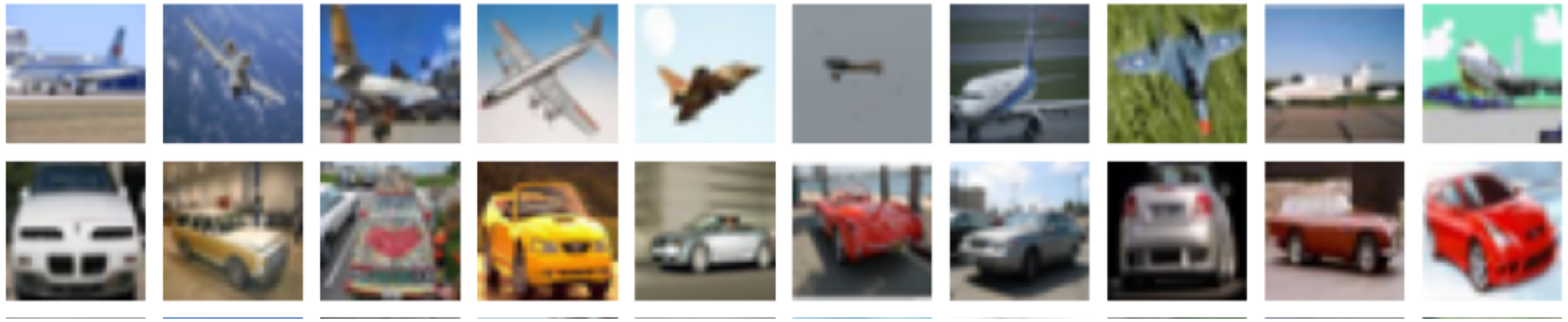


Learning:

$$\arg\min_{\mathbf{w}} \mathscr{L}(\mathbf{w})$$

$y = 0$

$y = 1$



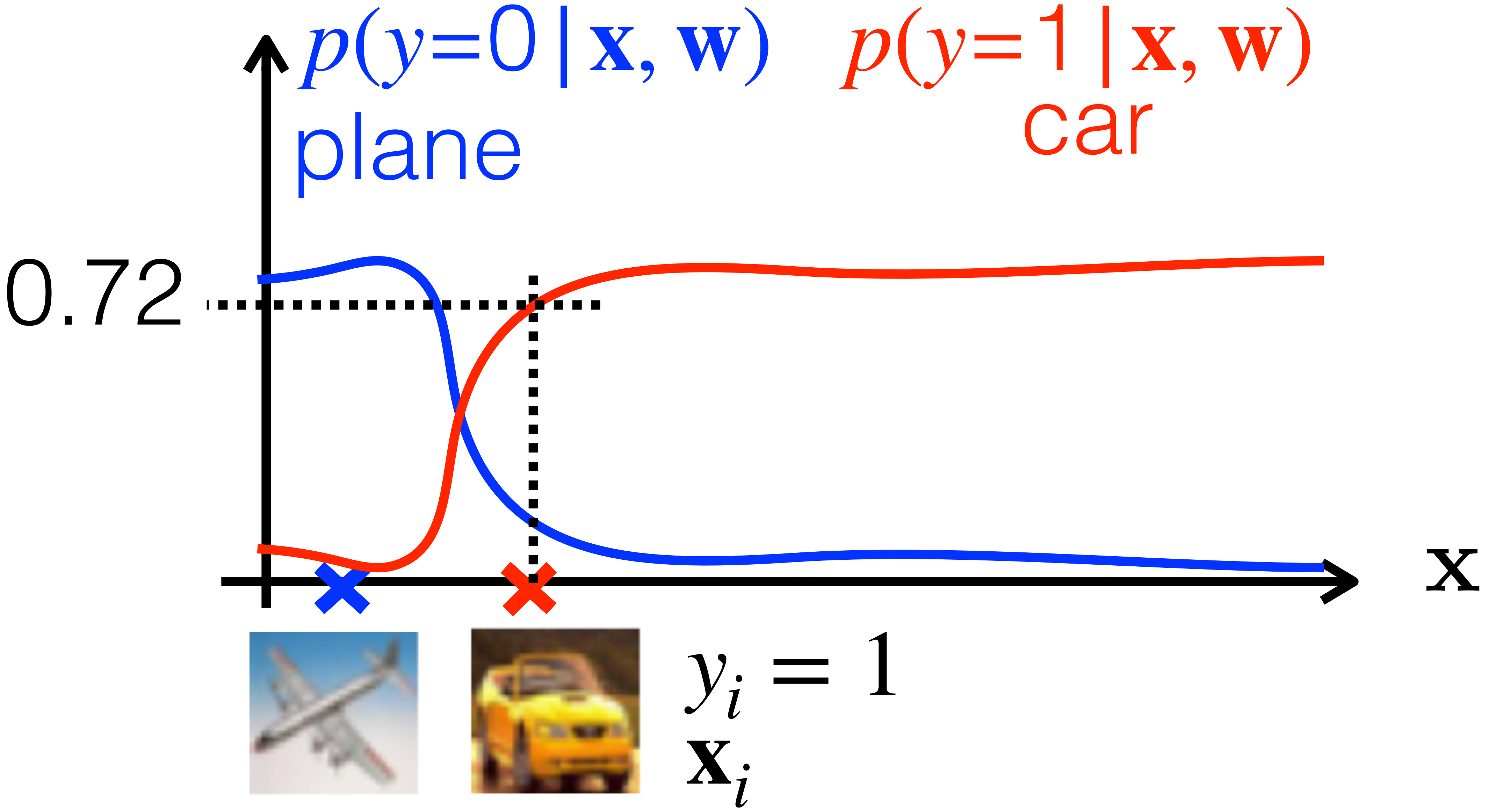
$$p(y\,|\,\mathbf{x},\mathbf{w}) = \begin{cases} \sigma(f(\mathbf{x},\mathbf{w})) & y = 1 \\ 1 - \sigma(f(\mathbf{x},\mathbf{w})) & y = 0 \end{cases} = y \cdot \sigma(f(\mathbf{x},\mathbf{w})) + (1-y)\cdot(1-\sigma(f(\mathbf{x},\mathbf{w})))$$

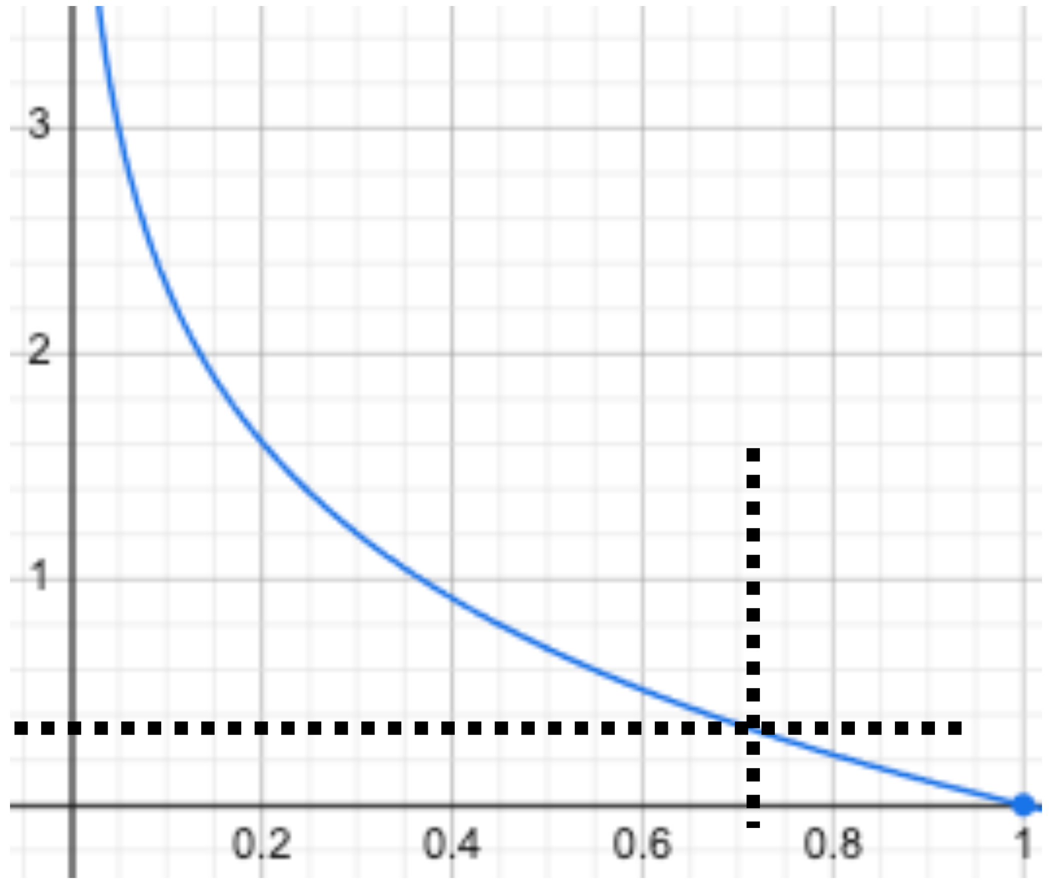
Loss function (subst. car ✖):

$$\mathscr{L}(\mathbf{w}) = \; -\log p(y_i\,|\,\mathbf{x}_i,\mathbf{w})$$

$$= -\log\big(y_i \cdot \sigma(f(\mathbf{x},\mathbf{w})) + (1-y_i)\cdot \sigma(f(\mathbf{x},\mathbf{w}))\big)$$

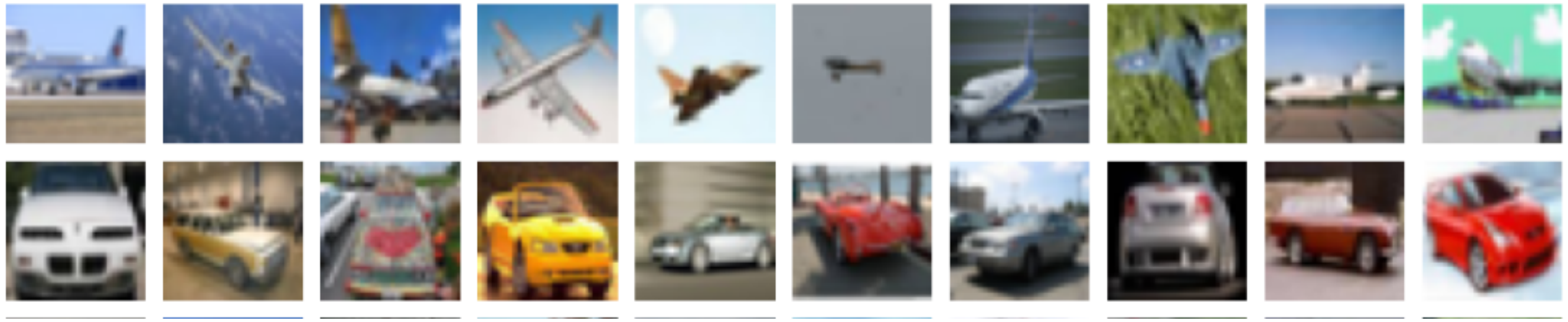
$$= -\log\big(1 \cdot 0.21 + (1-1)\cdot 0.21\big) = 1.56$$





Learning:

