Introduction to Machine Learning

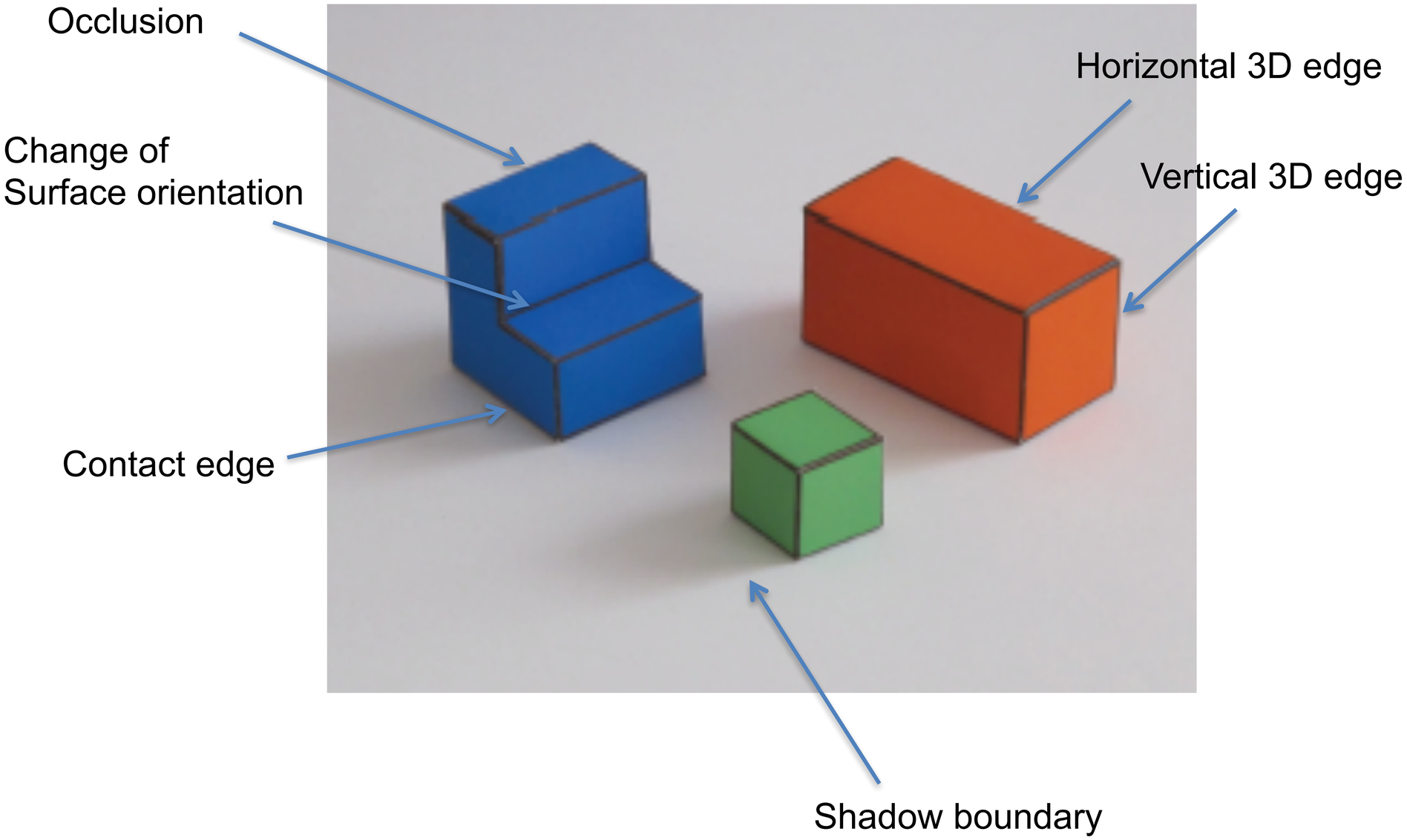
Bill Freeman, Antonio Torralba, Phillip Isola
6.819 / 6.869

[Some slides modified from Alexei Efros]

Announcements

- Final project proposals due 10/25
  - List your team members (1-3 members if fine, but 2 is highly recommended)
  - Either:
    - State which suggested project you will work on
    - Write your own ~1 page proposal (see guidelines on the website)

Edges


Occlusion
Change of
Surface orientation
Contact edge
Horizontal 3D edge
Vertical 3D edge
Shadow boundary

Violations of simple world assumptions

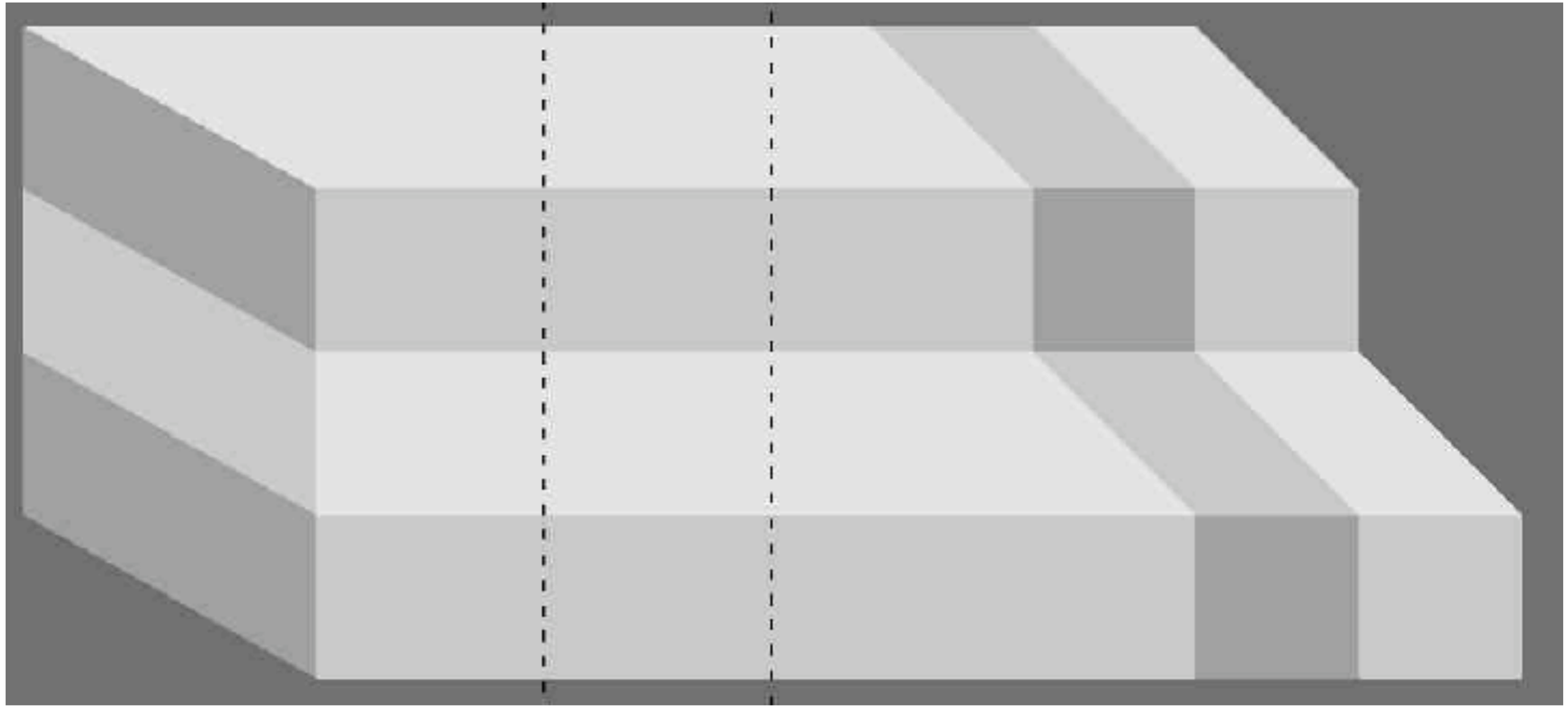
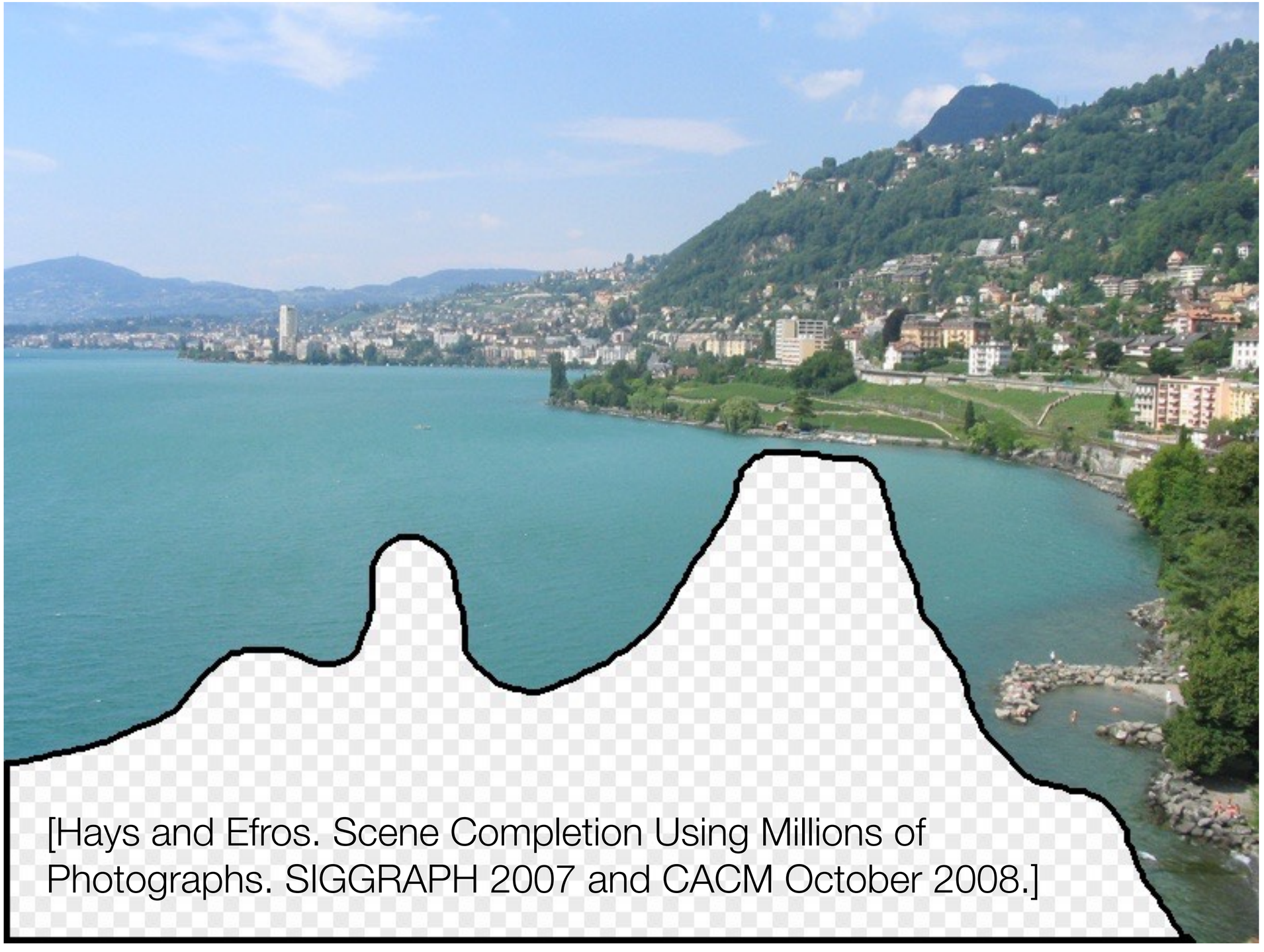


FIGURE 24.9 The impossible steps. On the left, the horizontal stripes appear to be due to paint; on the right, they appear to be due to shading.

Adelson, E.H. Lightness Perception and Lightness Illusions. In *The New Cognitive Neurosciences*, 2nd ed., M. Gazzaniga, ed. Cambridge, MA: MIT Press, pp. 339-351, (2000).

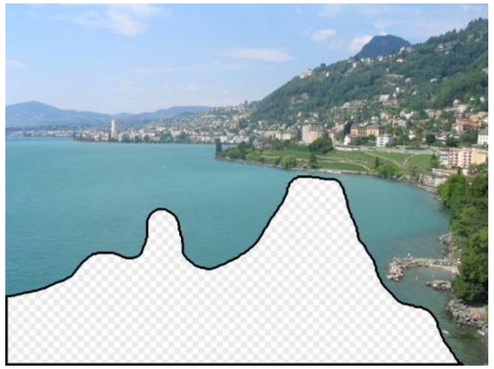
24 Lightness Perception and Lightness Illusions

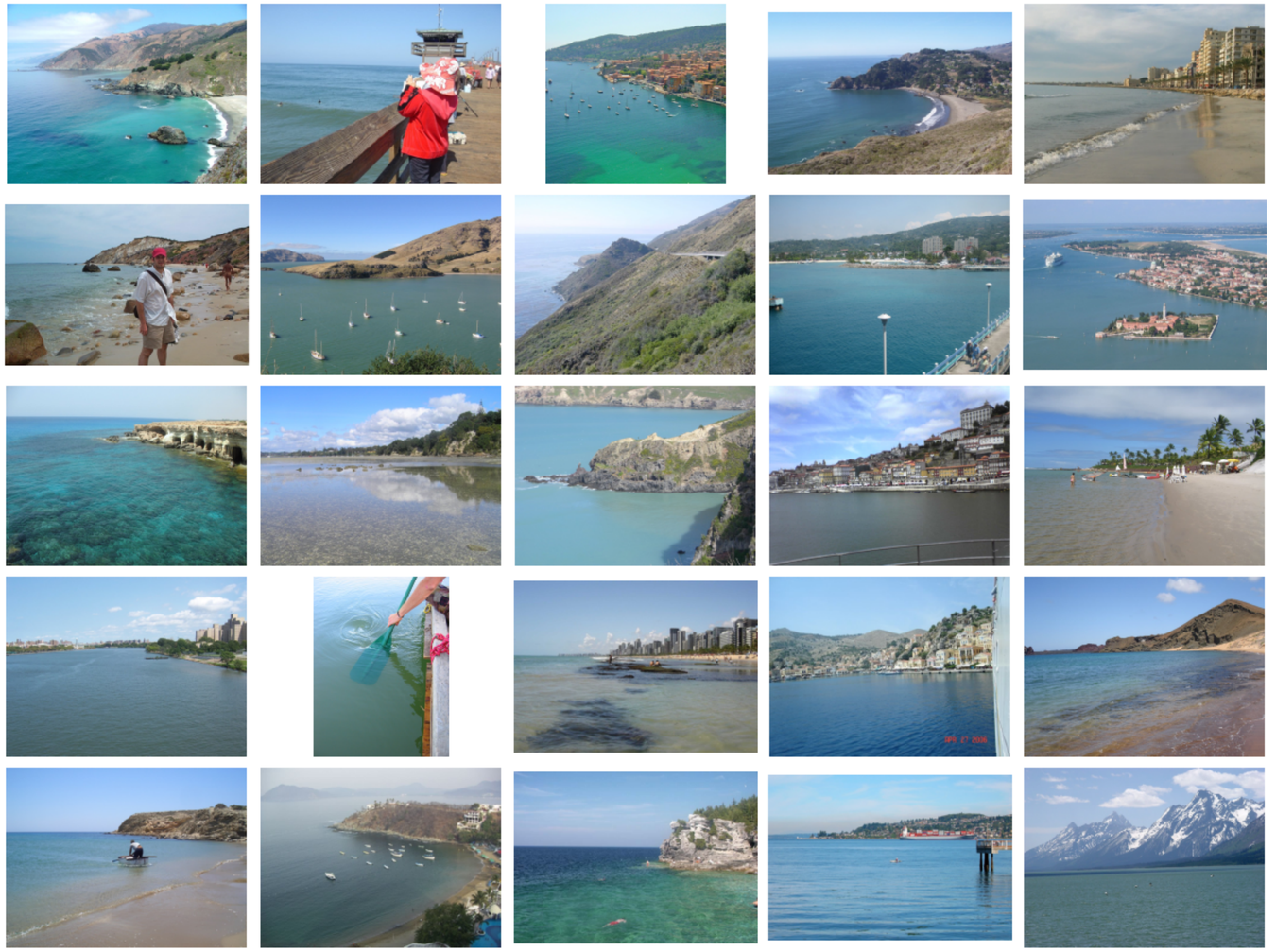
EDWARD H. ADELSON



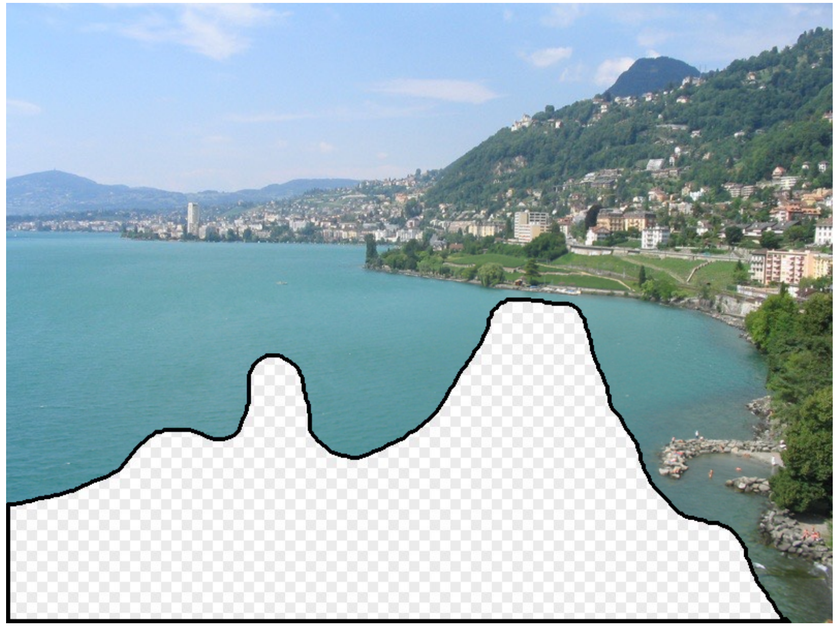
[Hays and Efros. Scene Completion Using Millions of Photographs. SIGGRAPH 2007 and CACM October 2008.]

