DeepRob

Lecture 3
Linear Classifiers
University of Minnesota

Slides adapted from Justin Johnson's Deep Learning for Computer Vision course

Project 0

- Instructions and code available on the website
  - Here: https://rpm-lab.github.io/CSCI5980-F24-DeepRob/projects/project0/

- **Autograder will be made available today!**

- **Due Sept 16, 11:59 PM CT**

Project 1

- Instructions and code will be available on the website today.
- Classification using K-Nearest Neighbors and Linear Models
- Will be due on Sept 25th, 11:59 pm CT.

Recap: Image Classification — A Core Computer Vision Task

Input: image

HERSHEY'S
COOKIES 'N' CREME
DiPPeDPreTzels
BITE SIZE PRETZELS DIPPED IN WHITE CREME AND SPRINKLED WITH COOKIE BITS

→

Output: assign image to one of a fixed set of categories

Chocolate Pretzels

Granola Bar

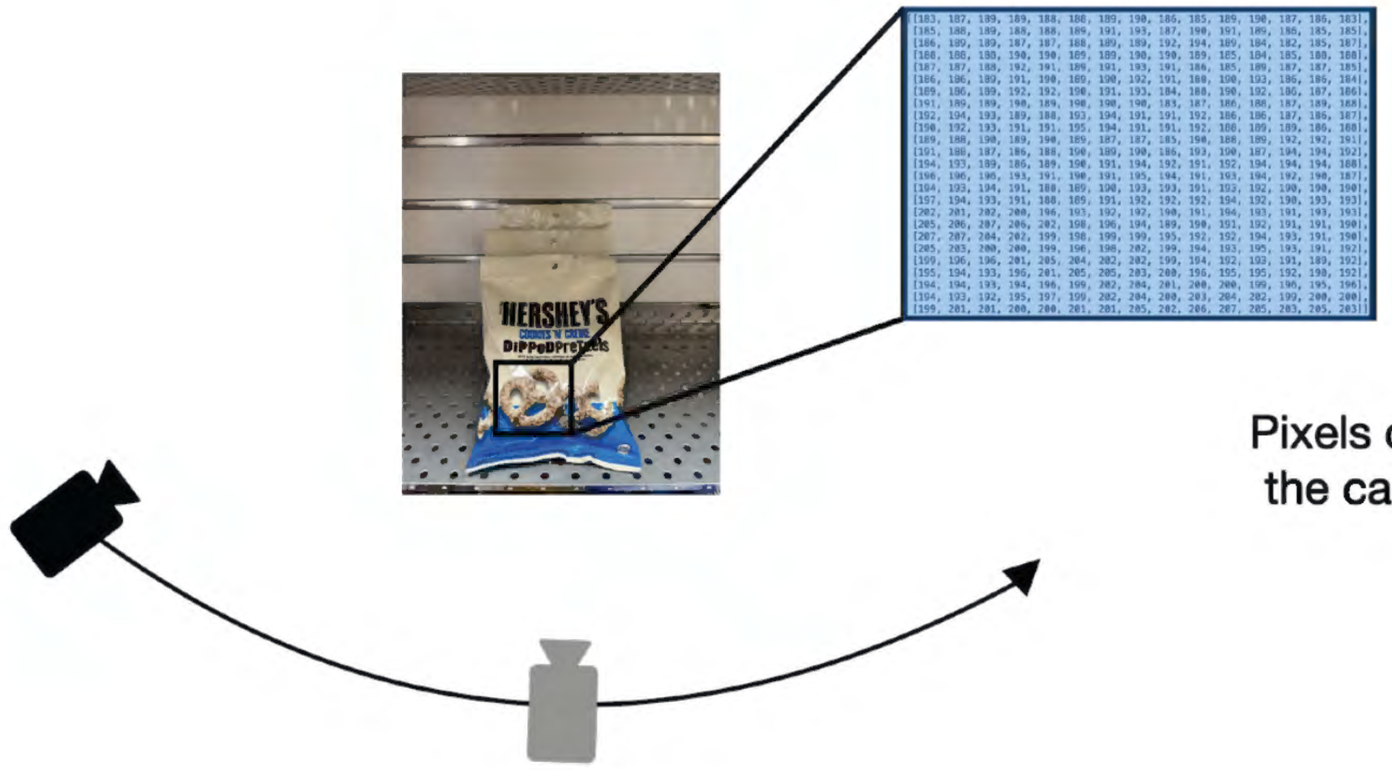
Potato Chips

Water Bottle

Popcorn

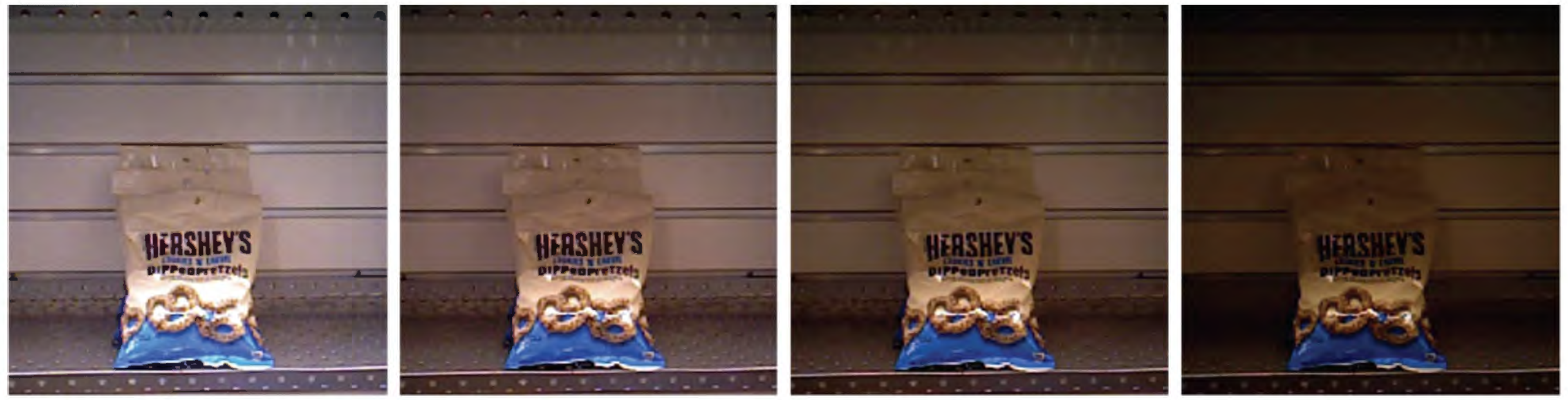
Recap: Image Classification Challenges

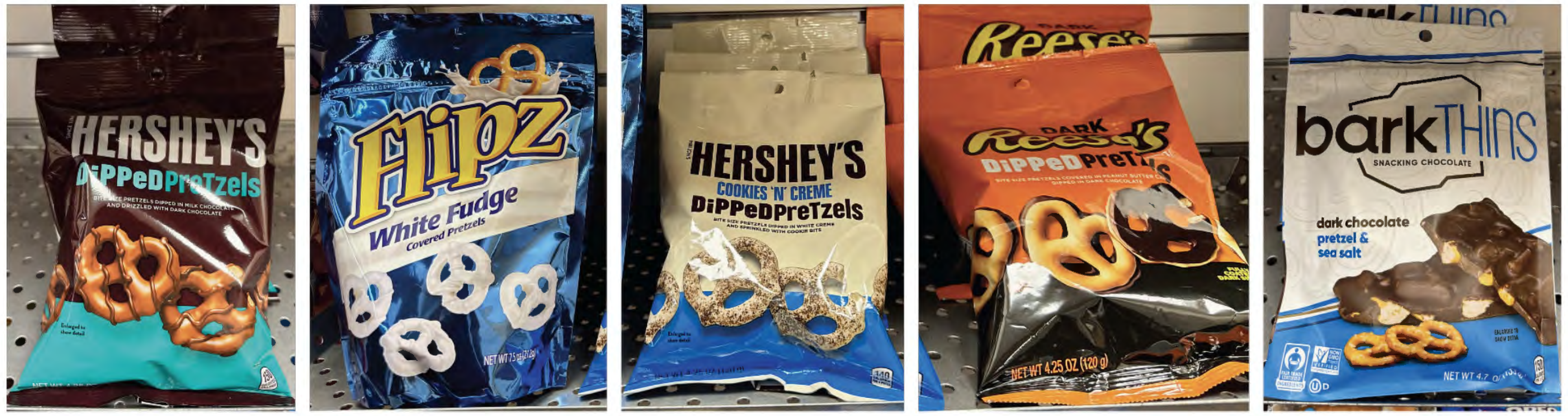
Viewpoint Variation & Semantic Gap



Pixels change when the camera moves

Illumination Changes





Milk Chocolate

White Chocolate

Cookies N' Creme

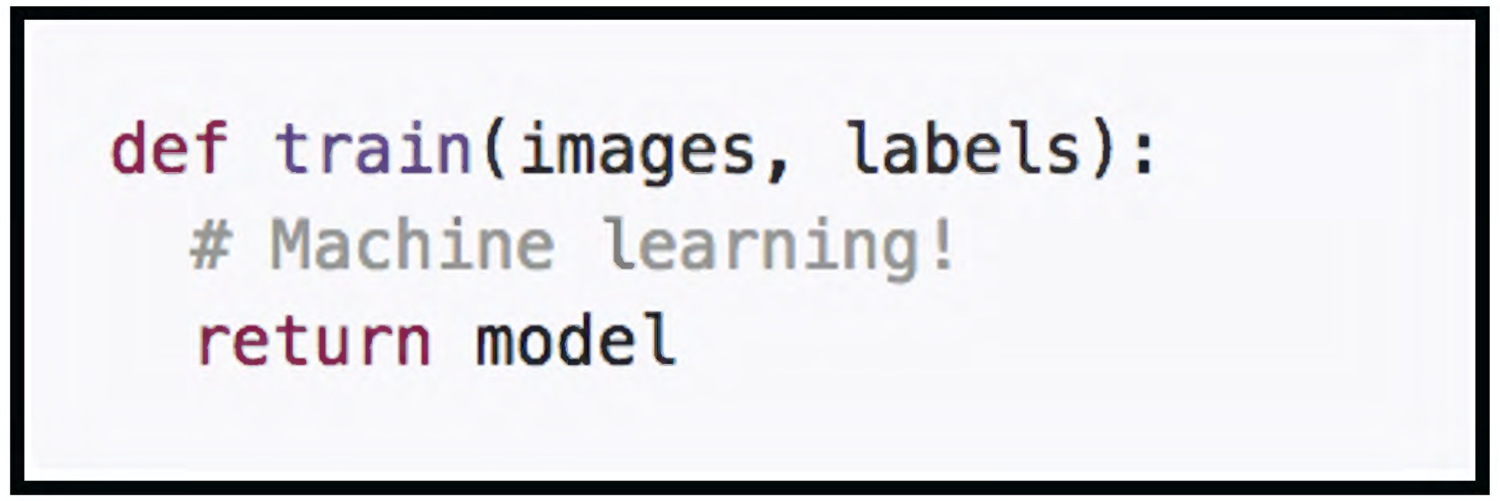
Peanut Butter

Ambiguous Category

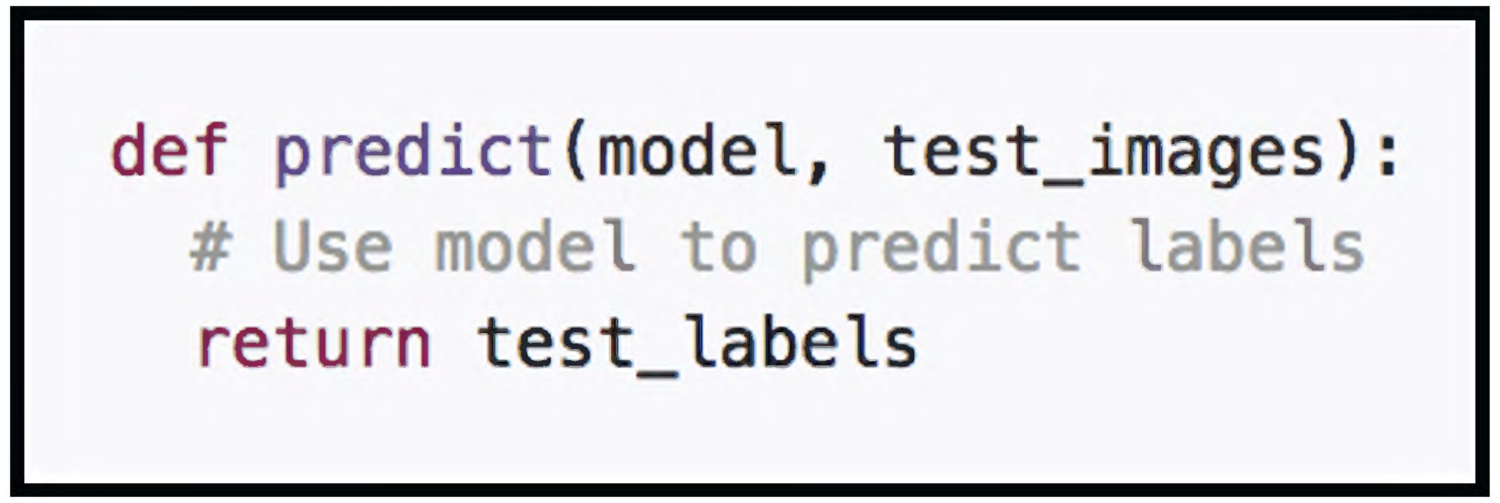
Intraclass Variation

Recap: Machine Learning—Data-Driven Approach

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

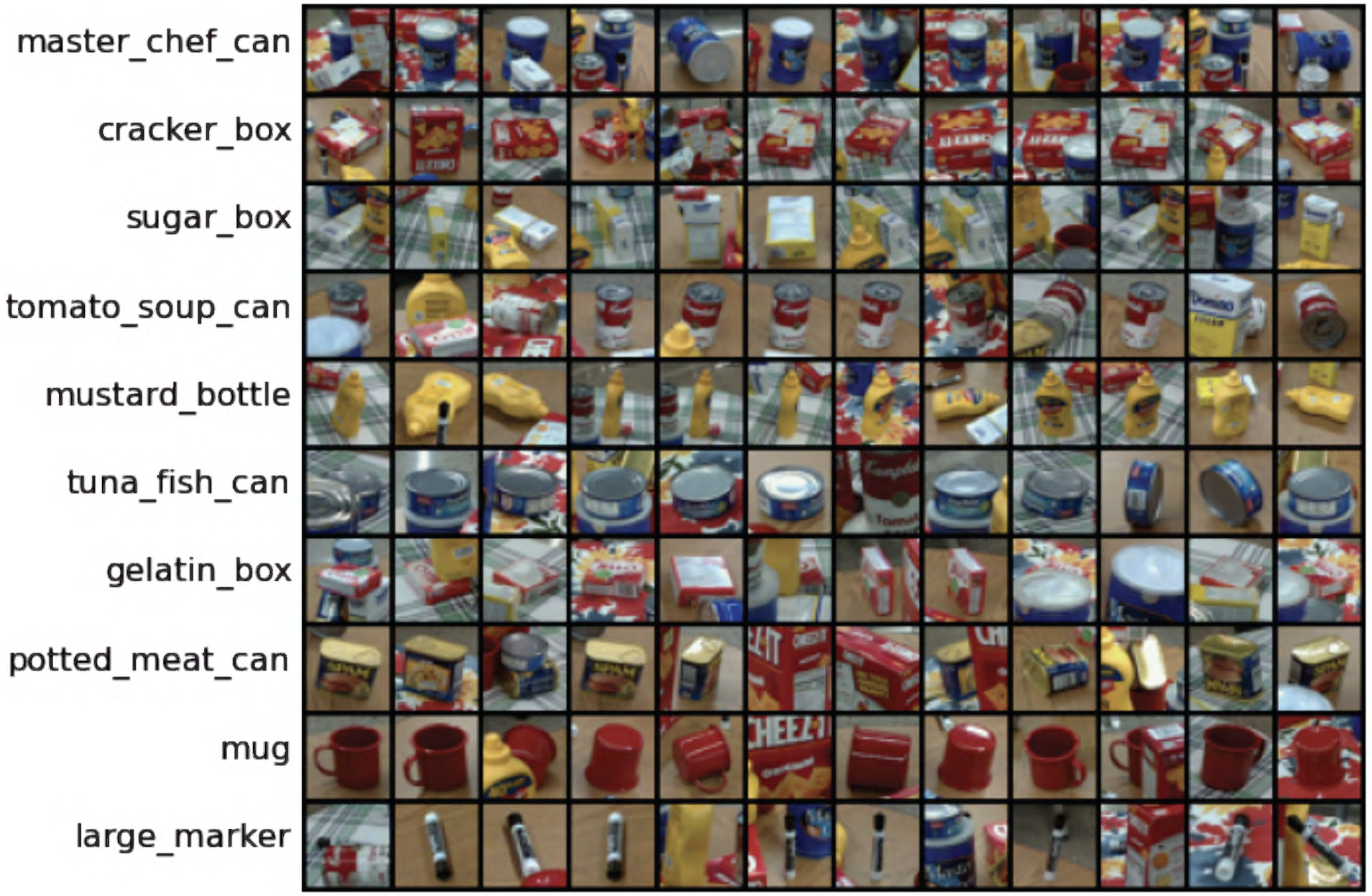


```python
def train(images, labels):
  # Machine learning!
  return model
```



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

Example training set



Linear Classifiers

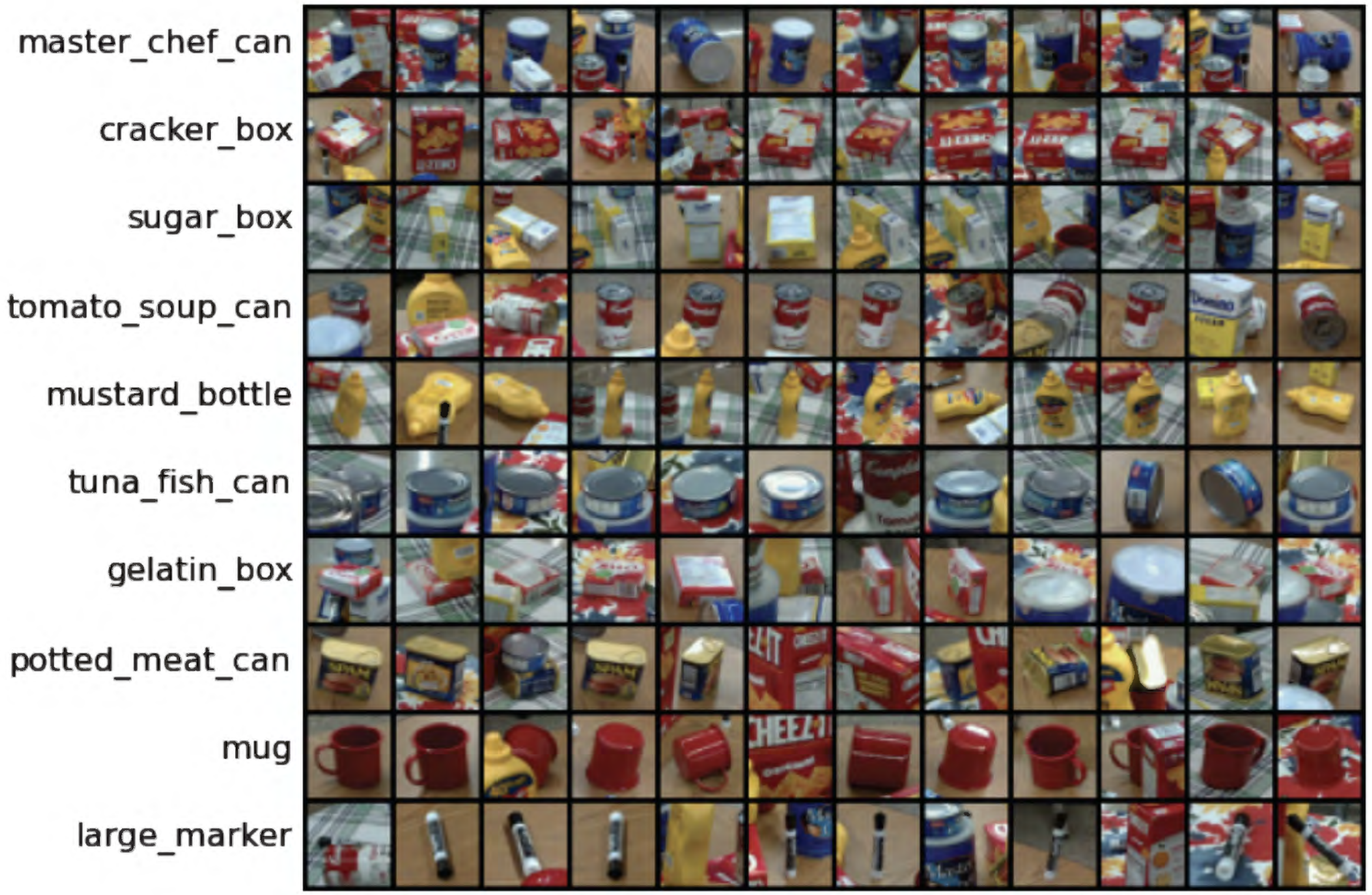
Building Block of Neural Networks

Linear classifiers

This image is CC0 1.0 public domain

Recall PROPS

Progress **R**obot **O**bject **P**erception **S**amples **D**ataset



Chen et al., "ProgressLabeller: Visual Data Stream Annotation for Training Object-Centric 3D Perception", IROS, 2022.

10 classes

32x32 RGB images

50k training images (5k per class)

10k test images (1k per class)

Parametric Approach

Image

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

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

10 numbers giving class scores

W

parameters
or weights

Parametric Approach—Linear Classifier

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

Image

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

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

10 numbers giving
class scores

W

parameters
or weights

Parametric Approach—Linear Classifier

$$\underbrace{\boxed{f(x,W)}}_{(10,)} = \underbrace{\boxed{W}}_{(10,\ 3072)}\overset{(3072,)}{\boxed{x}}$$

Image

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

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

10 numbers giving
class scores

W

parameters
or weights

Parametric Approach—Linear Classifier

$$\underset{(10,)}{\boxed{f(x,W)}} = \underset{(10,\ 3072)}{\boxed{W}}\overset{(3072,)}{\boxed{x}} + \boxed{b}\quad (10,)$$

Image

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

$\longrightarrow$ f(**x**,**W**) $\longrightarrow$ **10** numbers giving class scores

$\uparrow$

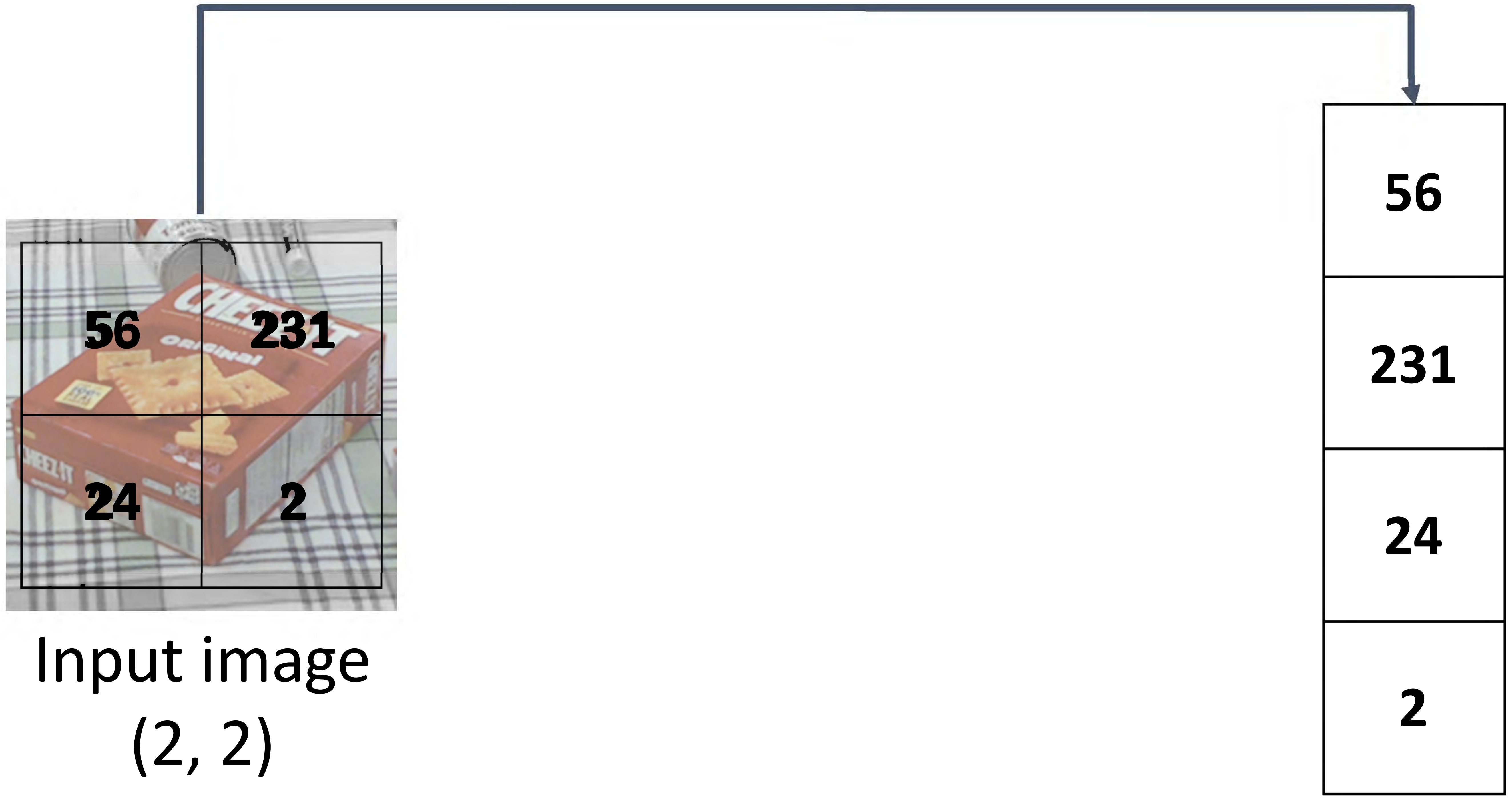
W

parameters
or weights

Example for 2x2 Image, 3 classes (crackers/mug/sugar)