$$\arg\min_{\mathbf{w}} \mathscr{L}(\mathbf{w})$$

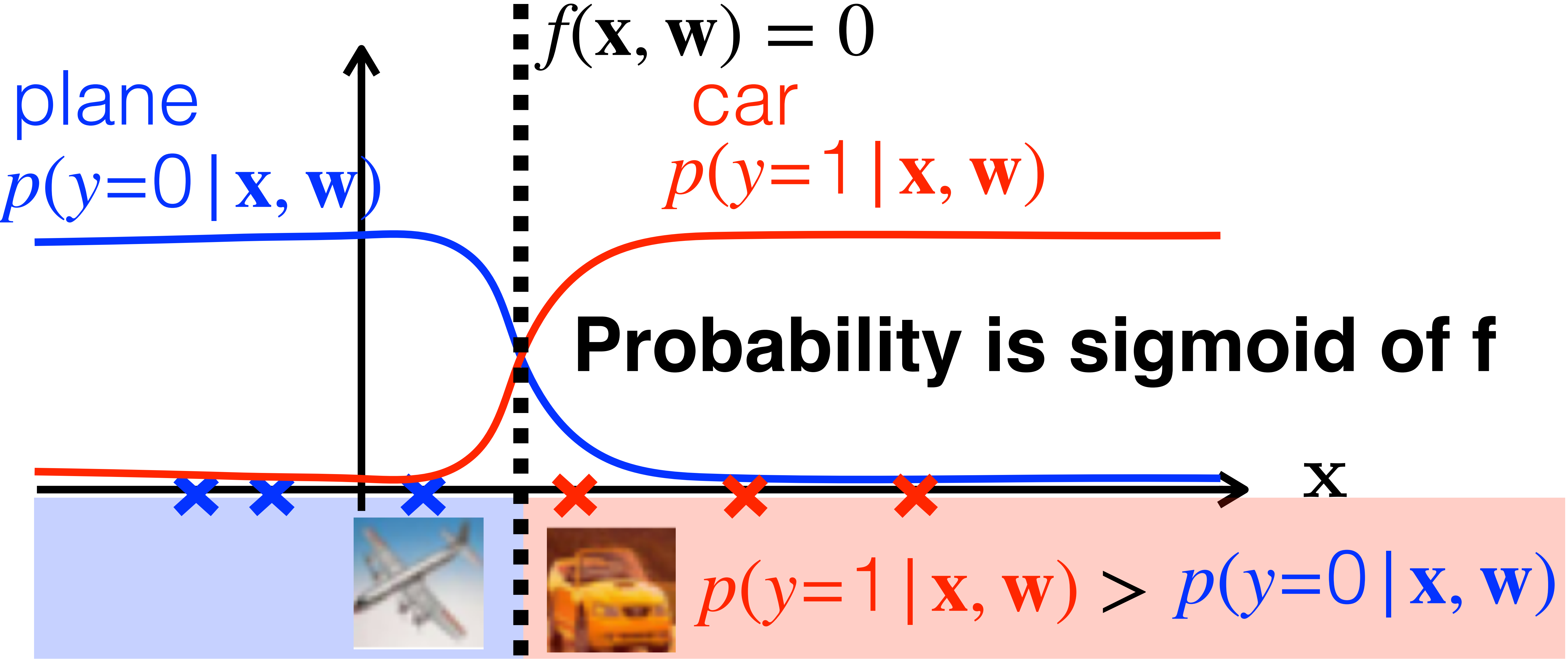


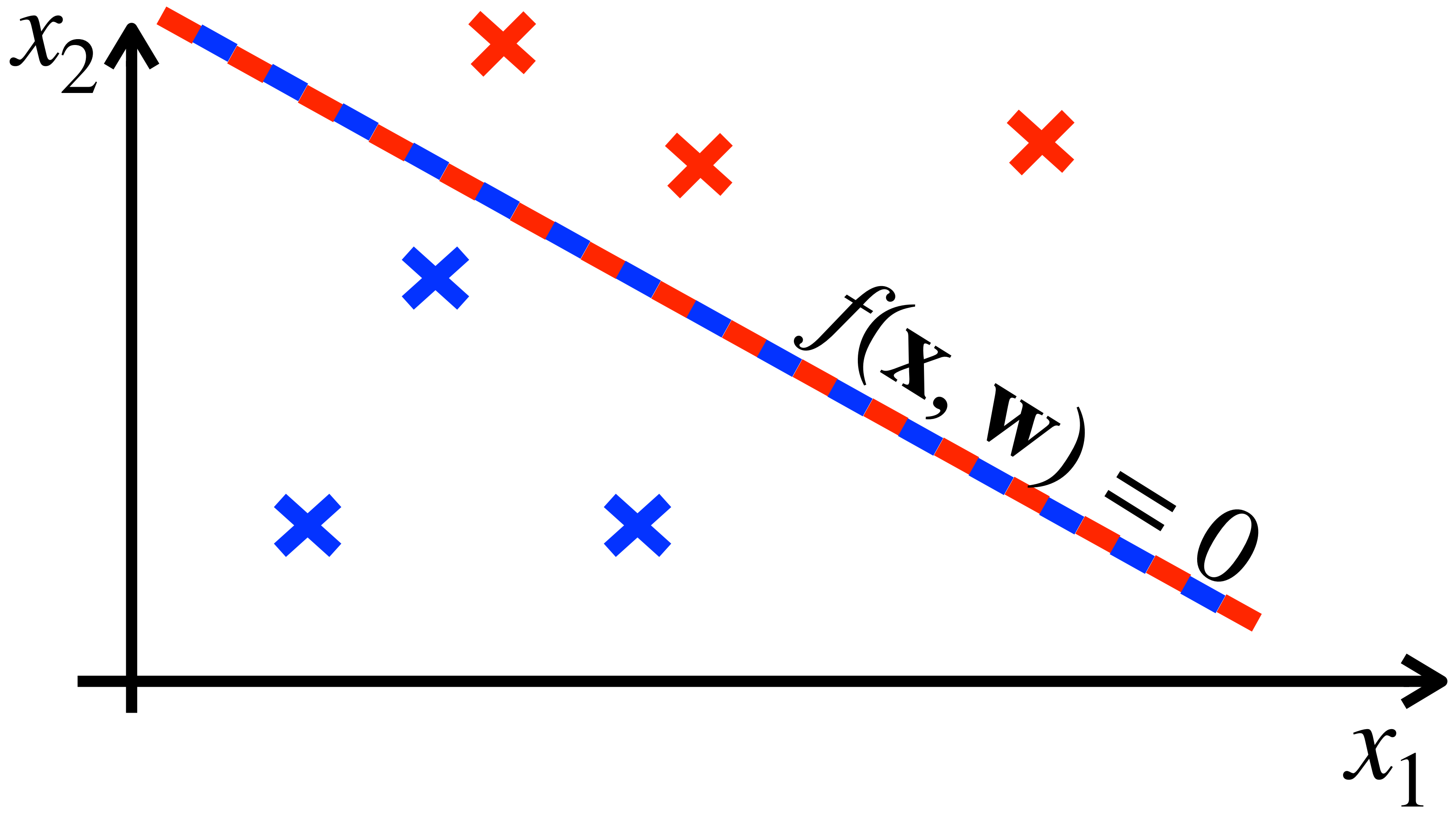
$$p(y\,|\,\mathbf{x},\mathbf{w}) = \begin{cases} \sigma(f(\mathbf{x},\mathbf{w})) & y = 1 \\ 1 - \sigma(f(\mathbf{x},\mathbf{w})) & y = 0 \end{cases} = y \cdot \sigma(f(\mathbf{x},\mathbf{w})) + (1-y)\cdot(1-\sigma(f(\mathbf{x},\mathbf{w})))$$

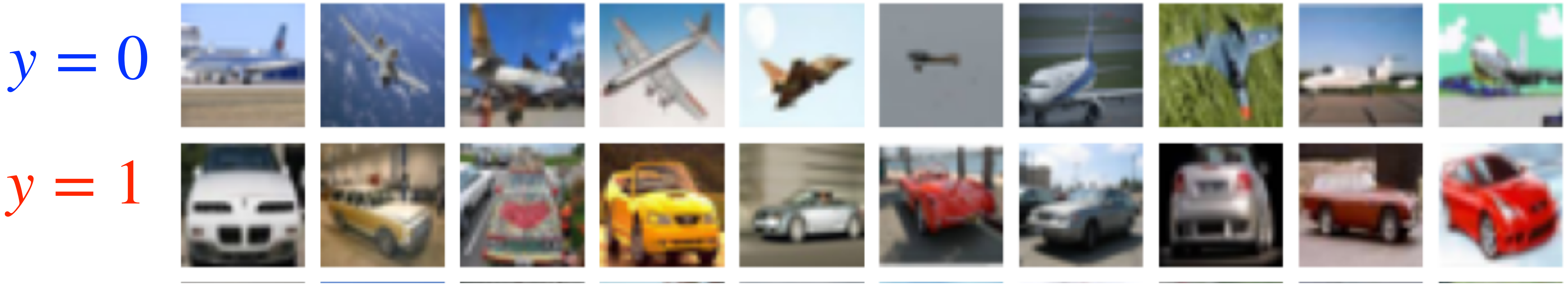
1D linear classifier

2D linear classifier

Which value of f corresponds to this threshold?



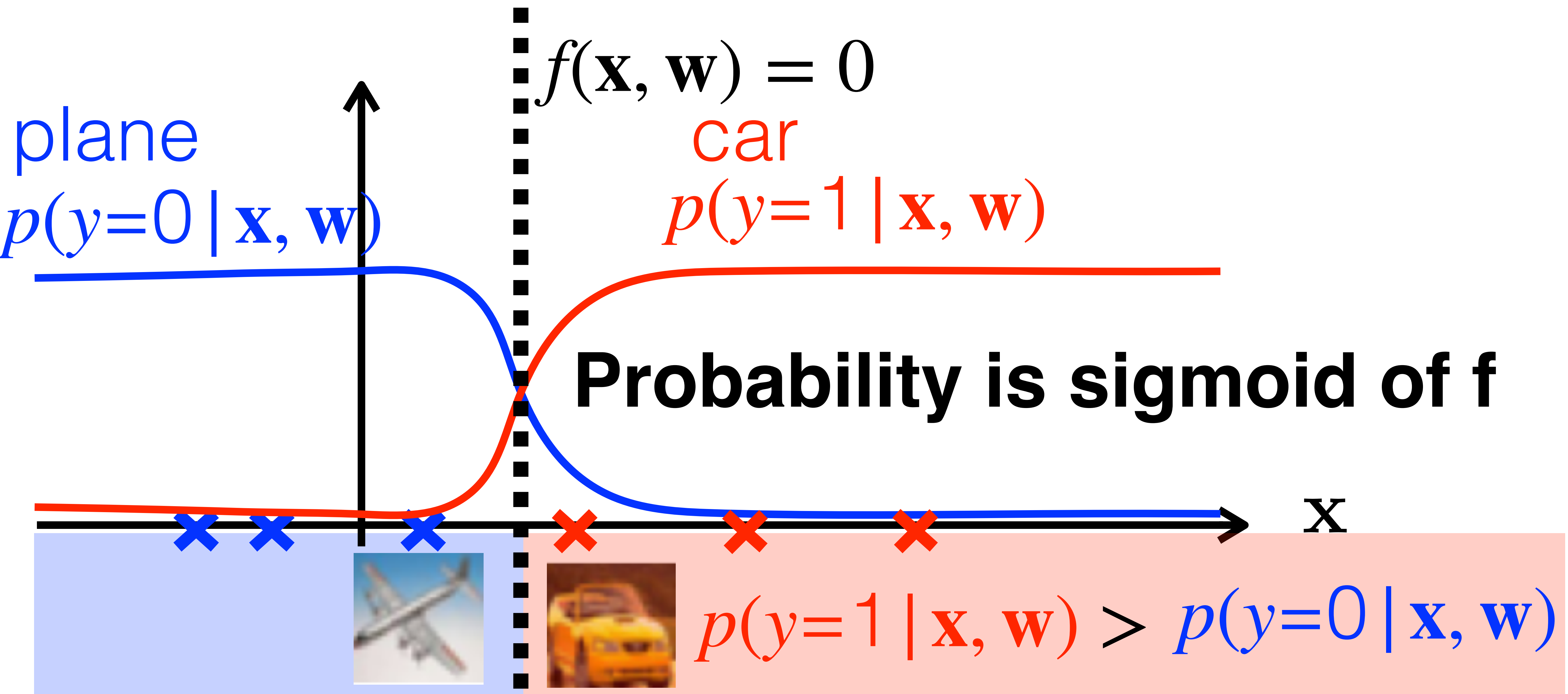




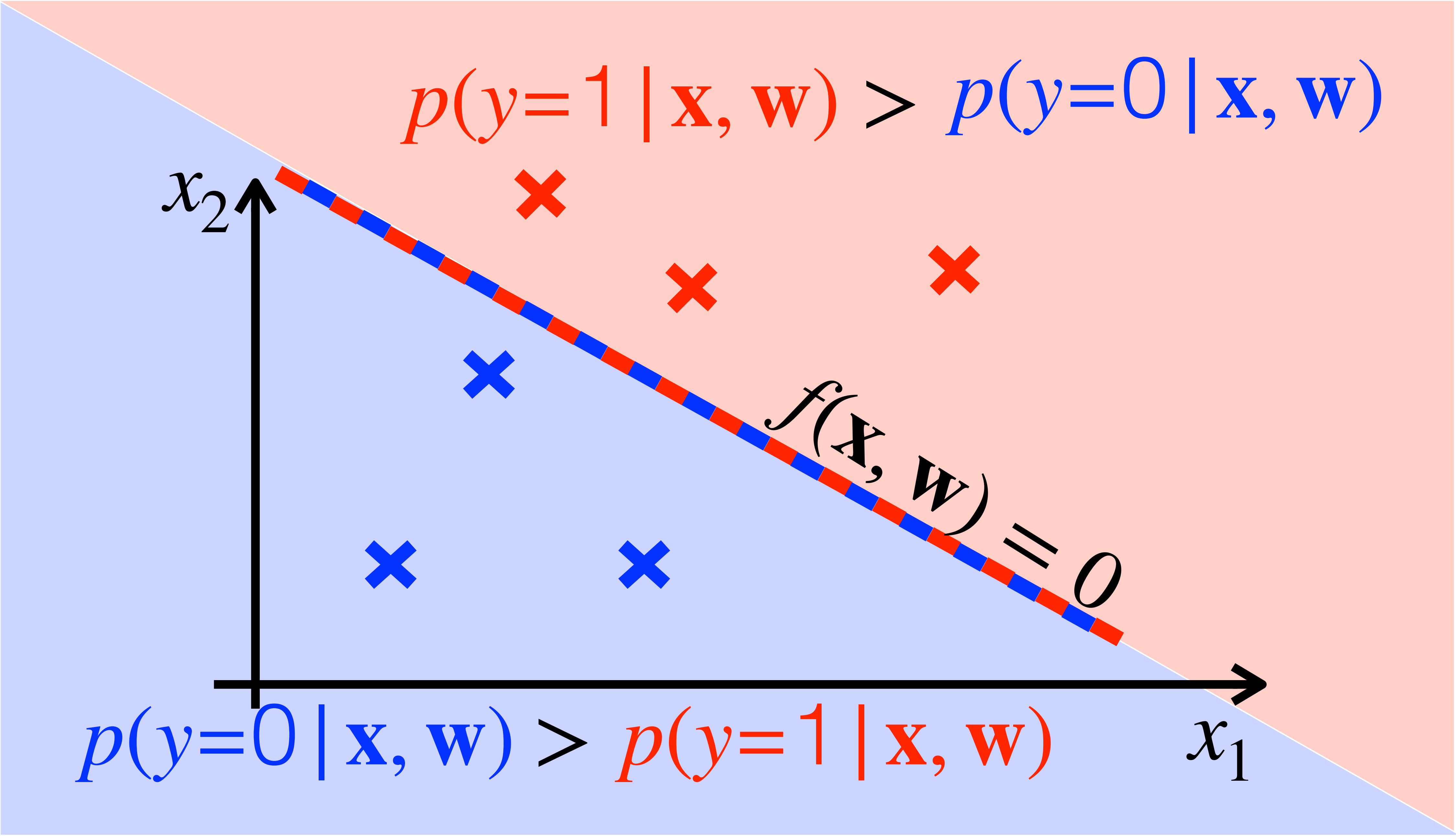
$$p(y\,|\,\mathbf{x},\mathbf{w}) = \begin{cases} \sigma(f(\mathbf{x},\mathbf{w})) & y = 1 \\ 1 - \sigma(f(\mathbf{x},\mathbf{w})) & y = 0 \end{cases} = y \cdot \sigma(f(\mathbf{x},\mathbf{w})) + (1-y)\cdot(1-\sigma(f(\mathbf{x},\mathbf{w})))$$

Let's implement it !!!