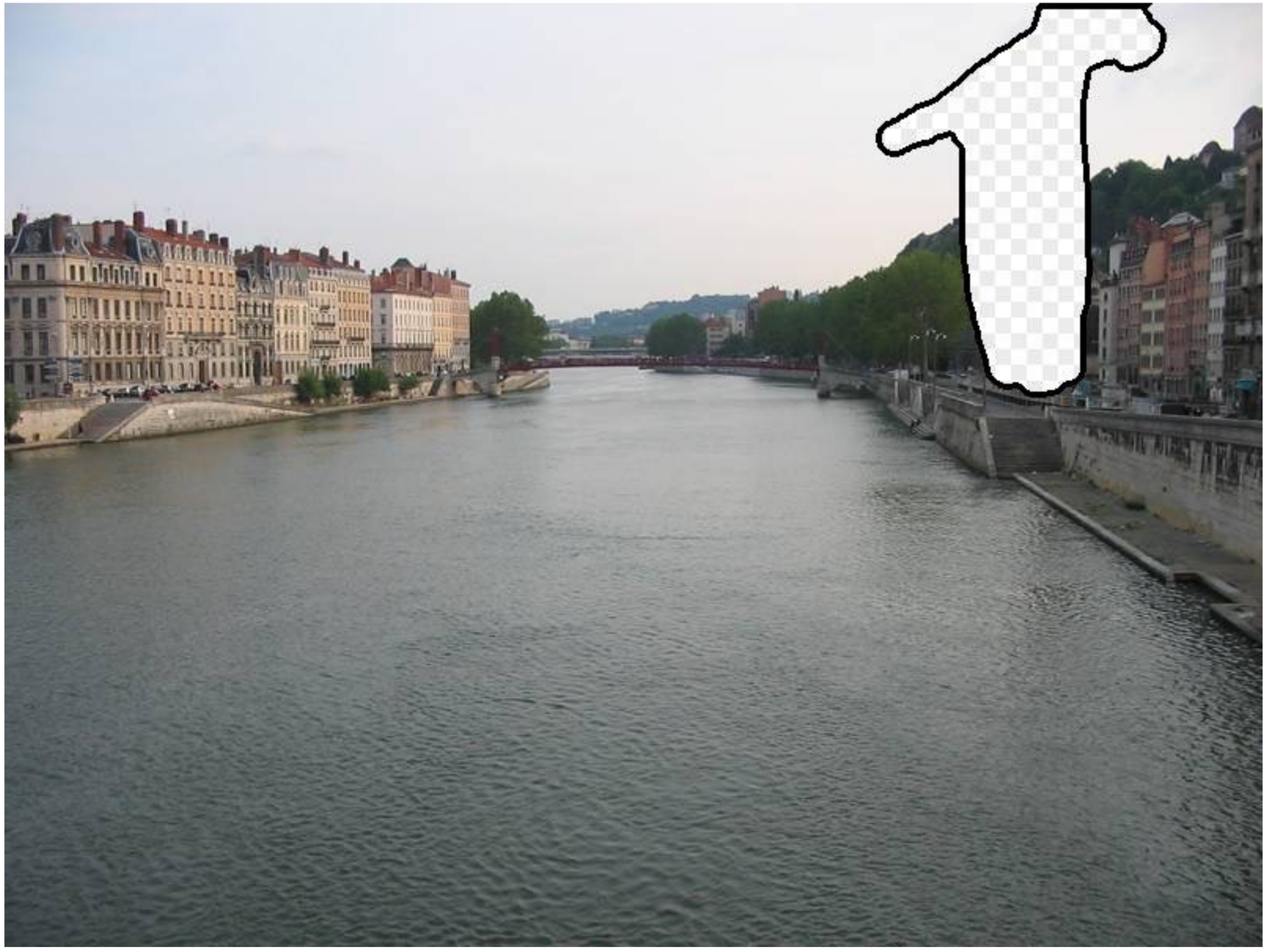

2 Million Flickr Images

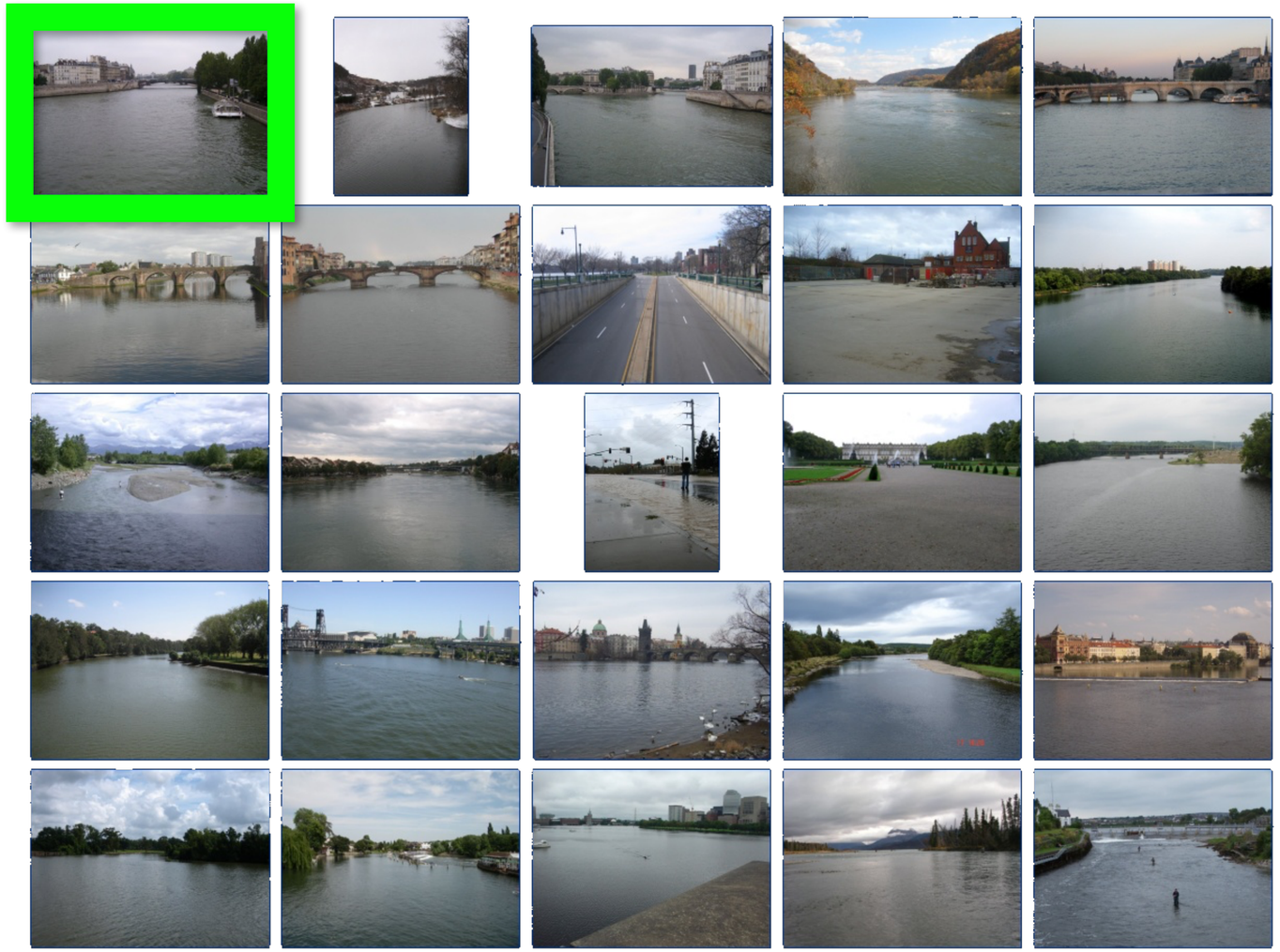




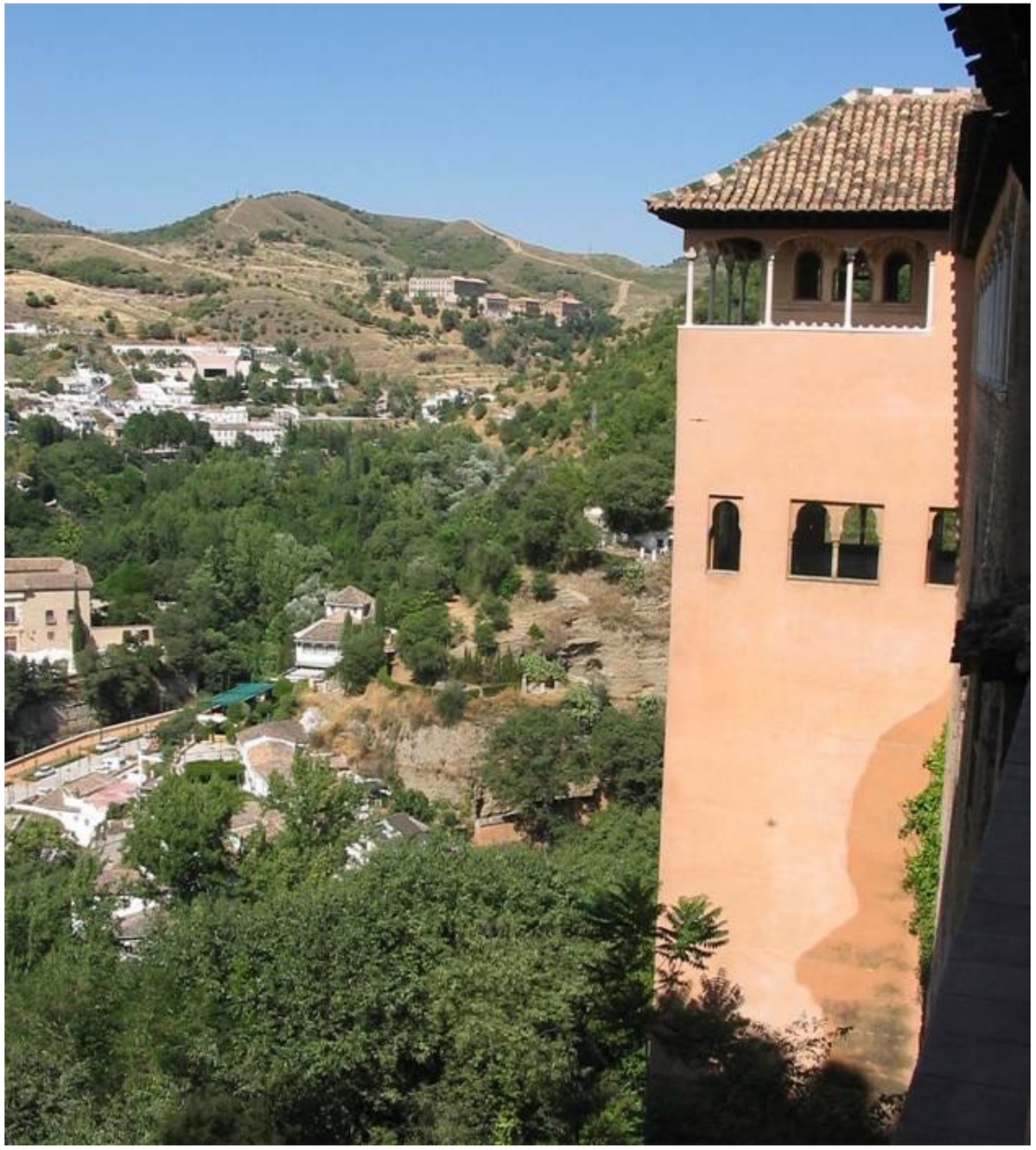
… 200 total

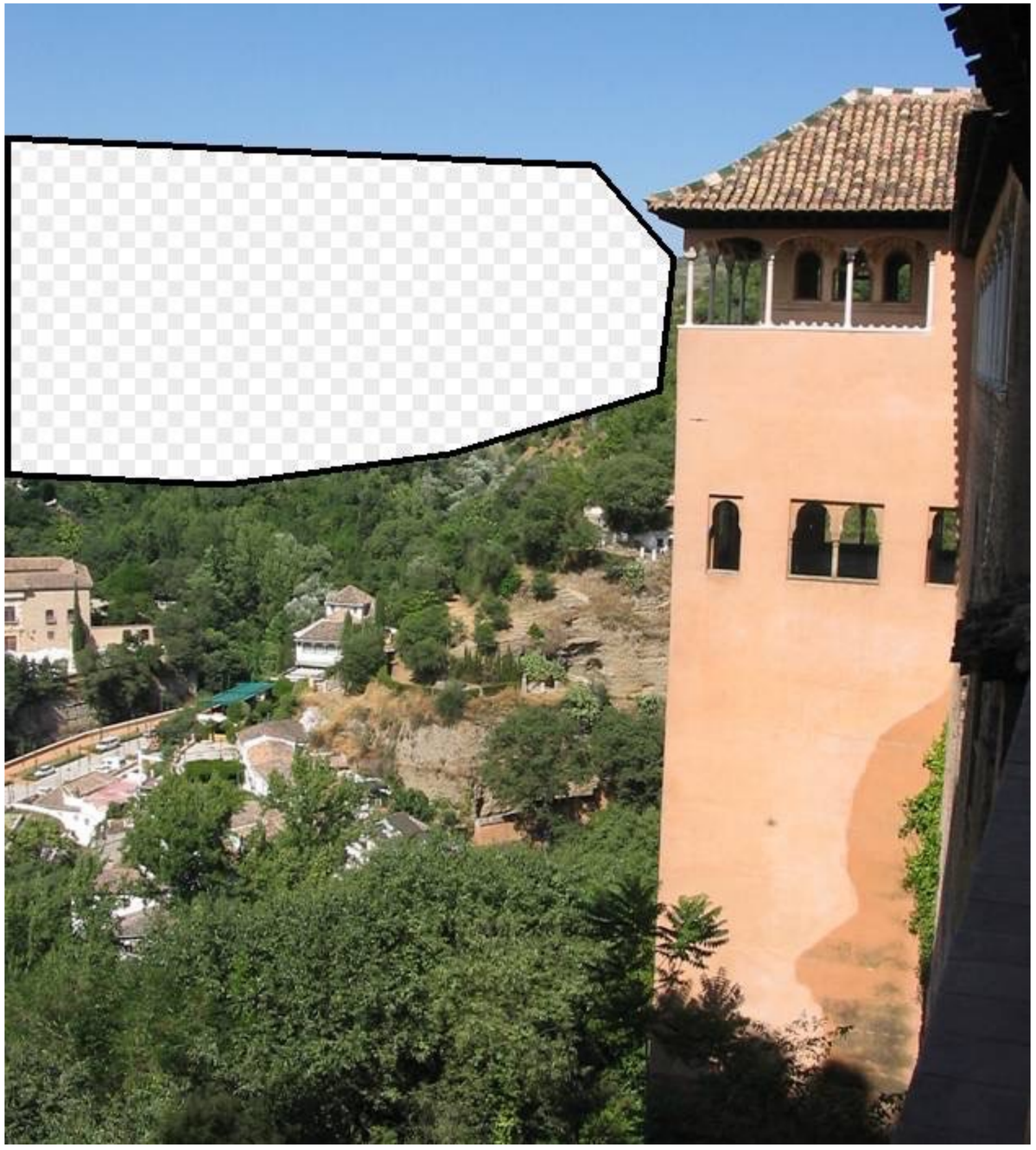




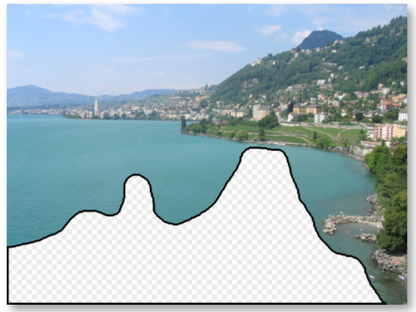


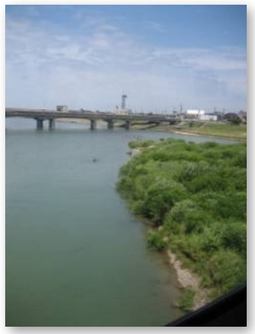
… 200 scene matches





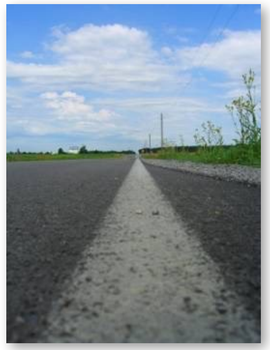
Why does it work?