Stretch pixels into column

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



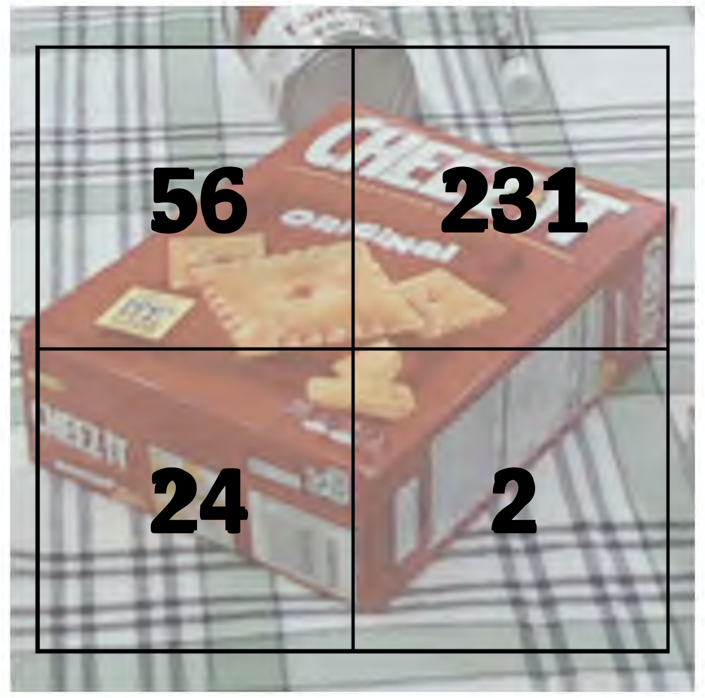
Input image
(2, 2)

(4,)

Example for 2x2 Image, 3 classes (crackers/mug/sugar)

Stretch pixels into column

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



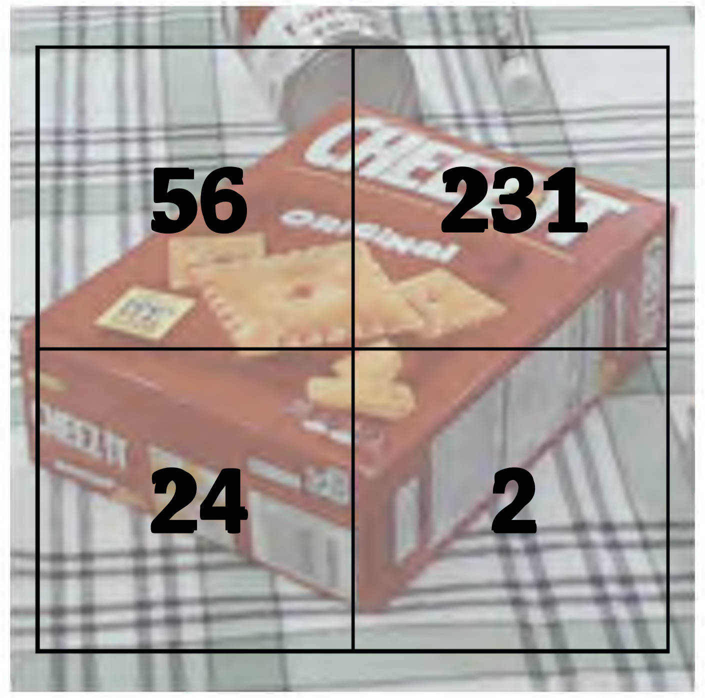
Input image
(2, 2)

$$\underbrace{\begin{bmatrix} 0.2 & -0.5 & 0.1 & 2.0 \\ 1.5 & 1.3 & 2.1 & 0.0 \\ 0 & 0.25 & 0.2 & -0.3 \end{bmatrix}}_{W\ (3,\,4)} \underbrace{\begin{bmatrix} 56 \\ 231 \\ 24 \\ 2 \end{bmatrix}}_{(4,)} + \underbrace{\begin{bmatrix} 1.1 \\ 3.2 \\ -1.2 \end{bmatrix}}_{b\ (3,)} = \underbrace{\begin{bmatrix} -96.8 \\ 437.9 \\ 61.95 \end{bmatrix}}_{(3,)}$$

Linear Classifier—Algebraic Viewpoint

Stretch pixels into column

f(x,W) = Wx + b

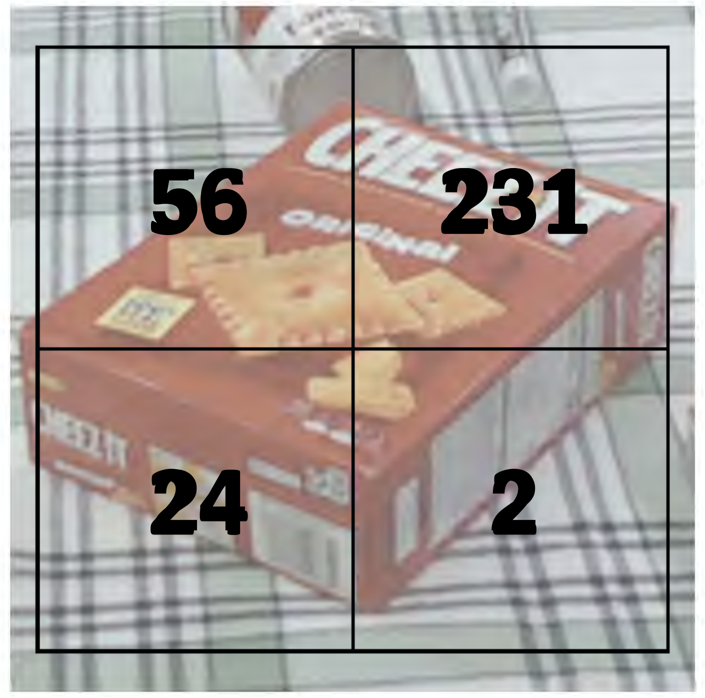


Input image
(2, 2)

$$\underbrace{\begin{bmatrix} 0.2 & -0.5 & 0.1 & 2.0 \\ 1.5 & 1.3 & 2.1 & 0.0 \\ 0 & 0.25 & 0.2 & -0.3 \end{bmatrix}}_{W\ (3,\ 4)} \underbrace{\begin{bmatrix} 56 \\ 231 \\ 24 \\ 2 \end{bmatrix}}_{(4,)} + \underbrace{\begin{bmatrix} 1.1 \\ 3.2 \\ -1.2 \end{bmatrix}}_{b\ (3,)} = \underbrace{\begin{bmatrix} -96.8 \\ 437.9 \\ 61.95 \end{bmatrix}}_{(3,)}$$

Linear Classifier—Bias Trick

Stretch pixels into column



Input image
(2, 2)

$$\underbrace{\left[\begin{array}{cccc|c} 0.2 & -0.5 & 0.1 & 2.0 & 1.1 \\ 1.5 & 1.3 & 2.1 & 0.0 & 3.2 \\ 0 & 0.25 & 0.2 & -0.3 & -1.2 \end{array}\right]}_{W\ (3,\,5)} \underbrace{\begin{bmatrix} 56 \\ 231 \\ 24 \\ 2 \\ 1 \end{bmatrix}}_{(5,)} = \underbrace{\begin{bmatrix} -96.8 \\ 437.9 \\ 61.95 \end{bmatrix}}_{(3,)}$$

Add extra one to data vector; bias is absorbed into last column of weight matrix

Linear Classifier—Predictions are Linear

f(x, W) = Wx (ignore bias)

f(cx, W) = W(cx) = c * f(x, W)

Linear Classifier—Predictions are Linear

$$f(x, W) = Wx \quad \text{(ignore bias)}$$

$$f(cx, W) = W(cx) = c * f(x, W)$$

Image

→

Scores
-96.8
437.8
62.0

0.5 * Image

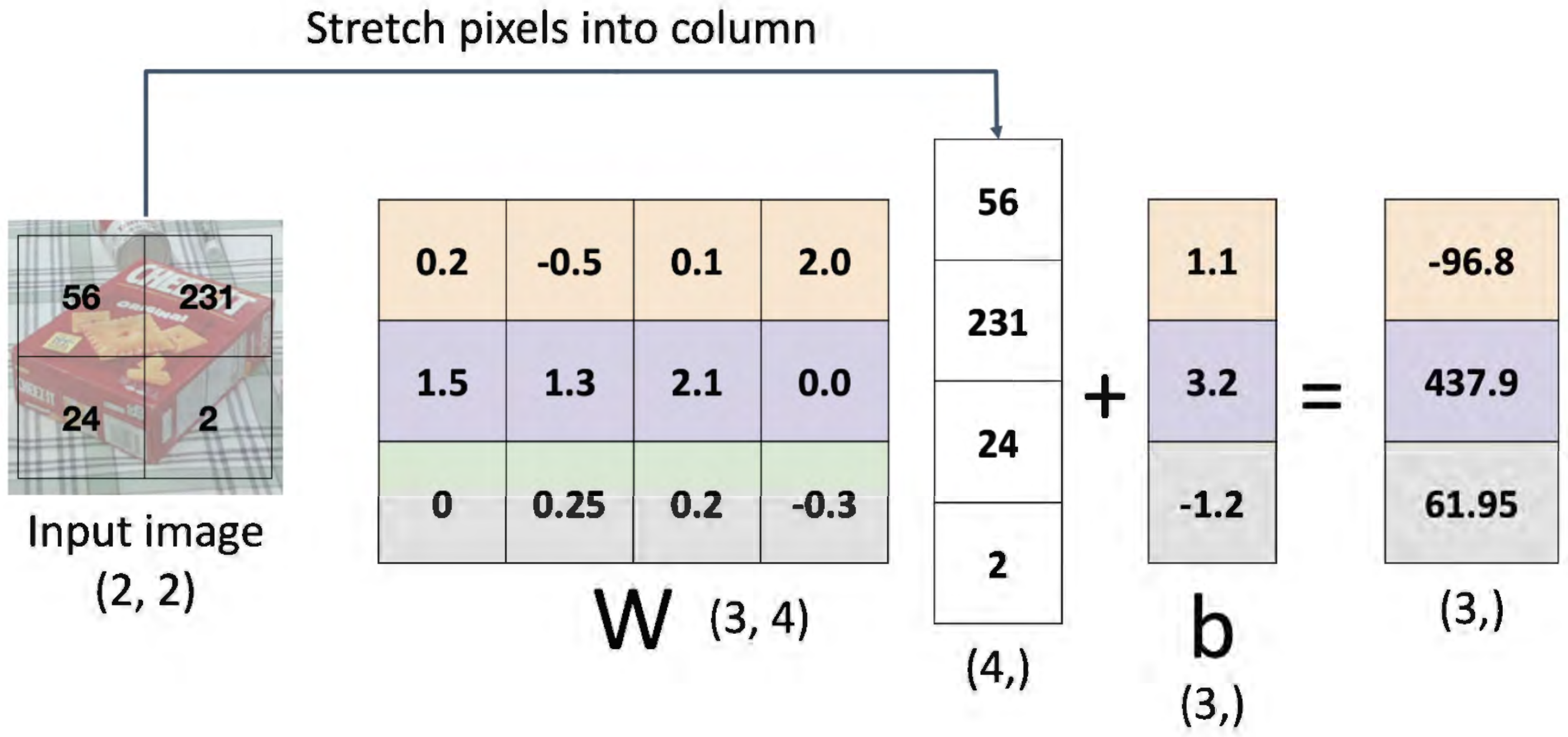
→

0.5 * Scores
-48.4
218.9
31.0

Interpreting a Linear Classifier

Algebraic Viewpoint

f(x,W) = Wx + b

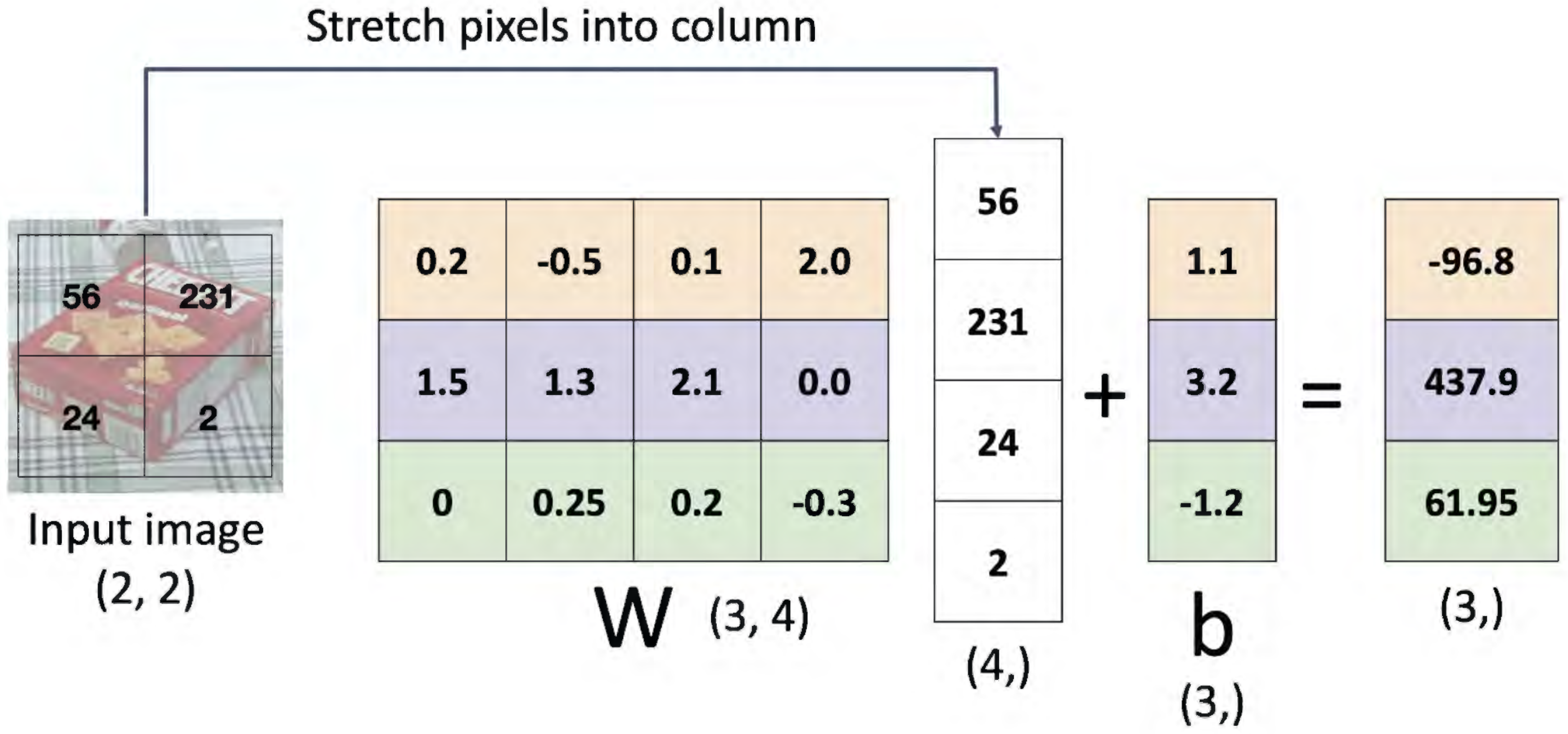

Stretch pixels into column

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

Interpreting a Linear Classifier

Algebraic Viewpoint

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



Stretch pixels into column

56	231
24	2

Input image (2, 2)

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

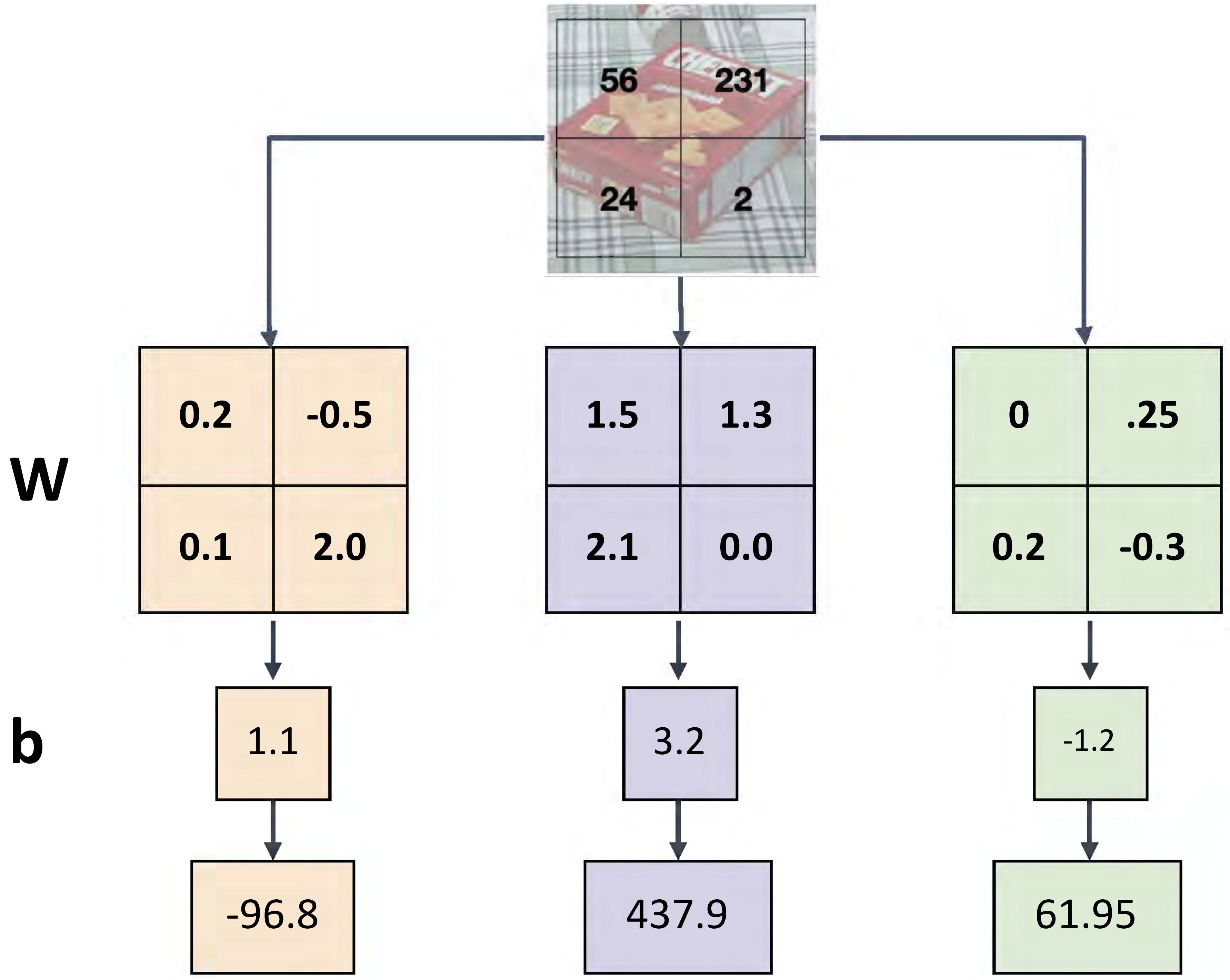
W (3, 4)

(4,)

b (3,)

(3,)

Instead of stretching pixels into columns, we can equivalently stretch rows of W into images!