1D linear classifier



2D linear classifier



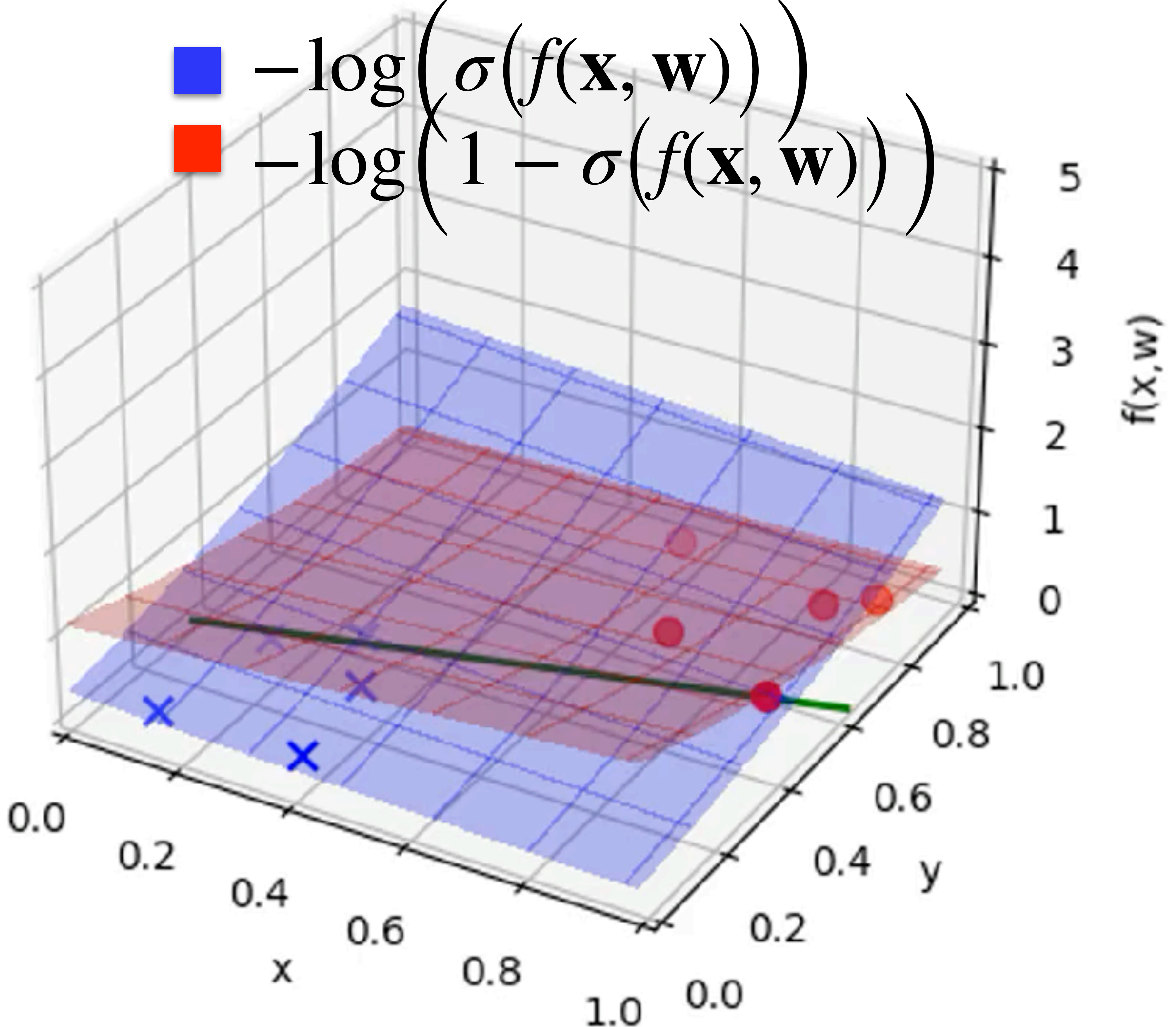
Numpy implementation

```
for i in range(30):
    f = w[0] * x[:, 0] + w[1] * x[:, 1] + w[2]
    p = sigmoid(u)
    loss = (-np.log(p)*y + -np.log(1-p)*(1-y)).sum()
    grad = -1/p * sigmoid(f) * (1-sigmoid(f)) * …
    w = w - 0.1 * grad
```

$$\mathscr{L}(\mathbf{W}) = -\log(y \cdot \sigma(f(\mathbf{x}, \mathbf{w})) + (1 - y) \cdot (1 - \sigma(f(\mathbf{x}, \mathbf{w})))$$

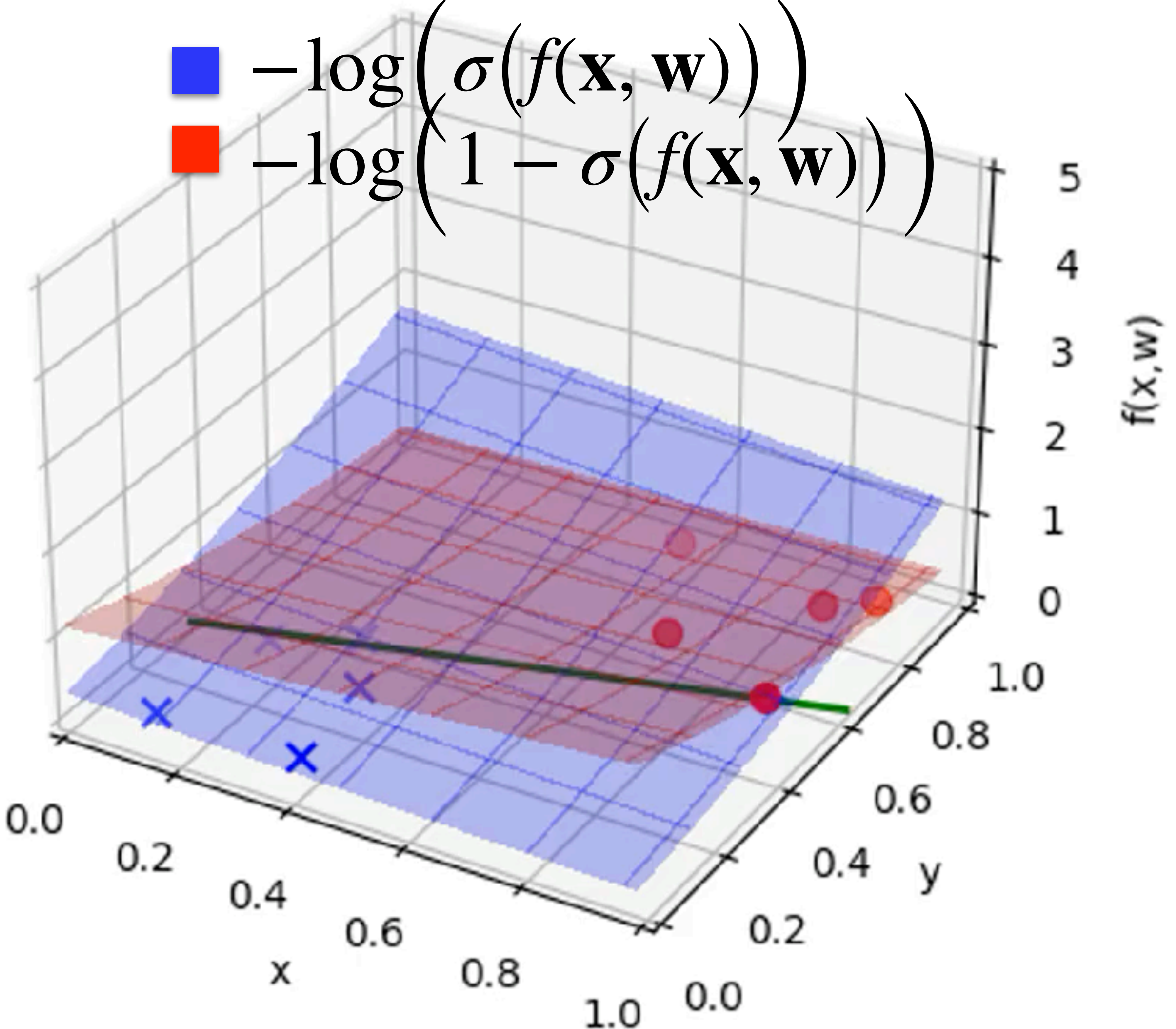
Numpy implementation

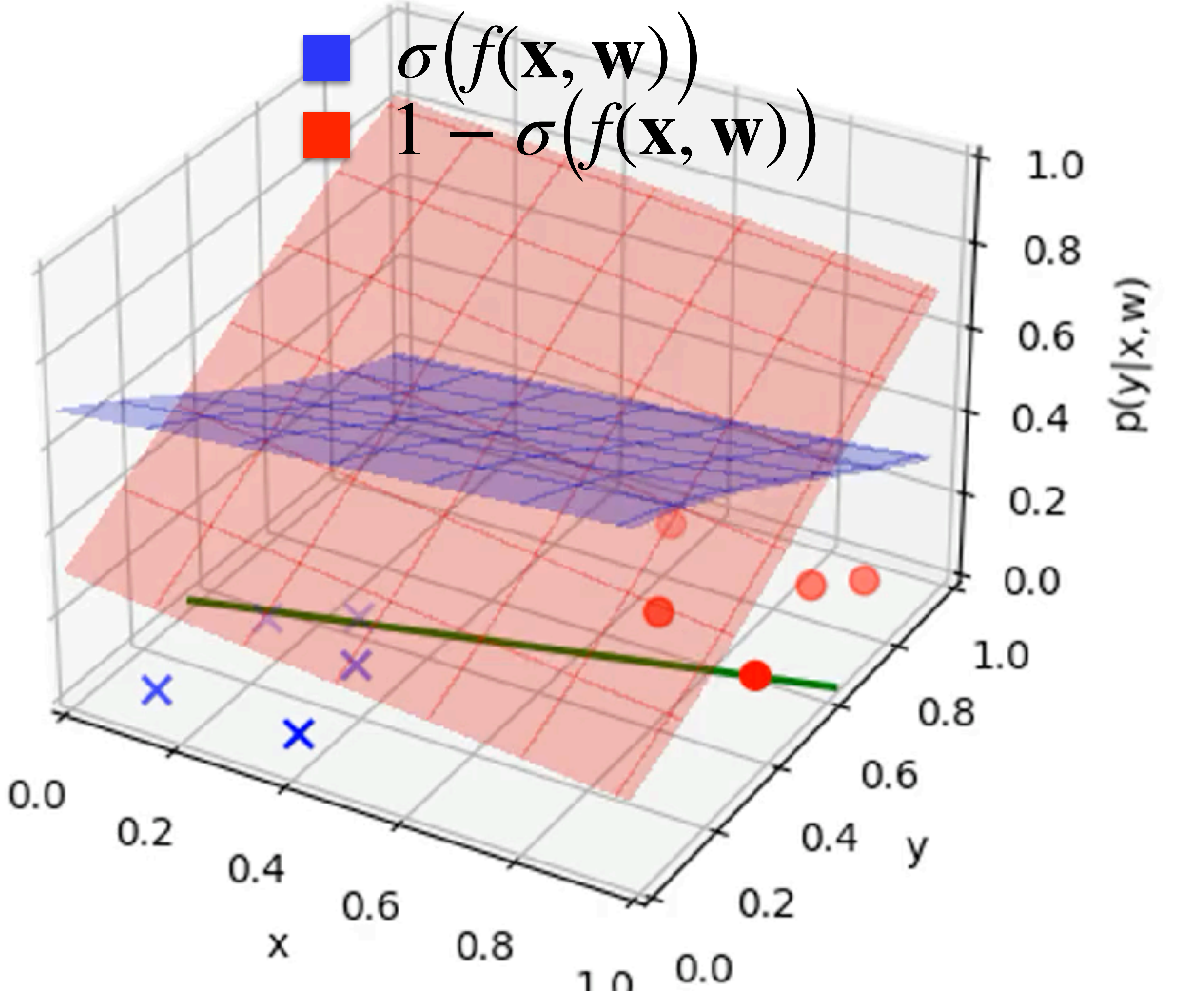
```
for i in range(30):
    f = w[0] * x[:, 0] + w[1] * x[:, 1] + w[2]
    p = sigmoid(u)
    loss = (-np.log(p)*y + -np.log(1-p)*(1-y)).sum()
    grad = -1/p * sigmoid(f) * (1-sigmoid(f)) * …
    w = w - 0.1 * grad
```

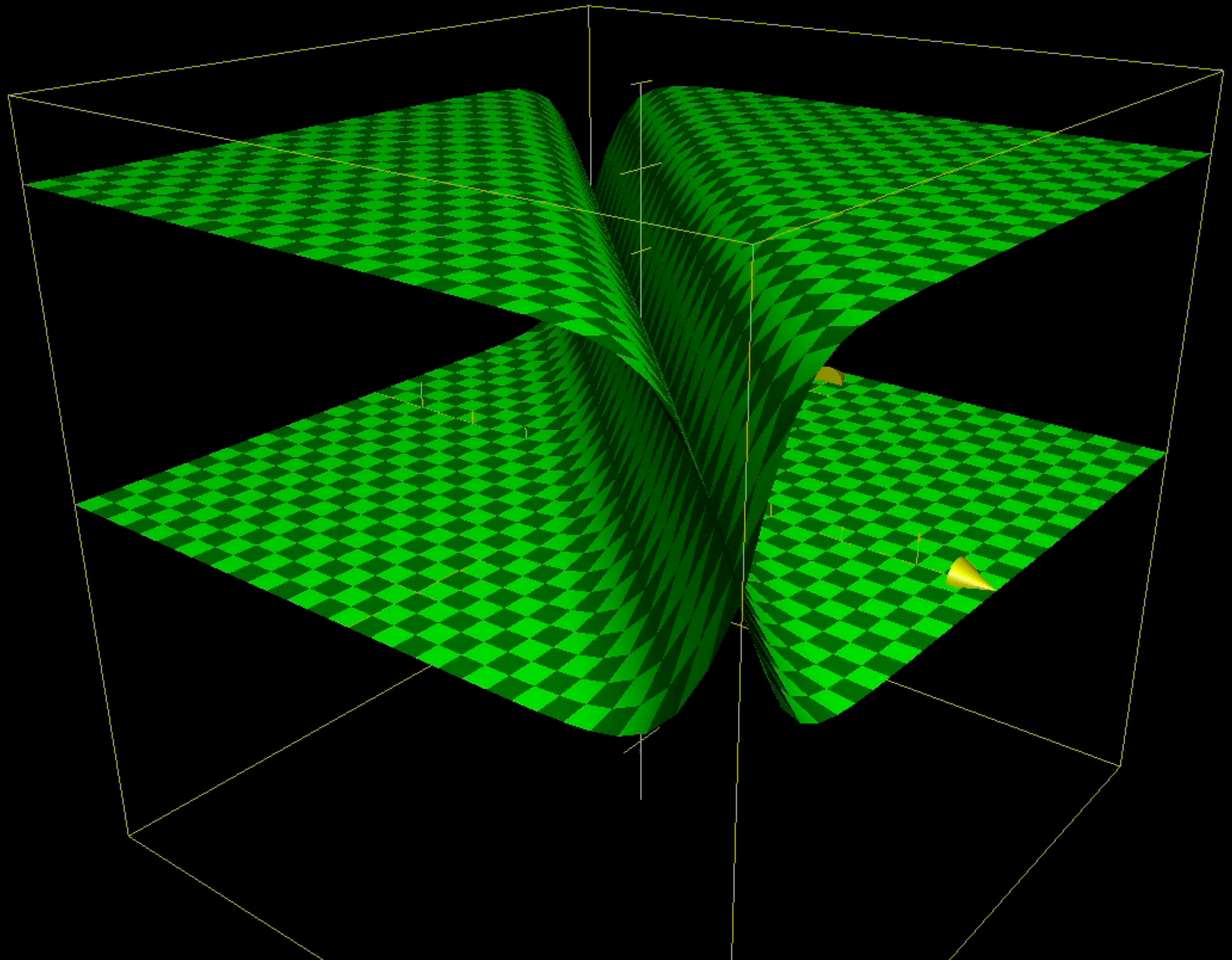


Numpy implementation

```
for i in range(30):
    f = w[0] * x[:, 0] + w[1] * x[:, 1] + w[2]
    p = sigmoid(u)
    loss = (-np.log(p)*y + -np.log(1-p)*(1-y)).sum()
    grad = -1/p * sigmoid(f) * (1-sigmoid(f)) * …
    w = w - 0.1 * grad
```