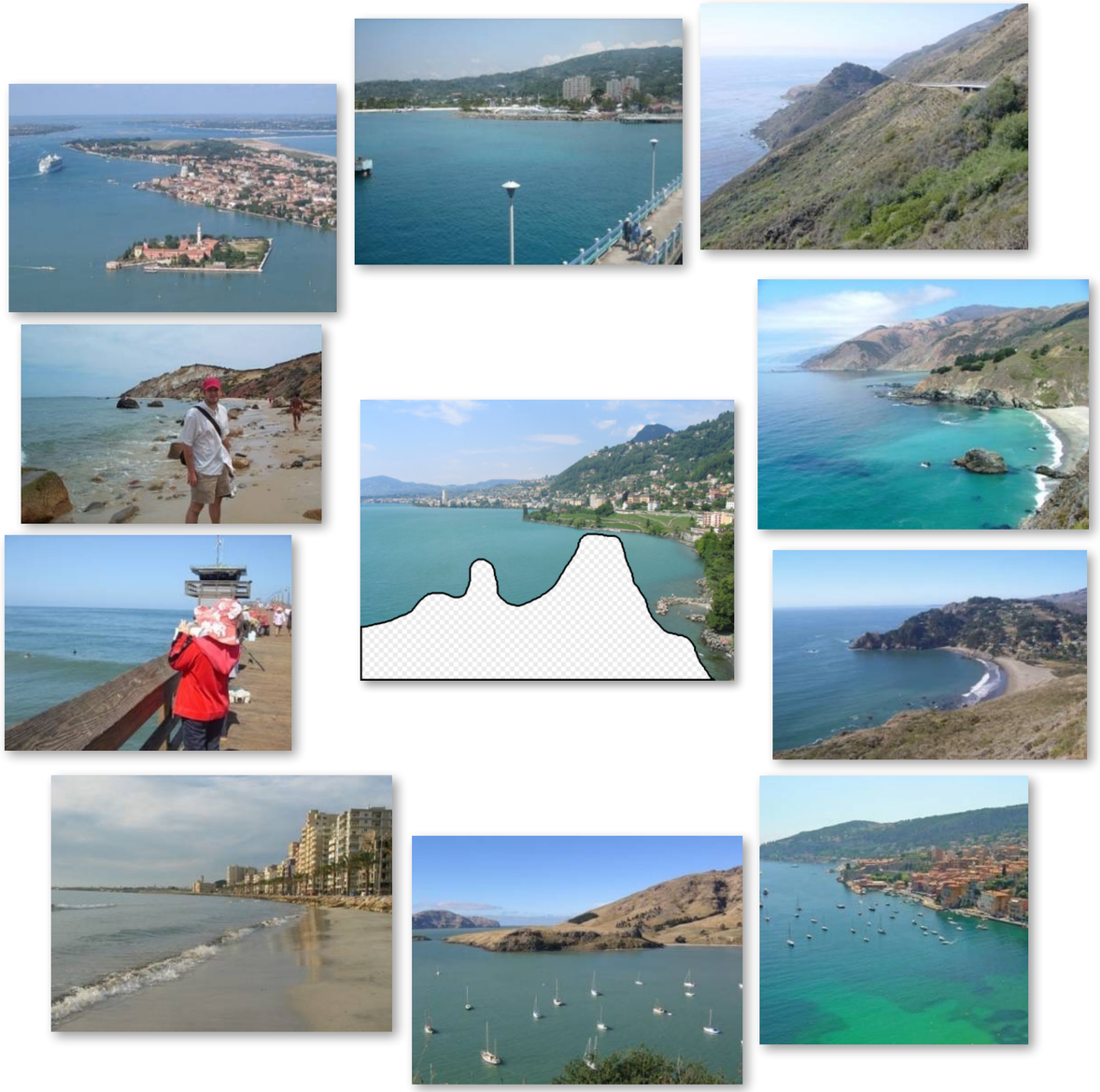







Nearest neighbors from a
collection of 20 thousand images



Nearest neighbors from a
collection of 2 million images

“Unreasonable Effectiveness of Data”

[Halevy, Norvig, Pereira 2009]

Parts of our world can be explained by elegant mathematics

physics, chemistry, astronomy, etc.

But much cannot

psychology, economics, genetics, etc.

Enter The Data!

Great advances in several fields:

e.g., speech recognition, machine translation

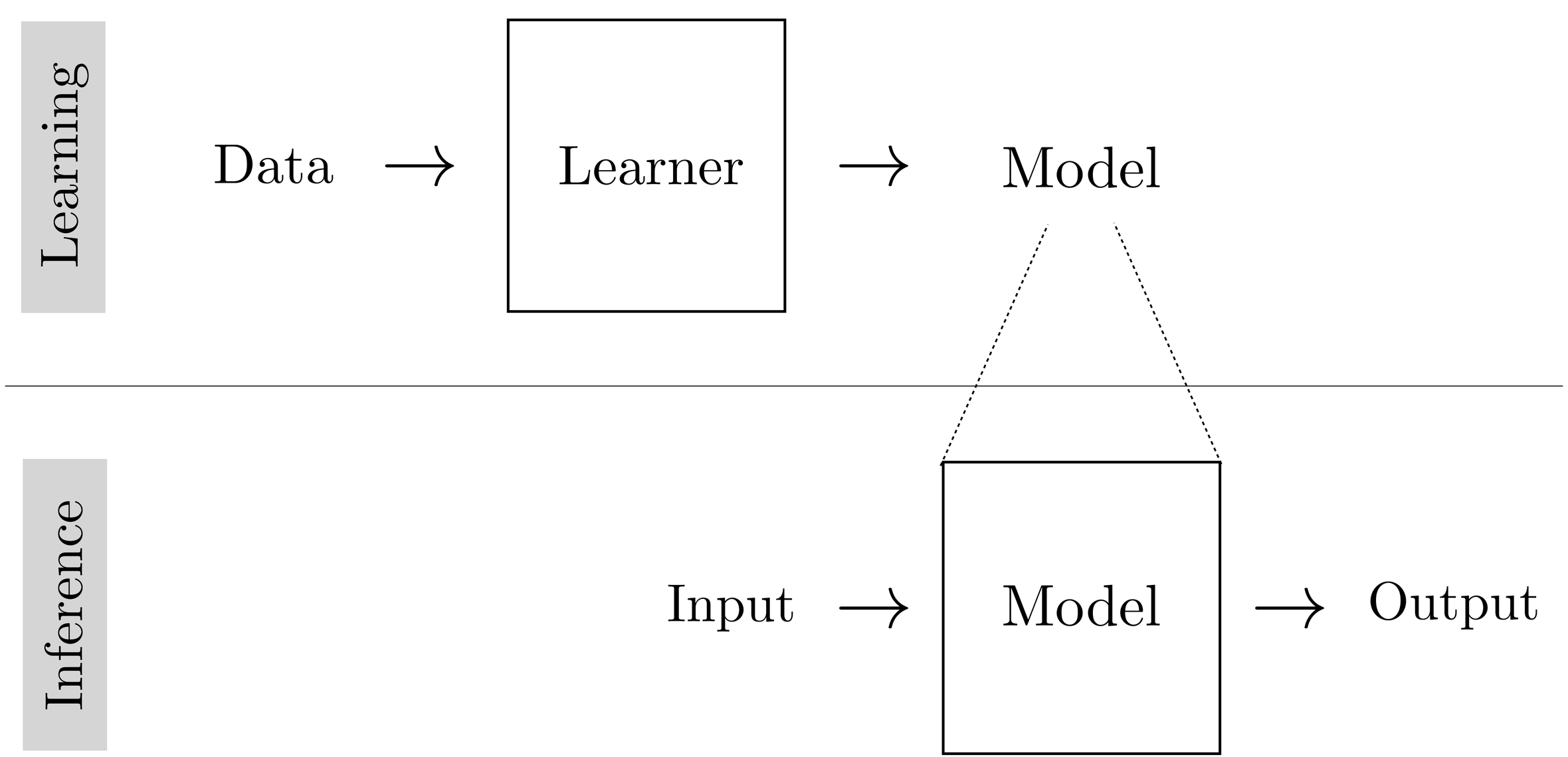
Case study: Google

> “For many tasks, once we have a billion or so examples, we essentially have a closed set that represents (or at least approximates) what we need…”

Learning versus inference

Last lecture: **Inference**
Given a model $P(x_1, x_2, x_3, \ldots)$, make inferences such as $\arg\max_{x_1} P(x_1|x_2, x_3, \ldots)$

This lecture: **Learning**
Given data, automatically come up with the model that best explains it


Learning
Data
Learner
Model
Inference
Input
Model
Output

What does ☆ do?

$$2 \star 3 = 36$$

$$7 \star 1 = 49$$

$$5 \star 2 = 100$$

$$2 \star 2 = 16$$

Training

2 ☆ 3 = 36

7 ☆ 1 = 49

5 ☆ 2 = 100

2 ☆ 2 = 16

→ Your brain → $x \star y \rightarrow (xy)^2$

Testing

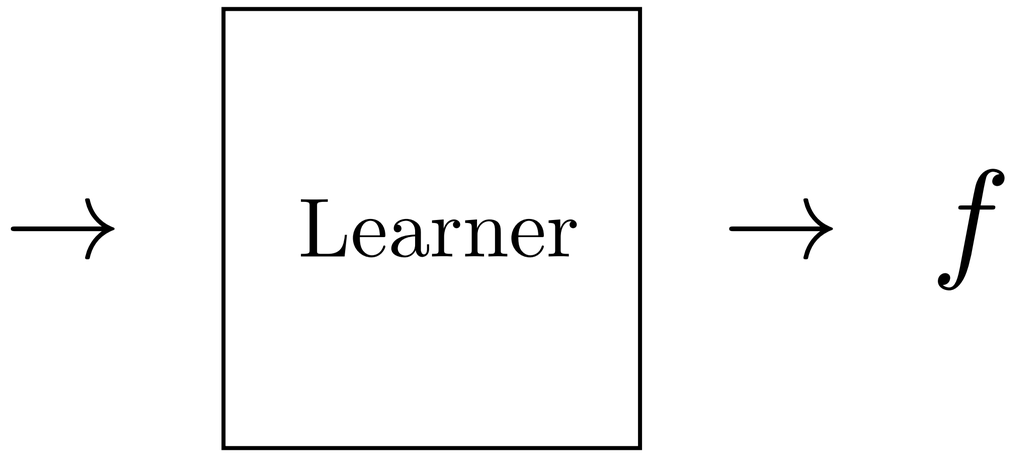
3 ☆ 5 → $x \star y \rightarrow (xy)^2$ → 225

Learning from examples

(aka **supervised learning**)

Training data

$\{\texttt{input}:[2,3],\texttt{output}:36\}$
$\{\texttt{input}:[7,1],\texttt{output}:49\}$
$\{\texttt{input}:[5,2],\texttt{output}:100\}$
$\{\texttt{input}:[2,2],\texttt{output}:16\}$



Learning from examples

(aka **supervised learning**)

Training data

$\{x_1, y_1\}$
$\{x_2, y_2\}$
$\{x_3, y_3\}$
$\ldots$

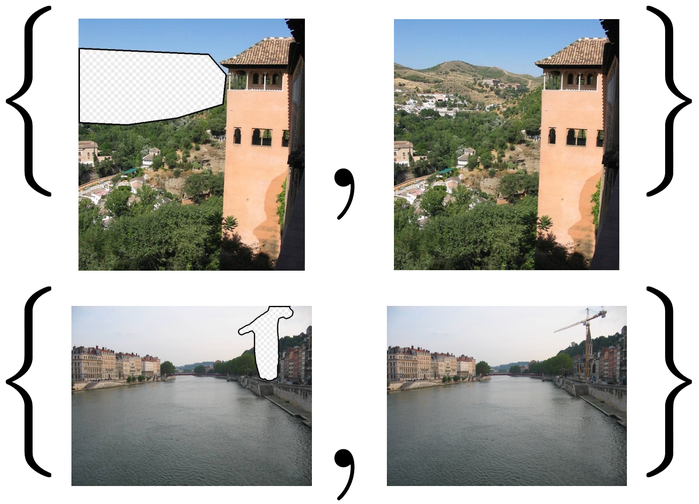
$\rightarrow$ Learner $\rightarrow$ $f : X \rightarrow Y$

Learning from examples

(aka **supervised learning**)