56	231
24	2

W

0.2	-0.5
0.1	2.0

1.5	1.3
2.1	0.0

0	.25
0.2	-0.3

b

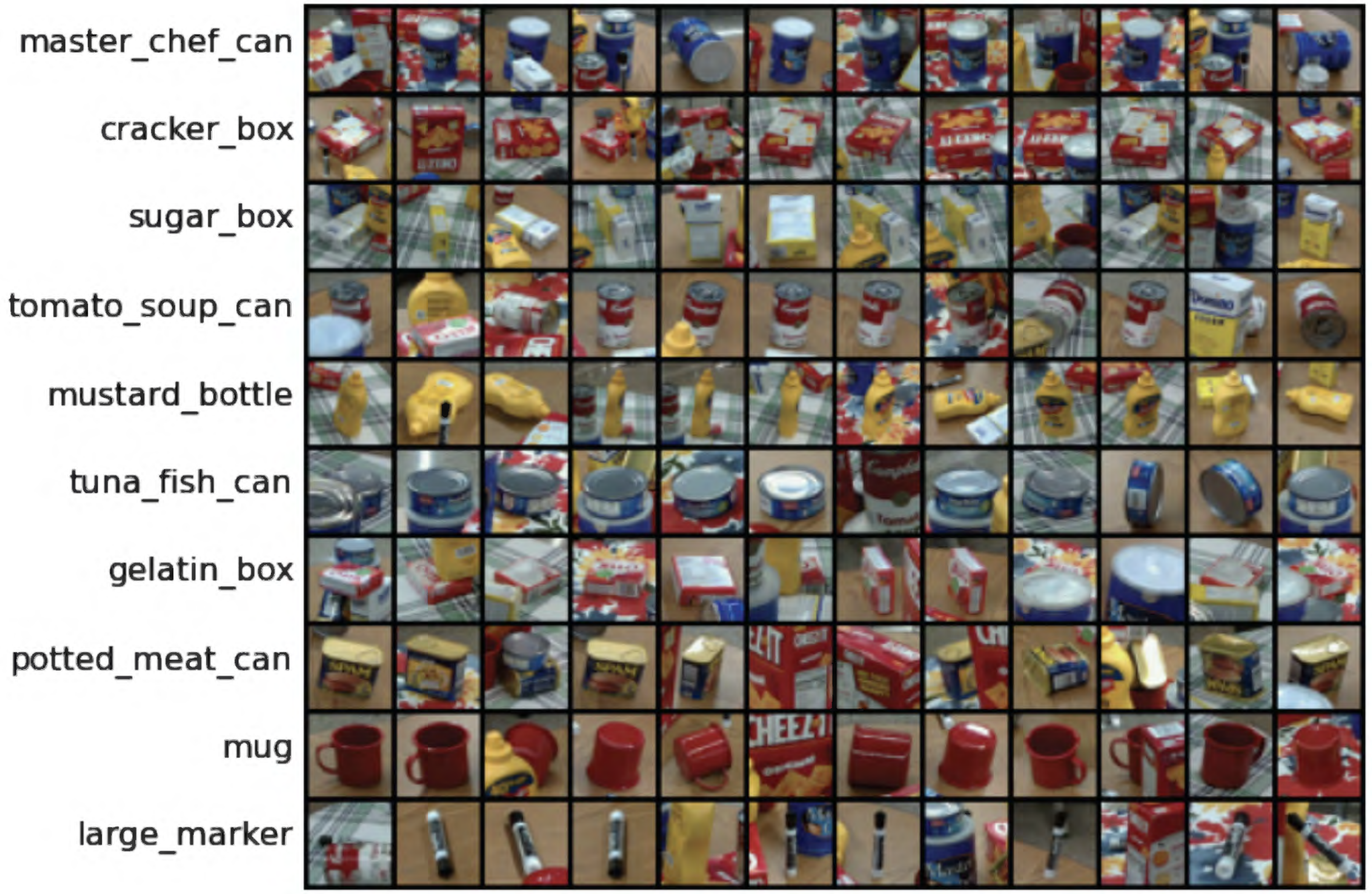
1.1 → -96.8

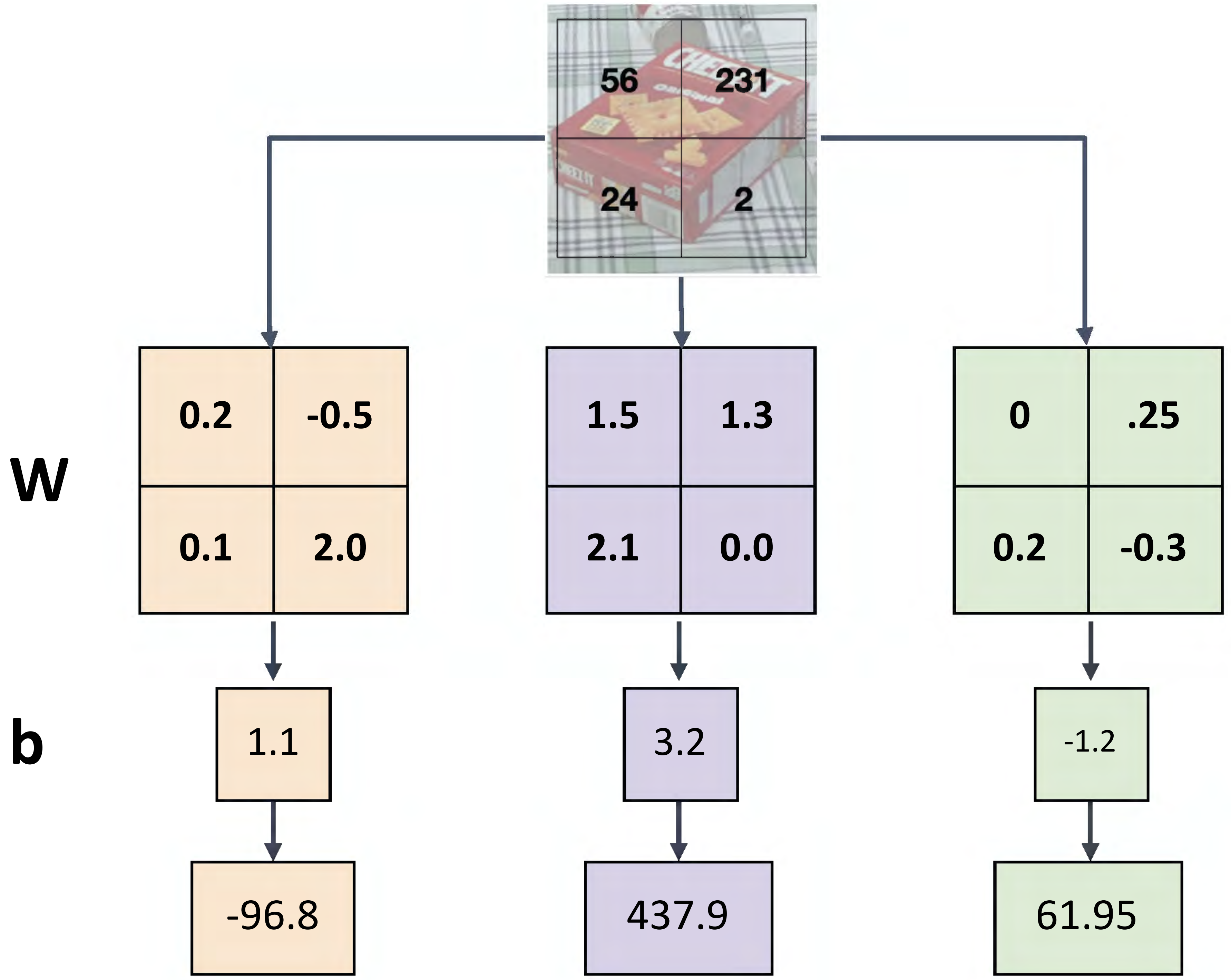
3.2 → 437.9

-1.2 → 61.95

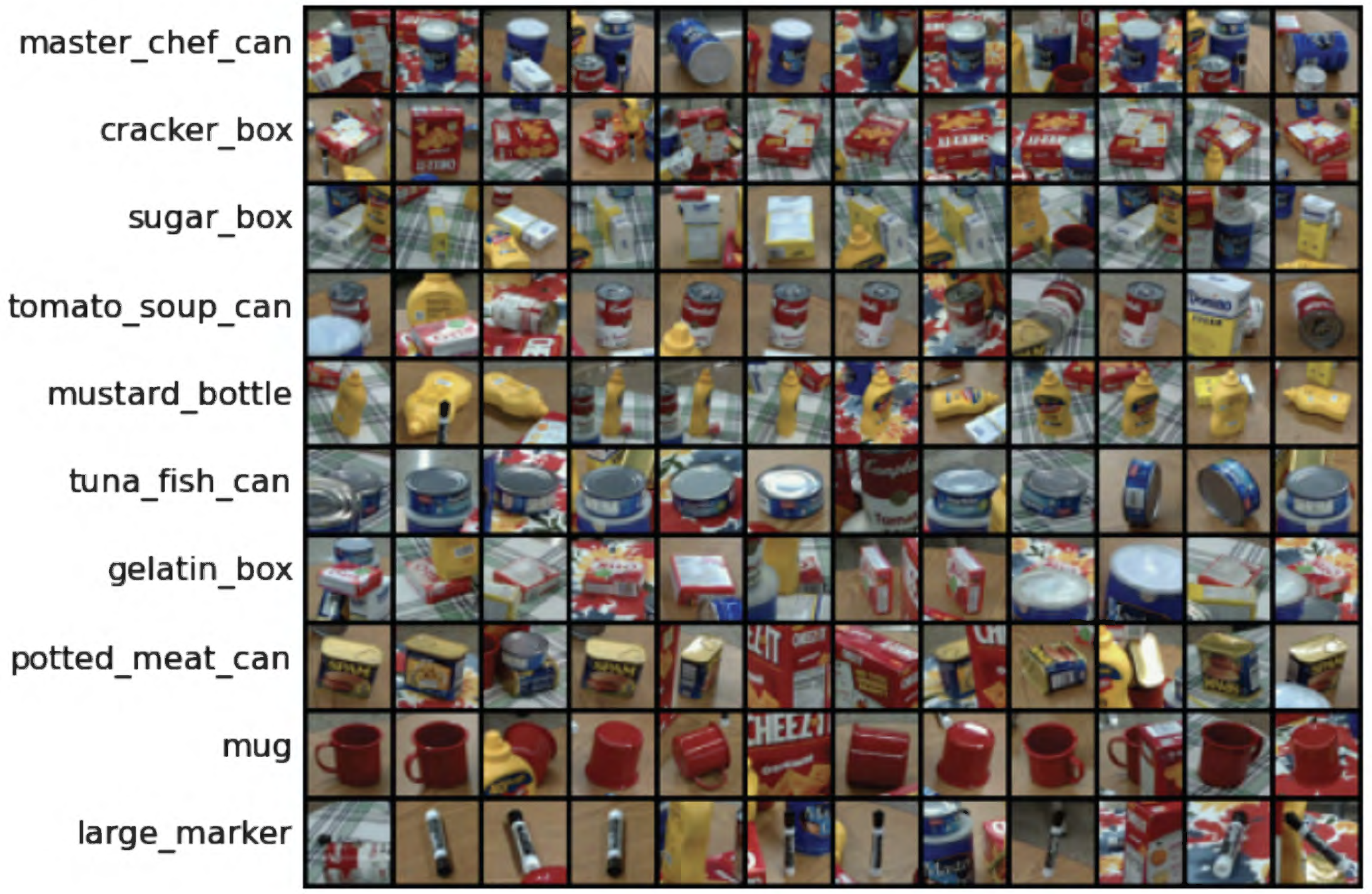
Interpreting a Linear Classifier

Instead of stretching pixels into columns, we can equivalently stretch rows of W into images!

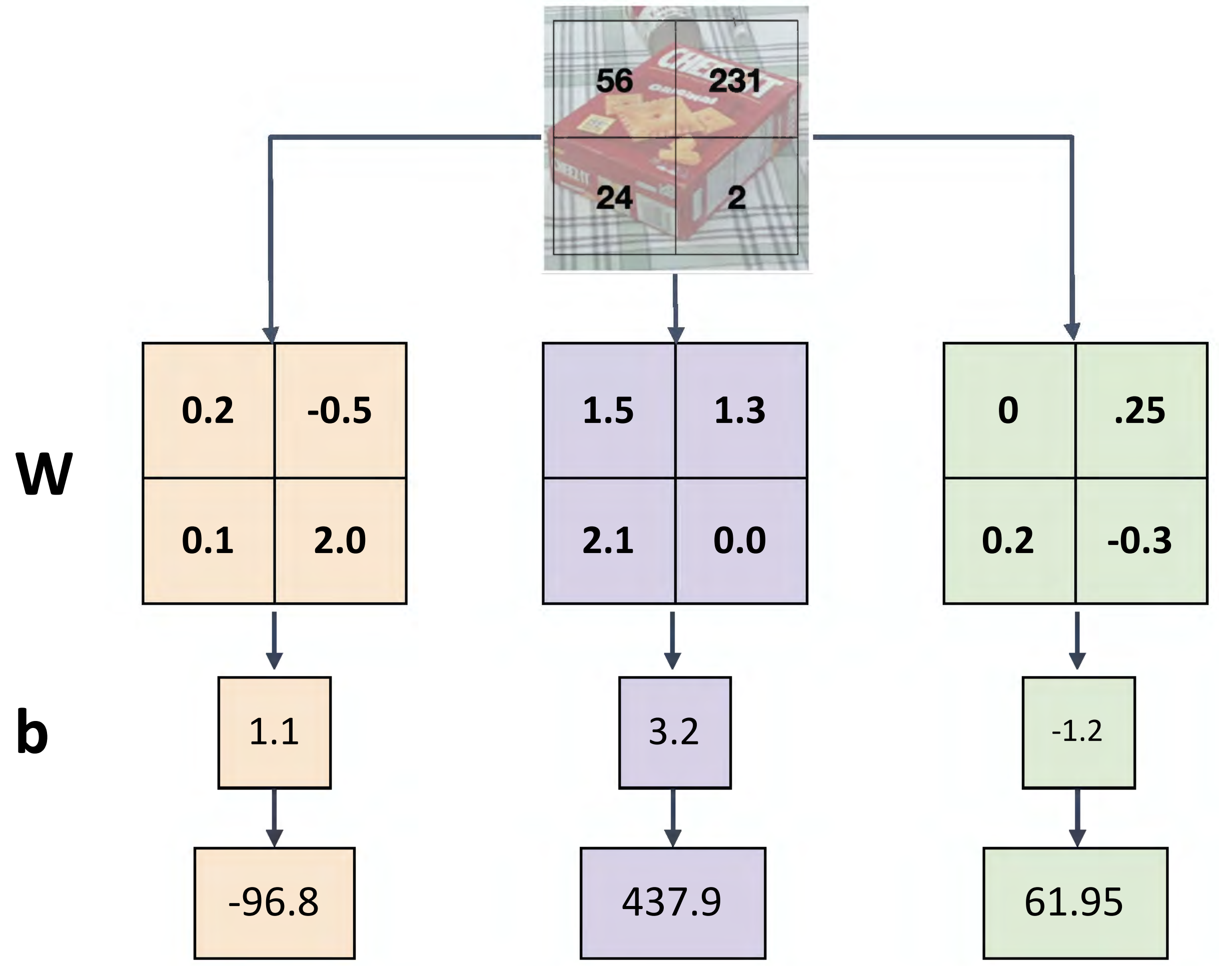




Interpreting a Linear Classifier



Instead of stretching pixels into columns, we can equivalently stretch rows of W into images!



56 231
24 2

W

0.2 -0.5
0.1 2.0

1.5 1.3
2.1 0.0

0 .25
0.2 -0.3

b

1.1

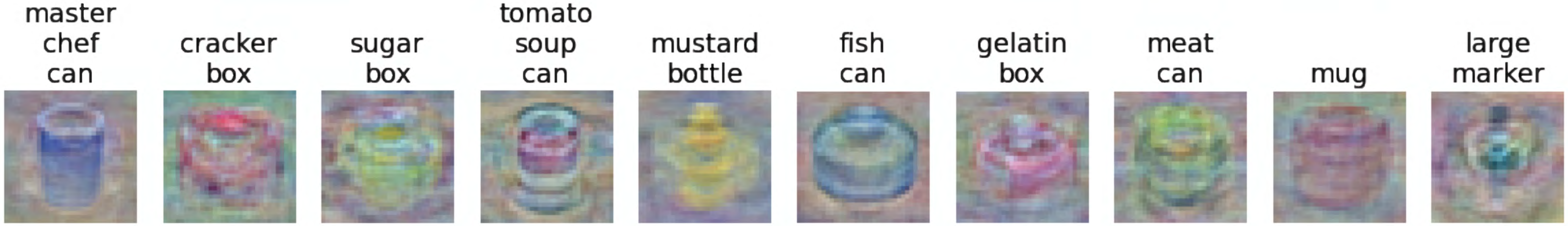
3.2

-1.2

-96.8

437.9

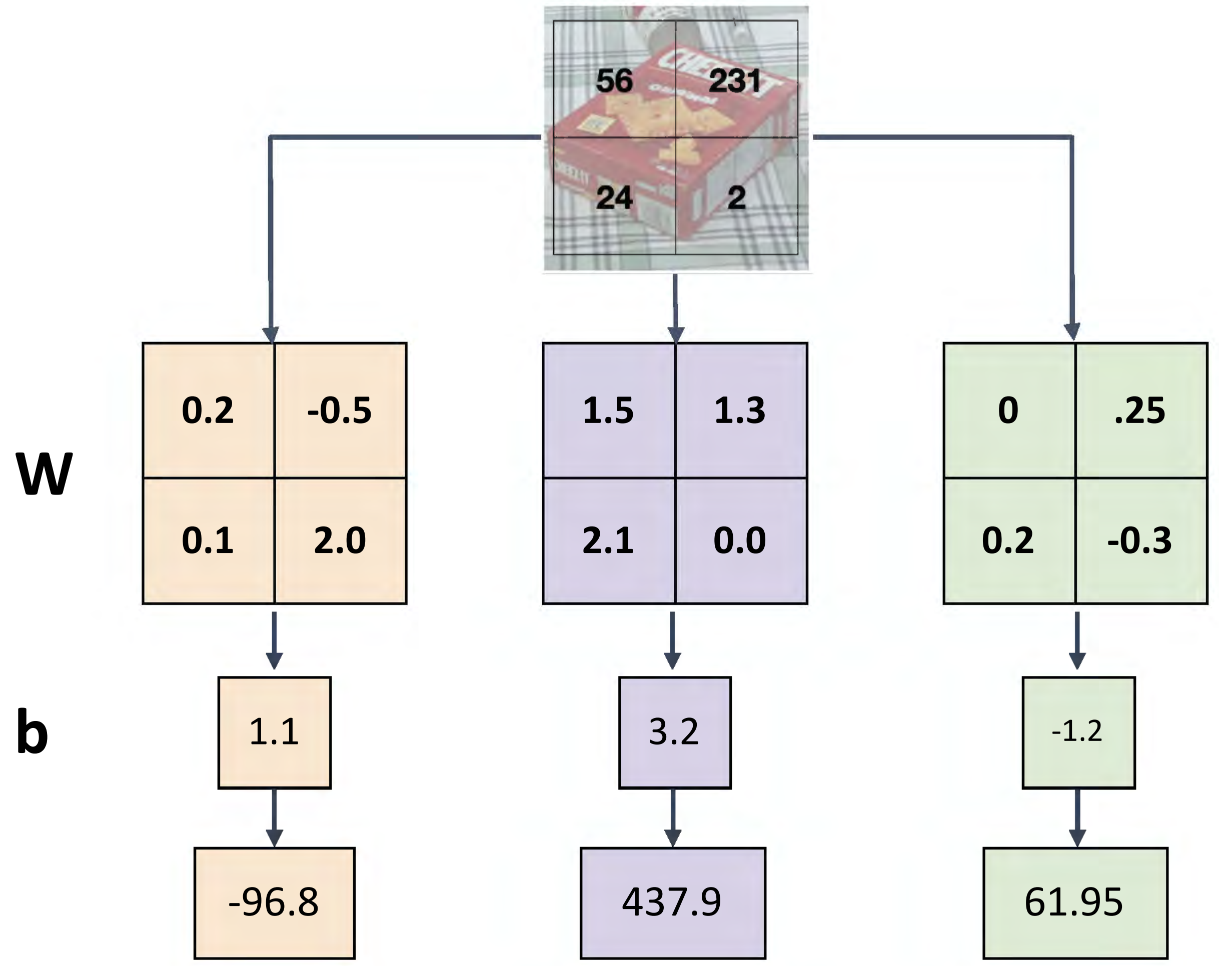
61.95

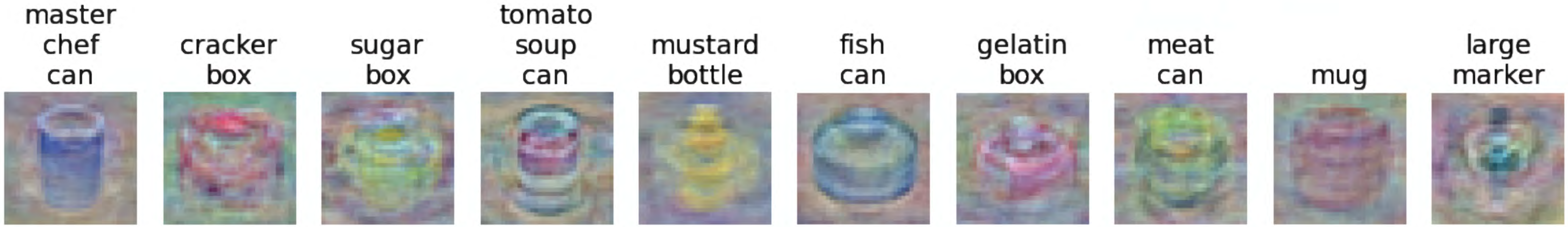


Interpreting a Linear Classifier—Visual Viewpoint

Instead of stretching pixels into columns, we can equivalently stretch rows of W into images!

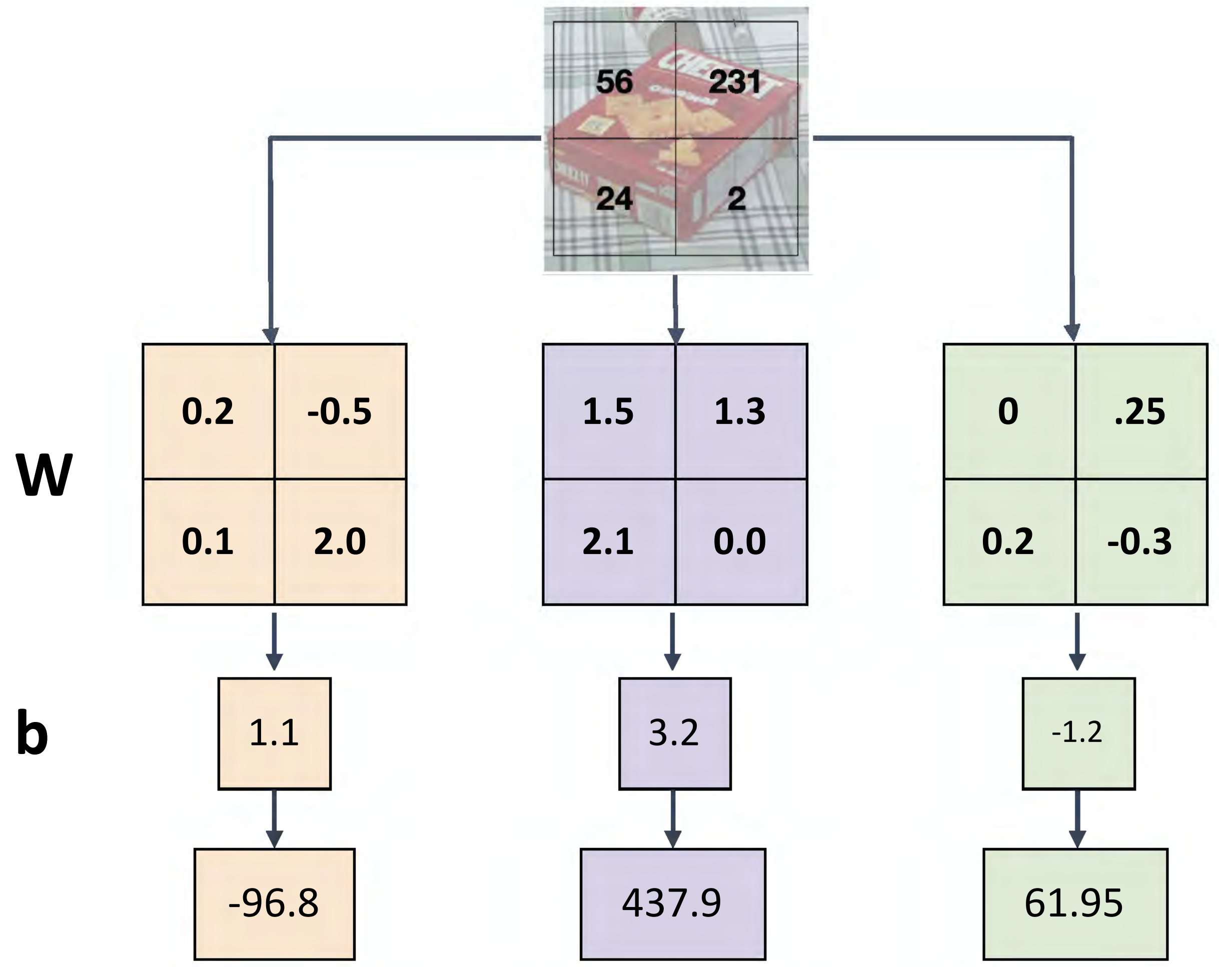
Linear classifier has one "template" per category





Interpreting a Linear Classifier—Visual Viewpoint

Instead of stretching pixels into columns, we can equivalently stretch rows of W into images!



56 231
24 2

W

0.2 -0.5
0.1 2.0

1.5 1.3
2.1 0.0

0 .25
0.2 -0.3

b

1.1

3.2

-1.2

-96.8

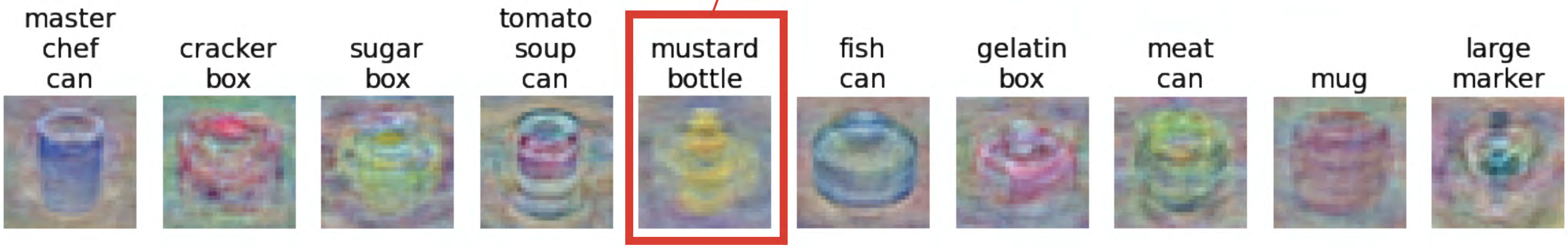
437.9

61.95

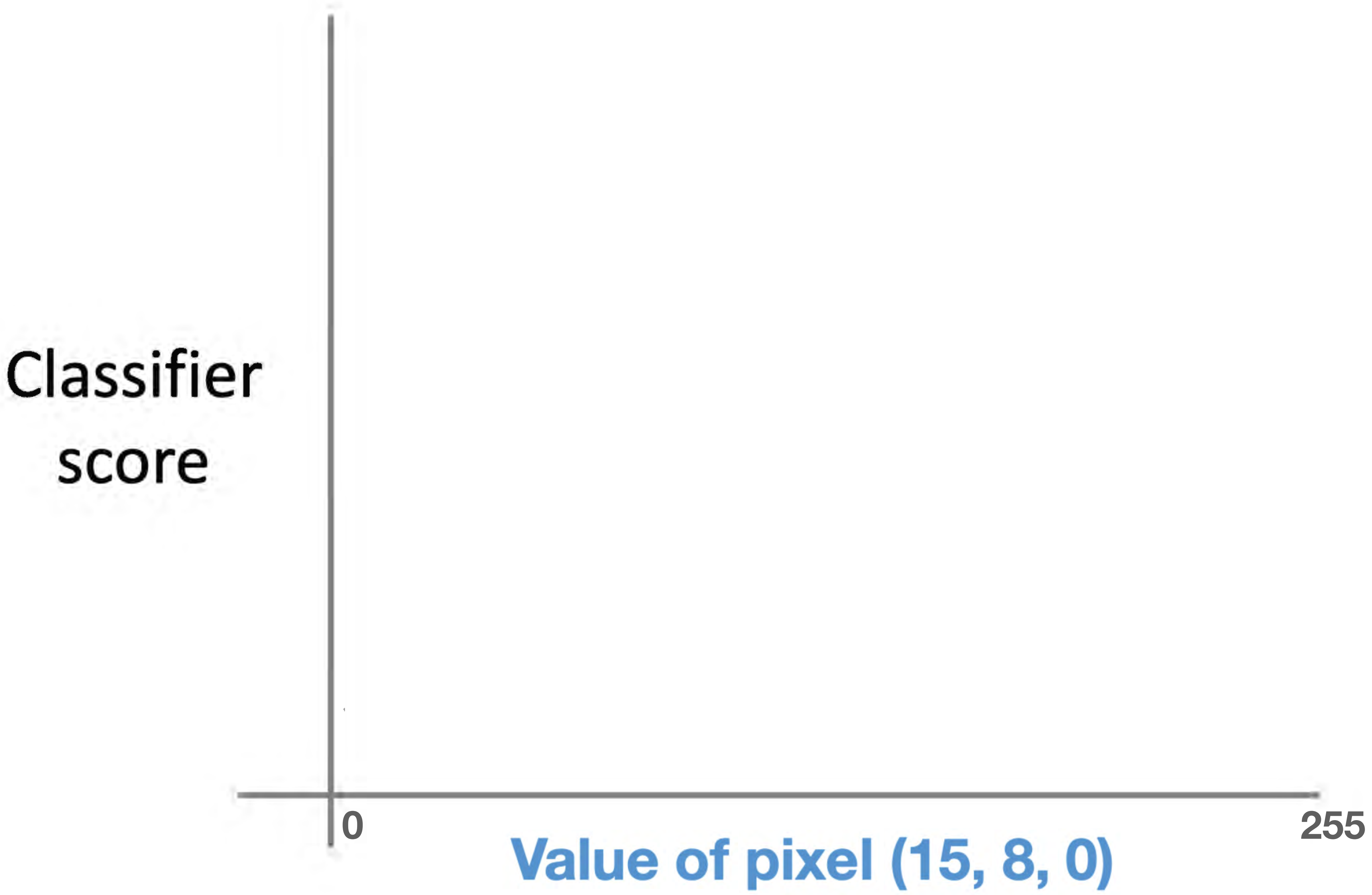
Linear classifier has one "template" per category