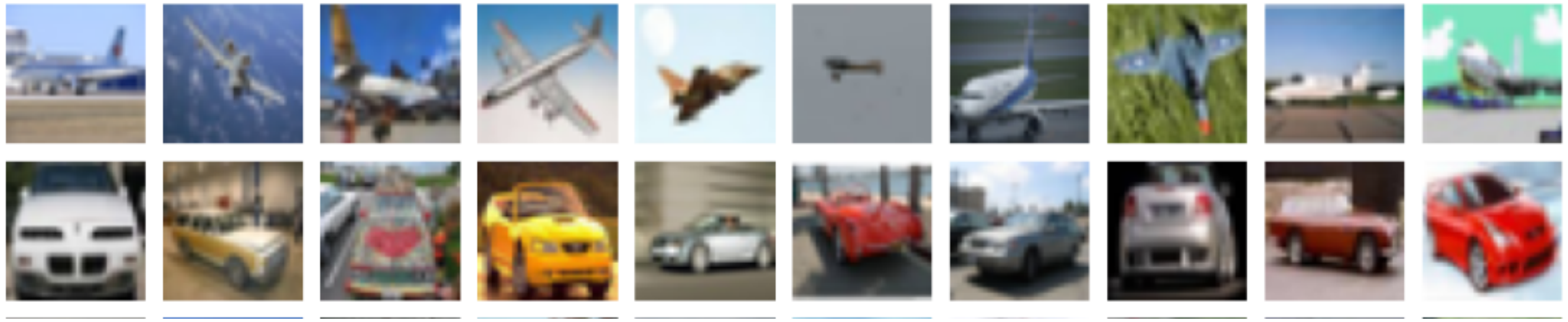






$y = 0$

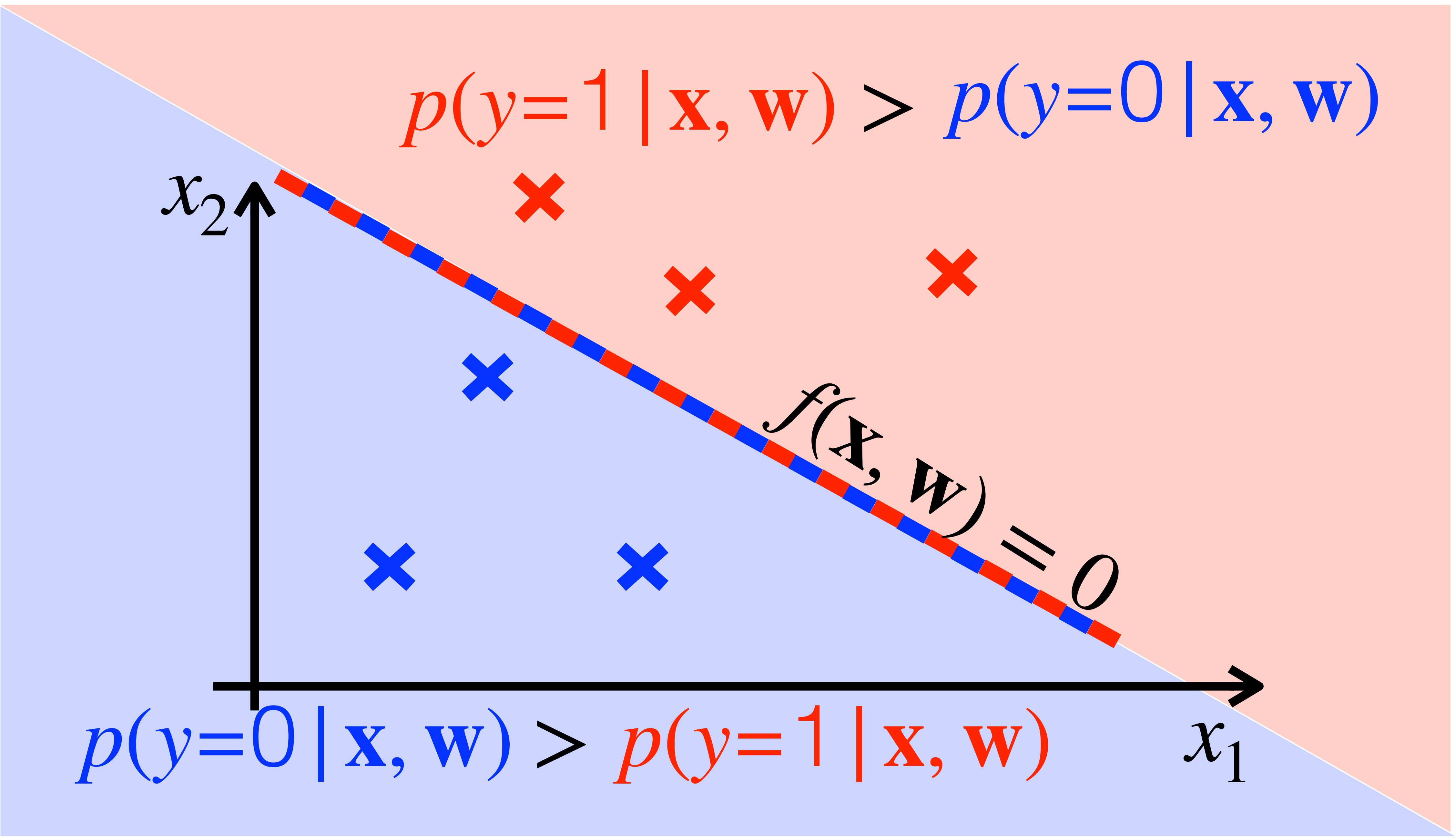
$y = 1$



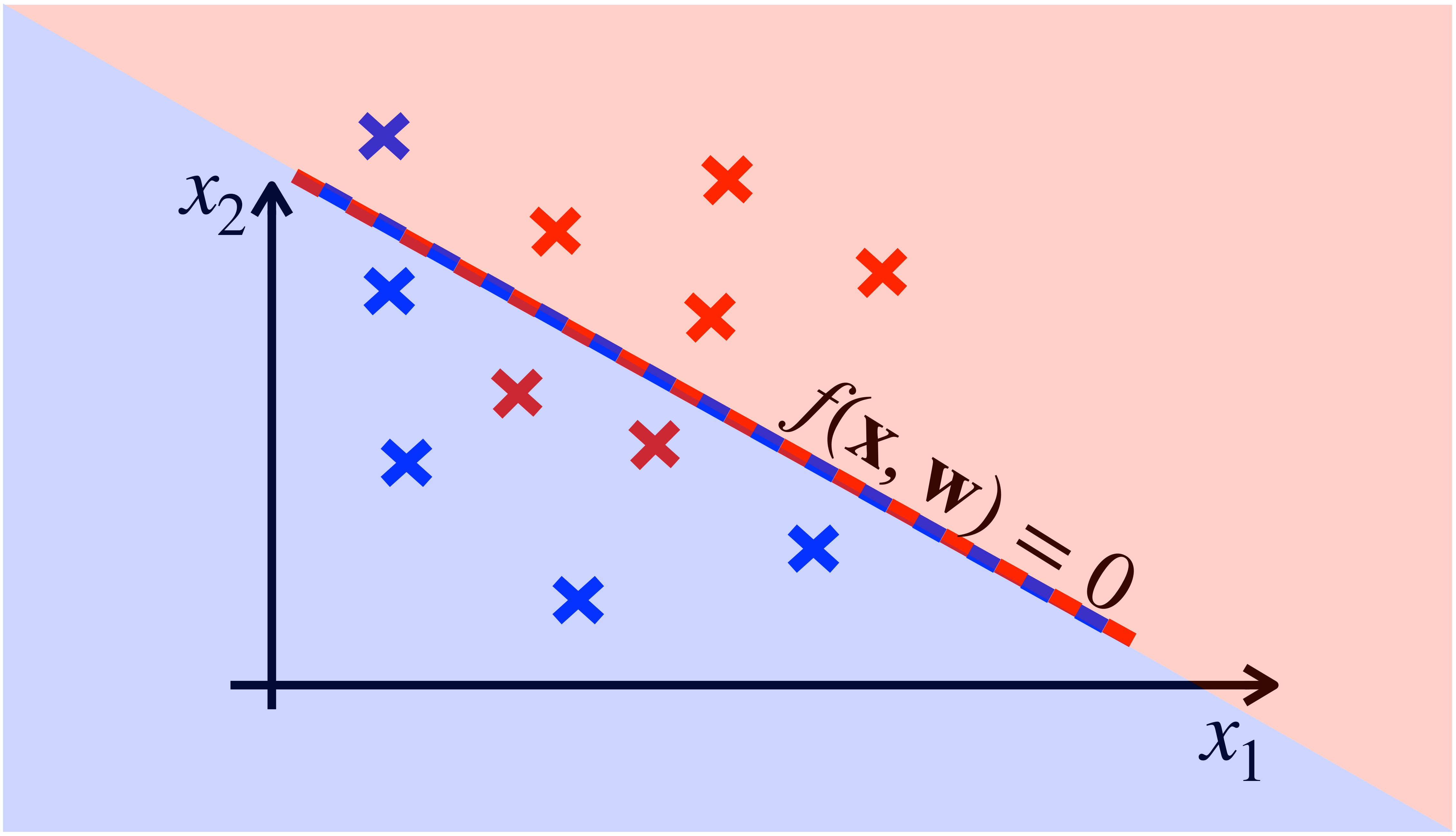
$$p(y\,|\,\mathbf{x}, \mathbf{w}) = \begin{cases} \sigma(f(\mathbf{x}, \mathbf{w})) & y = 1 \\ 1 - \sigma(f(\mathbf{x}, \mathbf{w})) & y = 0 \end{cases} = y_i \cdot \sigma(f(\mathbf{x}, \mathbf{w})) + (1 - y_i) \cdot (1 - \sigma(f(\mathbf{x}, \mathbf{w})))$$

Draw training data that cannot be separated by linear classifier????