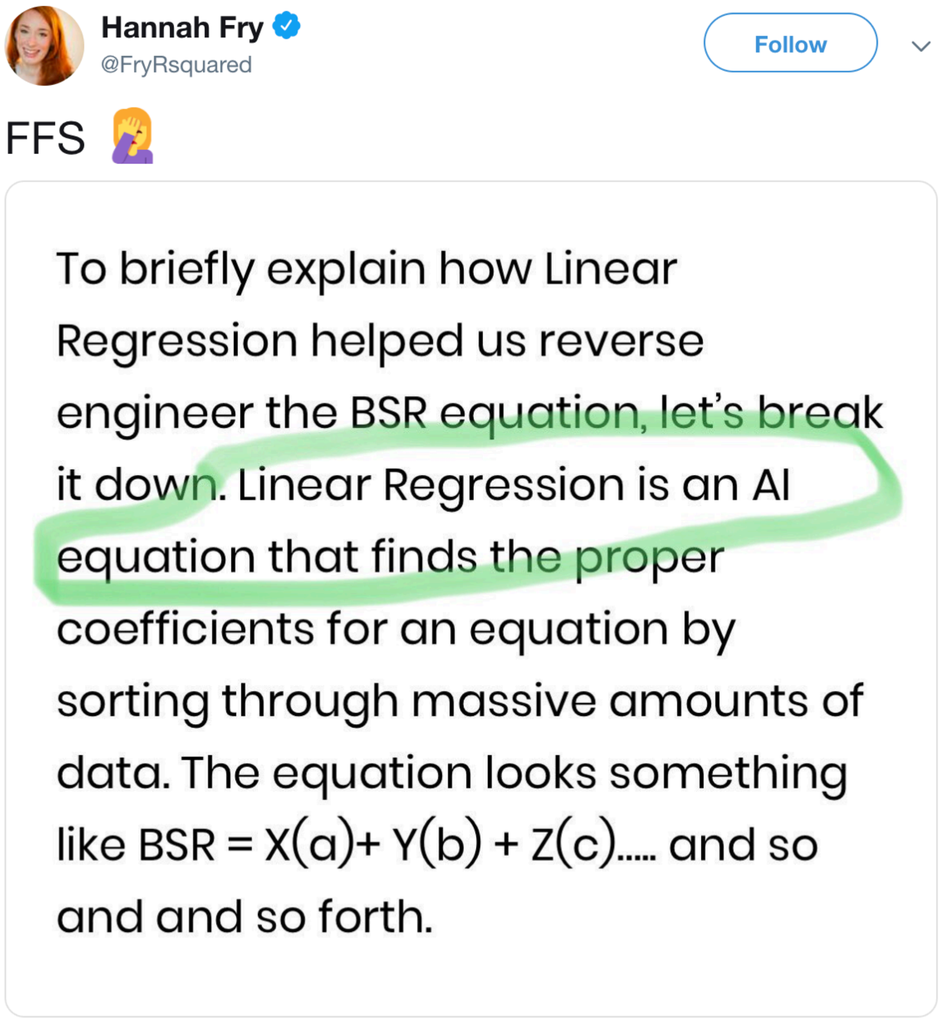
Training data

X Y

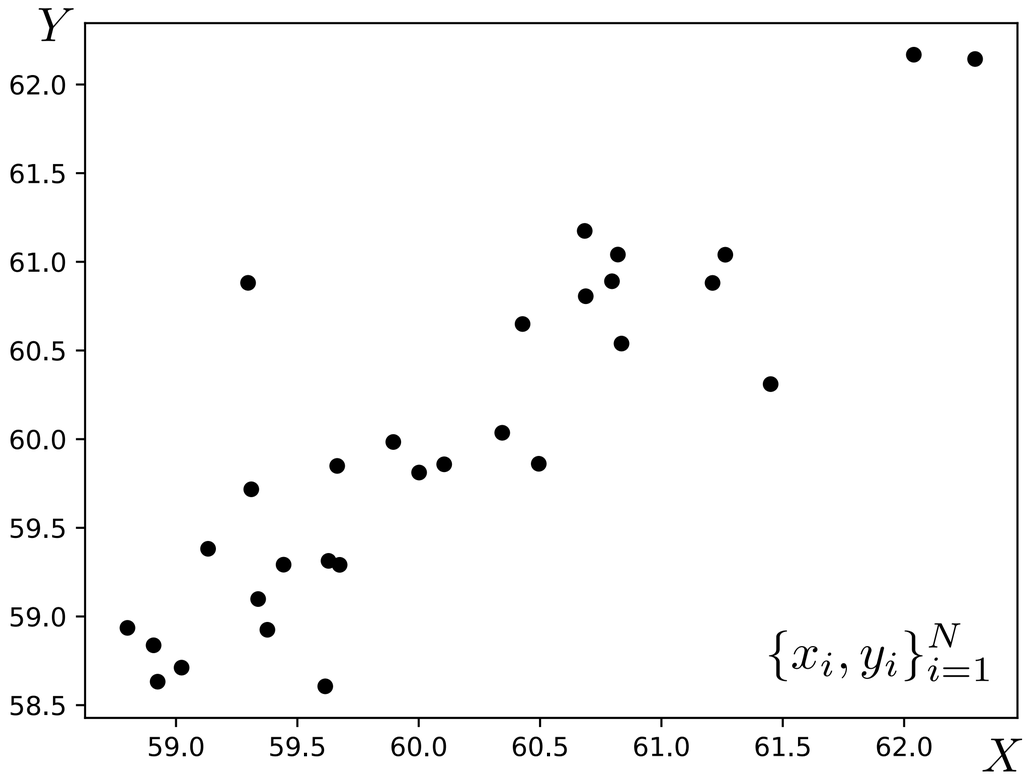


$\rightarrow$ Learner $\rightarrow$ $f : X \rightarrow Y$

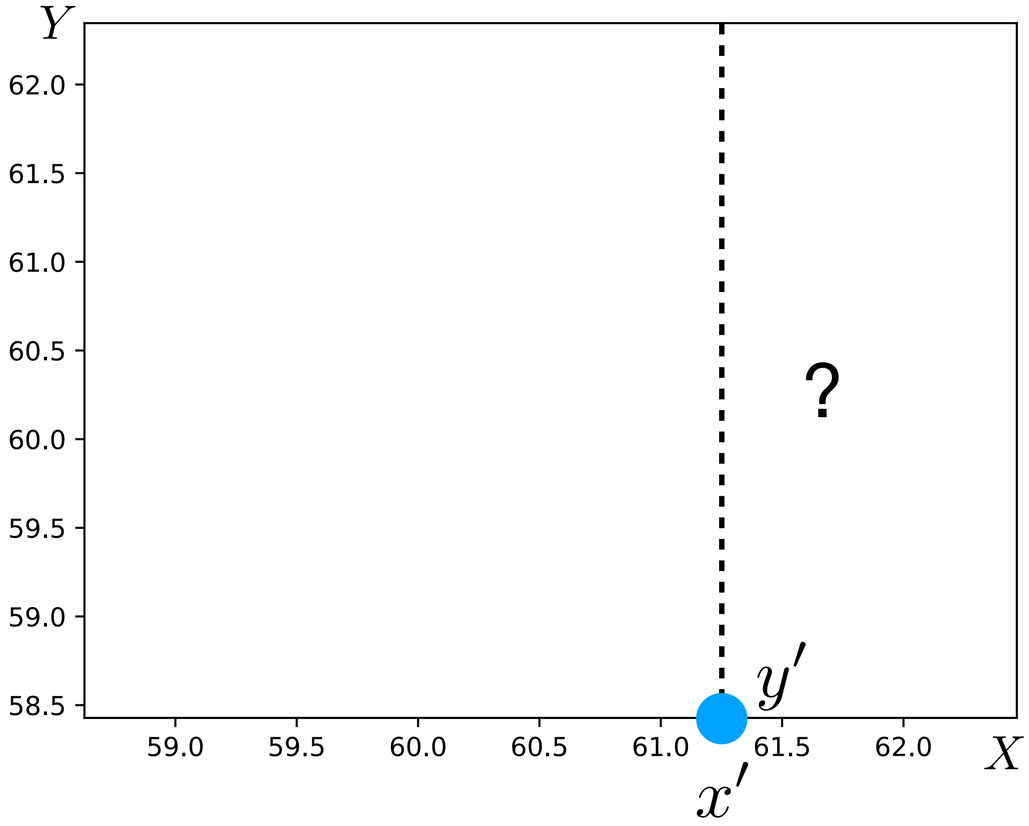
Example 1: Linear least squares



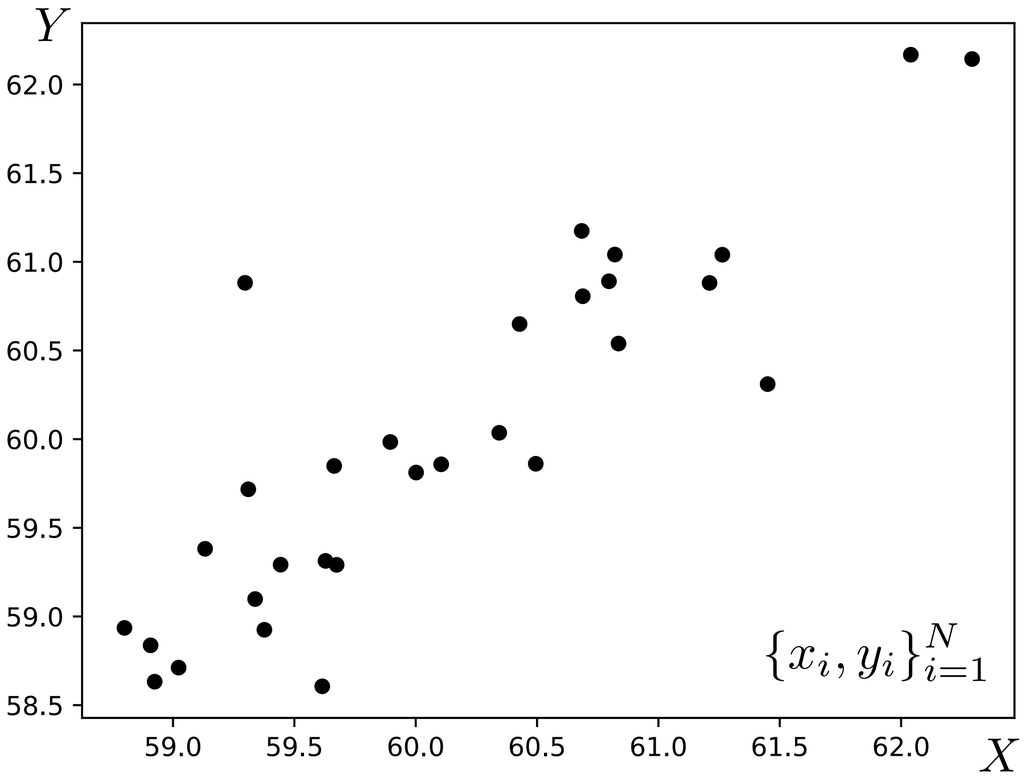
Training data



Test query



Training data

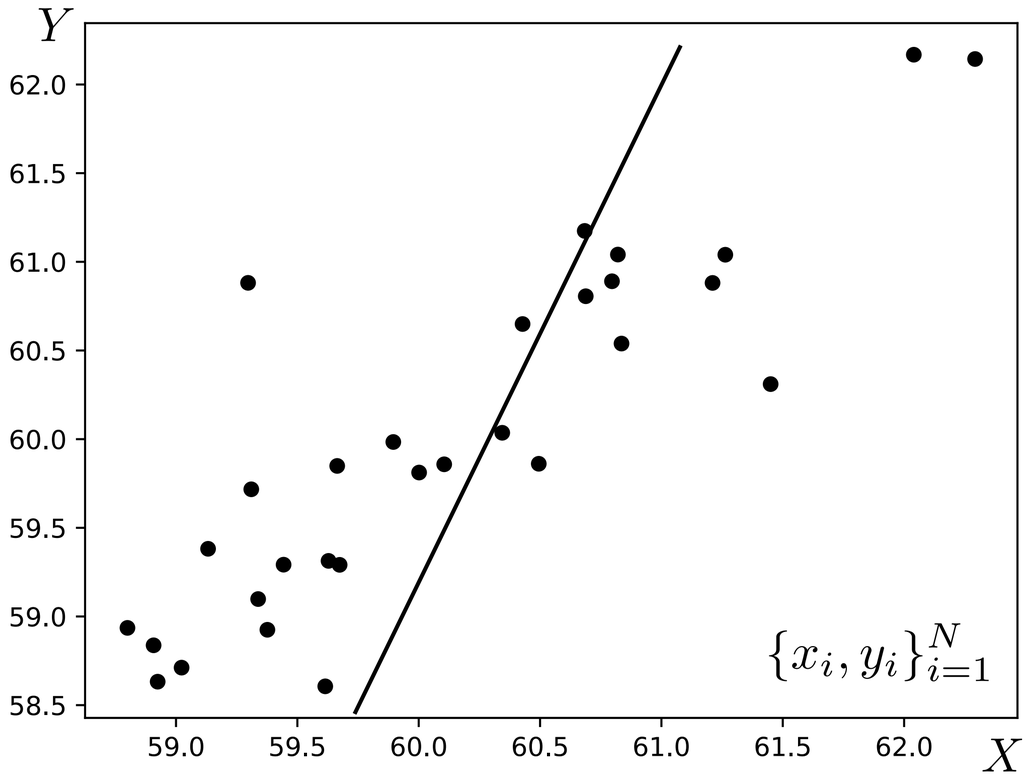


$\longrightarrow$ Learner $\longrightarrow$ $f_\theta(x) = \theta_1 x + \theta_0$

Hypothesis space

The relationship between X and Y is roughly linear: $y \approx \theta_1 x + \theta_0$

Training data

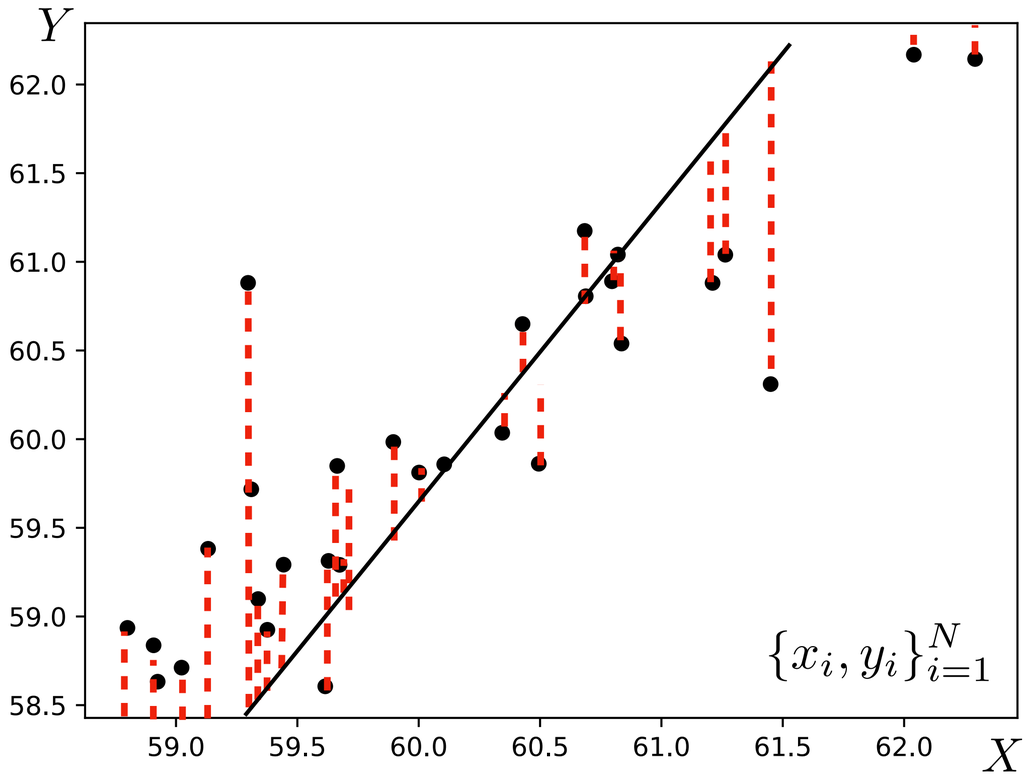


Search for the **parameters**, $\theta = \{\theta_0, \theta_1\}$, that best fit the data.

$$f_\theta(x) = \theta_1 x + \theta_0$$

Best fit in what sense?

Training data



Search for the **parameters**, $\theta = \{\theta_0, \theta_1\}$, that best fit the data.

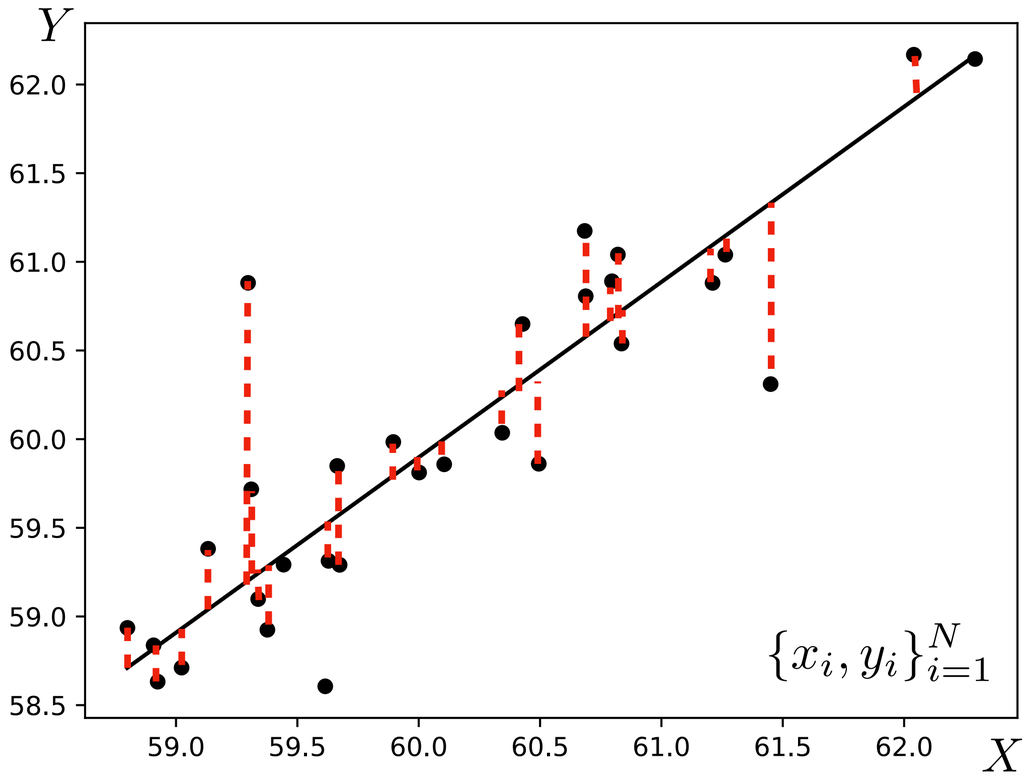
$$f_\theta(x) = \theta_1 x + \theta_0$$

Best fit in what sense?

The least-squares **objective** (aka **loss**) says the best fit is the function that minimizes the squared error between predictions and target values:

$$\mathcal{L}(\hat{y}, y) = (\hat{y} - y)^2 \quad \hat{y} \equiv f_\theta(x)$$

Training data



Search for the **parameters**, $\theta = \{\theta_0, \theta_1\}$, that best fit the data.