A single template cannot capture multiple modes of the data

e.g. mustard bottles can rotate



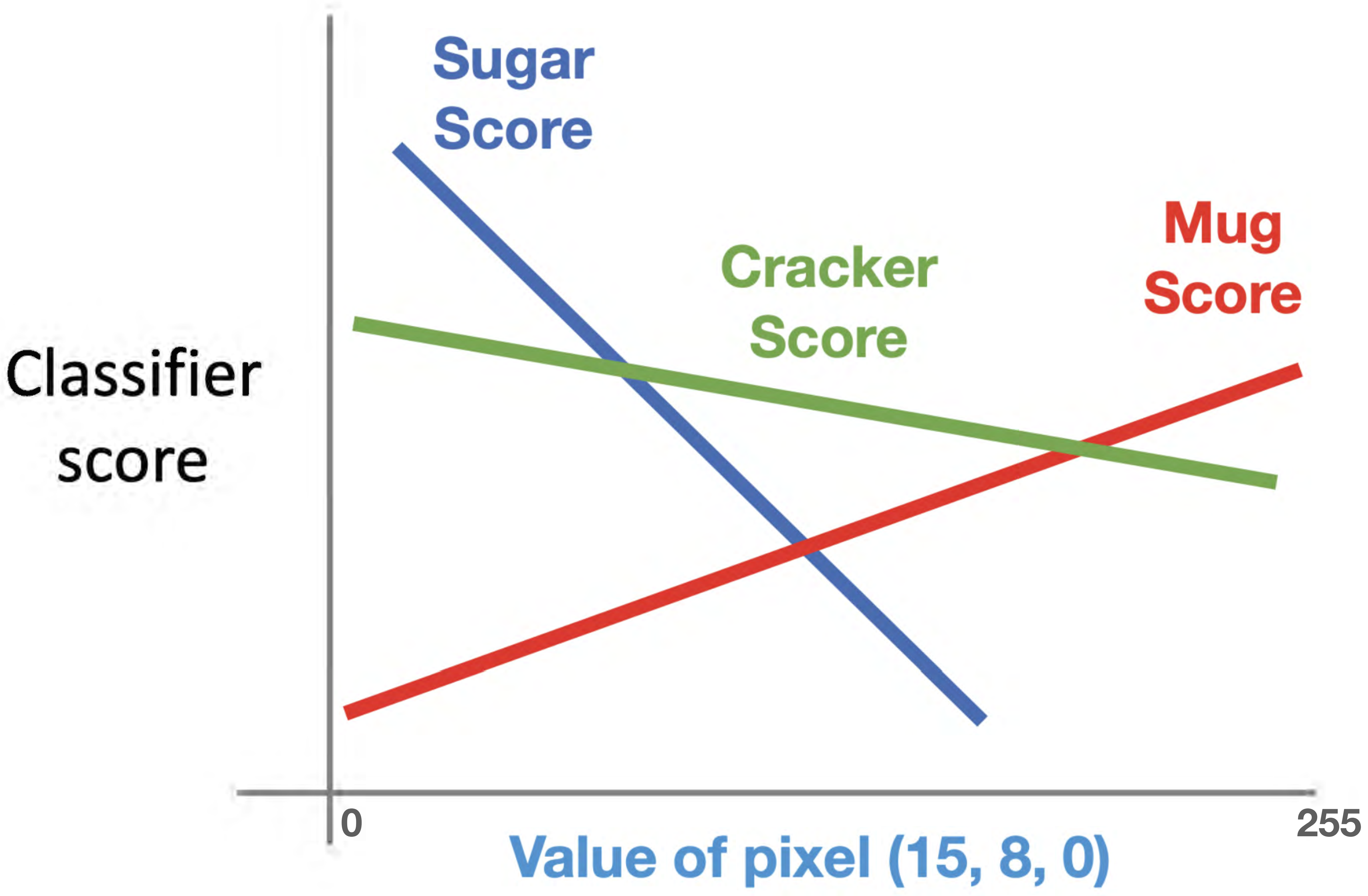
Interpreting a Linear Classifier—Geometric Viewpoint



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

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

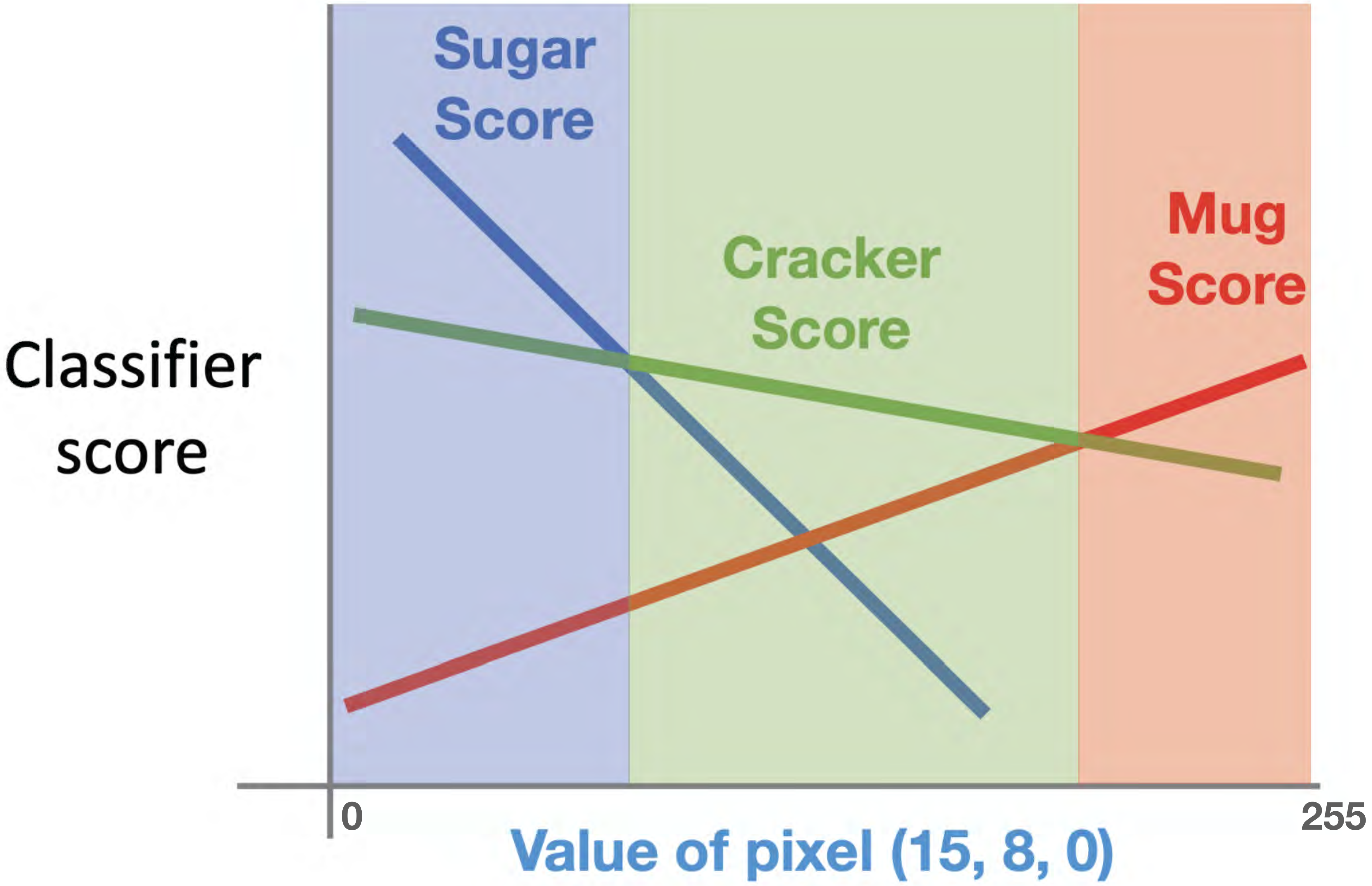
Interpreting a Linear Classifier—Geometric Viewpoint



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

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

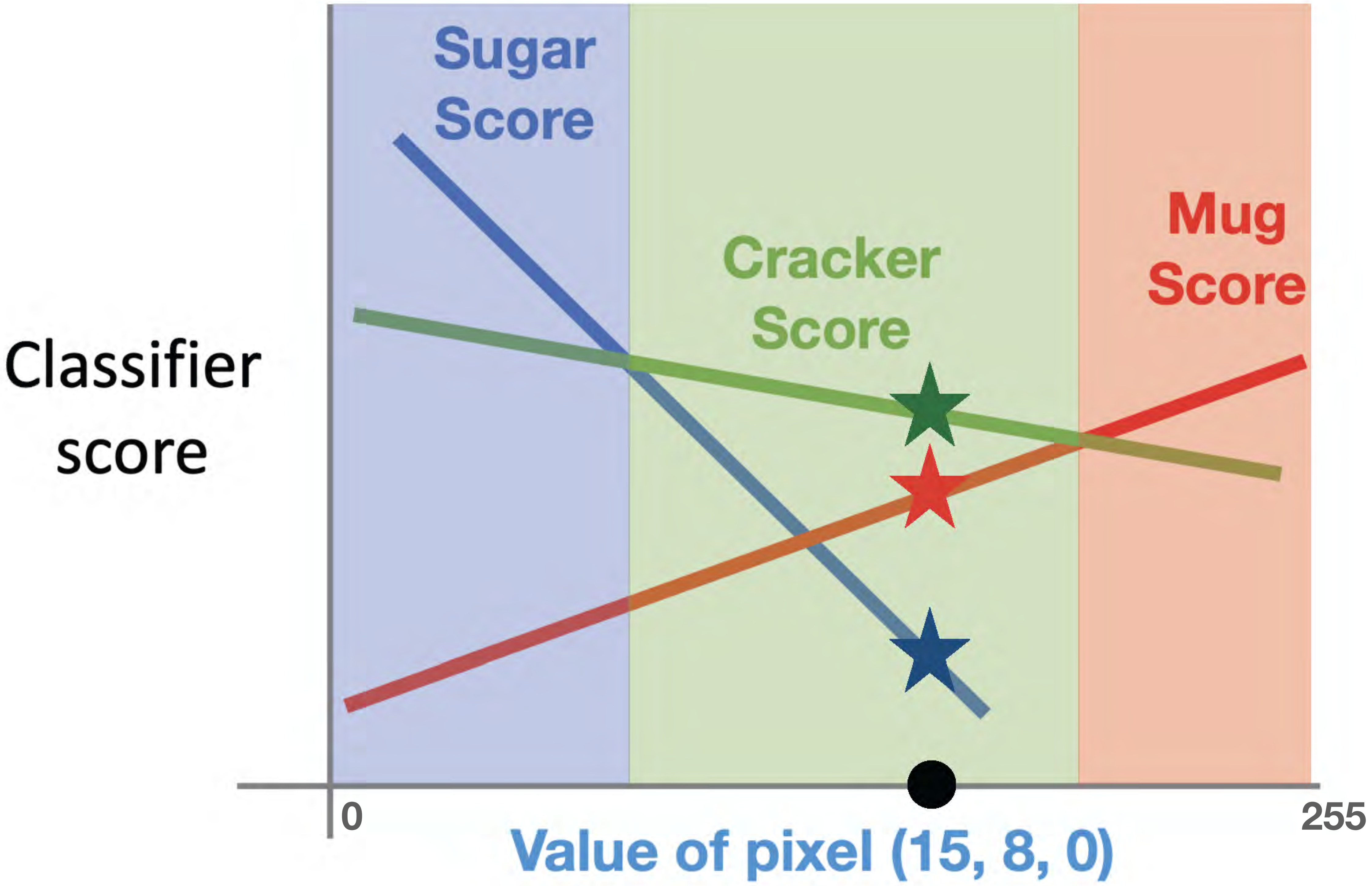
Interpreting a Linear Classifier—Geometric Viewpoint



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

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

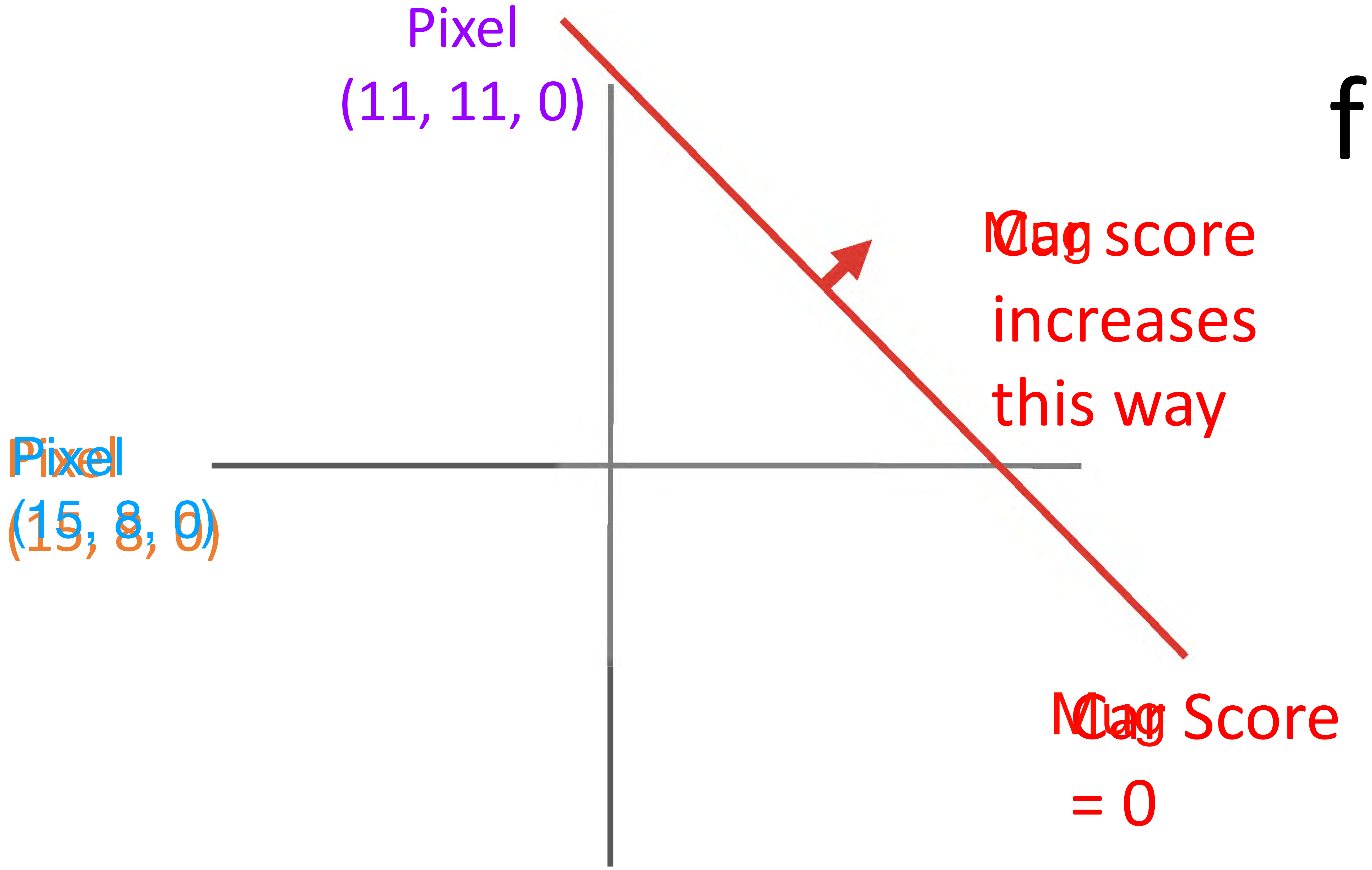
Interpreting a Linear Classifier—Geometric Viewpoint



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

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

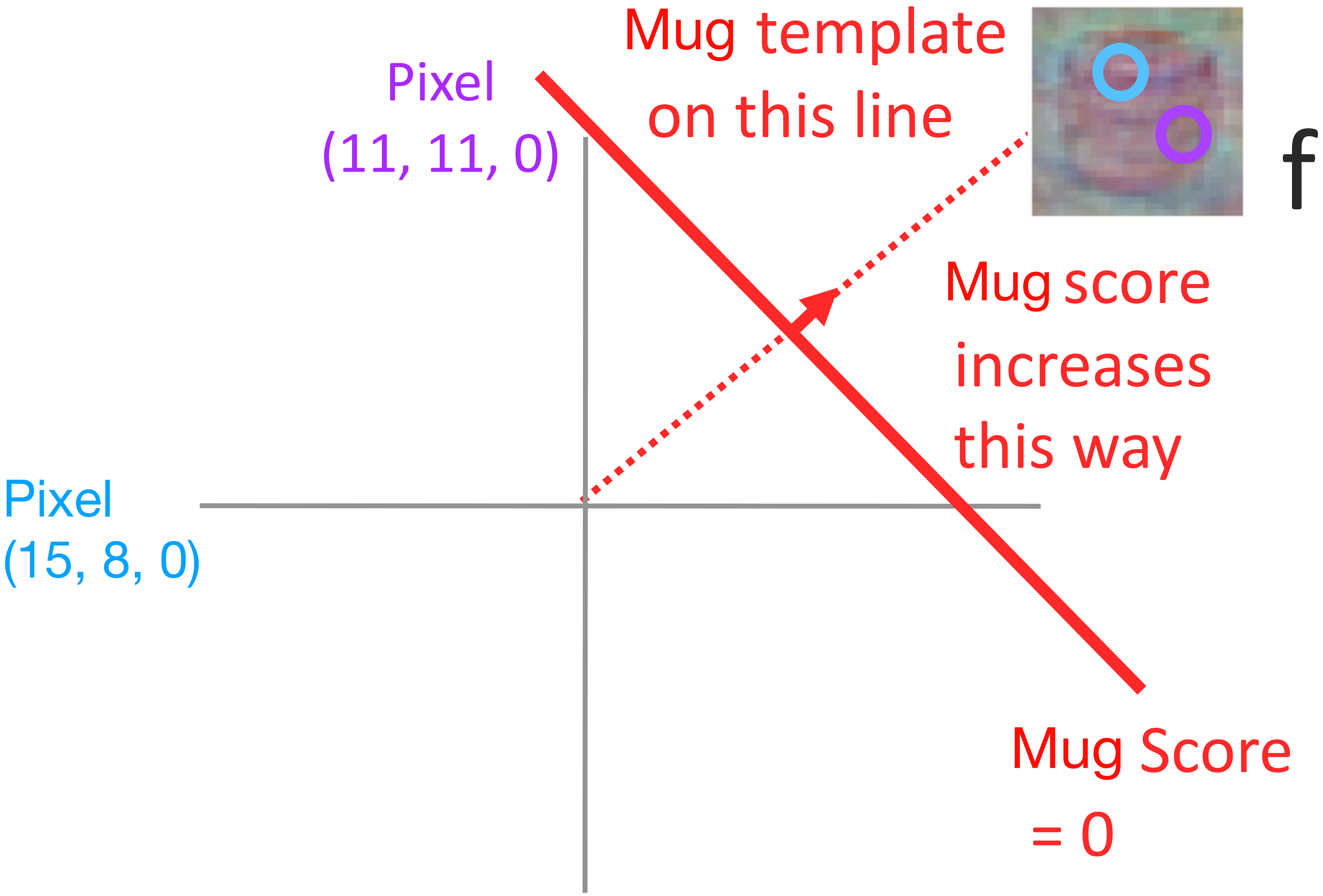
Interpreting a Linear Classifier—Geometric Viewpoint



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

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

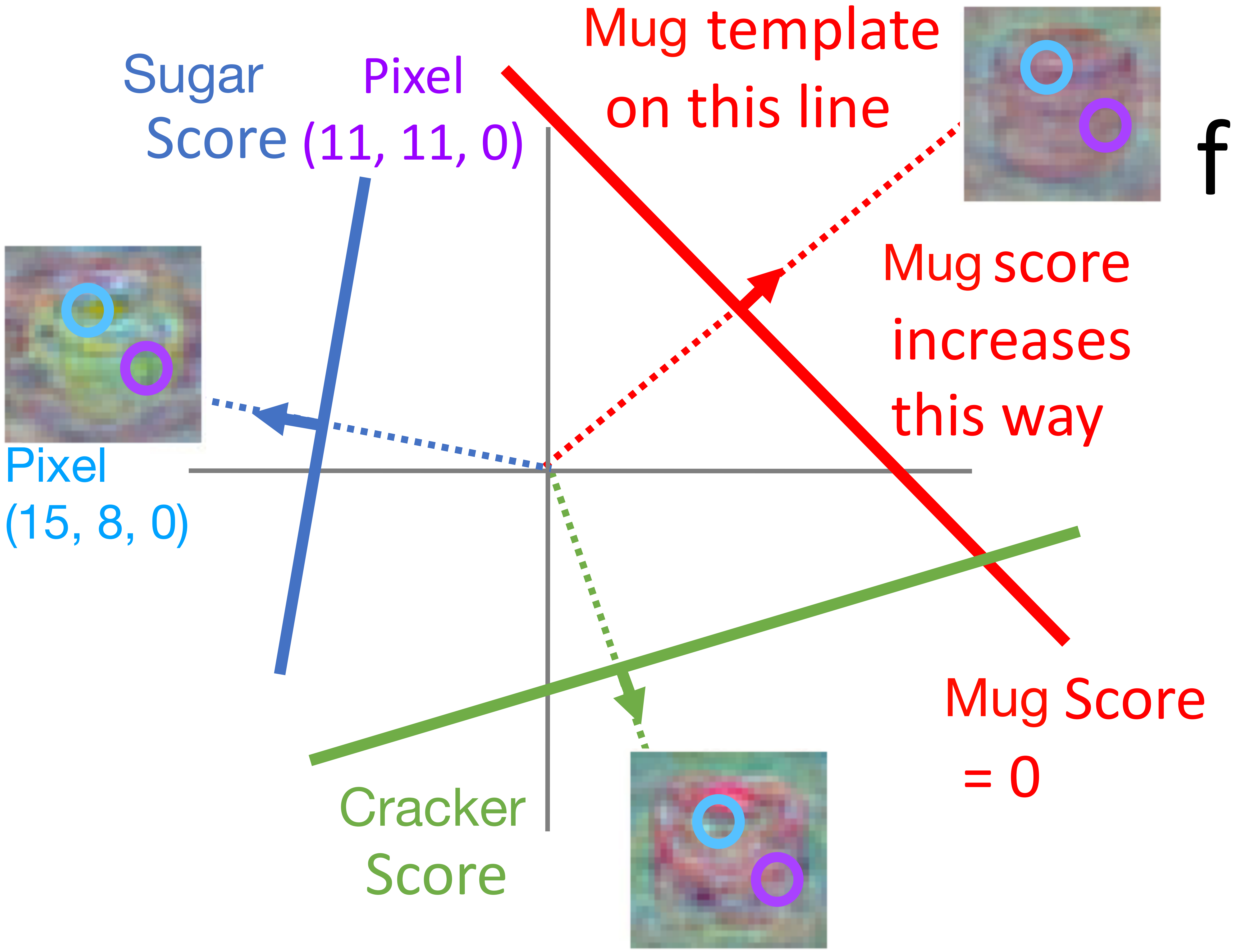
Interpreting a Linear Classifier—Geometric Viewpoint



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

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

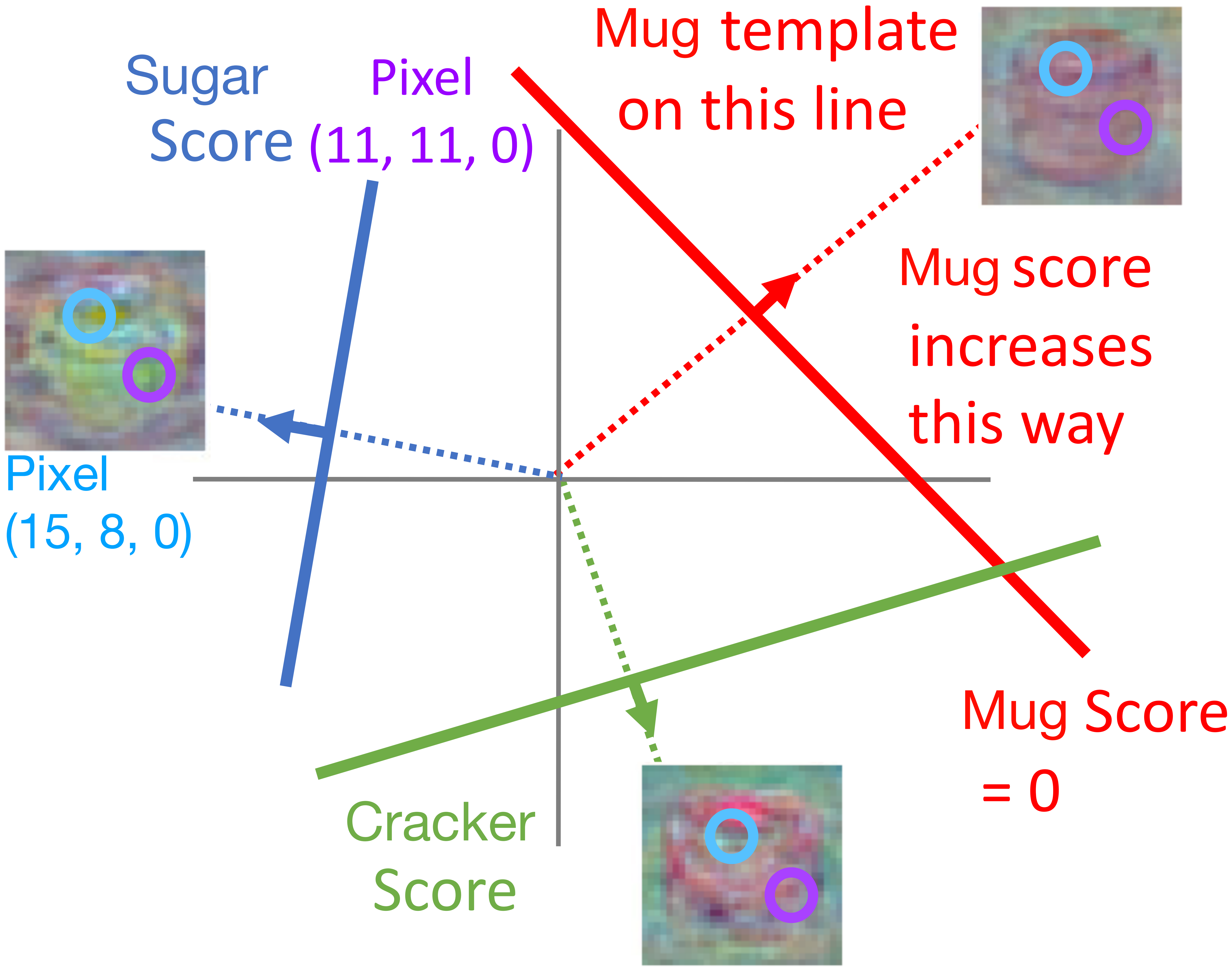
Interpreting a Linear Classifier—Geometric Viewpoint