2D linear classifier

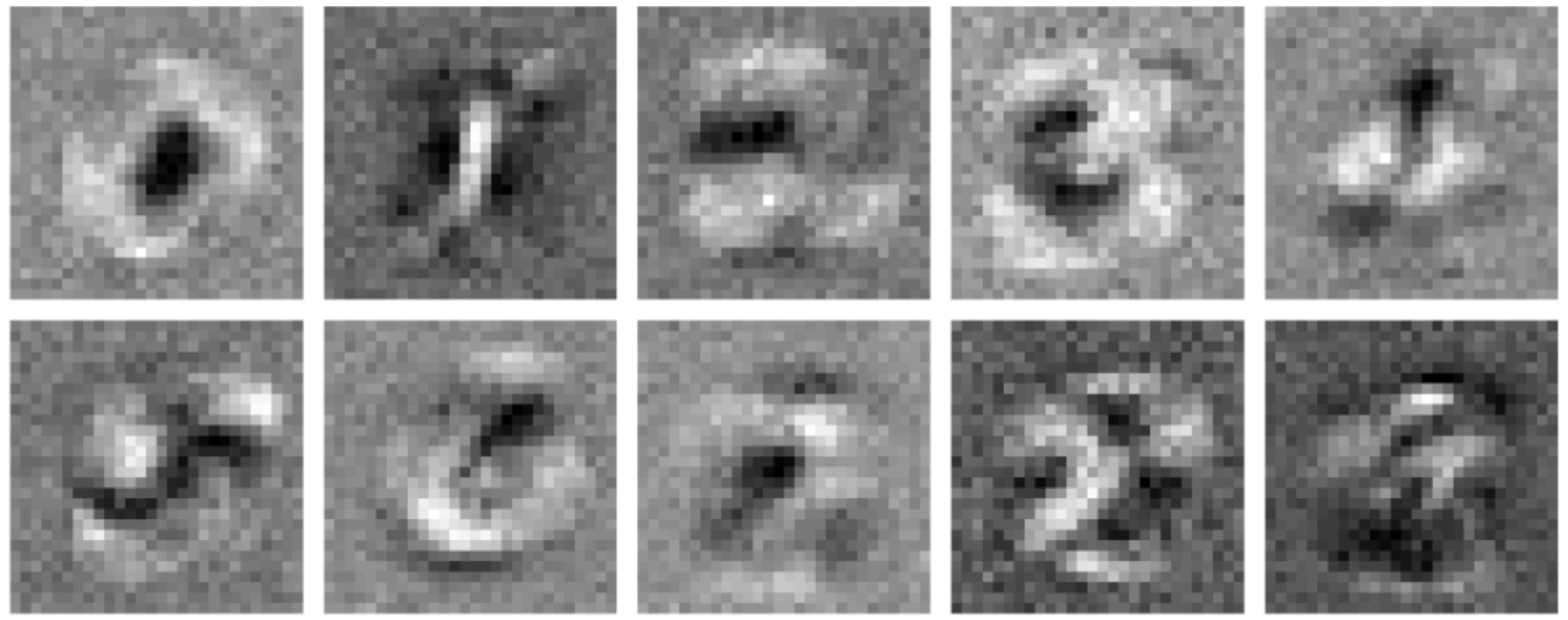


2D linear classifier

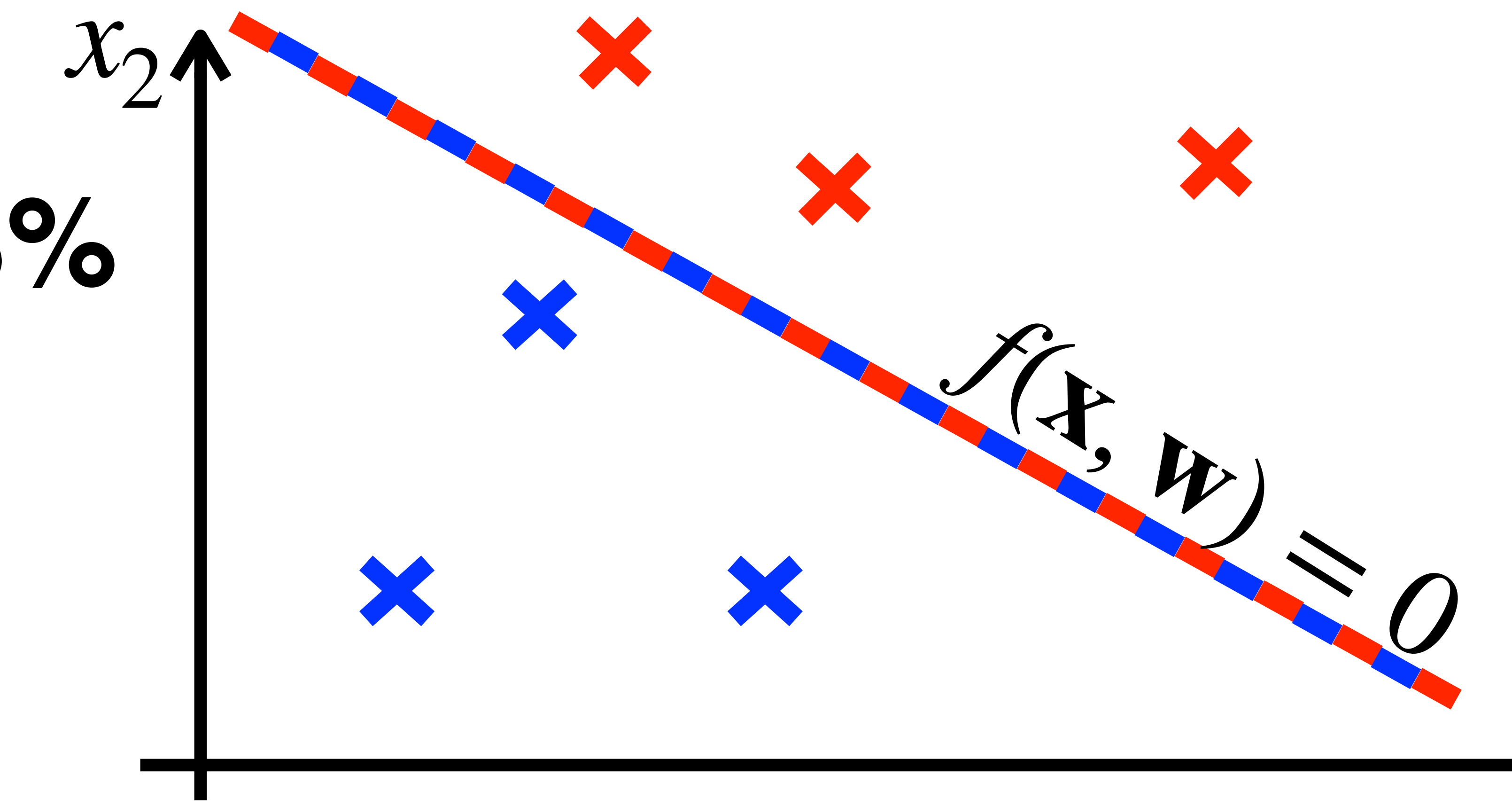


Label | Images | Learned weights | Error | 2D linear classifier

"0"
"1"
"2"
"3"
"4"
"5"
"6"
"7"
"8"
"9"