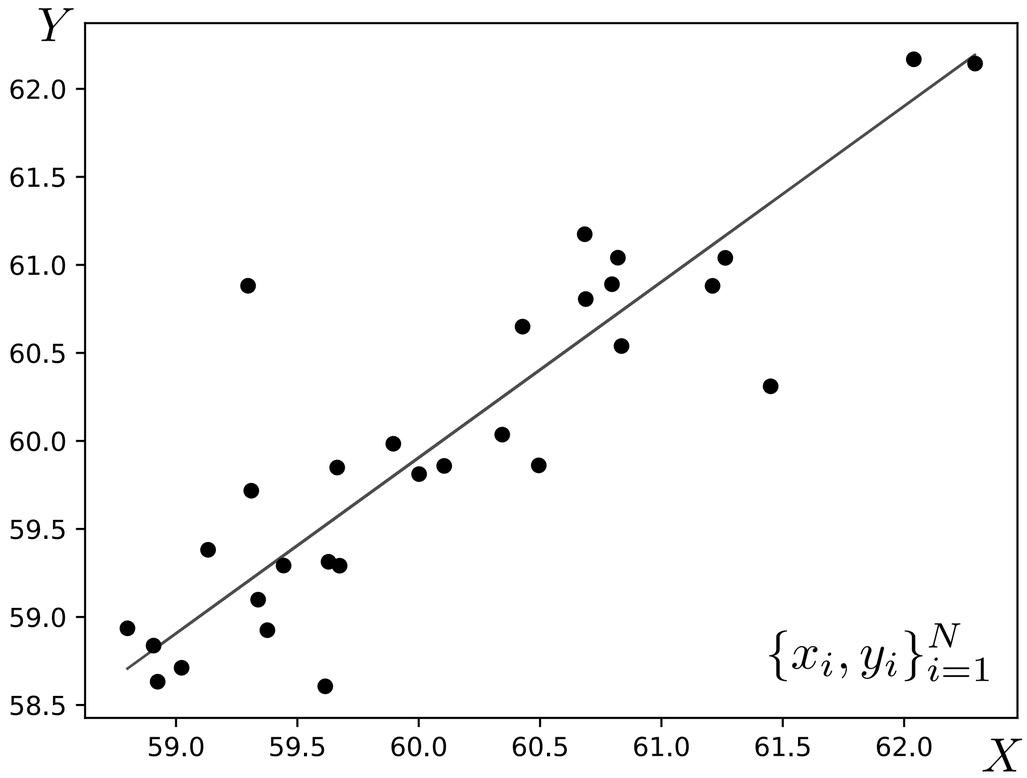
$$f_\theta(x) = \theta_1 x + \theta_0$$

Best fit in what sense?

The least-squares **objective** (aka **loss**) says the best fit is the function that minimizes the squared error between predictions and target values:

$$\mathcal{L}(\hat{y}, y) = (\hat{y} - y)^2 \quad \hat{y} \equiv f_\theta(x)$$

Training data

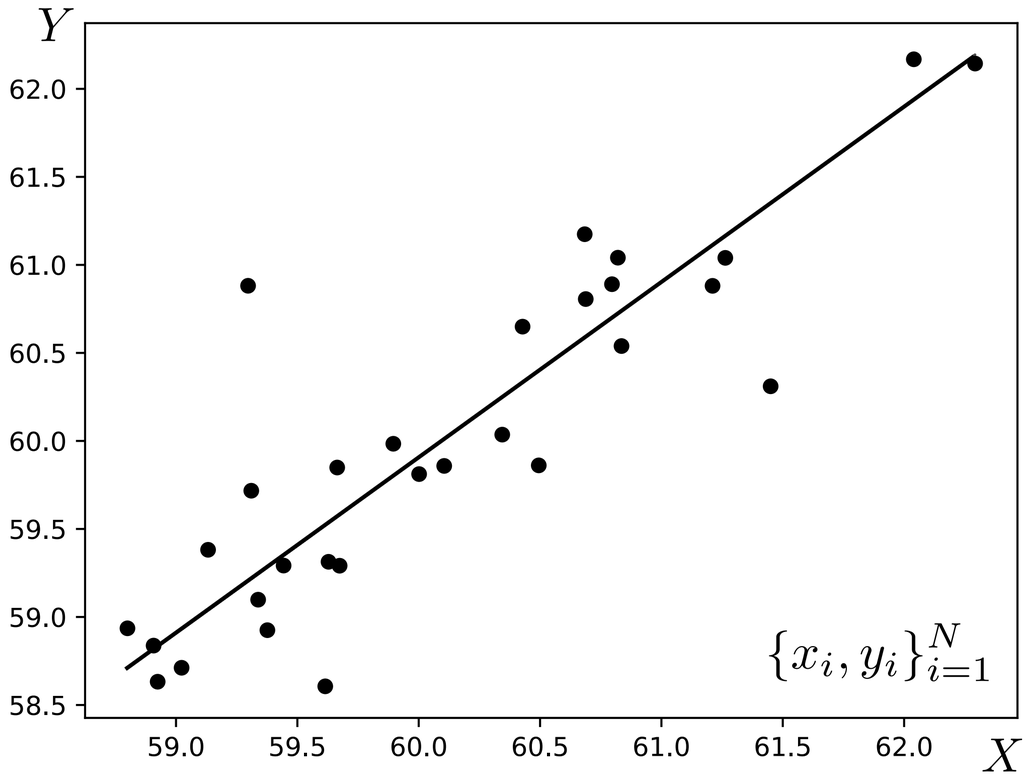


Complete learning problem:

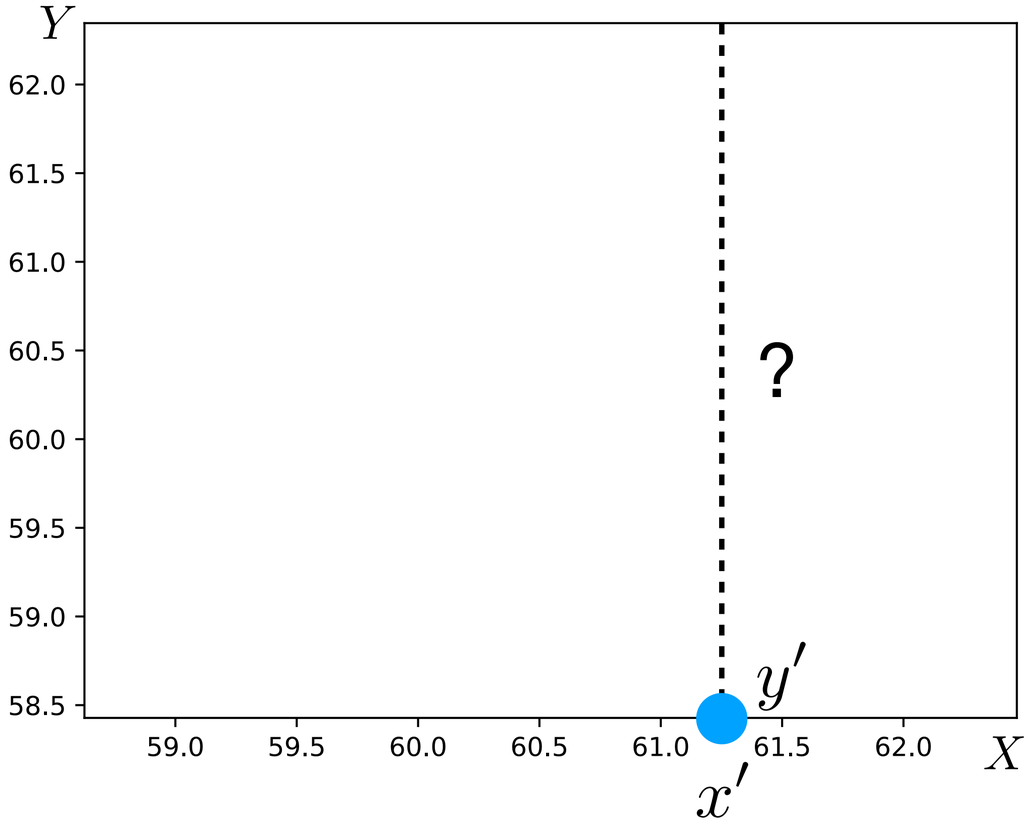
$$\theta^* = \arg\min_\theta \sum_{i=1}^{N} (f_\theta(x_i) - y_i)^2$$

$$= \arg\min_\theta \sum_{i=1}^{N} (\theta_1 x_i + \theta_0 - y_i)^2$$

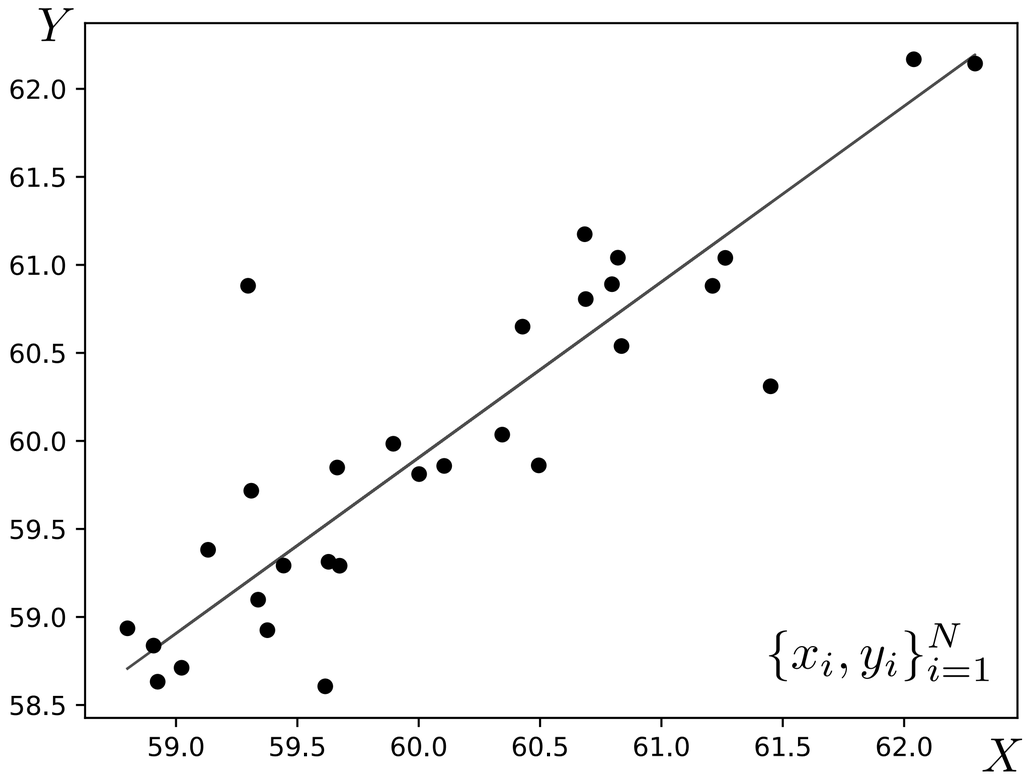
Training data



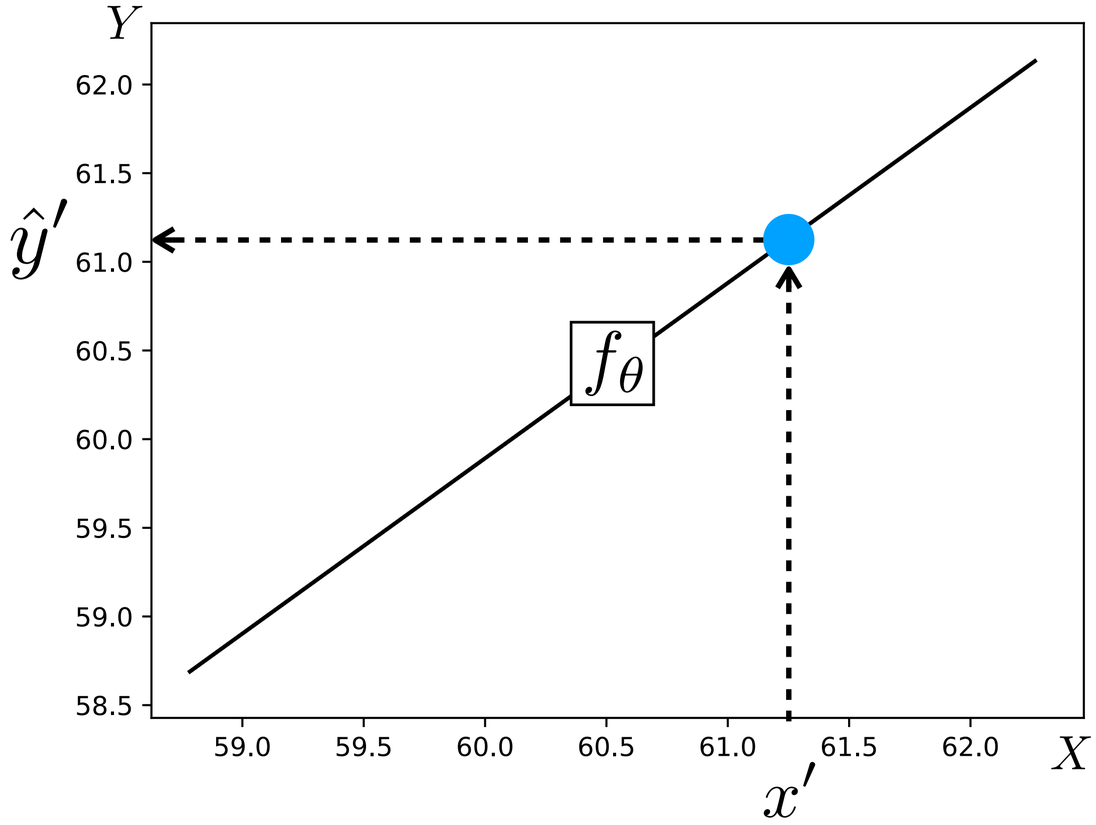
Test query



Training data

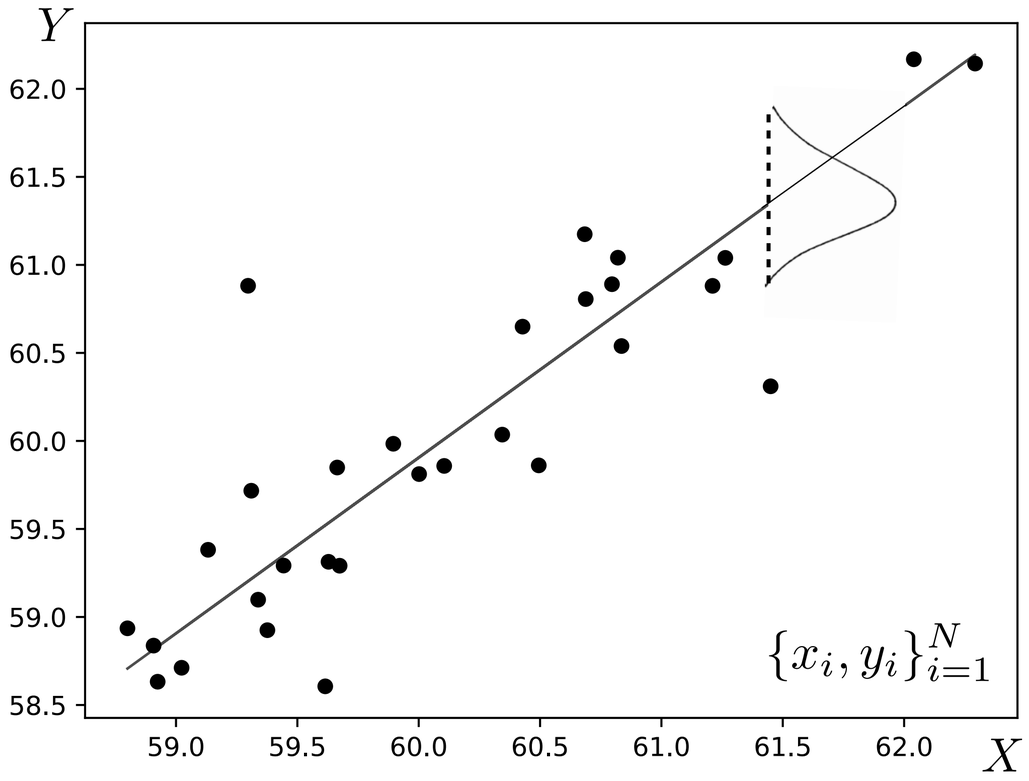


Test query



$$x' \dashrightarrow \boxed{f_\theta} \dashrightarrow \hat{y}'$$

Why is squared error a good objective?