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

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

Interpreting a Linear Classifier—Geometric Viewpoint



Sugar Score

Pixel (11, 11, 0)

Mug template on this line

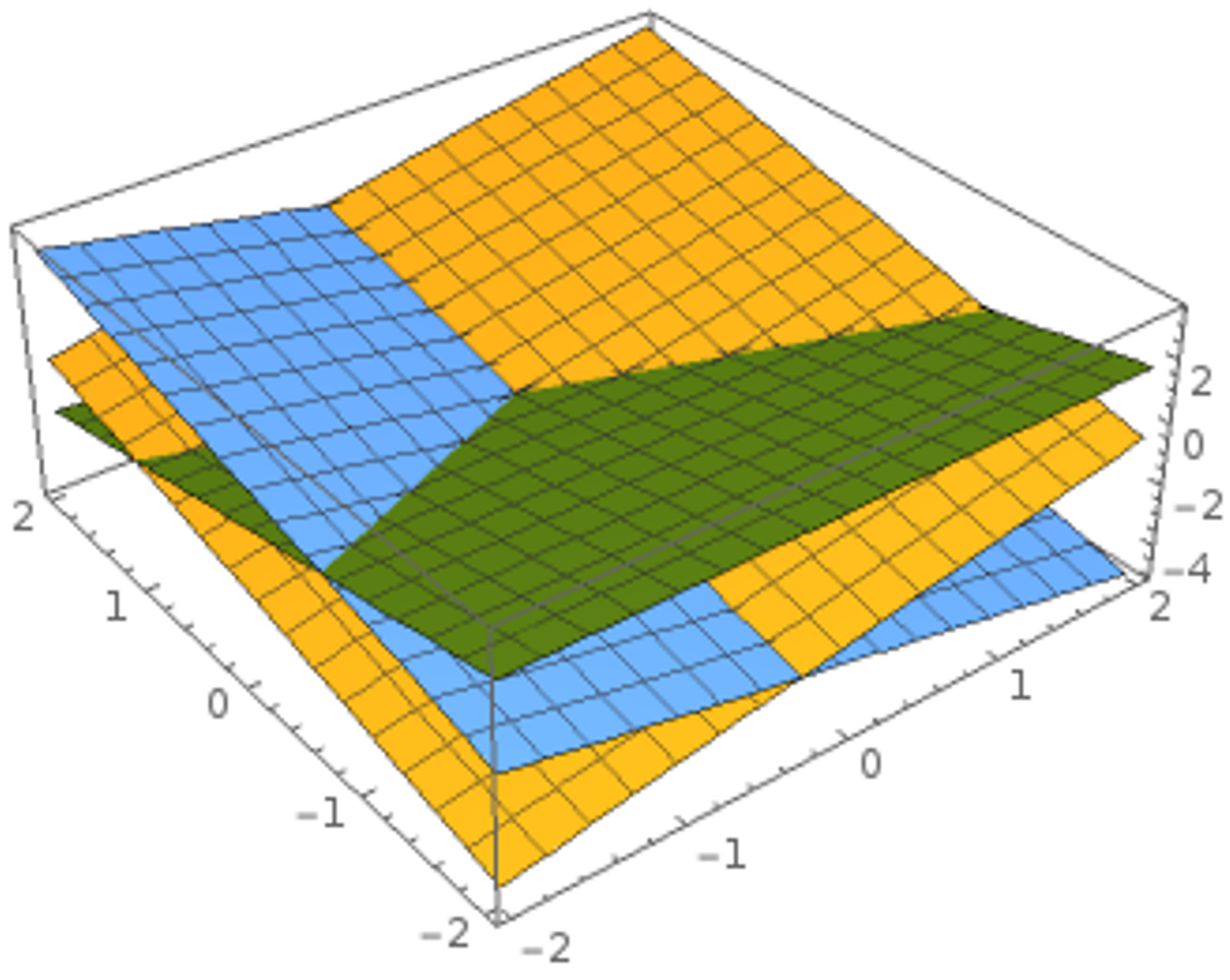
Pixel (15, 8, 0)