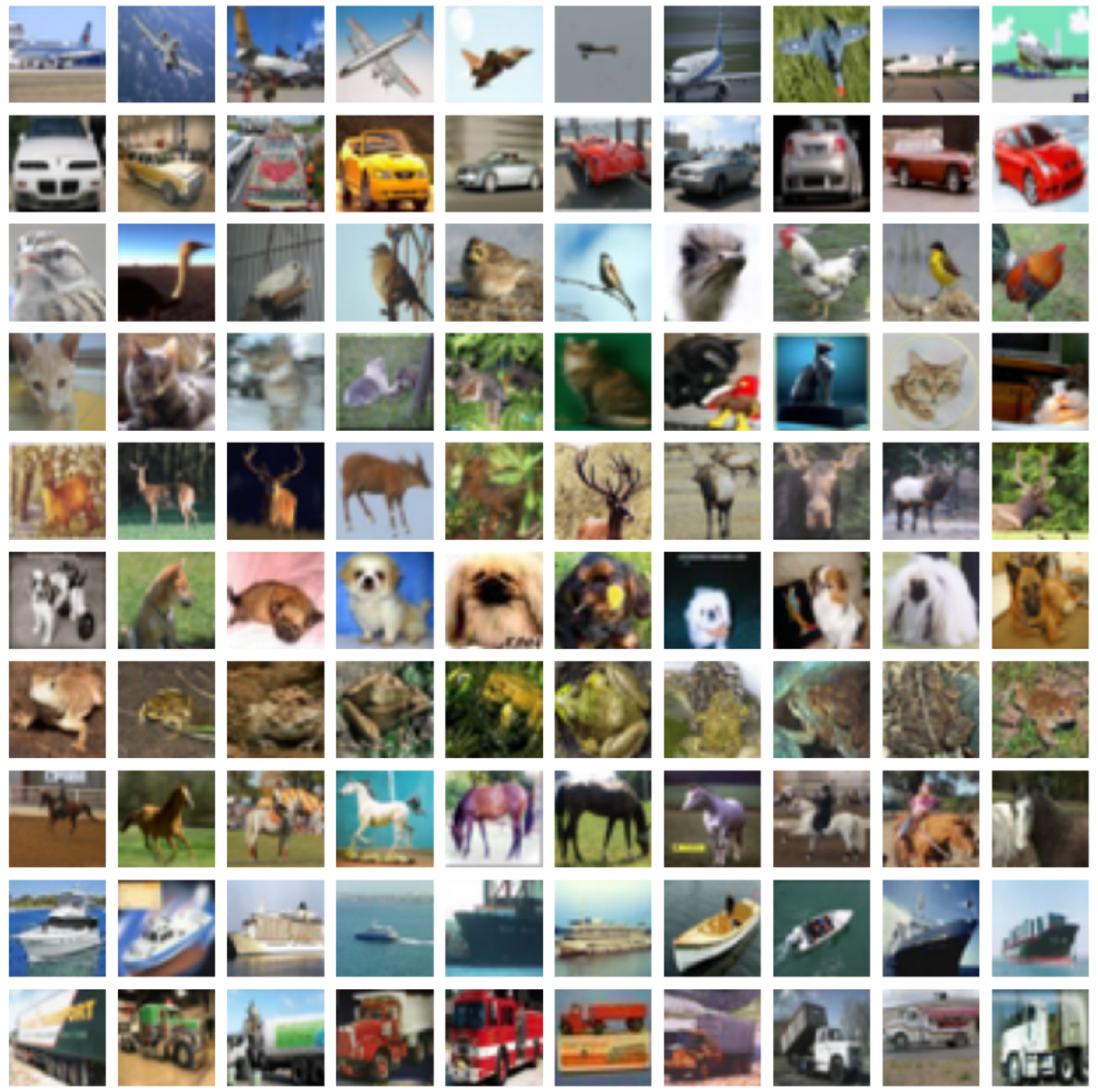


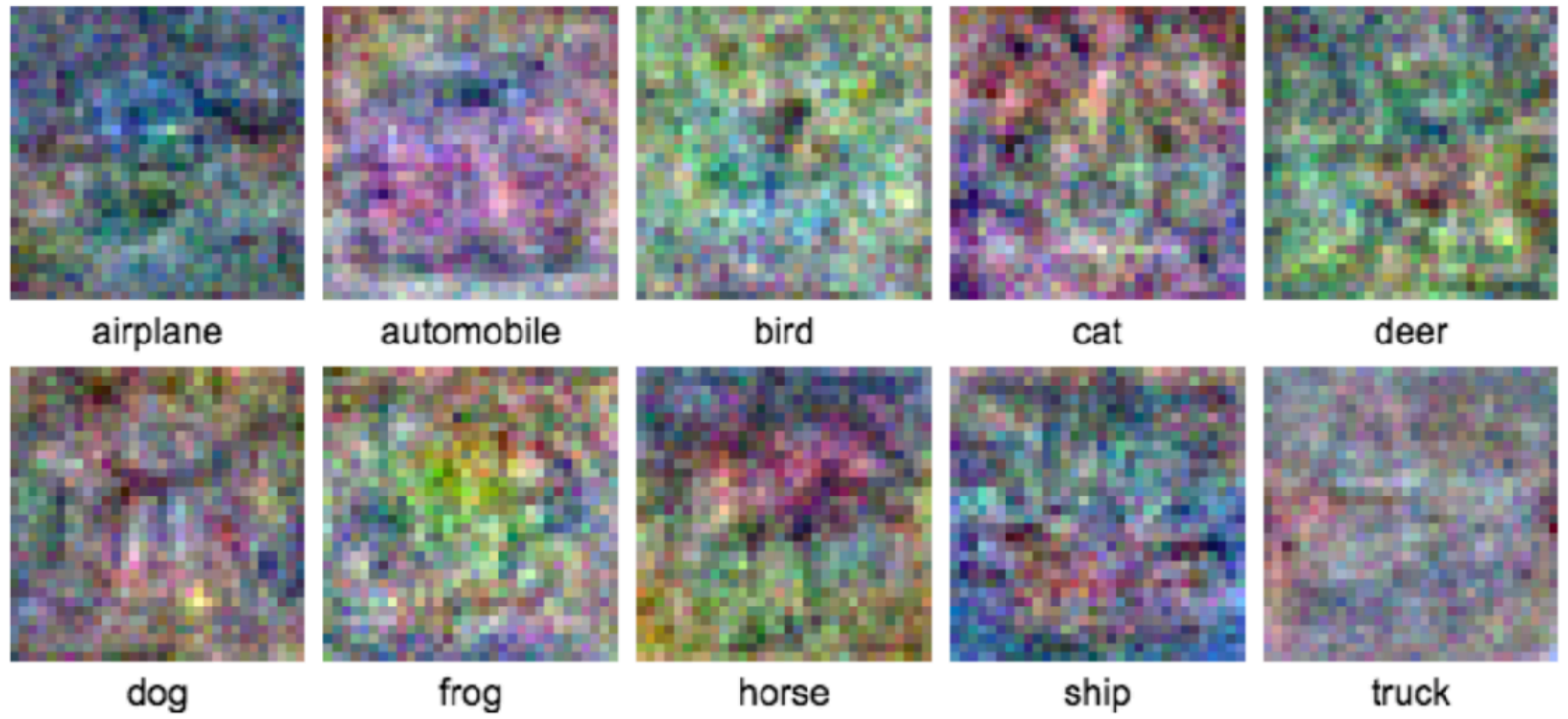
8%



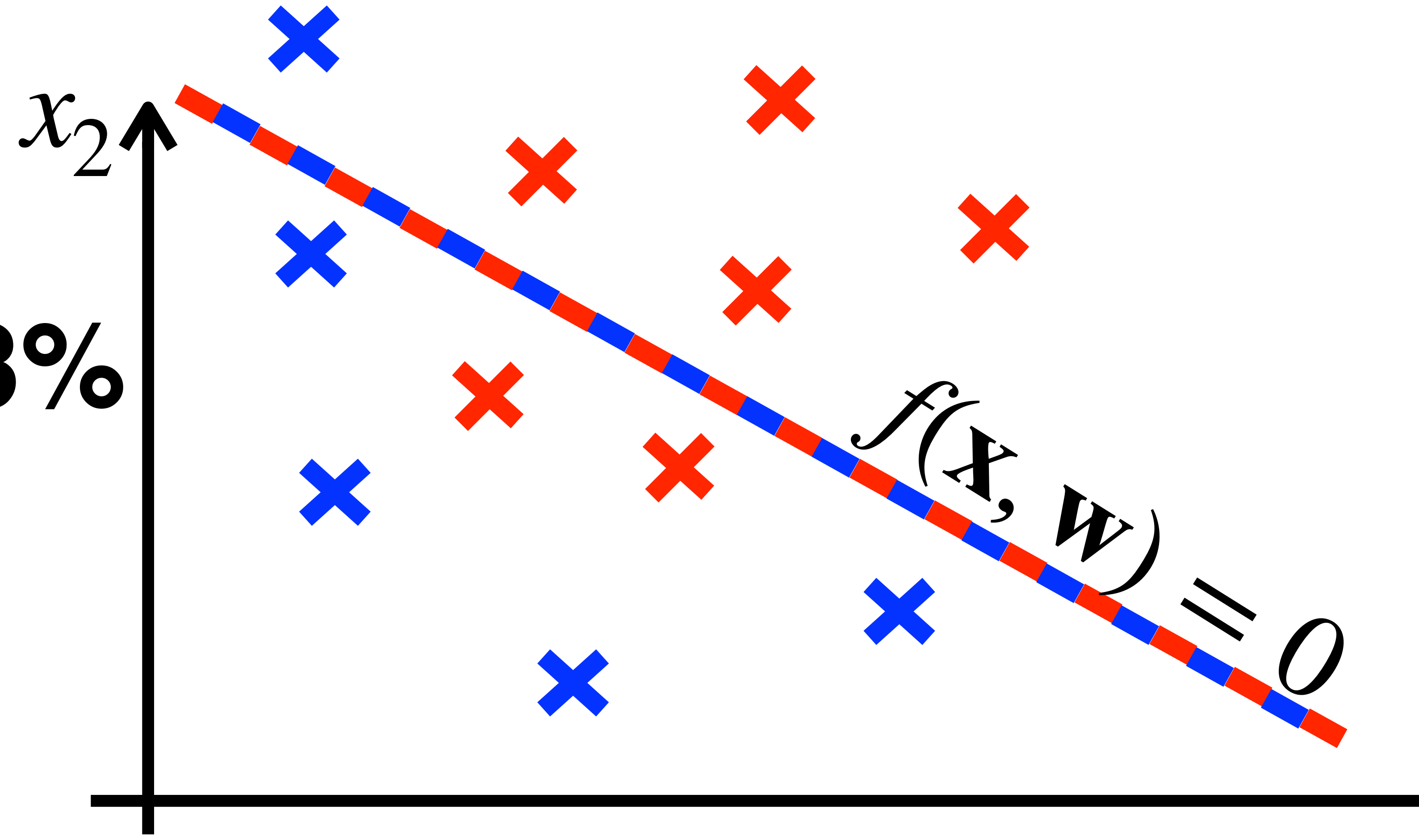
2D linear classifier

airplane
automobile
bird
cat
deer
dog
frog
horse
ship
truck



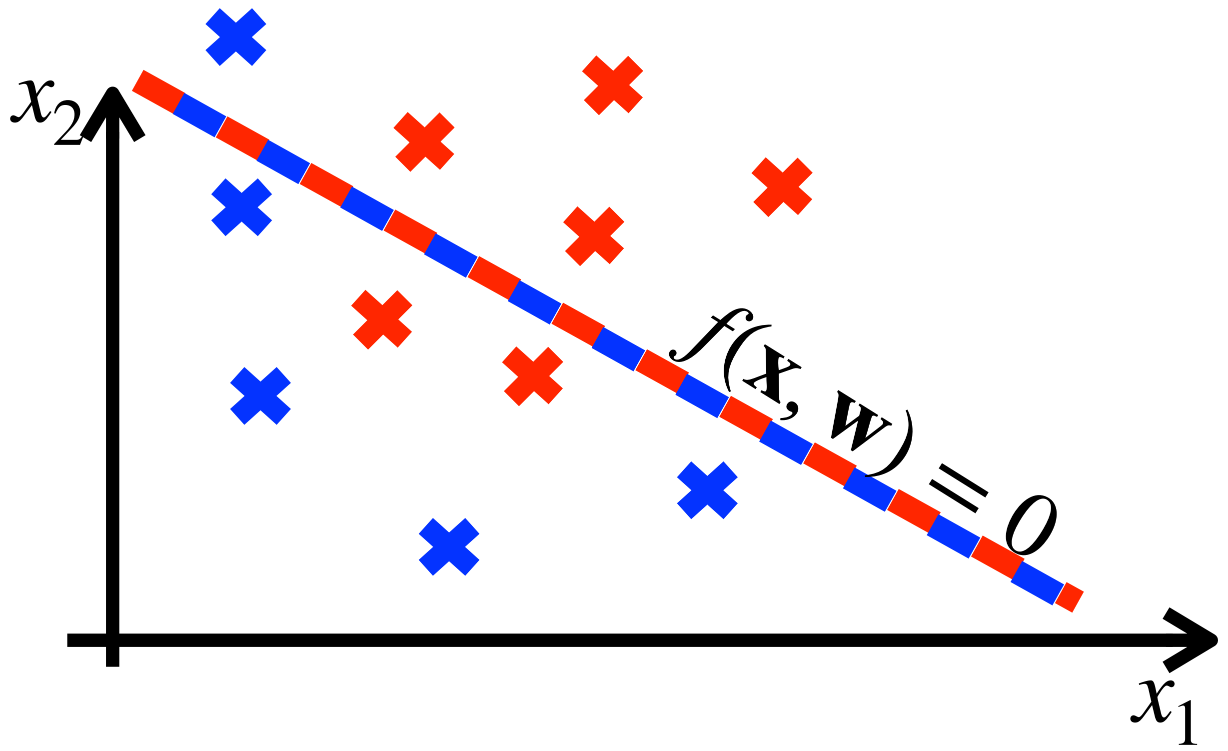


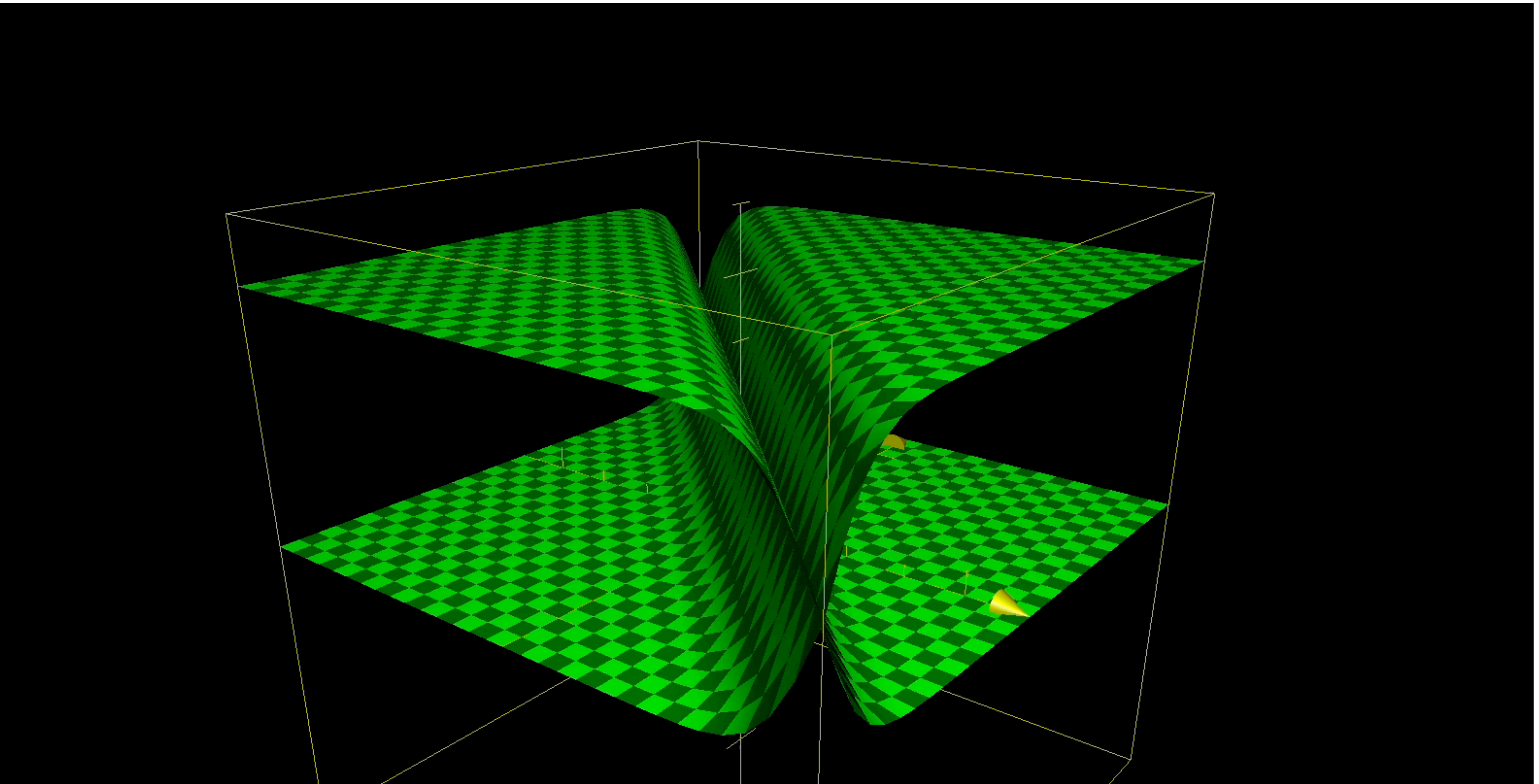
63%



2D linear classifier

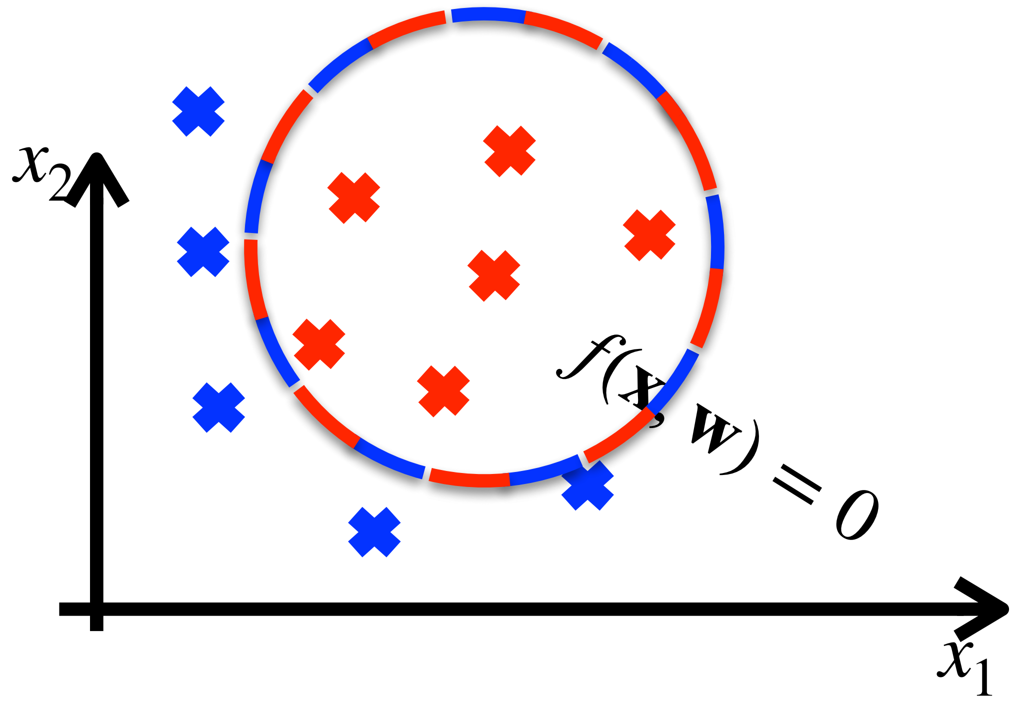
$$f([x_1, x_2], \mathbf{w}) = w_1 x_1 + w_2 x_2 + w_0$$





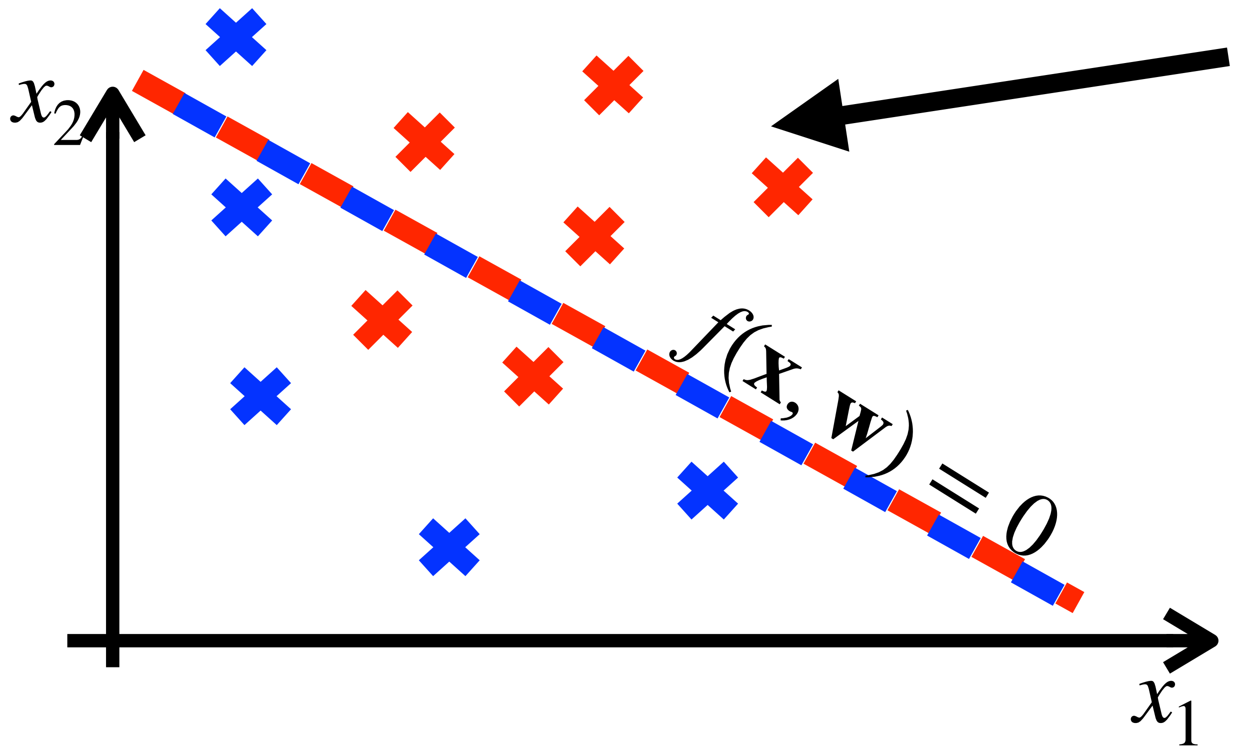
2D non-linear classifier

$$f([x_1, x_2], \mathbf{w}) =$$
$$= w_1 x_1 + w_2 x_2 + w_3 x_1^2 + w_4 x_2^2 + w_5 x_1 x_2 + w_0$$



2D linear classifier

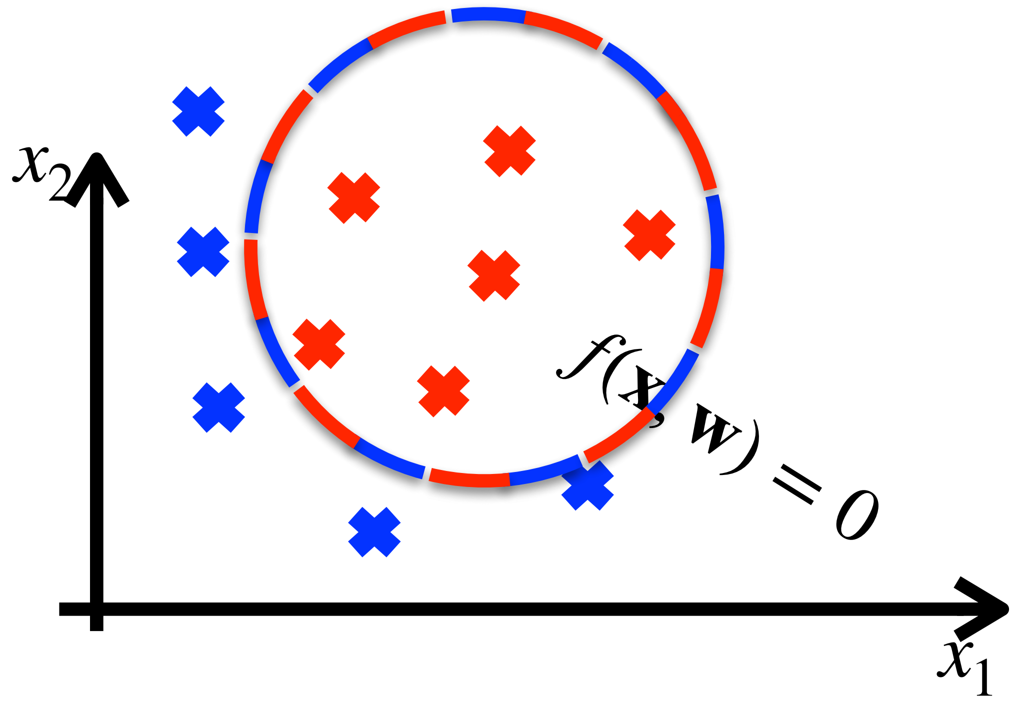
$$f([x_1, x_2], \mathbf{w}) = w_1 x_1 + w_2 x_2 + w_0$$