Follows from two *modeling assumptions*

1. Prediction errors follow a normal distribution:

$$Y \approx f_\theta(X)$$

$$\epsilon = Y - f_\theta(X) \qquad \epsilon \sim \mathcal{N}(0, \sigma)$$

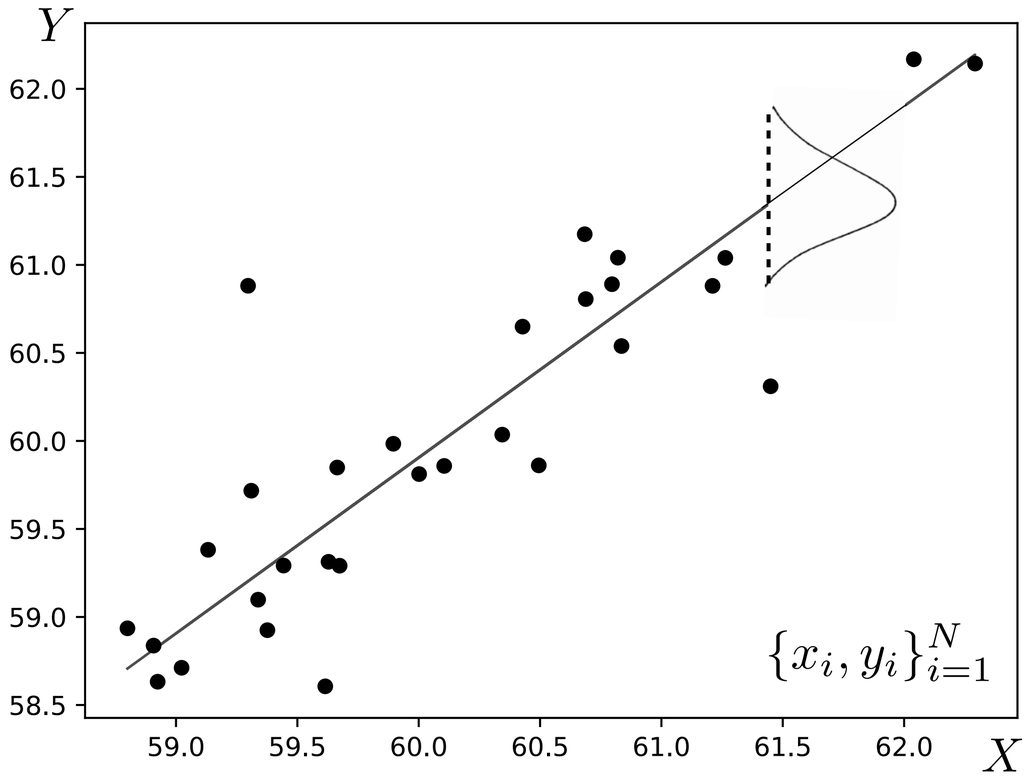
$$Y = f_\theta(X) + \epsilon$$

$$P_\theta(Y = y | X = x) \propto \exp \frac{-(y - f_\theta(x))^2}{2\sigma^2}$$

2. Training datapoints are independently and identically distributed (**iid**):

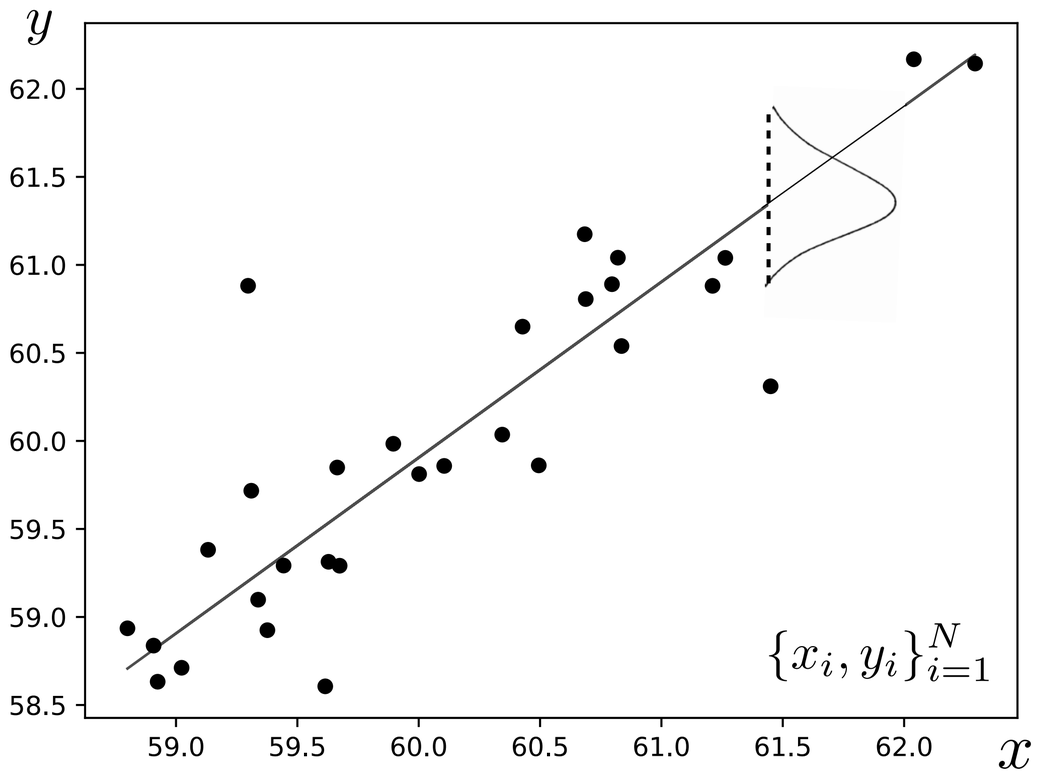
$$P_\theta(\{y_i\}_{i=1}^N | \{x_i\}_{i=1}^N) = \prod_{i=1}^N P_\theta(y_i | x_i)$$

Why is squared error a good objective?



$$
\begin{aligned}
\theta^* &= \arg\max_\theta \prod_{I=1}^{N} P_\theta(y_i|x_i) \\
&= \arg\max_\theta \prod_{I=1}^{N} \exp -\frac{(y - f_\theta(x))^2}{2\sigma^2} \\
&= \arg\max_\theta \log \prod_{I=1}^{N} \exp -\frac{(y - f_\theta(x))^2}{2\sigma^2} \\
&= \arg\max_\theta \sum_{i=1}^{N} -\frac{(y - f_\theta(x))^2}{2\sigma^2} \\
&= \arg\min_\theta \sum_{i=1}^{N} \frac{(y - f_\theta(x))^2}{2\sigma^2} \\
&= \arg\min_\theta \sum_{i=1}^{N} (y - f_\theta(x))^2
\end{aligned}
$$

Why is squared error a good objective?



Under the assumption that the data is distributed as:

$$Y = f_\theta(X) + \epsilon \qquad \epsilon \sim \mathcal{N}(0, \sigma)$$

$$P_\theta(\{y_i\}_{i=1}^N | \{x_i\}_{i=1}^N) = \prod_{i=1}^N P_\theta(y_i|x_i)$$

The most probable estimate of θ is:

$$\theta^* = \arg\min_\theta \sum_{i=1}^N (f_\theta(x_i) - y_i)^2$$

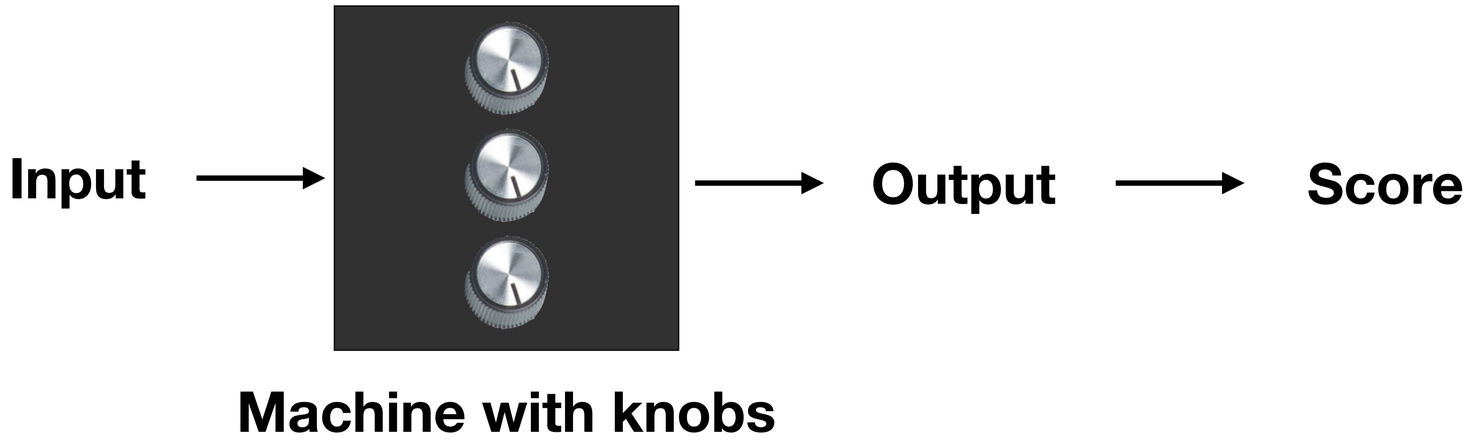
This is called the **max likelihood** estimator of θ.

Notice that this does not depend on the functional form of f!

How to minimize the objective w.r.t. θ?

$$\theta^* = \arg\min_{\theta} \sum_{i=1}^{N} (f_\theta(x_i) - y_i)^2$$

Use an **optimizer**!



How to minimize the objective w.r.t. θ?

In the linear case:

Learning problem

$$\theta^* = \arg\min_\theta \sum_{i=1}^{N} (\theta_1 x_i + \theta_0 - y_i)^2$$

$$E(\theta) = \sum_{i=1}^{N} (\theta_1 x_i + \theta_0 - y_i)^2$$
$$= (\mathbf{y} - \mathbf{X}\theta)^T (\mathbf{y} - \mathbf{X}\theta)$$

$$\mathbf{X} = \begin{pmatrix} x_1 & 1 \\ x_2 & 1 \\ \vdots & \vdots \\ x_N & 1 \end{pmatrix} \qquad \theta = \begin{pmatrix} \theta_1 & \theta_0 \end{pmatrix} \qquad \mathbf{y} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{pmatrix}$$

$$\theta^* = \arg\min_\theta E(\theta)$$

$$\frac{\partial E(\theta)}{\partial \theta} = 0$$

$$\frac{\partial E(\theta)}{\partial \theta} = 2(\mathbf{X}^T\mathbf{X}\theta - \mathbf{X}^T\mathbf{y})$$

$$2(\mathbf{X}^T\mathbf{X}\theta^* - \mathbf{X}^T\mathbf{y}) = 0$$

$$\mathbf{X}^T\mathbf{X}\theta^* = \mathbf{X}^T\mathbf{y}$$

Solution

$$\theta^* = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$$

Empirical Risk Minimization

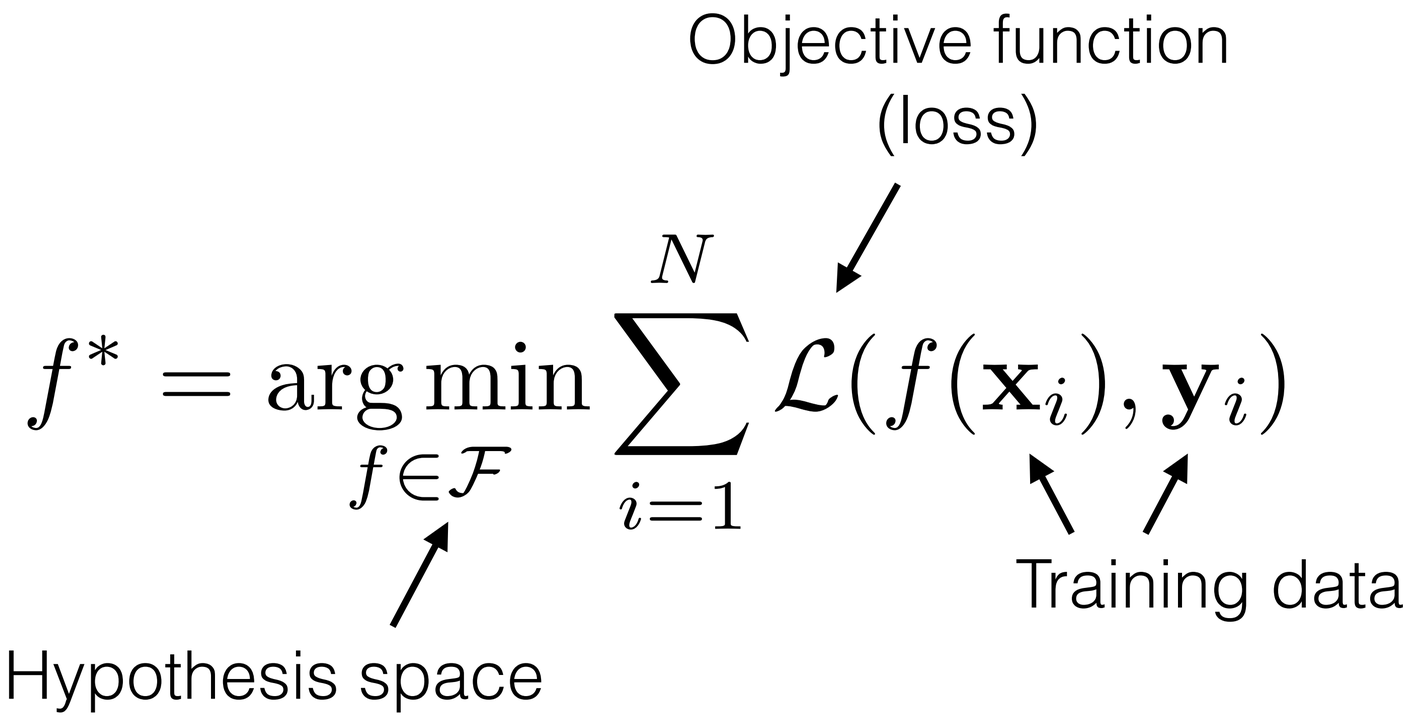
(formalization of supervised learning)

Linear least squares learning problem

$$\theta^* = \arg\min_{\theta} \sum_{i=1}^{N} (\theta_1 x_i + \theta_0 - y_i)^2$$

Empirical Risk Minimization

(formalization of supervised learning)



Example 1: Linear least squares

Data

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

Learner

Objective

$\mathcal{L}(f_\theta(x), y) = (f_\theta(x) - y)^2$

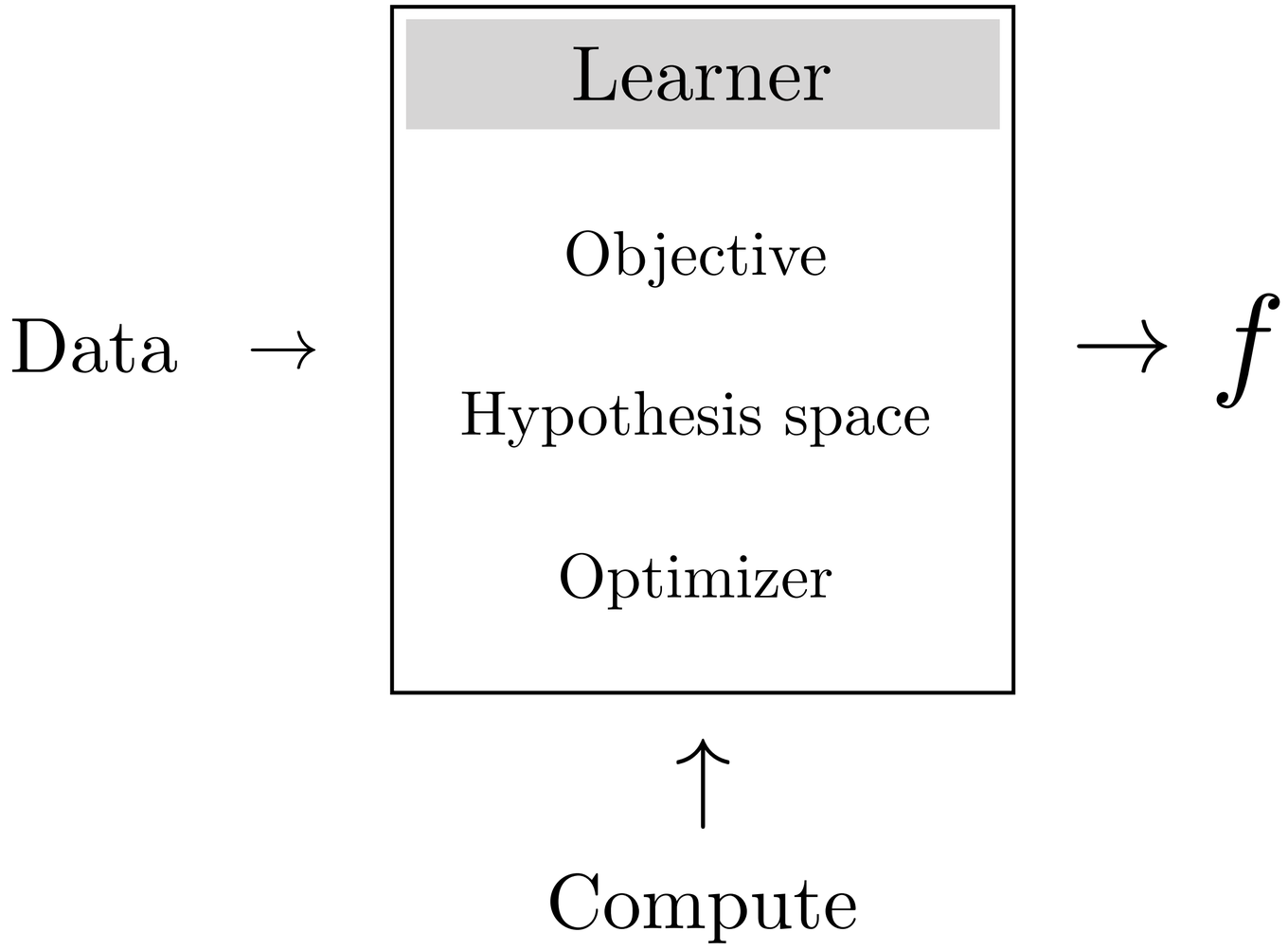
Hypothesis space

$f_\theta(x) = \theta_1 x + \theta_0$

Optimizer

$\theta^* = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$

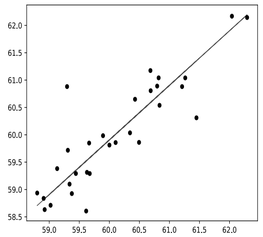
$\rightarrow f$


Learner
Objective
Hypothesis space
Optimizer
Data
$\rightarrow$
$\rightarrow f$
$\uparrow$
Compute