Mug score increases this way

Mug Score = 0

Cracker Score

Hyperplanes carving up a high-dimensional space

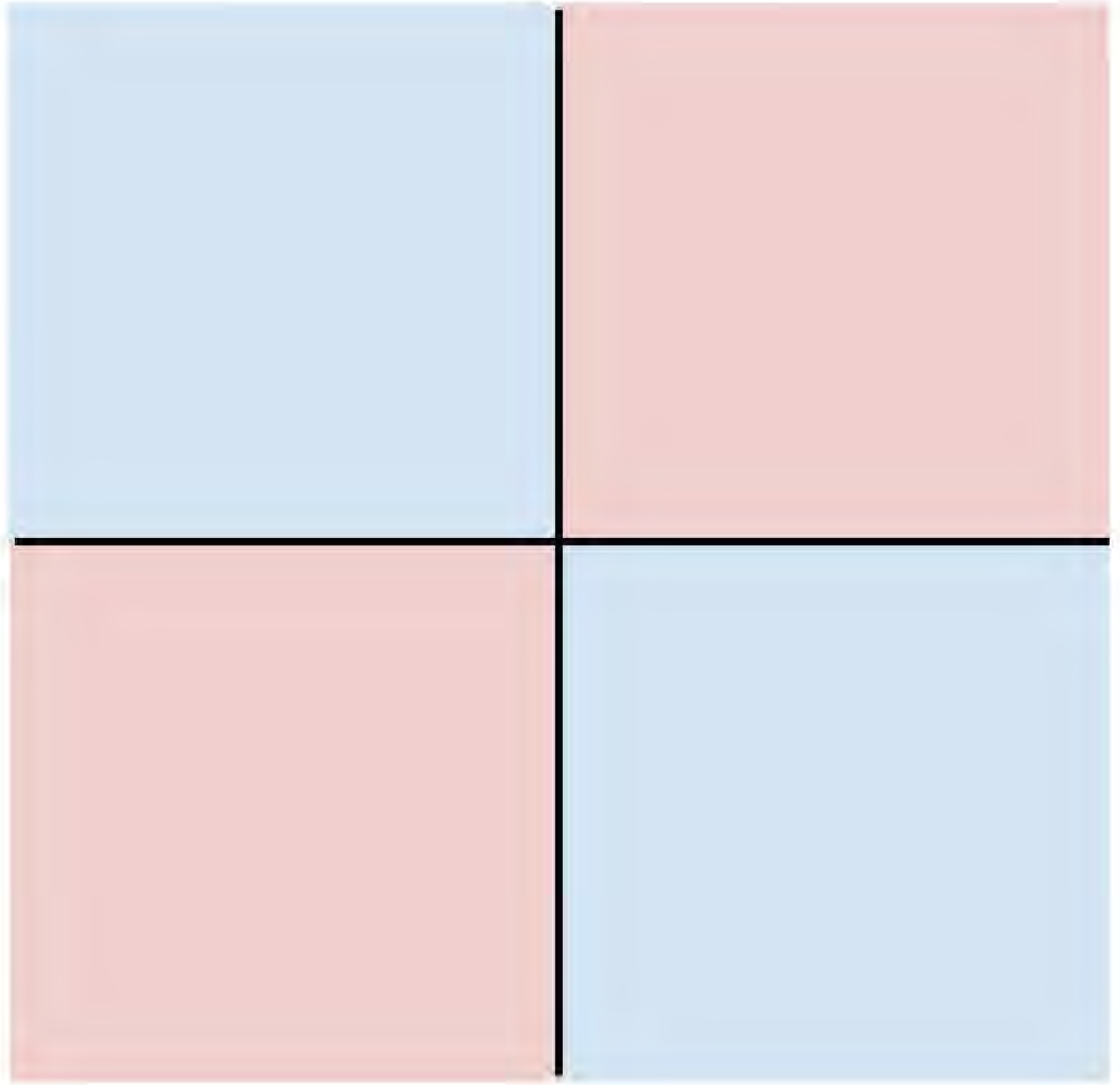


Plot created using Wolfram Cloud

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

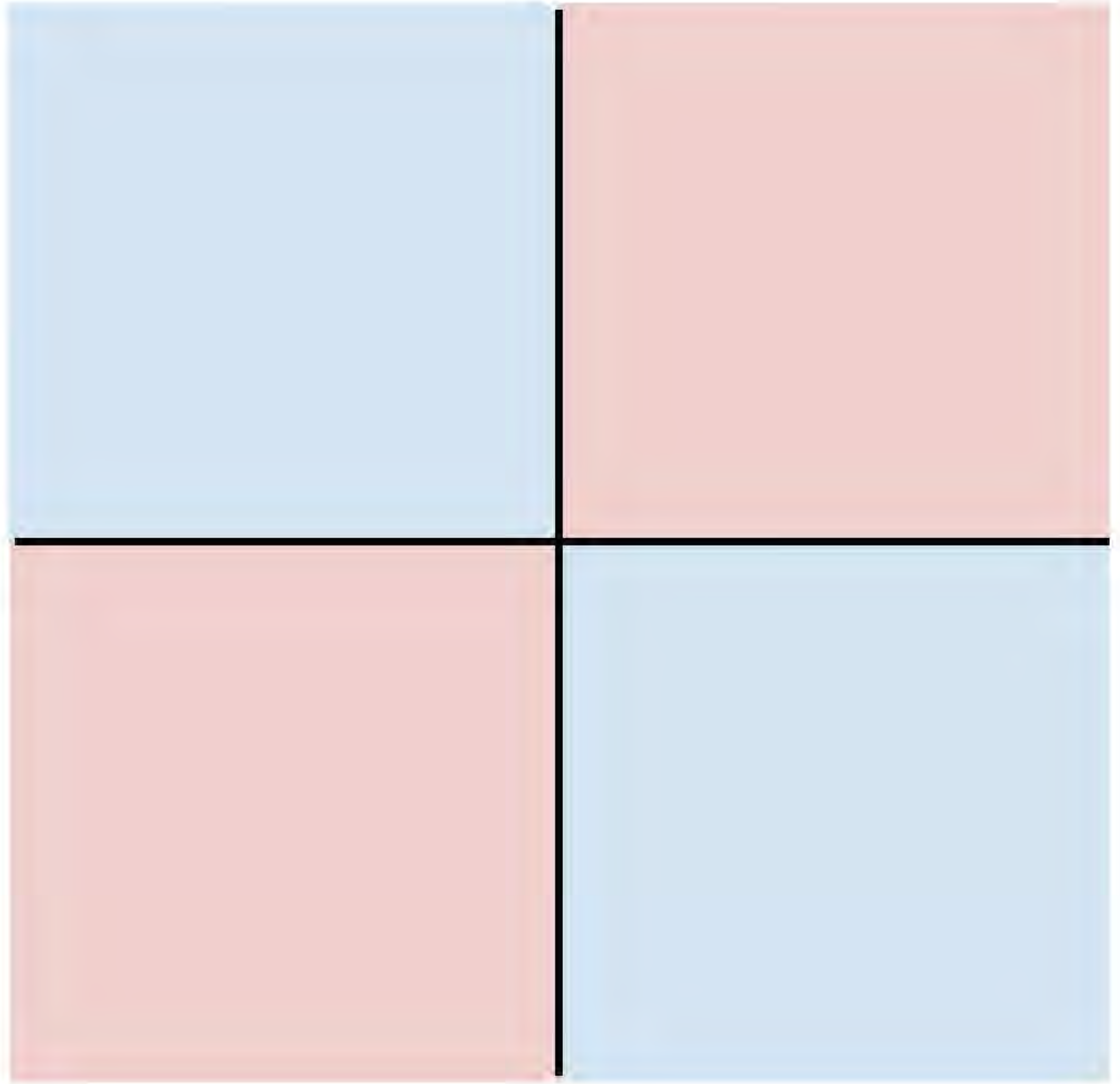
Class 1:
Three modes

Class 2:
Everything else

Hard Cases for a Linear Classifier

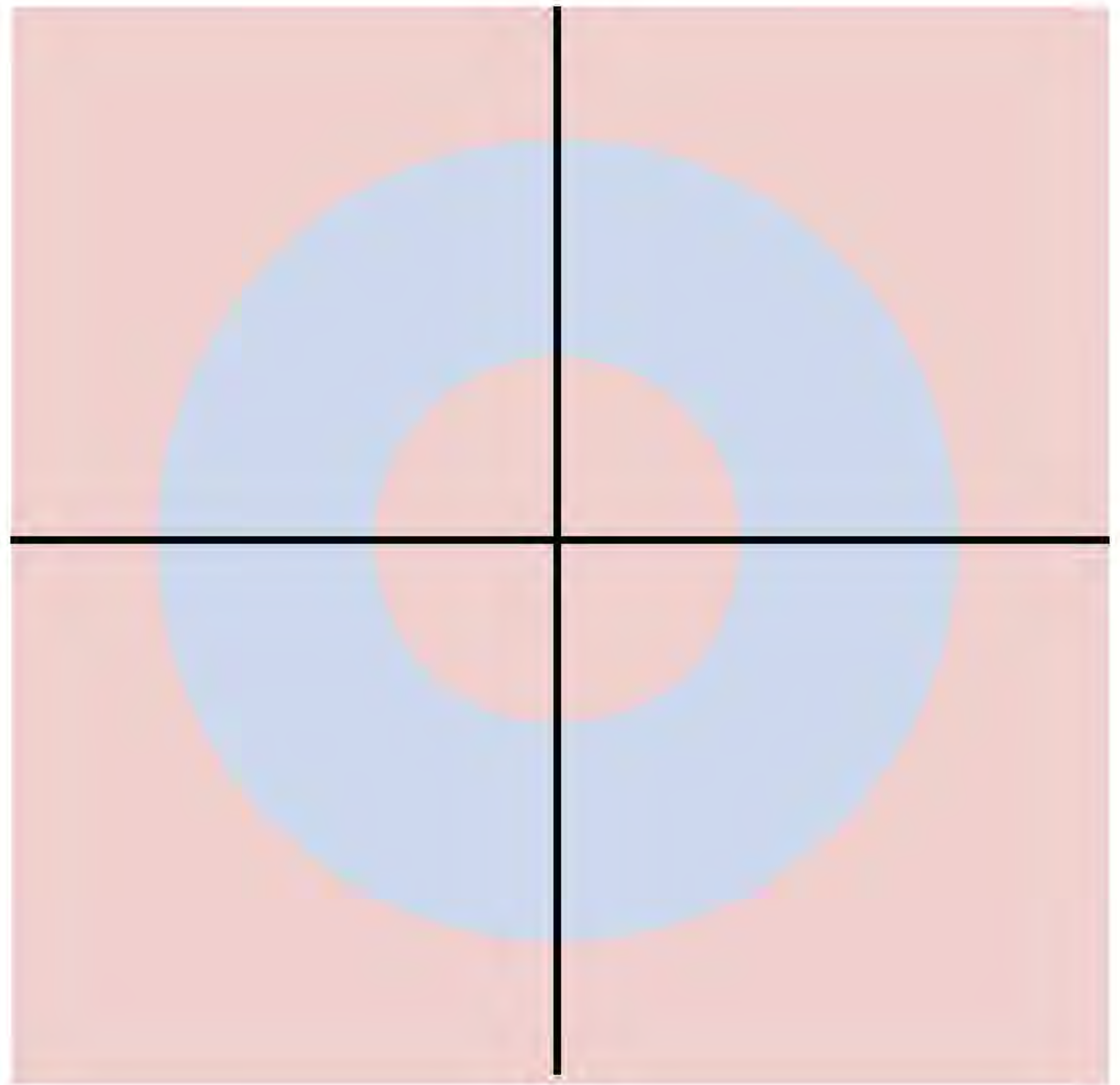
Class 1:
First and third quadrants

Class 2:
Second and fourth quadrants



Class 1:
1 <= L2 norm <= 2

Class 2:
Everything else



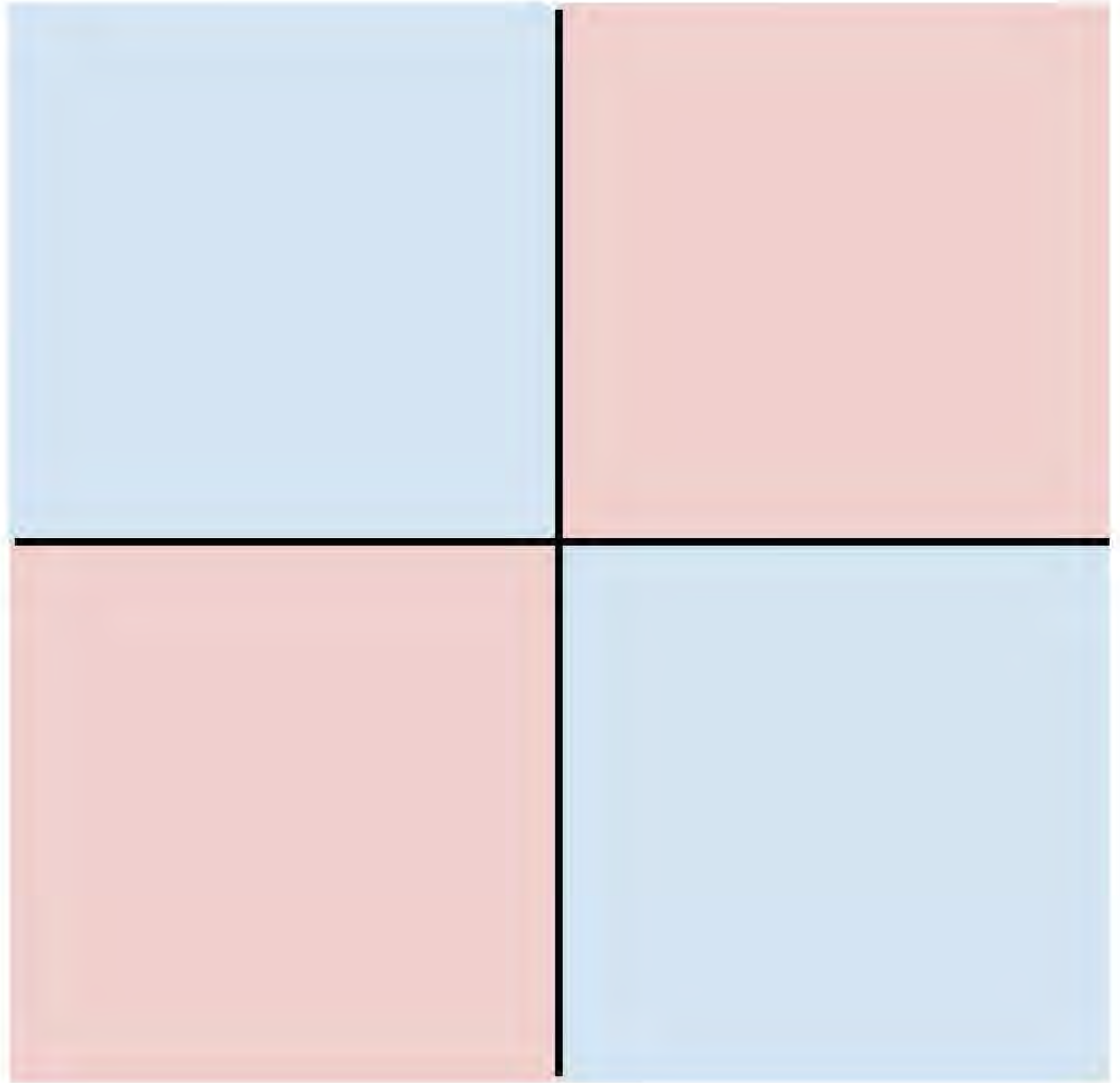
Class 1:
Three modes

Class 2:
Everything else

Hard Cases for a Linear Classifier

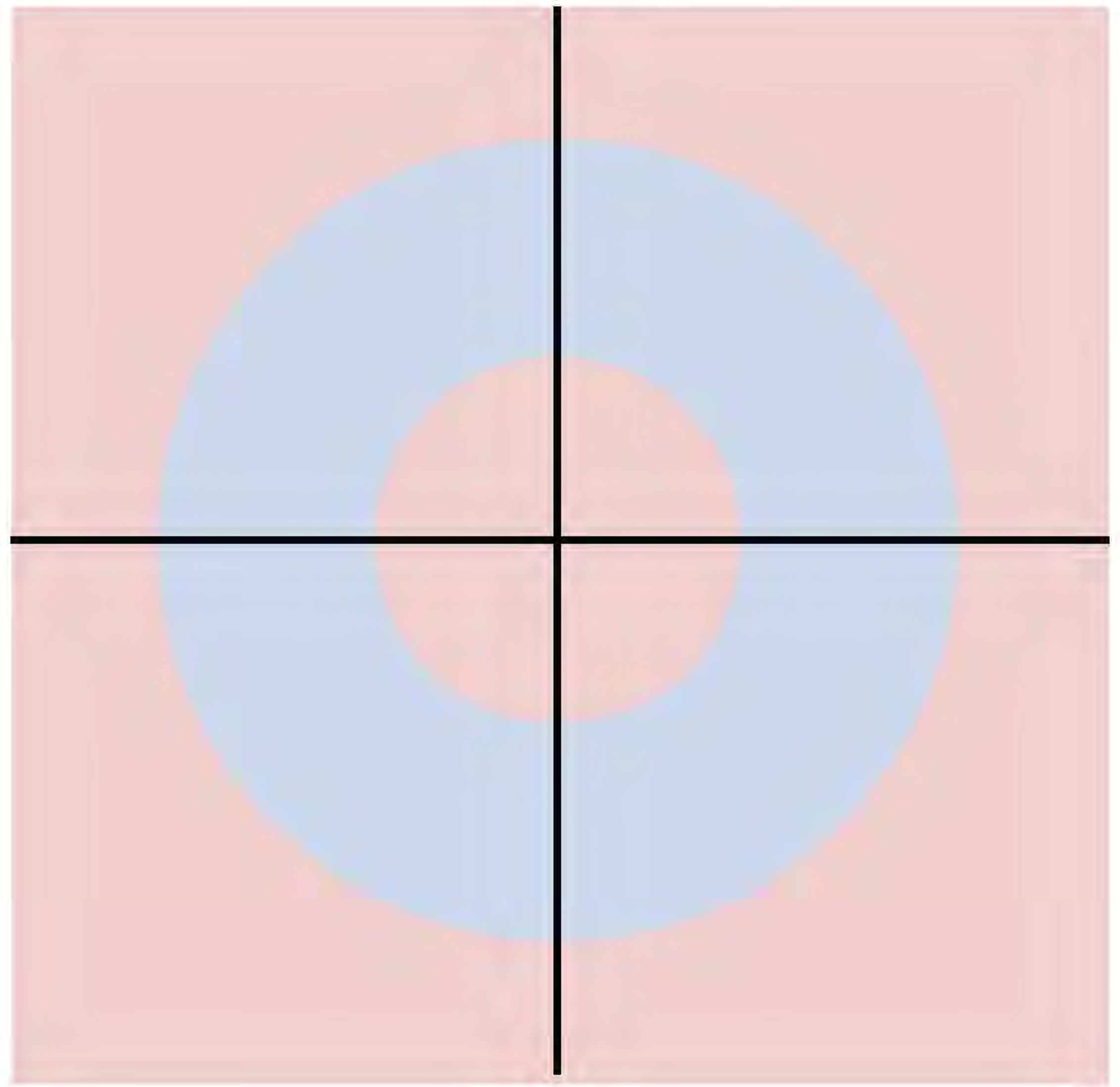
Class 1:
First and third quadrants

Class 2:
Second and fourth quadrants



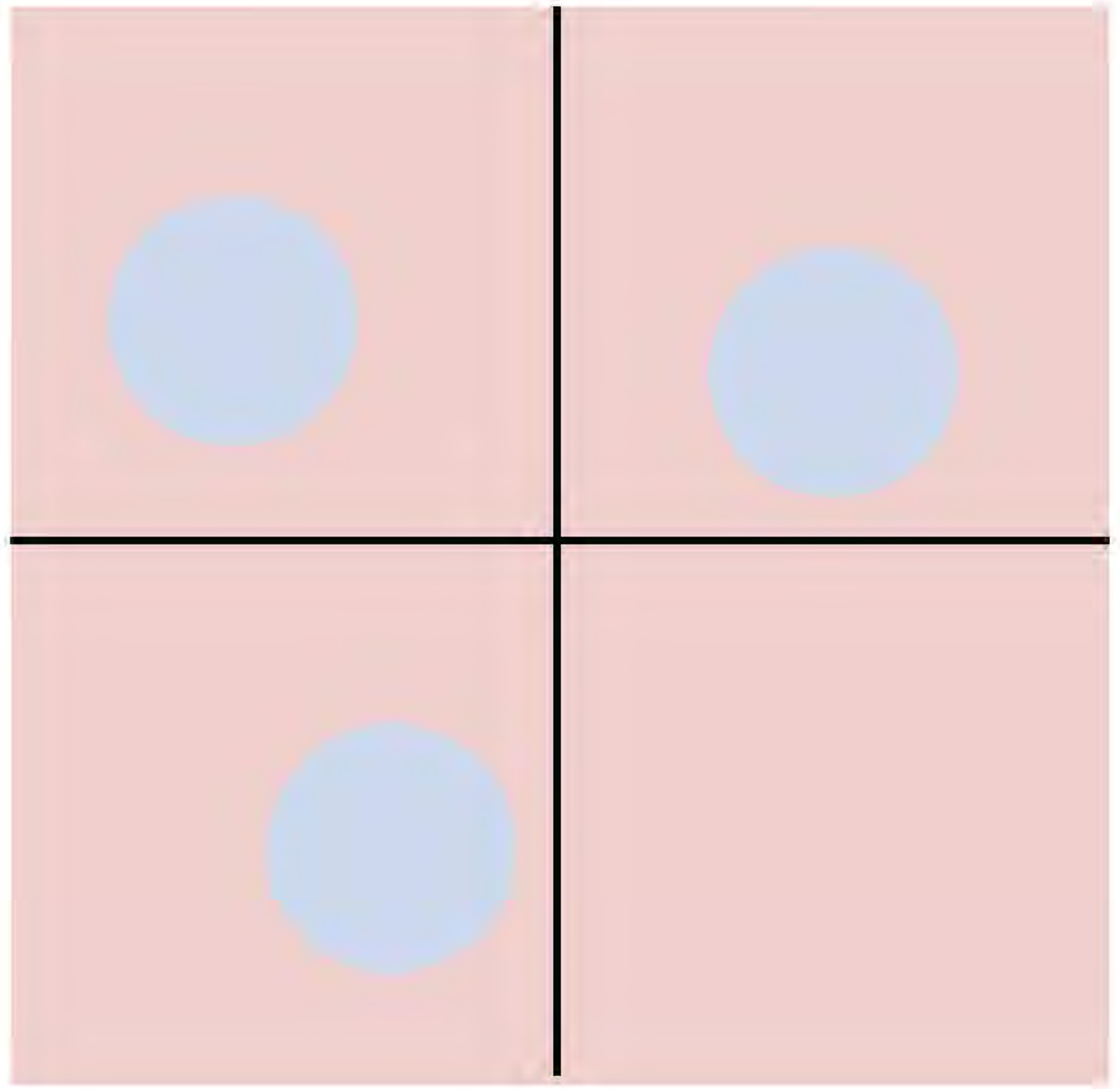
Class 1:
1 <= L2 norm <= 2

Class 2:
Everything else



Class 1:
Three modes

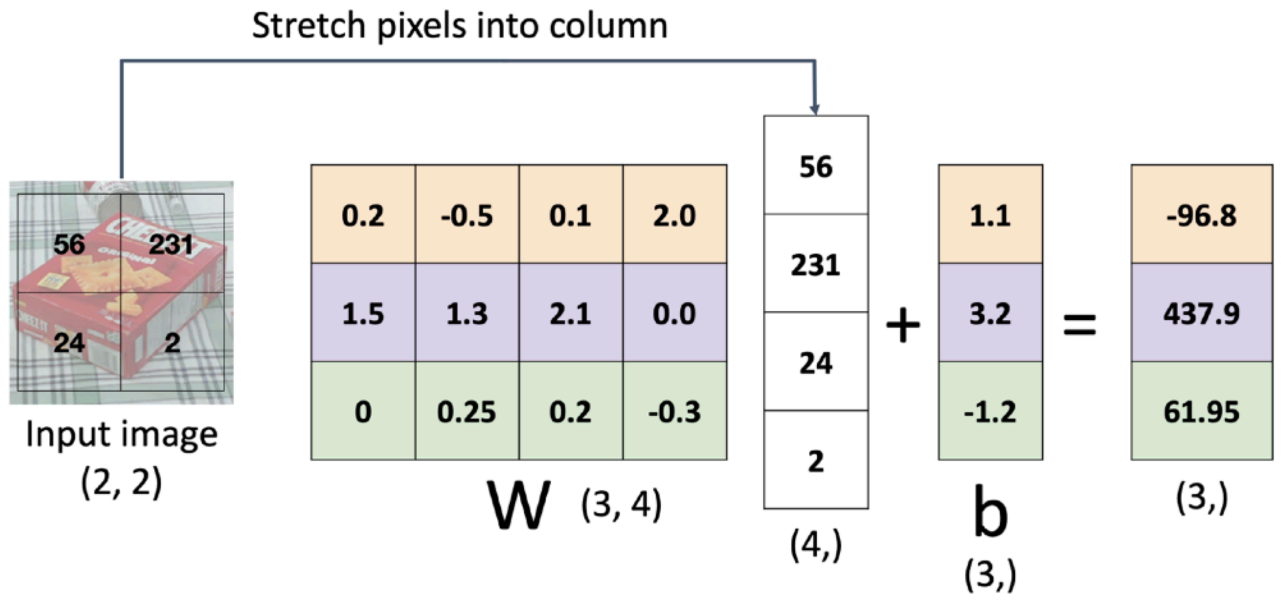
Class 2:
Everything else



Linear Classifier—Three Viewpoints

Algebraic Viewpoint

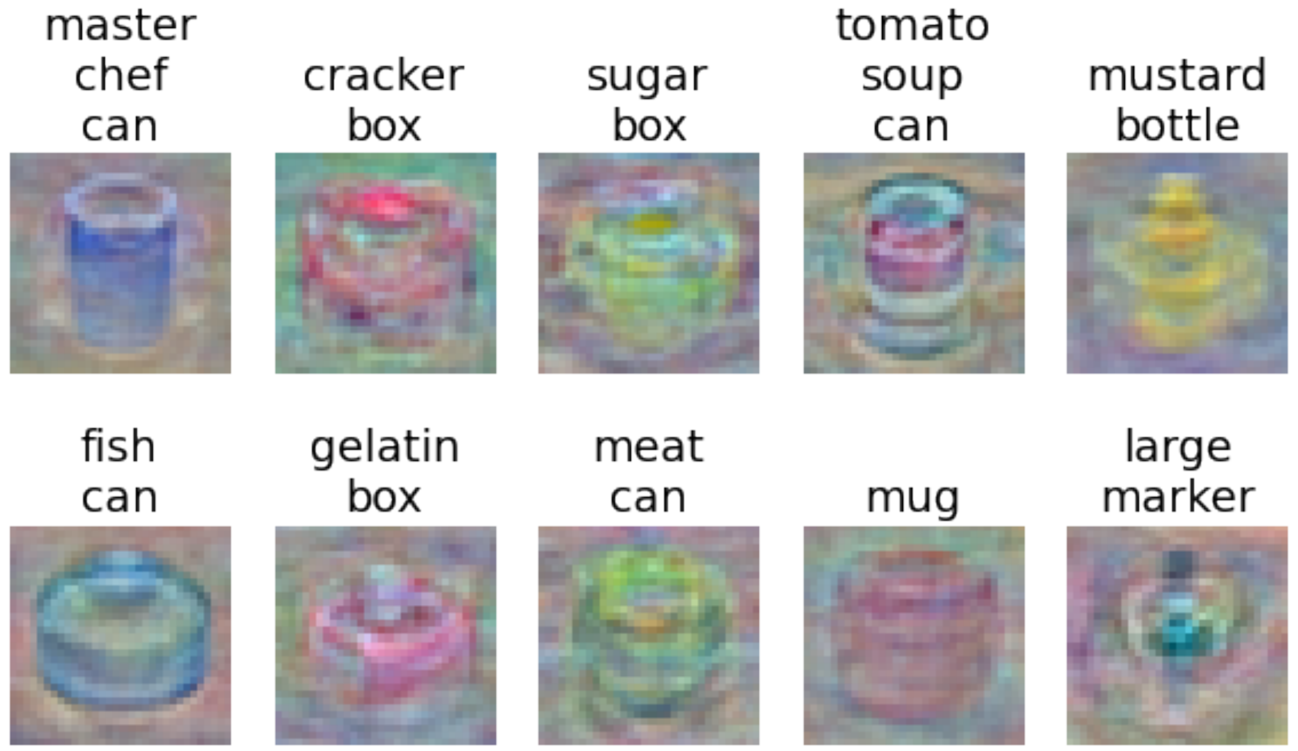
$$f(x,W) = Wx$$



$$\underset{W\ (3,4)}{\begin{bmatrix}0.2 & -0.5 & 0.1 & 2.0\\ 1.5 & 1.3 & 2.1 & 0.0\\ 0 & 0.25 & 0.2 & -0.3\end{bmatrix}} \underset{(4,)}{\begin{bmatrix}56\\ 231\\ 24\\ 2\end{bmatrix}} + \underset{\substack{b\\ (3,)}}{\begin{bmatrix}1.1\\ 3.2\\ -1.2\end{bmatrix}} = \underset{(3,)}{\begin{bmatrix}-96.8\\ 437.9\\ 61.95\end{bmatrix}}$$

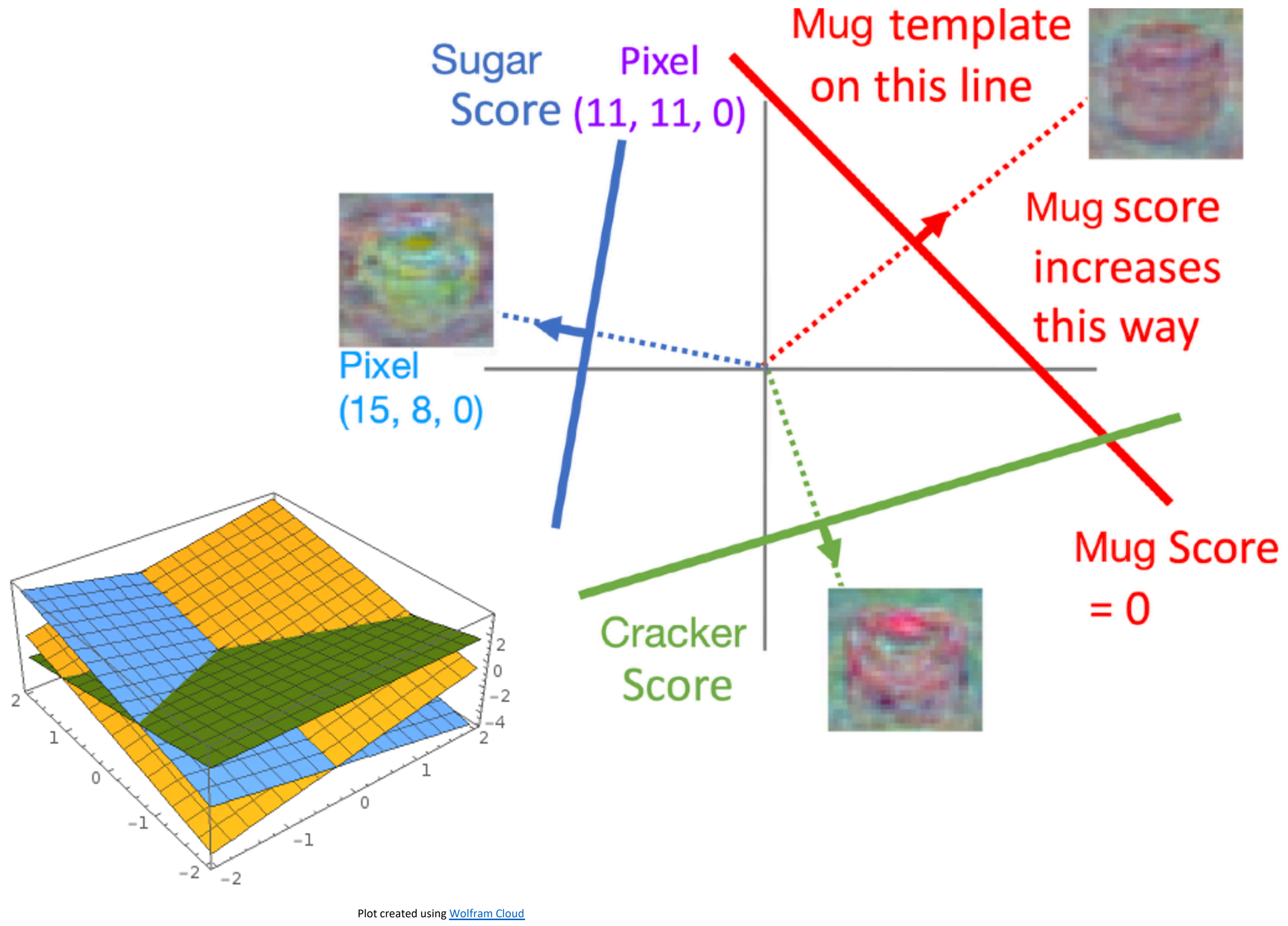
Visual Viewpoint

One template per class



Geometric Viewpoint

Hyperplanes cutting up space


Sugar Score
Pixel (11, 11, 0)
Mug template on this line
Mug score increases this way
Pixel (15, 8, 0)
Mug Score = 0
Cracker Score
Plot created using Wolfram Cloud

So far—Defined a Score Function

master chef can	-3.45	-0.51	3.42
mug	-8.87	**6.04**	4.64
tomato soup can	0.09	5.31	2.65
cracker box	**2.9**	-4.22	5.1
mustard bottle	4.48	-4.19	2.64
tuna fish can	8.02	3.58	5.55
sugar box	3.78	4.49	**-4.34**
gelatin box	1.06	-4.37	-1.5
potted meat can	-0.36	-2.09	-4.79
large marker	-0.72	-2.93	6.14

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

Given a W, we can compute class scores for an image, x.

But how can we actually choose a good W?

So far—Choosing a Good W

master chef can	-3.45	-0.51	3.42
mug	-8.87	**6.04**	4.64
tomato soup can	0.09	5.31	2.65
cracker box	**2.9**	-4.22	5.1
mustard bottle	4.48	-4.19	2.64
tuna fish can	8.02	3.58	5.55
sugar box	3.78	4.49	**-4.34**
gelatin box	1.06	-4.37	-1.5
potted meat can	-0.36	-2.09	-4.79
large marker	-0.72	-2.93	6.14

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

TODO:

1. Use a **loss function** to quantify how good a value of W is

2. Find a W that minimizes the loss function (**optimization**)

Loss Function

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

Low loss = good classifier
High loss = bad classifier

Also called: **objective function, cost function**

Loss Function

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

Low loss = good classifier
High loss = bad classifier

Also called: **objective function, cost function**

Negative loss function sometimes called **reward function, profit function, utility function, fitness function,** etc.

Loss Function

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

Low loss = good classifier
High loss = bad classifier

Also called: **objective function, cost function**

Negative loss function sometimes called **reward function, profit function, utility function, fitness function,** etc.

Given a dataset of examples

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

where x_i is an image and
y_i is a (discrete) label

Loss Function

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

Low loss = good classifier
High loss = bad classifier

Also called: **objective function, cost function**

Negative loss function sometimes called **reward function, profit function, utility function, fitness function,** etc.

Given a dataset of examples

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

where x_i is an image and
y_i is a (discrete) label

Loss for a single example is

$$L_i(f(x_i, W), y_i)$$

Loss Function

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

Low loss = good classifier
High loss = bad classifier

Also called: **objective function, cost function**

Negative loss function sometimes called **reward function, profit function, utility function, fitness function,** etc.

Given a dataset of examples

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

where x_i is an image and
y_i is a (discrete) label

Loss for a single example is

$$L_i(f(x_i, W), y_i)$$

Loss for the dataset is average of per-example losses:

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

Cross-Entropy Loss
Multinomial Logistic Regression

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

cracker	**3.2**
mug	5.1
sugar	-1.7

Cross-Entropy Loss
Multinomial Logistic Regression

cracker	**3.2**
mug	5.1
sugar	-1.7

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

$$s = f(x_i; W) \qquad P(Y = k \mid X = x_i) = \frac{\exp(s_k)}{\sum_j \exp(s_j)}$$

Softmax function

Cross-Entropy Loss
Multinomial Logistic Regression

cracker	**3.2**
mug	5.1
sugar	-1.7

Unnormalized log-probabilities (logits)

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

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

$$P(Y = k \mid X = x_i) = \frac{\exp(s_k)}{\sum_j \exp(s_j)}$$

Softmax function

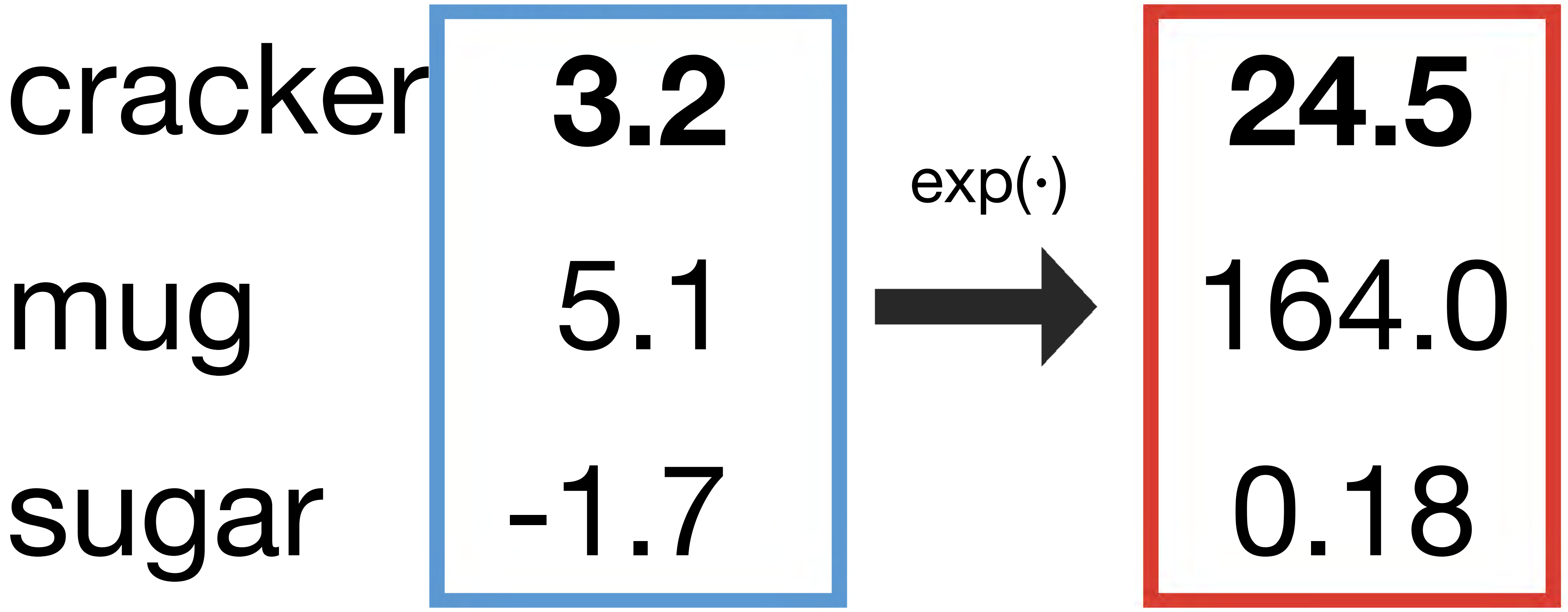
Cross-Entropy Loss
Multinomial Logistic Regression

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

$$s = f(x_i; W) \qquad P(Y = k \mid X = x_i) = \frac{\exp(s_k)}{\sum_j \exp(s_j)}$$

Softmax function

Probabilities must be >=0



cracker	**3.2**	exp(·)	**24.5**
mug	5.1		164.0
sugar	-1.7		0.18

Unnormalized log-probabilities (logits)

Unnormalized probabilities

Cross-Entropy Loss
Multinomial Logistic Regression

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

$$s = f(x_i; W) \qquad P(Y = k \mid X = x_i) = \frac{\exp(s_k)}{\sum_j \exp(s_j)}$$

Softmax function

Probabilities must be >=0

Probabilities must sum to 1

cracker	**3.2**	**24.5**	**0.13**
mug	5.1	164.0	0.87
sugar	-1.7	0.18	0.00
	Unnormalized log-probabilities (logits)	Unnormalized probabilities	Probabilities

$$\xrightarrow{\exp(\cdot)} \qquad \xrightarrow{\text{normalize}}$$

Cross-Entropy Loss
Multinomial Logistic Regression

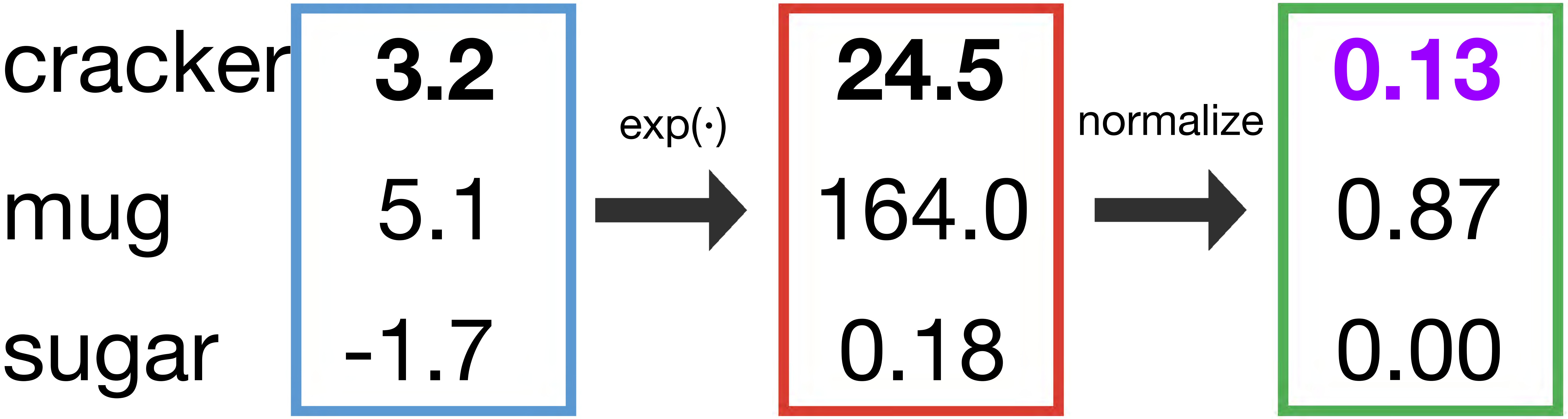
Want to interpret raw classifier scores as **probabilities**

$$s = f(x_i; W) \qquad P(Y = k \mid X = x_i) = \frac{\exp(s_k)}{\sum_j \exp(s_j)}$$

Softmax function

Probabilities must be >=0

Probabilities must sum to 1



Unnormalized log-probabilities (logits)

Unnormalized probabilities

Probabilities

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

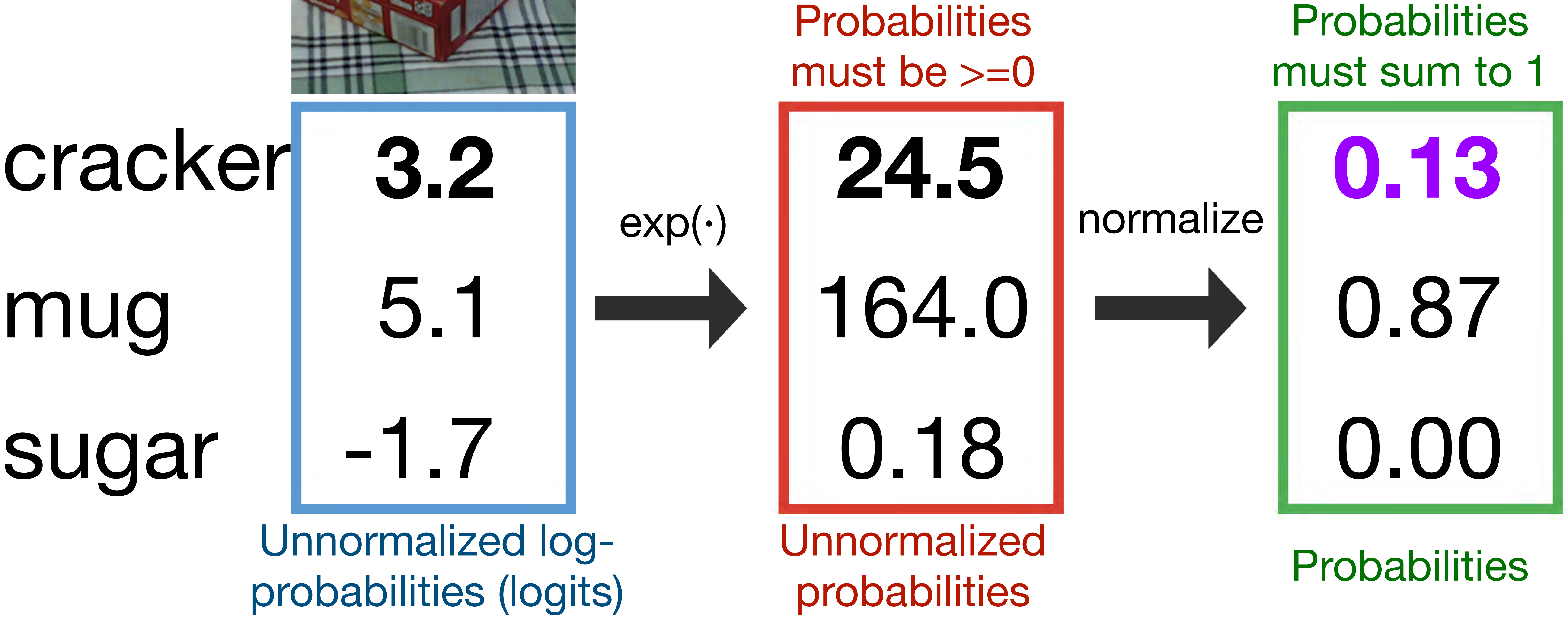
$$L_i = -\log(0.13)$$
$$= 2.04$$

Cross-Entropy Loss
Multinomial Logistic Regression

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

$$s = f(x_i; W) \qquad P(Y = k \mid X = x_i) = \frac{\exp(s_k)}{\sum_j \exp(s_j)}$$