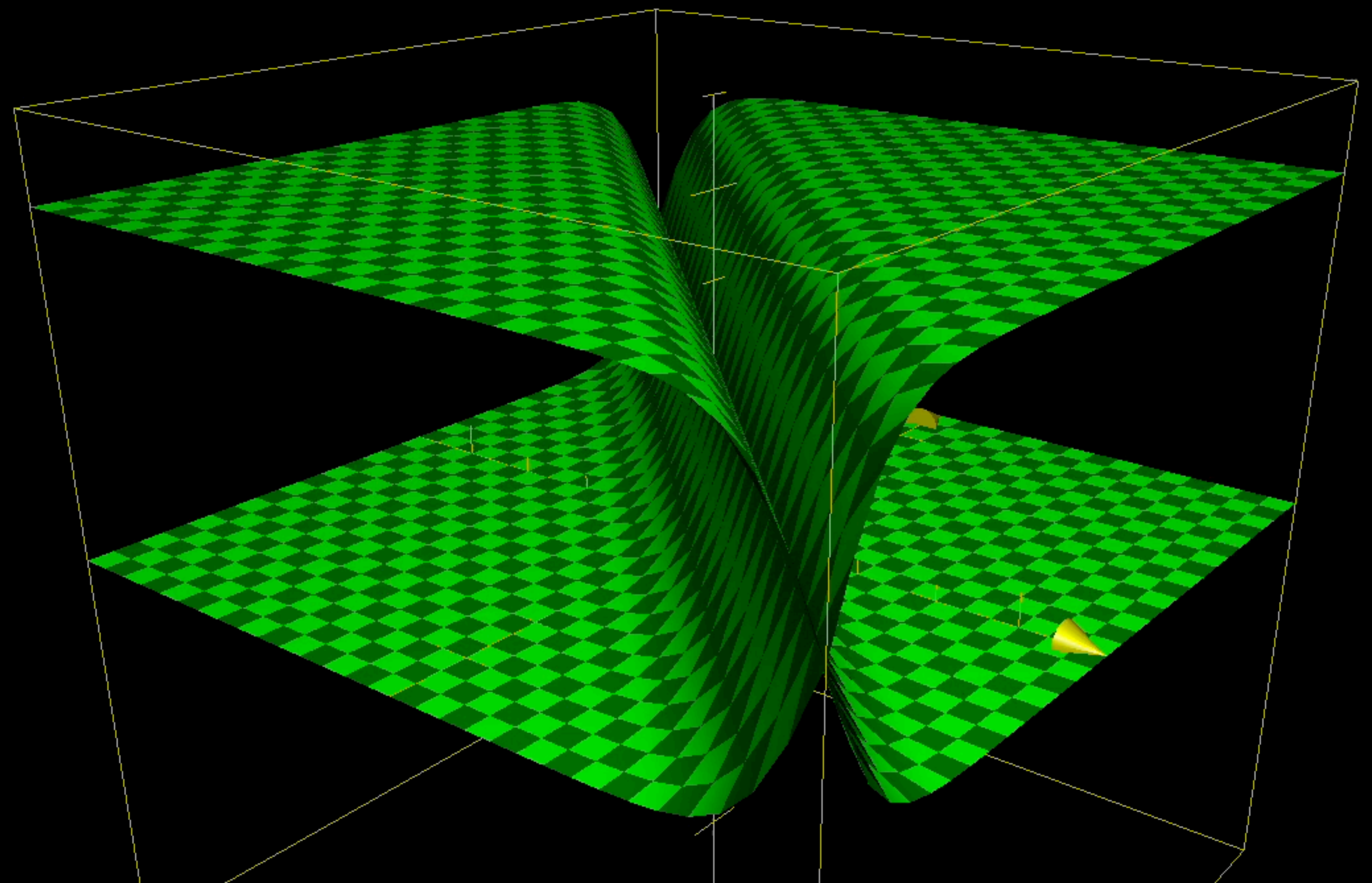


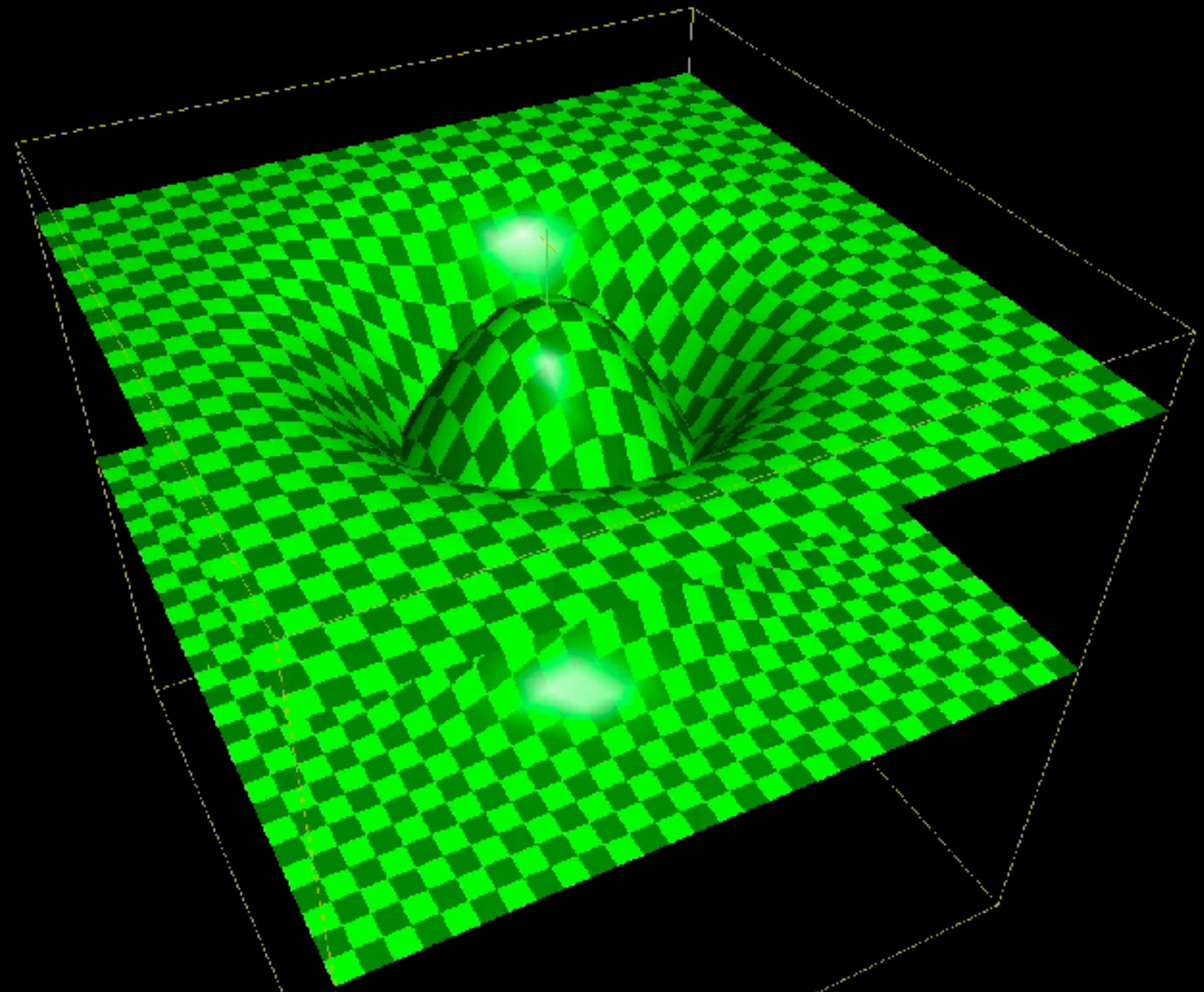
Is there a linear classifier that separates this data?

2D non-linear classifier

$$f([x_1, x_2], \mathbf{w}) =$$
$$= w_1 x_1 + w_2 x_2 + w_3 x_1^2 + w_4 x_2^2 + w_5 x_1 x_2 + w_0$$







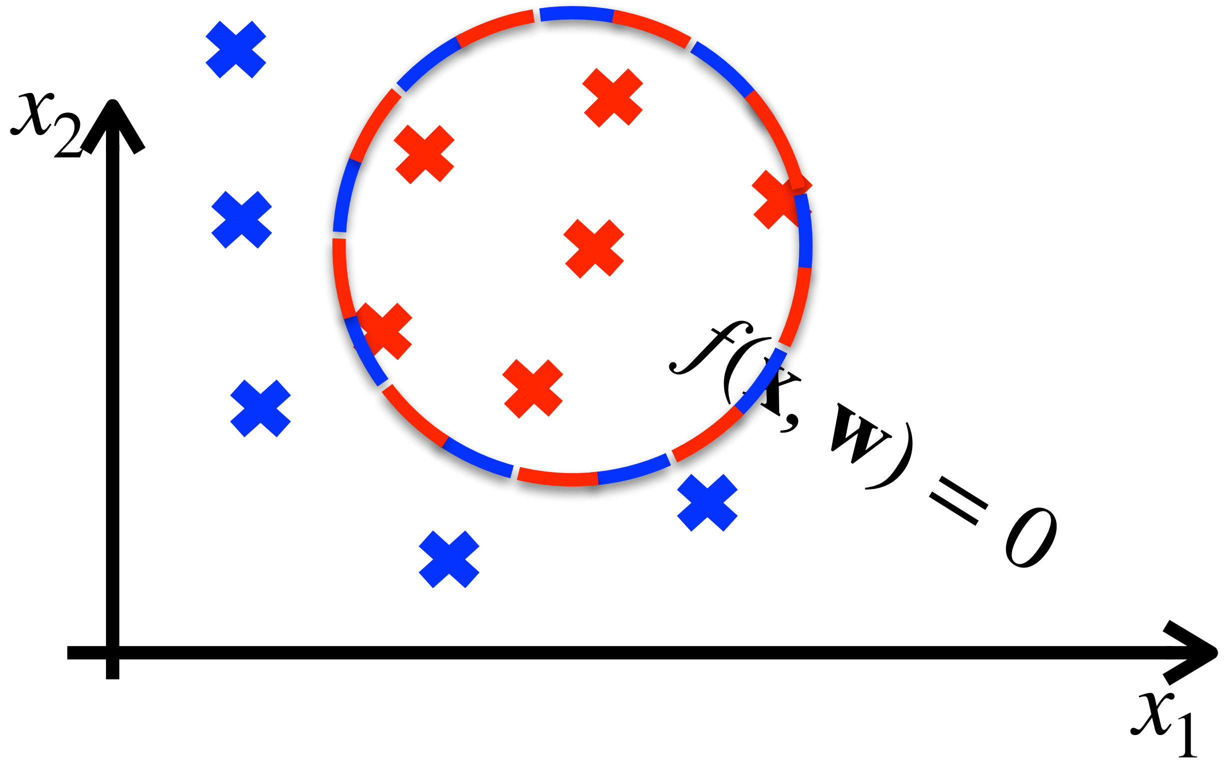
5D linear classifier

$$f([x_1, x_2, x_1^2, x_2^2, x_1x_2], \mathbf{w}) =$$
$$= w_1x_1 + w_2x_2 + w_3x_1^2 + w_4x_2^2 + w_5x_1x_2 + w_0$$

$\Leftrightarrow$

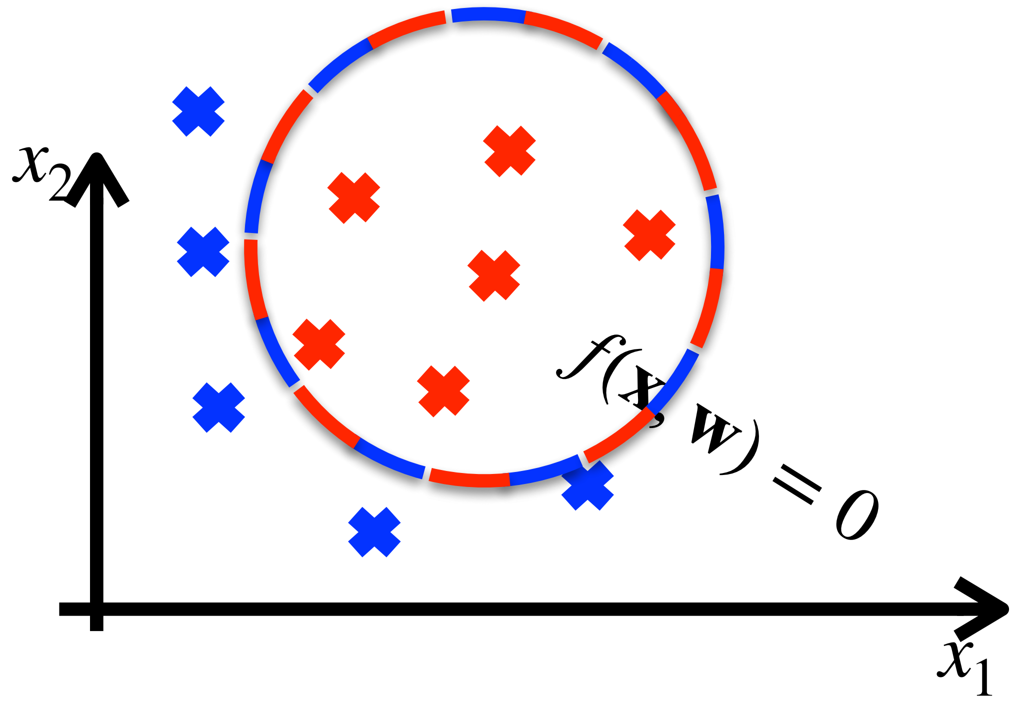
2D non-linear classifier

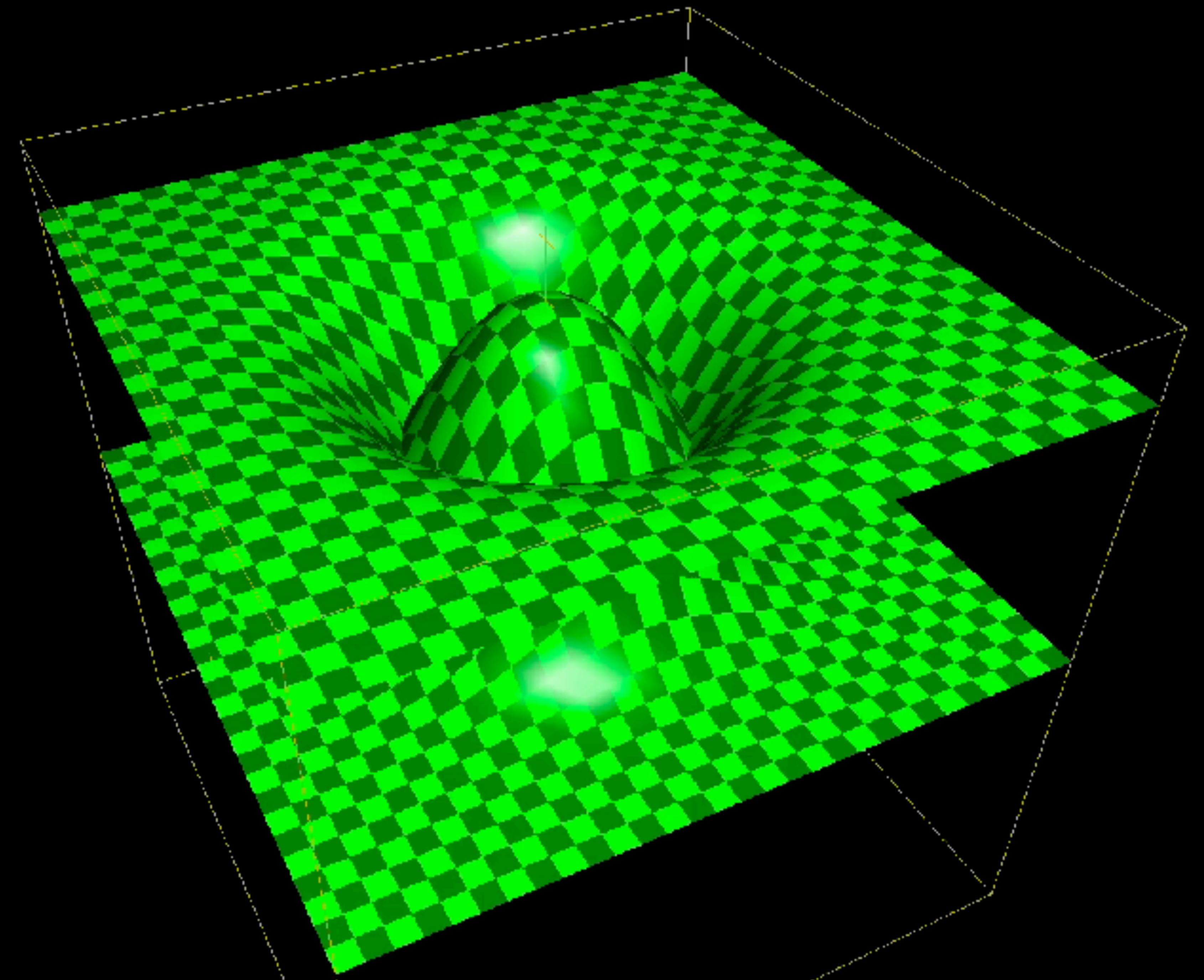
$$f([x_1, x_2], \mathbf{w}) =$$
$$= w_1x_1 + w_2x_2 + w_3x_1^2 + w_4x_2^2 + w_5x_1x_2 + w_0$$