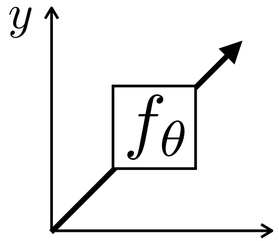
Example 1: Linear least squares

Training

Data

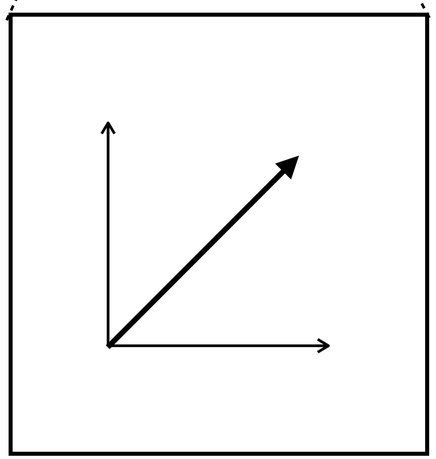


→ Learner →



Testing

Input →

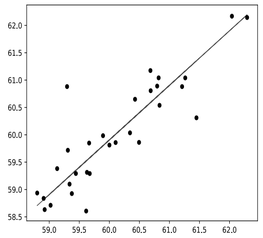


→ Output

Example 2: Program induction

Training

Data



→ Learner →

```
def predict(x):
    y = 0.8*x + 2
    return y
```

Testing

Input →

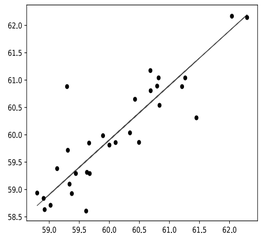
```
def predict(x):
    y = 0.8*x + 2
    return y
```

→ Output

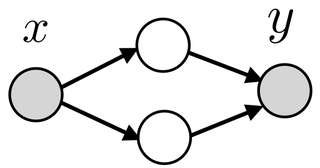
Example 3: Deep learning

Training

Data

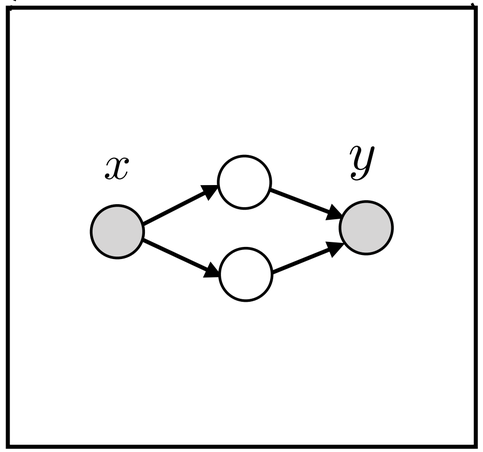


→ Learner →



Testing

Input →



→ Output

Learning for vision

Big questions:

1. How do you represent the input and output?

2. What is the objective?

3. What is the hypothesis space? (e.g., linear, polynomial, neural net?)

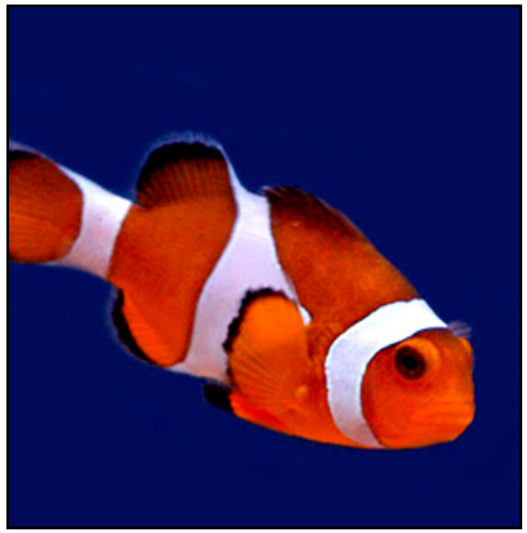
4. How do you optimize? (e.g., gradient descent, Newton's method?)

5. What data do you train on?

Case study 1: Image classification

1. **How do you represent the input and output?**
2. **What is the objective?**
3. Assume hypothesis space is sufficienly expressive
4. Assume we optimize perfectly
5. Assume we train on exactly the data we care about

Image classification



Classifier

“Fish”

image **x**

label y

Image classification

Classifier

“Fish”

image **x**

label y

Image classification

Classifier

“Fish”

image **x**

label y

Image classification

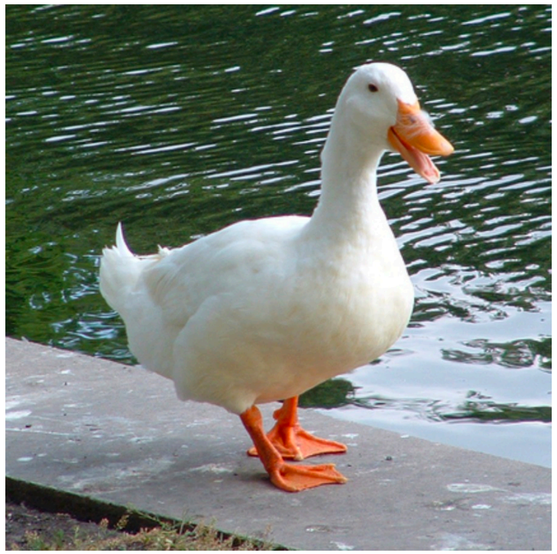


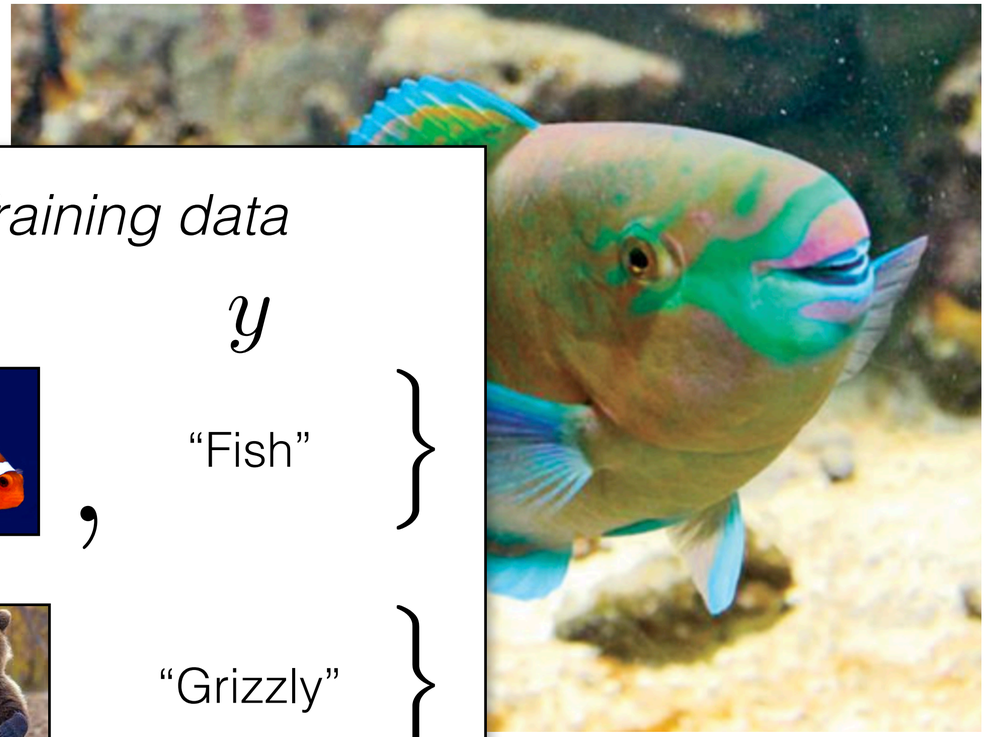
image **x**

Classifier

"Duck"

label y

$\mathbf{x}$



Training data

$\mathbf{x}$	y
	“Fish”
	“Grizzly”
	“Chameleon”
⋮	

f

y

“Fish”

$$\arg\min_{f \in \mathcal{F}} \sum_{i=1}^{N} \mathcal{L}(f(\mathbf{x}_i), y_i)$$

How to represent class labels?

One-hot vector

Training data

$\mathbf{x}$ y

{  , "Fish" }

{  , "Grizzly" }

{  , "Chameleon" }

⋮

→

Training data

$\mathbf{x}$ y

{  , 1 }

{  , 2 }

{  , 3 }

⋮

→

Training data

$\mathbf{x}$ $\mathbf{y}$

{  , [0,0,1] }

{  , [0,1,0] }

{  , [1,0,0] }

⋮

What should the loss be?

0-1 loss (number of misclassifications)

$$\mathcal{L}(\hat{\mathbf{y}}, \mathbf{y}) = \mathbb{1}(\hat{\mathbf{y}} = \mathbf{y})$$ ← discrete, NP-hard to optimize!

Cross entropy

$$\mathcal{L}(\hat{\mathbf{y}}, \mathbf{y}) = H(\mathbf{y}, \hat{\mathbf{y}}) = -\sum_{k=1}^{K} y_k \log \hat{y}_k$$ ← continuous, differentiable

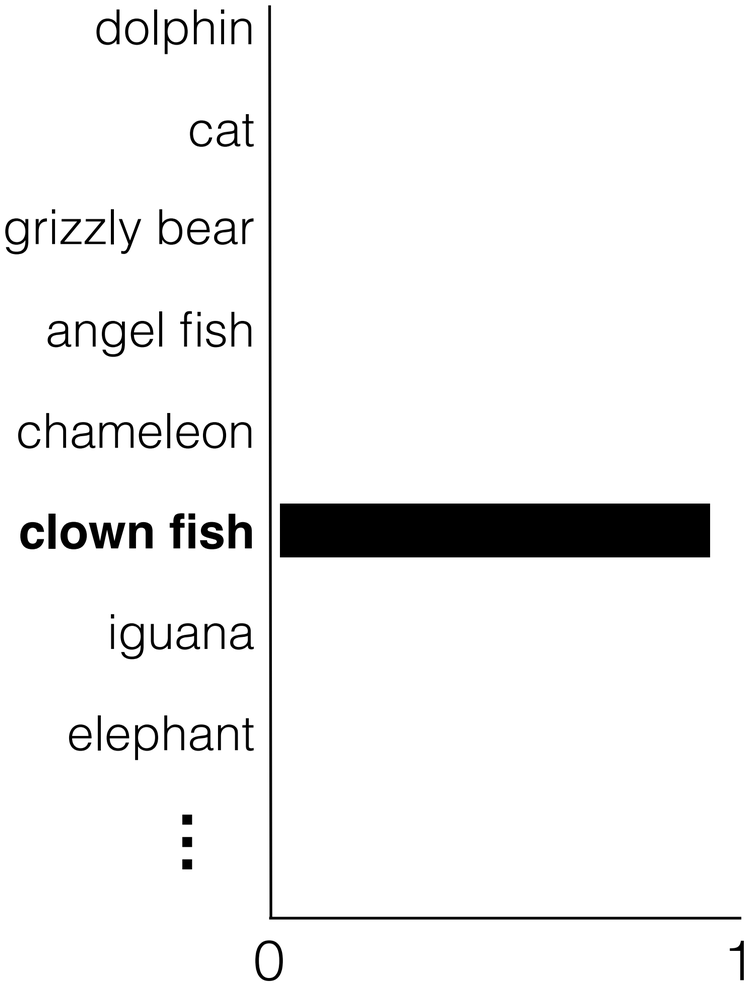
Ground truth label $\mathbf{y}$

$\mathbf{x}$

[0,0,0,0,0,1,0,0,…]