Softmax function



Probabilities must be >=0

Probabilities must sum to 1

		exp(·) →		normalize →	
cracker	**3.2**		**24.5**		**0.13**
mug	5.1		164.0		0.87
sugar	-1.7		0.18		0.00
	Unnormalized log-probabilities (logits)		Unnormalized probabilities		Probabilities

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

$$L_i = -\log(0.13)$$
$$= 2.04$$

Maximum Likelihood Estimation
Choose weights to maximize the likelihood of the observed data (see CSCI 5521)

Cross-Entropy Loss
Multinomial Logistic Regression

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

$$s = f(x_i; W) \qquad P(Y = k \mid X = x_i) = \frac{\exp(s_k)}{\sum_j \exp(s_j)}$$

Softmax function

Probabilities must be >=0

Probabilities must sum to 1

	Unnormalized log-probabilities (logits)	exp(·) →	Unnormalized probabilities	normalize →	Probabilities	compare	Correct probabilities
cracker	**3.2**		**24.5**		**0.13**	→ compare ←	**1.00**
mug	5.1		164.0		0.87		0.00
sugar	-1.7		0.18		0.00		0.00

Cross-Entropy Loss
Multinomial Logistic Regression

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

$$s = f(x_i; W) \qquad P(Y = k \mid X = x_i) = \frac{\exp(s_k)}{\sum_j \exp(s_j)}$$

Softmax function

Probabilities must be >=0

Probabilities must sum to 1

	Unnormalized log-probabilities (logits)	Unnormalized probabilities	Probabilities	Correct probabilities
cracker	**3.2**	**24.5**	**0.13**	**1.00**
mug	5.1	164.0	0.87	0.00
sugar	-1.7	0.18	0.00	0.00

exp(·) →

normalize →

→ compare ←

Kullback-Leibler divergence

$$D_{KL}(P\,||\,Q) = \sum_y P(y) \log \frac{P(y)}{Q(y)}$$

Cross-Entropy Loss
Multinomial Logistic Regression

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

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

$$P(Y = k \mid X = x_i) = \frac{\exp(s_k)}{\sum_j \exp(s_j)}$$ Softmax function

Probabilities must be >=0

Probabilities must sum to 1

cracker 3.2 → exp(·) → 24.5 → normalize → 0.13 → compare ← 1.00

mug 5.1 → 164.0 → 0.87 — 0.00

sugar -1.7 → 0.18 → 0.00 — 0.00

Unnormalized log-probabilities (logits)

Unnormalized probabilities

Probabilities

Correct probabilities

Cross Entropy

$$H(P, Q) = H(P) + D_{KL}(P\,||\,Q)$$

Cross-Entropy Loss
Multinomial Logistic Regression

cracker	**3.2**
mug	5.1
sugar	-1.7

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

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

$$P(Y = k \mid X = x_i) = \frac{\exp(s_k)}{\sum_j \exp(s_j)}$$

Softmax function

Maximize probability of correct class

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

Putting it all together

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

Cross-Entropy Loss
Multinomial Logistic Regression

cracker	**3.2**
mug	5.1
sugar	-1.7

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

$$s = f(x_i; W) \qquad P(Y = k \mid X = x_i) = \frac{\exp(s_k)}{\sum_j \exp(s_j)}$$

Softmax function

Maximize probability of correct class

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

Putting it all together

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

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

Cross-Entropy Loss
Multinomial Logistic Regression

cracker	**3.2**
mug	5.1
sugar	-1.7

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

$$s = f(x_i; W) \qquad P(Y = k \mid X = x_i) = \frac{\exp(s_k)}{\sum_j \exp(s_j)}$$

Softmax function

Maximize probability of correct class

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

Putting it all together

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

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

A: Min: 0, Max: $+\infty$

Cross-Entropy Loss
Multinomial Logistic Regression

cracker	**3.2**
mug	5.1
sugar	-1.7

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

$$s = f(x_i; W) \qquad P(Y = k \mid X = x_i) = \frac{\exp(s_k)}{\sum_j \exp(s_j)}$$

Softmax function

Maximize probability of correct class

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

Putting it all together

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

Q: If all scores are small random values, what is the loss?

Cross-Entropy Loss
Multinomial Logistic Regression

cracker	**3.2**
mug	5.1
sugar	-1.7

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

$$s = f(x_i; W) \qquad P(Y = k \mid X = x_i) = \frac{\exp(s_k)}{\sum_j \exp(s_j)}$$

Softmax function

Maximize probability of correct class

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

Putting it all together

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

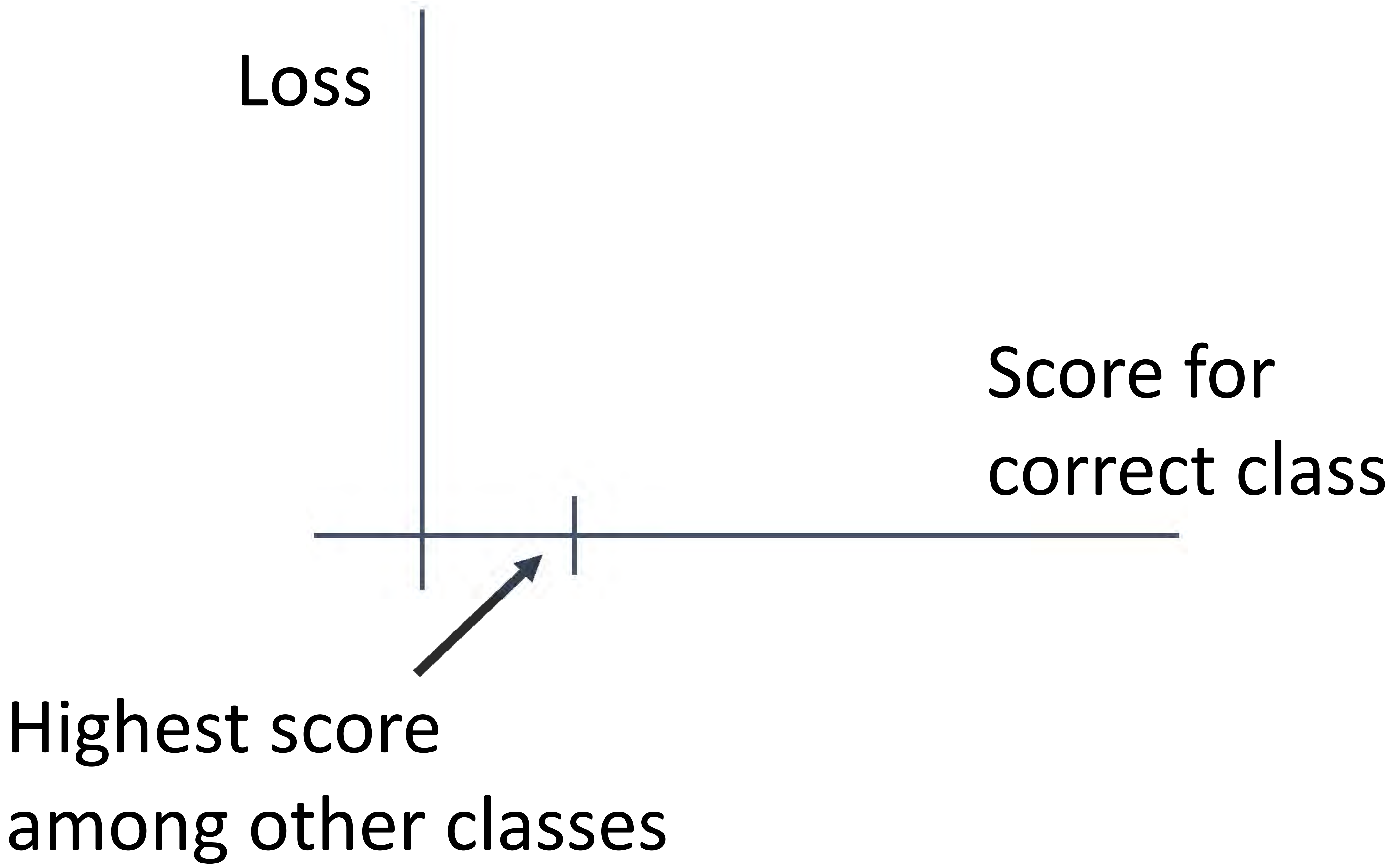
Q: If all scores are small random values, what is the loss?

A: $-\log(\frac{1}{C})$

$$\log(\frac{1}{10}) \approx 2.3$$

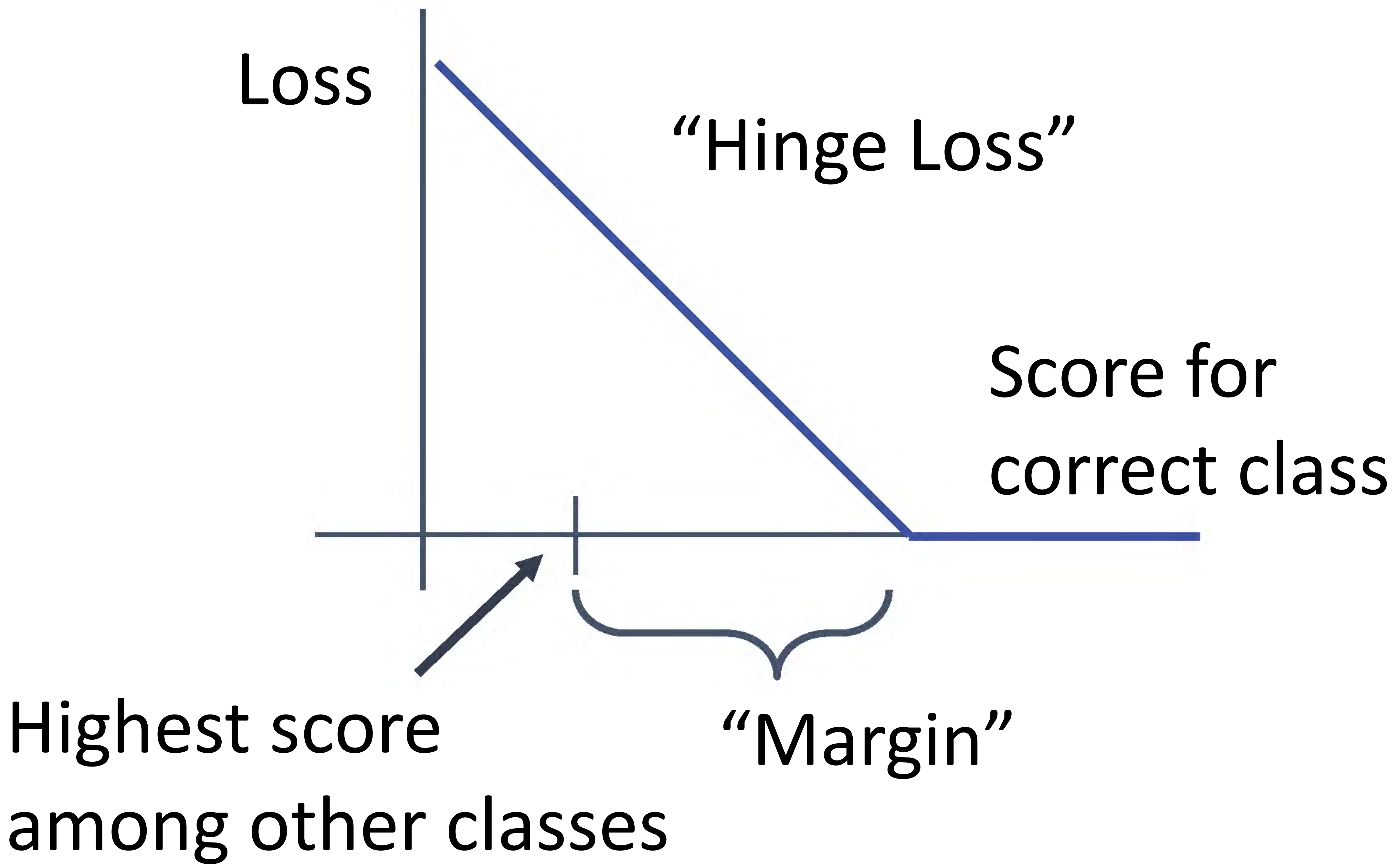
Multiclass SVM Loss

”The score of the correct class should be higher than all the other scores”



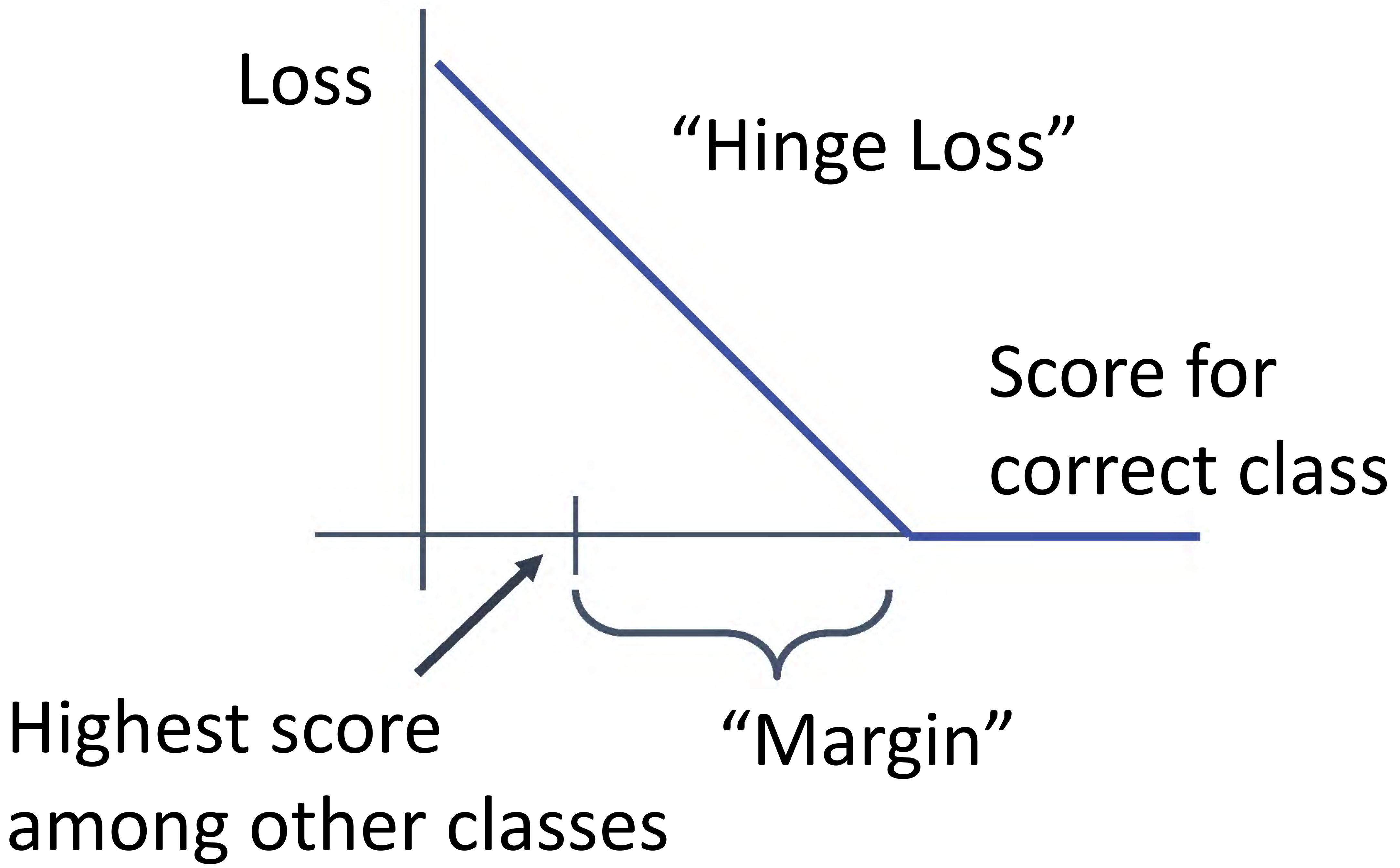
Multiclass SVM Loss

”The score of the correct class should be higher than all the other scores”



Multiclass SVM Loss

”The score of the correct class should be higher than all the other scores”



Given an example (x_i, y_i)
(x_i is image, y_i is label)

Let $s = f(x_i, W)$ be scores

Then the SVM loss has the form:

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

Multiclass SVM Loss

cracker	**3.2**	1.3	2.2
mug	5.1	**4.9**	2.5
sugar	-1.7	2.0	**-3.1**

Given an example (x_i, y_i)
(x_i is image, y_i is label)

Let $s = f(x_i, W)$ be scores

Then the SVM loss has the form:

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

Multiclass SVM Loss

cracker	**3.2**	1.3	2.2
mug	5.1	**4.9**	2.5
sugar	-1.7	2.0	**-3.1**
Loss	2.9		

Given an example (x_i, y_i)
(x_i is image, y_i is label)

Let $s = f(x_i, W)$ be scores

Then 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

Multiclass SVM Loss

cracker	**3.2**	1.3	2.2
mug	5.1	**4.9**	2.5
sugar	-1.7	2.0	**-3.1**
Loss	2.9	0	

Given an example (x_i, y_i)
(x_i is image, y_i is label)

Let $s = f(x_i, W)$ be scores

Then the SVM loss has the form:

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

= 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

Multiclass SVM Loss

cracker	**3.2**	1.3	2.2
mug	5.1	**4.9**	2.5
sugar	-1.7	2.0	**-3.1**
Loss	2.9	0	12.9

Given an example (x_i, y_i)
(x_i is image, y_i is label)

Let $s = f(x_i, W)$ be scores

Then the SVM loss has the form:

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

= 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

Multiclass SVM Loss

cracker	**3.2**	1.3	2.2
mug	5.1	**4.9**	2.5
sugar	-1.7	2.0	**-3.1**
Loss	2.9	0	12.9

Given an example (x_i, y_i)
(x_i is image, y_i is label)

Let $s = f(x_i, W)$ be scores

Then the SVM loss has the form:

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

Loss over the dataset is:

L = (2.9 + 0.0 + 12.9) / 3
= 5.27

Multiclass SVM Loss

cracker	**3.2**	1.3	2.2
mug	5.1	**4.9**	2.5
sugar	-1.7	2.0	**-3.1**
Loss	2.9	0	12.9

Given an example (x_i, y_i)
(x_i is image, y_i is label)

Let $s = f(x_i, W)$ be scores

Then 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 the loss if the scores for the mug image change a bit?

Multiclass SVM Loss

cracker	**3.2**	1.3	2.2
mug	5.1	**4.9**	2.5
sugar	-1.7	2.0	**-3.1**
Loss	2.9	0	12.9

Given an example (x_i, y_i)
(x_i is image, y_i is label)

Let $s = f(x_i, W)$ be scores

Then the SVM loss has the form:

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

Q2: What are the min
and max possible loss?

Multiclass SVM Loss

cracker	**3.2**	1.3	2.2
mug	5.1	**4.9**	2.5
sugar	-1.7	2.0	**-3.1**
Loss	2.9	0	12.9

Given an example (x_i, y_i)
(x_i is image, y_i is label)

Let $s = f(x_i, W)$ be scores

Then the SVM loss has the form:

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

Q3: If all the scores were random, what loss would we expect?

Cross-Entropy vs SVM Loss

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

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

assume scores:

[10, -2, 3]

[10, 9, 9]

[10, -100, -100]

and $\boxed{y_i = 0}$

Q: What is cross-entropy loss?
What is SVM loss?

Cross-Entropy vs SVM Loss

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

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

assume scores:

[10, -2, 3]

[10, 9, 9]

[10, -100, -100]

and $\boxed{y_i = 0}$

Q: What is cross-entropy loss?
What is SVM loss?

A: Cross-entropy loss > 0
SVM loss = 0

Cross-Entropy vs SVM Loss

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

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

assume scores:

[10, -2, 3]

[10, 9, 9]

[10, -100, -100]

and $\boxed{y_i = 0}$

Q: What happens to each loss if I slightly change the scores of the last datapoint?

Cross-Entropy vs SVM Loss

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

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

assume scores:
[10, -2, 3]
[10, 9, 9]
[10, -100, -100]
and $\boxed{y_i = 0}$

Q: What happens to each loss if I slightly change the scores of the last datapoint?

A: Cross-entropy loss will change;
SVM loss will stay the same
SVM loss will stay the same for 1st and 3rd example
SVM loss will change for the 2nd

Cross-Entropy vs SVM Loss

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

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

assume scores:
[10, -2, 3]
[10, 9, 9]
[10, -100, -100]
and $\boxed{y_i = 0}$

Q: What happens to each loss if I double the score of the correct class from 10 to 20?

Cross-Entropy vs SVM Loss

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

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

assume scores:

[10, -2, 3]

[10, 9, 9]

[10, -100, -100]

and $y_i = 0$

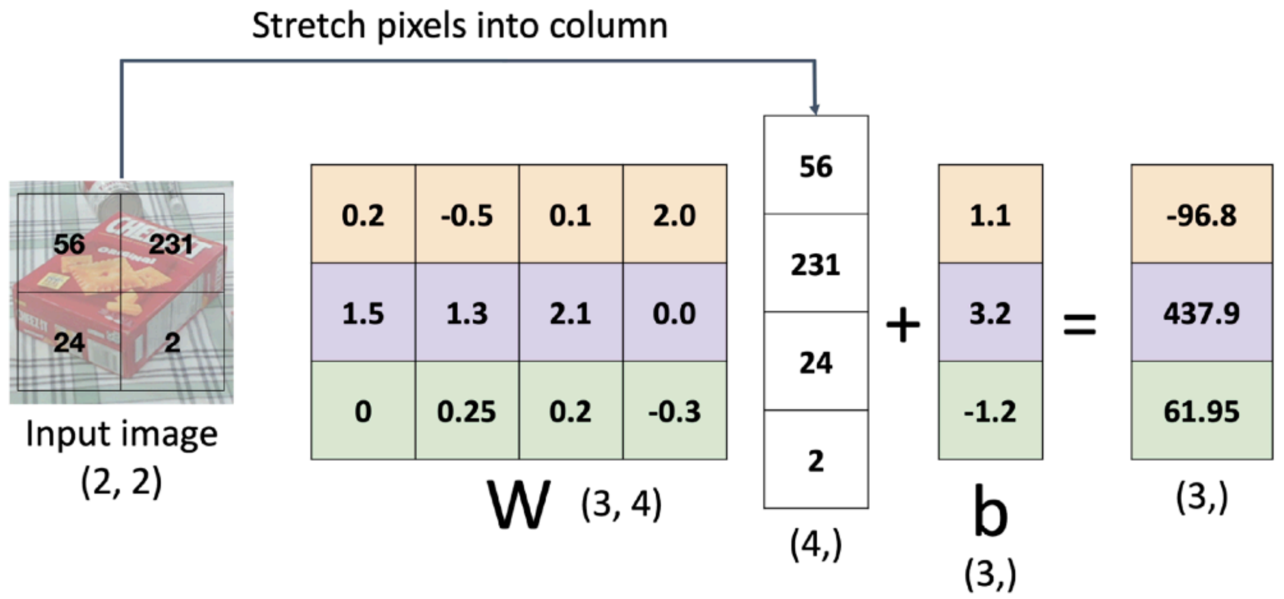
Q: What happens to each loss if I double the score of the correct class from 10 to 20?