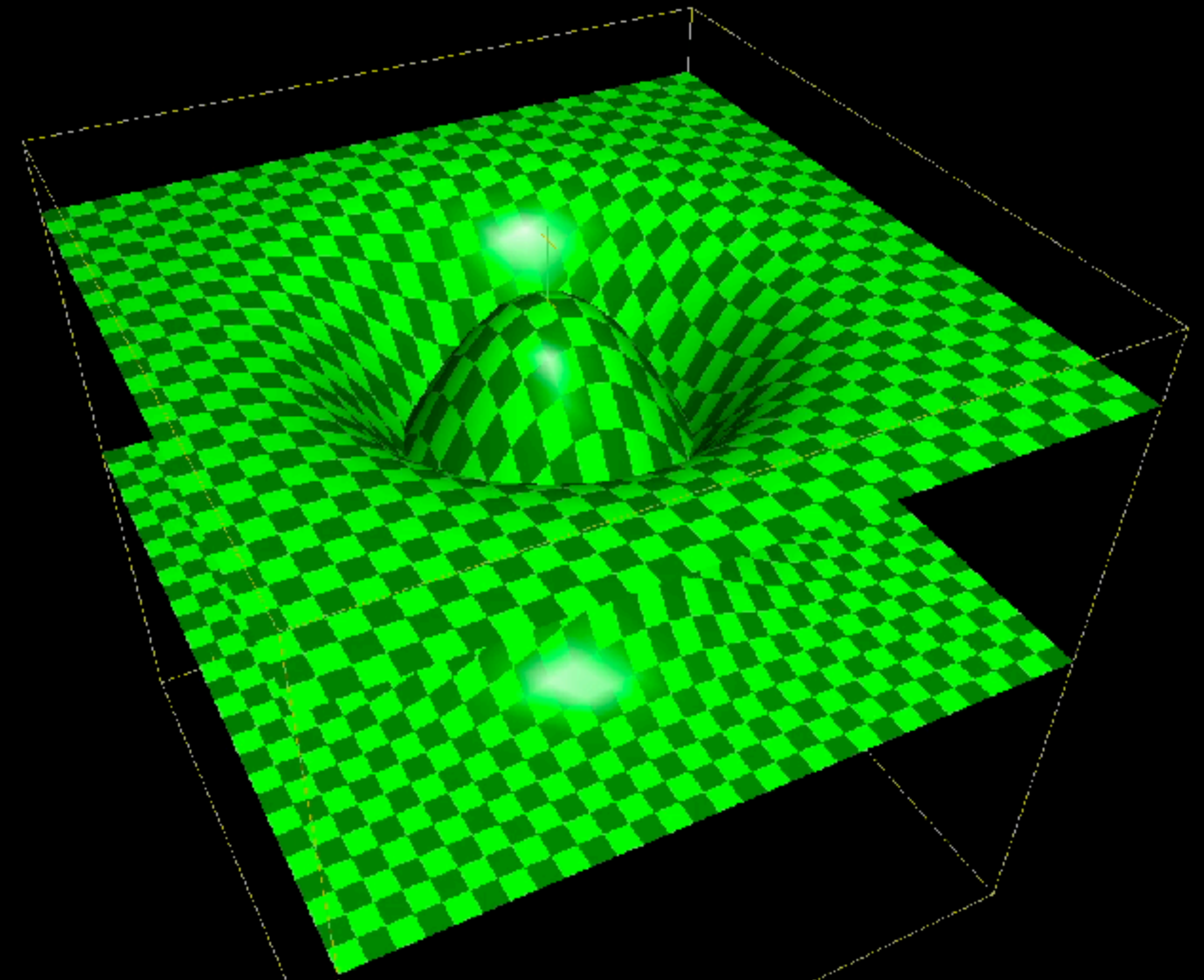


Is there a linear classifier that separates this data?

- Taylor polynoma
- Fourier transform
- Hilbert or Banach space







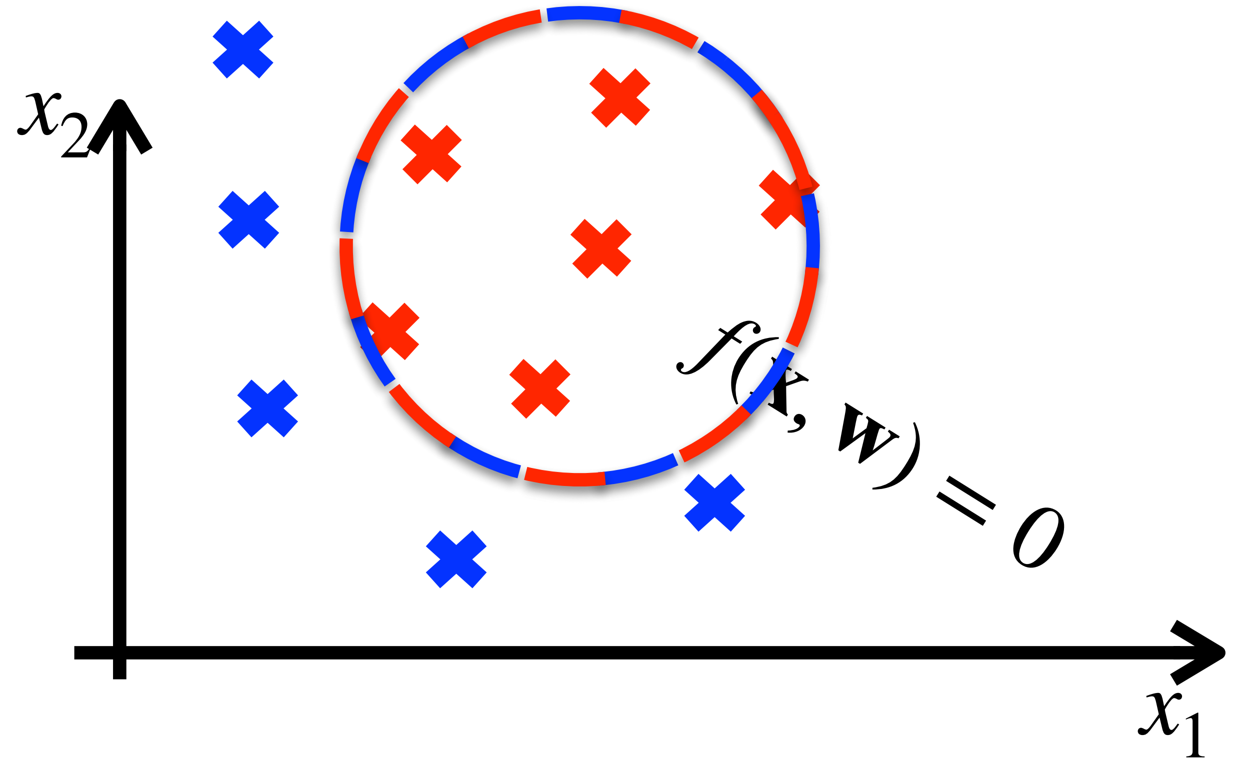
5D linear classifier

$$f([x_1, x_2, x_1^2, x_2^2, x_1 x_2], \mathbf{w}) =$$
$$= w_1 x_1 + w_2 x_2 + w_3 x_1^2 + w_4 x_2^2 + w_5 x_1 x_2 + w_0$$

$\Leftrightarrow$

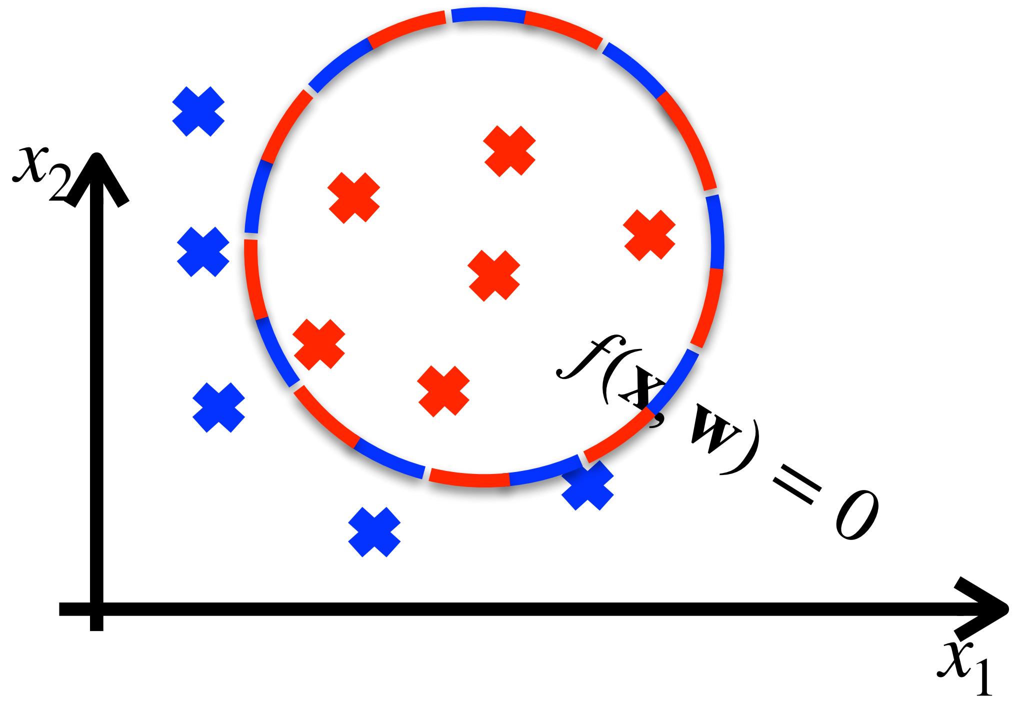
2D non-linear classifier

$$f([x_1, x_2], \mathbf{w}) =$$
$$= w_1 x_1 + w_2 x_2 + w_3 x_1^2 + w_4 x_2^2 + w_5 x_1 x_2 + w_0$$



Is there a linear classifier that separates this data?

- Taylor polynoma
- Fourier transform
- Hilbert or Banach space



There are two ways you can understand deep learning:

- Learn **non-linear classifier** that separates complicated areas in pixel space
- Learn **non-linear features** that makes classes linearly separable

Equivalence of common binary losses

Binary cross-entropy loss:

$$p(y\,|\,\mathbf{x},\mathbf{w}) = \begin{cases} \sigma(f(\mathbf{x},\mathbf{w})) & y=1 \\ 1-\sigma(f(\mathbf{x},\mathbf{w})) & y=0 \end{cases} \quad = y_i \cdot \sigma(f(\mathbf{x},\mathbf{w})) + (1-y_i)\cdot \sigma(f(\mathbf{x},\mathbf{w}))$$

$$\mathscr{L}(\mathbf{W}) = \sum_i -\log p(y_i\,|\,\mathbf{x}_i,\mathbf{w}) = \sum_i -\log\Big(y_i \cdot \sigma(f(\mathbf{x}_i,\mathbf{w})) + (1-y_i)\cdot \sigma(f(\mathbf{x}_i,\mathbf{w}))\Big)$$

```
loss = y*(-torch.log(torch.sigmoid(w@x)))
       + (1-y)*(-torch.log(1-torch.sigmoid(w@x)))
tensor(3.9876, requires_grad=True)
```

Logistic loss:

$$p(y\,|\,\mathbf{x},\mathbf{w}) = \begin{cases} \sigma(f(\mathbf{x},\mathbf{w})) & y=1 \\ 1-\sigma(f(\mathbf{x},\mathbf{w})) & y=-1 \end{cases} = \begin{cases} \sigma(f(\mathbf{x},\mathbf{w})) & y=1 \\ \sigma(-f(\mathbf{x},\mathbf{w})) & y=-1 \end{cases} \quad = \sigma\big(y_i f(\mathbf{x}_i,\mathbf{w})\big)$$

$$\mathscr{L}(\mathbf{W}) = \sum -\log p(y_i\,|\,\mathbf{x}_i,\mathbf{w}) = \sum -\log\Big(\sigma(y_i f(\mathbf{x}_i,\mathbf{w}))\Big) = \sum \log\Big(1+\exp(-y_i f(\mathbf{x}_i,\mathbf{w}))\Big)$$

```
loss = -torch.log(torch.sigmoid(y*(w@x)))
tensor(3.9876, requires_grad=True)
loss = torch.log(1+torch.exp(-y*(w@x)))
tensor(3.9876, requires_grad=True)
```

Competencies required for the test T1

- Classification loss for two-class and K-class classification problem.
- Equivalence of common binary classification losses
- Meaning of weights in linear classifier
- 2D visualization, decision boundary, features
- Drawback of linear classifier (too simple for some problems)