A: Cross-entropy loss will decrease, SVM loss still 0

Recap—Three Ways to Interpret Linear Classifiers

Algebraic Viewpoint

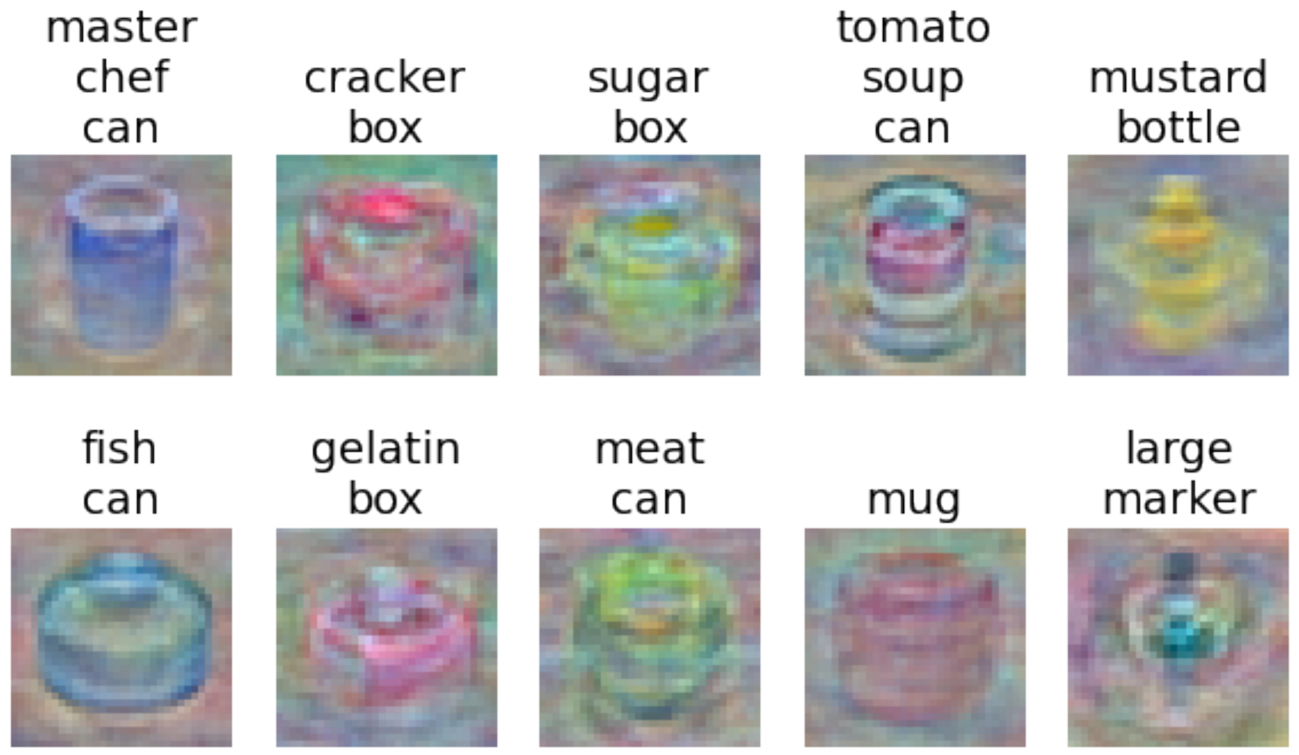
$$f(x,W) = Wx$$



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

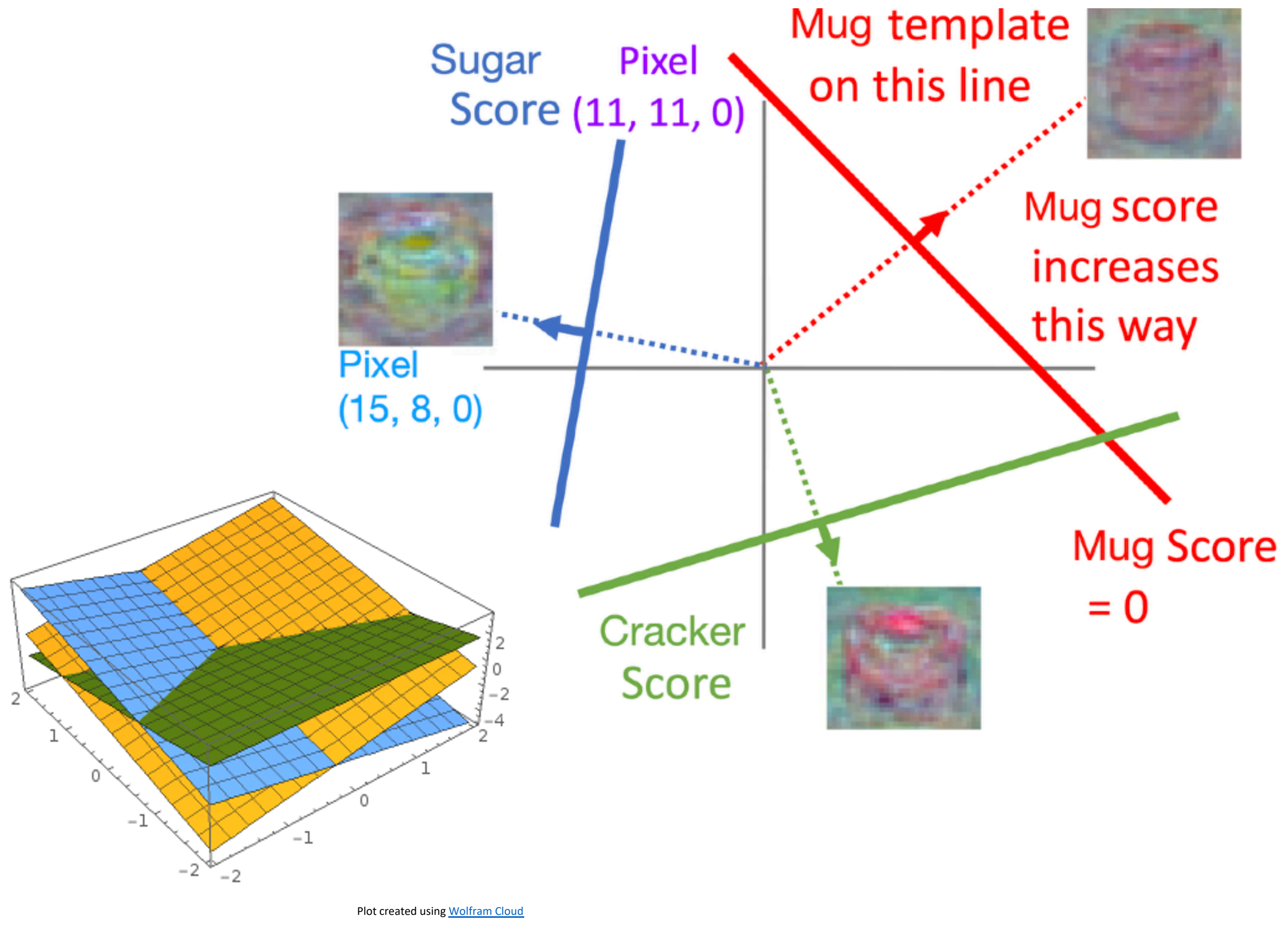
Visual Viewpoint

One template
per class



Geometric Viewpoint

Hyperplanes
cutting up space



Sugar Score
Pixel (11, 11, 0)
Mug template on this line
Mug score increases this way
Pixel (15, 8, 0)
Mug Score = 0
Cracker Score
Plot created using Wolfram Cloud

Recap—Loss Functions Quantify Preferences

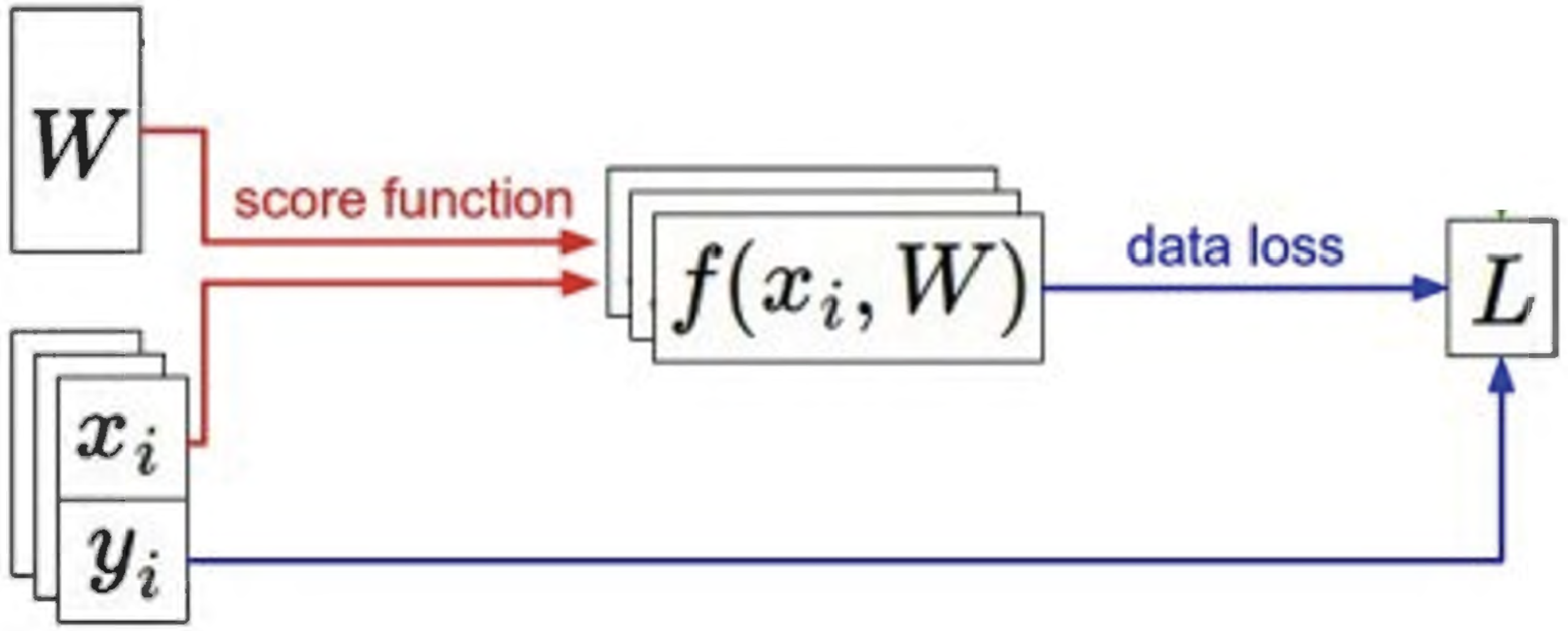
- We have some dataset of (x, y)
- We have a **score function:**
- We have a **loss function**:

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

Linear classifier

Softmax: $L_i = -\log\left(\frac{\exp(s_{y_i})}{\sum_j \exp(s_j)}\right)$

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



Recap—Loss Functions Quantify Preferences

- We have some dataset of (x, y)
- We have a **score function:**
- We have a **loss function**:

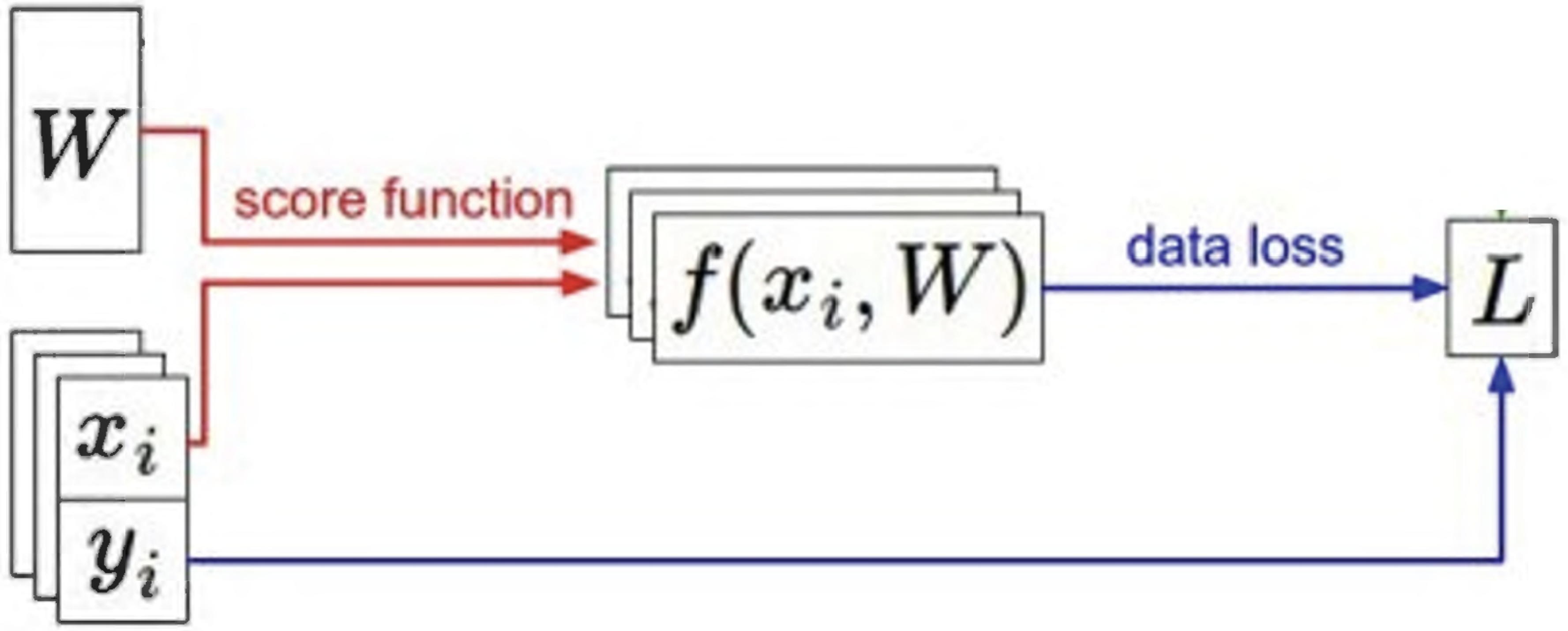
Q: How do we find the best W,b?

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

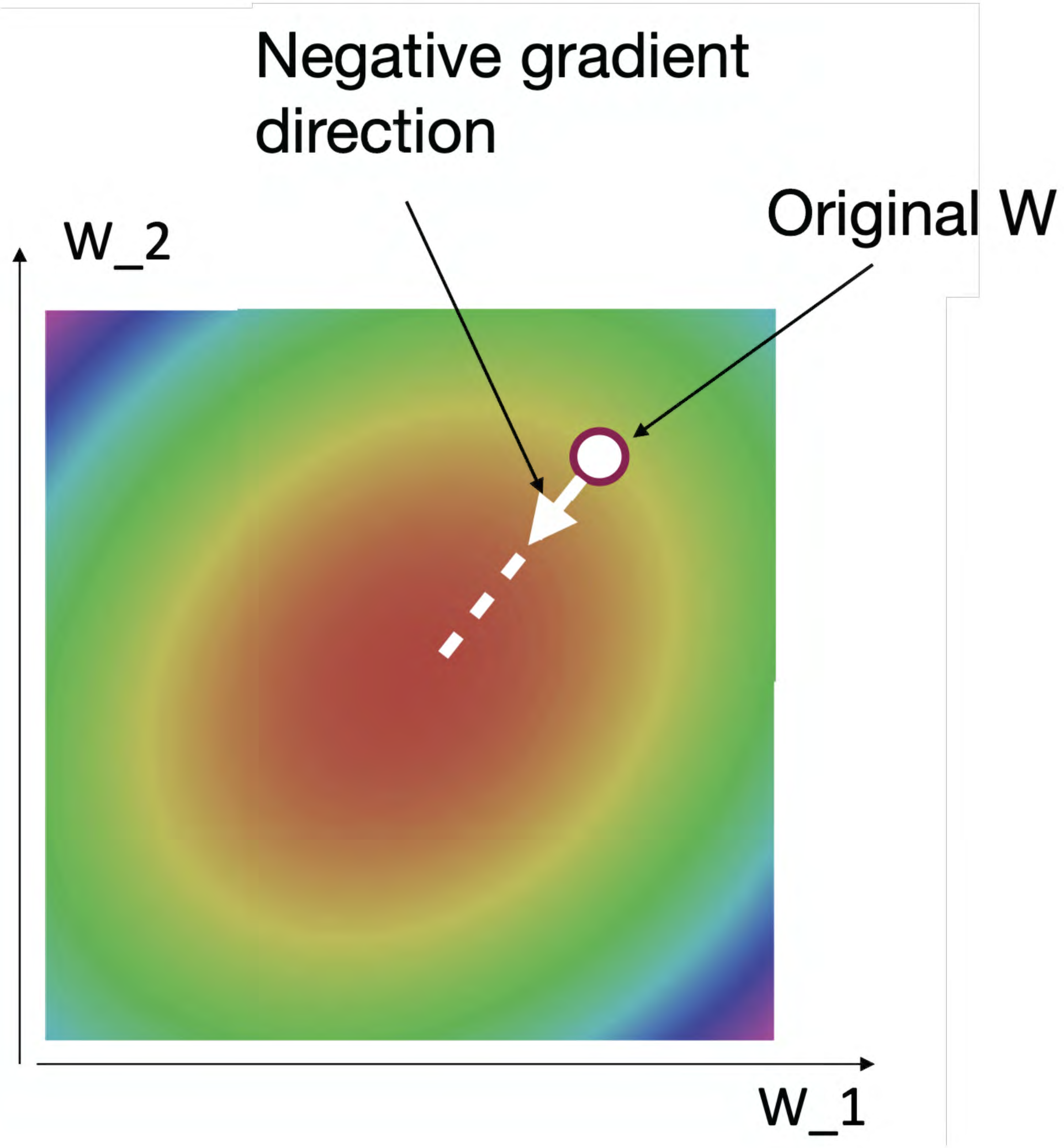
Linear classifier

Softmax: $L_i = -\log\left(\frac{\exp(s_{y_i})}{\sum_j \exp(s_j)}\right)$

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



Next time: Regularization + Optimization



Task brainstorming!

Robotic arms can assist people without arms

Robot Library Book Sorter

Robot ironing shirts

Robot that can fold clothes

Robot that be a steward

Robot Underwater Coral Reef Restoration

Robot for manipulating components on a spacecraft

Robotic toothbrush

Robot that can be a service dog

Robotic coffee maker

Robot that can tie someone's shoes

Robot in an Airplane

Robot that cooks spaghetti

Robot that retrieves basketballs

Robotic arm massages for people

Robot organizing a fridge

Robot that can administer first aid and CPR

Robot for helping paraplegia patients move

Robot that plays chess

Adaptive Puzzle Assembly Assistant

Robot to do laundry and fold my clothes

Robot Rinsing dishes and arranging in dishwasher

Robot to change a baby's diapers

Robot Recycling Electronic Waste

Robot that changes car oil

Grocery Shopper and stock refilling robot

Robot charging all the electronic devices in the home:

Robot for watering plants

Robot Chef Assistant

Robot Cutting Vegetables

Robot can assemble a smartphone with dexterous hands

Robot for feeding or grooming the pet

Monitoring a power loom

Robot Setting up Surgical Instruments in Operating Rooms

Robot calibrating piano

Micro-Soldering Precision Robot

Robotic Violinist

Robot for Disaster Response and Recovery

Robot Performing Minimally Invasive Surgery

Robot Assisting in Plant Harvesting

Robot that can wrap gifts

Robot Syringe Administrator

LEGO Sorting and Storage Automation System

Robot cooking dumplings

Robot-Assisted Bed Making

Robot Morning Assistant

Robot dispensing medication

Bedside Book Reading Assistant Robot

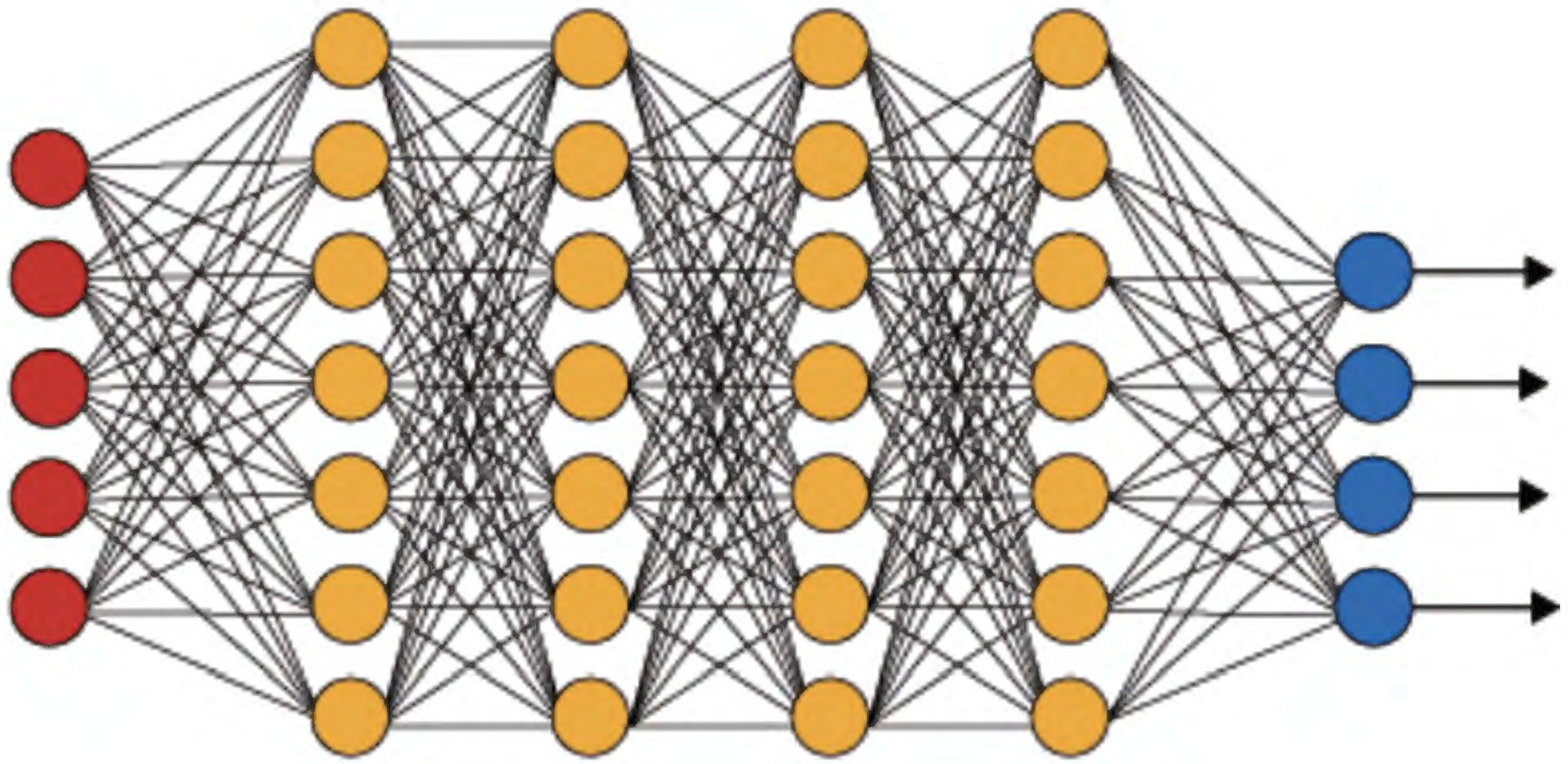
Domestic companion robot to play Table Tennis(TT)

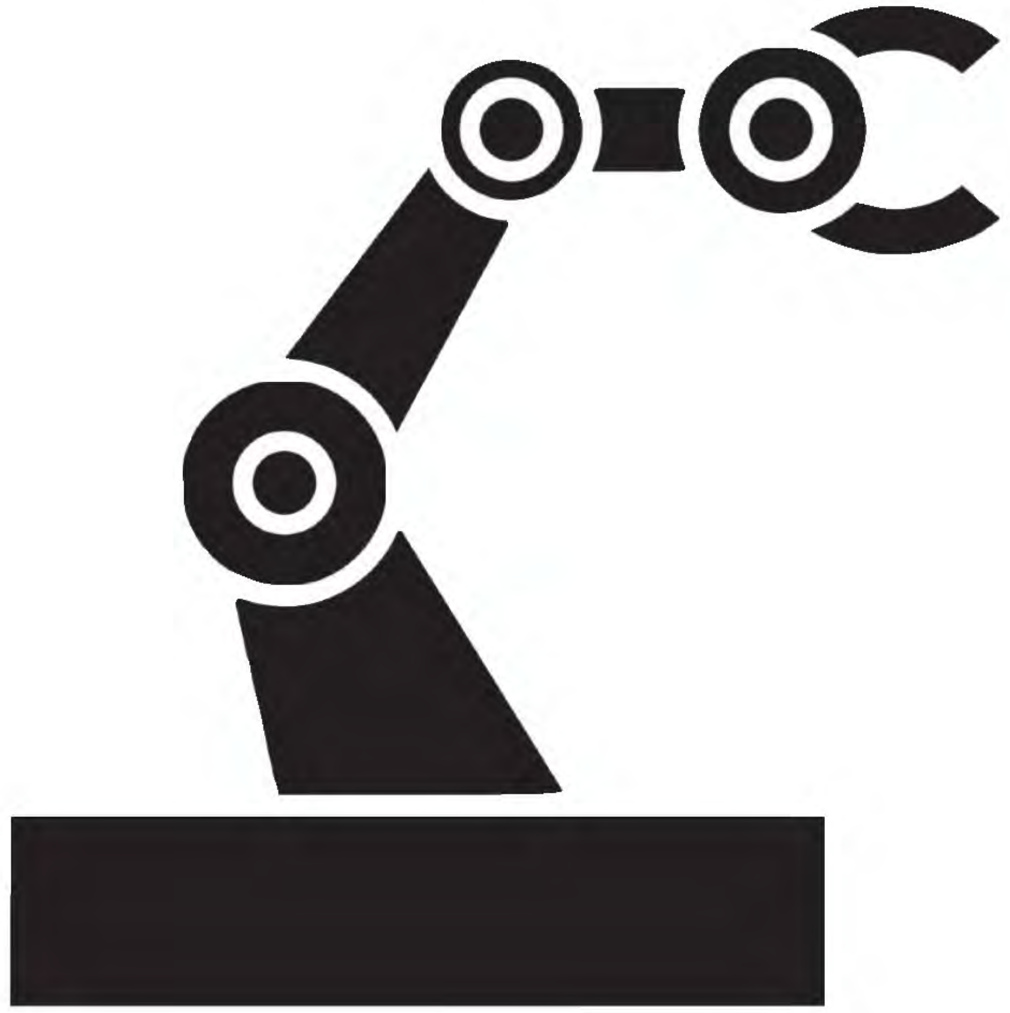
Robot calibrating piano

Robot for Multi-Surface Cleaning

Task brainstorming!

Deep Learning X Robot Manipulation





Next brainstorming exercise:

How will you collect data? What is the input to your DL? What is the output of your DL? …

DeepRob

Lecture 3
Linear Classifiers
University of Minnesota

Slides adapted from Justin Johnson's Deep Learning for Computer Vision course