Ground truth label $\mathbf{y}$

$\mathbf{x}$



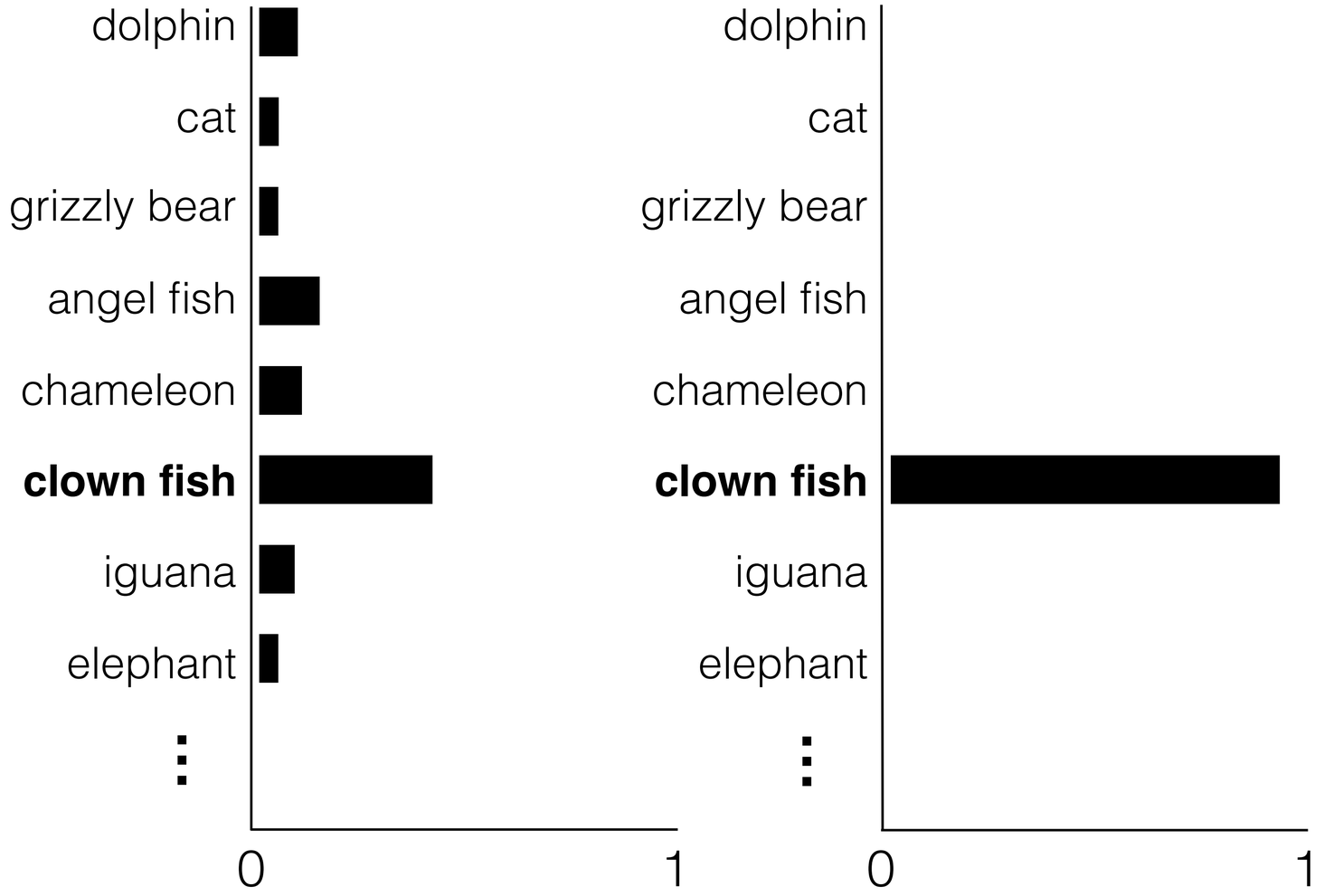
Prediction $\hat{\mathbf{y}}$

$f_\theta : X \to \mathbb{R}^K$

Ground truth label $\mathbf{y}$

$\mathbf{x}$

f



Prediction $\hat{\mathbf{y}}$

$$f_\theta : X \to \mathbb{R}^K$$

Ground truth label $\mathbf{y}$

Loss

$$H(\mathbf{y}, \hat{\mathbf{y}}) = -\sum_{k=1}^{K} y_k \log \hat{y}_k$$

$\mathbf{x}$

f

dolphin
cat
grizzly bear
angel fish
chameleon
clown fish
iguana
elephant
⋮

0 1

$\odot$

dolphin
cat
grizzly bear
angel fish
chameleon
clown fish
iguana
elephant
⋮

0 1

0 1

Prediction $\hat{\mathbf{y}}$

$$f_\theta : X \to \mathbb{R}^K$$

Ground truth label $\mathbf{y}$

Loss

$$H(\mathbf{y}, \hat{\mathbf{y}}) = -\sum_{k=1}^{K} y_k \log \hat{y}_k$$

$\mathbf{x}$

f

dolphin
cat
grizzly bear
angel fish
chameleon
clown fish
iguana
elephant
⋮

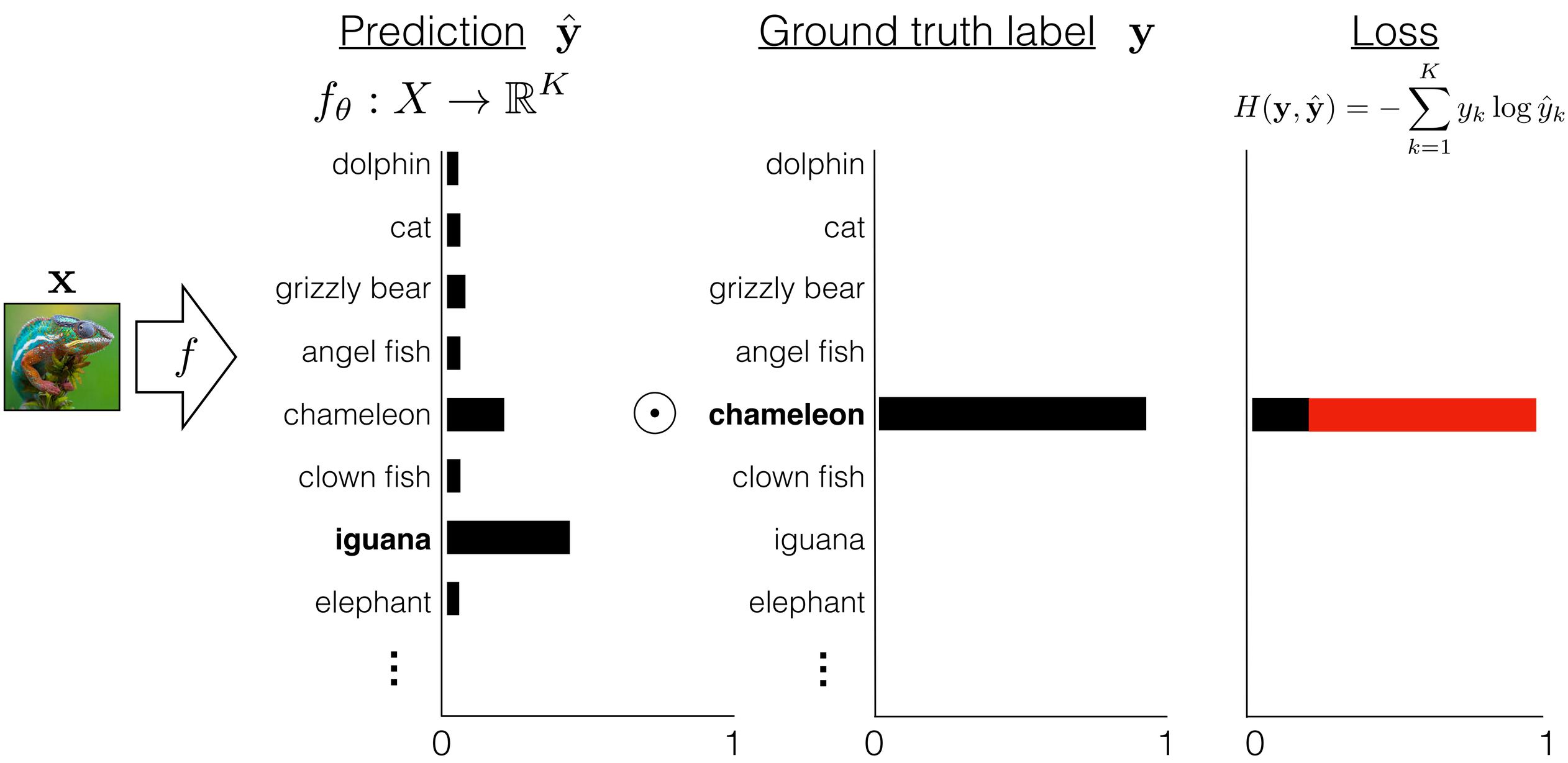
0 1

$\odot$

dolphin
cat
grizzly bear
angel fish
chameleon
clown fish
iguana
elephant
⋮

0 1

0 1


Prediction $\hat{\mathbf{y}}$
$f_\theta : X \to \mathbb{R}^K$
Ground truth label $\mathbf{y}$
Loss
$H(\mathbf{y}, \hat{\mathbf{y}}) = -\sum_{k=1}^{K} y_k \log \hat{y}_k$
$\mathbf{x}$
f
dolphin
cat
grizzly bear
angel fish
chameleon
clown fish
iguana
elephant
0
1
dolphin
cat
grizzly bear
angel fish
chameleon
clown fish
iguana
elephant
0
1
0
1

Softmax regression (a.k.a. multinomial logistic regression)

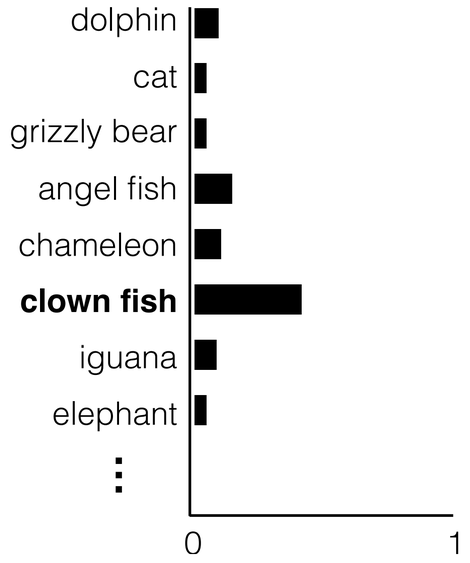
$$f_\theta : X \to \mathbb{R}^K$$

$$\mathbf{z} = f_\theta(\mathbf{x})$$ ← **logits**: vector of K scores, one for each class

$$\hat{\mathbf{y}} = \texttt{softmax}(\mathbf{z})$$ ← squash into a non-negative vector that sums to 1 — i.e. **a probability mass function!**

$$\hat{y}_j = \frac{e^{-z_j}}{\sum_{k=1}^{K} e^{-z_k}}$$

$\hat{\mathbf{y}} =$



Softmax regression (a.k.a. multinomial logistic regression)

Probabilistic interpretation:

$$\hat{\mathbf{y}} \equiv [P_\theta(Y = 1|X = \mathbf{x}), \ldots, P_\theta(Y = K|X = \mathbf{x})]$$ ← predicted probability of each class given input **x**

$$H(\mathbf{y}, \hat{\mathbf{y}}) = -\sum_{k=1}^{K} y_k \log \hat{y}_k$$ ← picks out the -log likelihood of the ground truth class $\mathbf{y}$ under the model prediction $\hat{\mathbf{y}}$

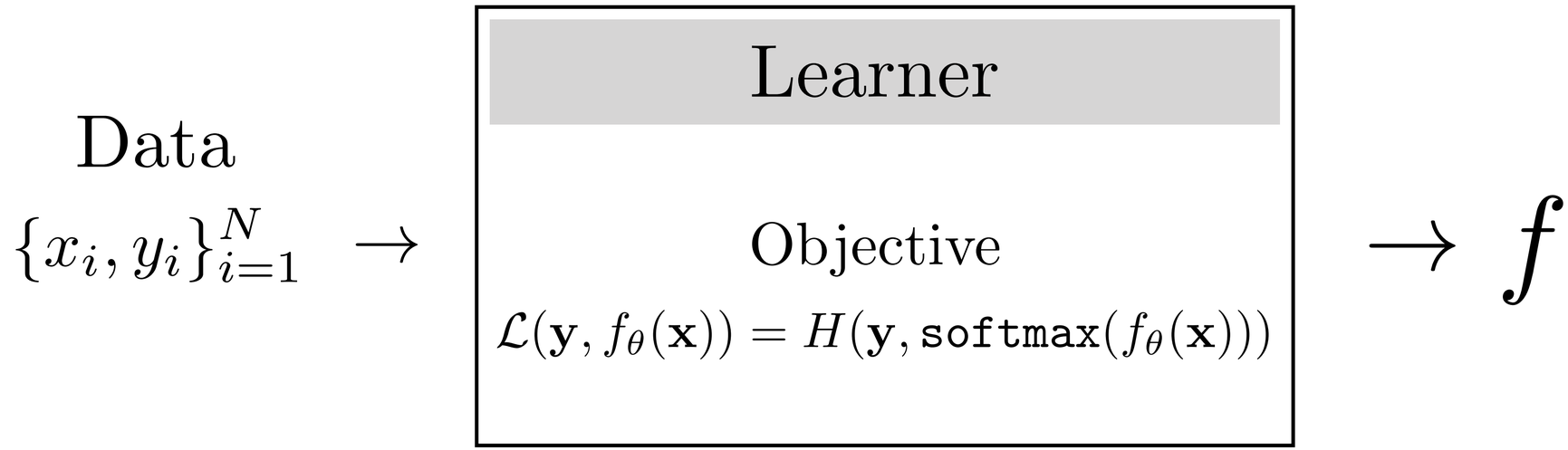
$$f^* = \arg\min_{f \in \mathcal{F}} \sum_{i=1}^{N} H(\mathbf{y}_i, \hat{\mathbf{y}}_i)$$ ← max likelihood learner!

Softmax regression (a.k.a. multinomial logistic regression)

$$f_\theta : X \to \mathbb{R}^K$$

$$\mathbf{z} = f_\theta(\mathbf{x})$$

$$\hat{\mathbf{y}} = \texttt{softmax}(\mathbf{z})$$



Case study 2: Image colorization

1. **How do you represent the input and output?**

2. **What is the objective?**

3. Assume hypothesis space is sufficienly expressive

4. Assume we optimize perfectly

5. **What data do we train on?**

Ansel Adams, Yosemite Valley Bridge


Result of [Zhang et al., ECCV 2016]

Image colorization

Input $\mathbf{x}$

Output $\mathbf{y}$

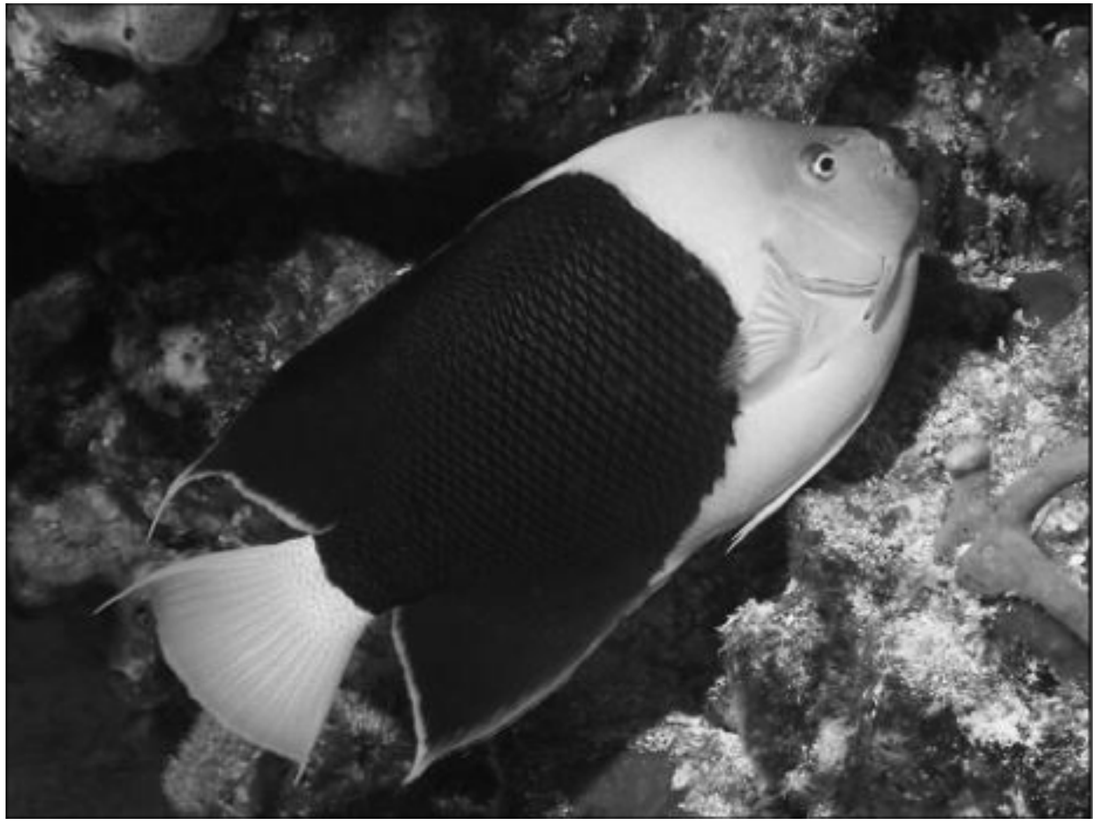
f

Training data

$\mathbf{x}$ $\mathbf{y}$

$$\arg\min_{f \in \mathcal{F}} \sum_{i=1}^{N} \mathcal{L}(f(\mathbf{x}_i), \mathbf{y}_i)$$

[Zhang et al., ECCV 2016]



f

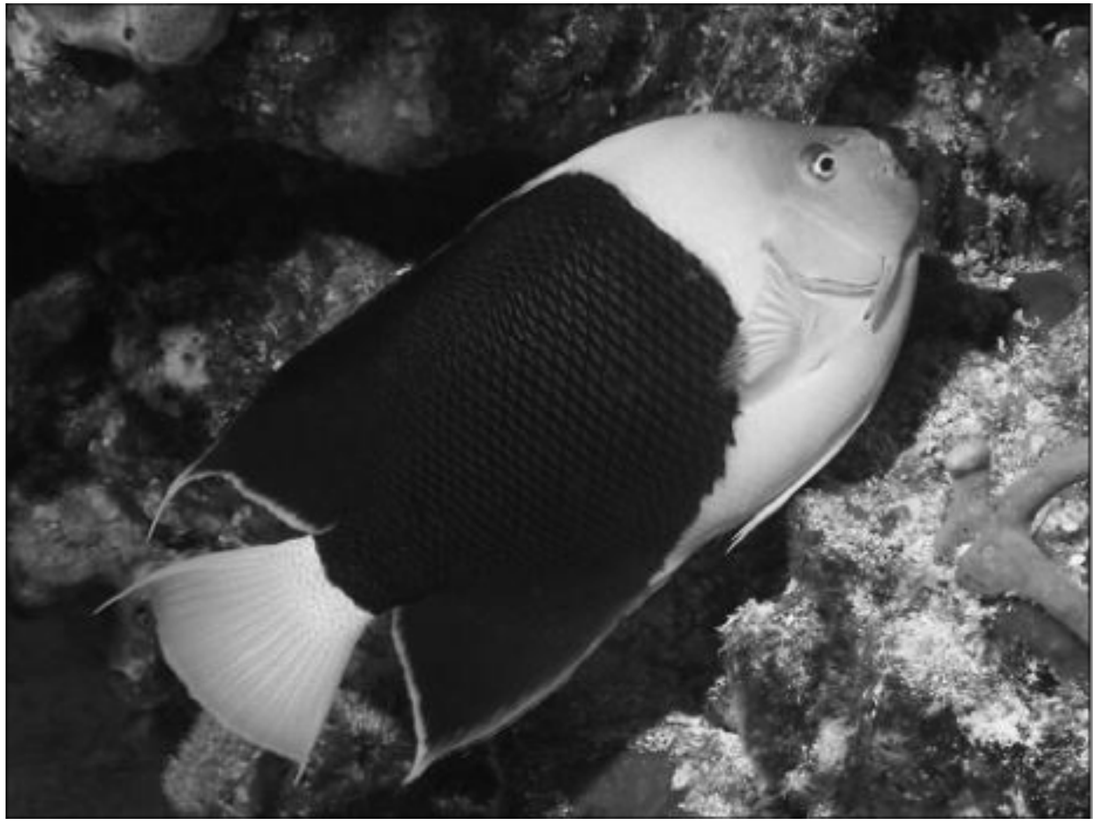
Grayscale image: **L channel**

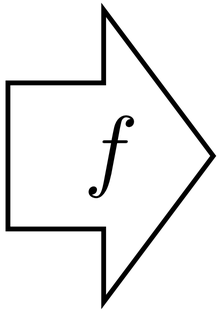
$$\mathbf{x} \in \mathbb{R}^{H \times W \times 1}$$

Color information: **ab channels**

$$\mathbf{y} \in \mathbb{R}^{H \times W \times 2}$$

[Zhang et al., ECCV 2016]





Grayscale image: **L channel**

$$\mathbf{x} \in \mathbb{R}^{H \times W \times 1}$$

Color information: **ab channels**

$$\mathbf{y} \in \mathbb{R}^{H \times W \times 2}$$

[Zhang et al., ECCV 2016]

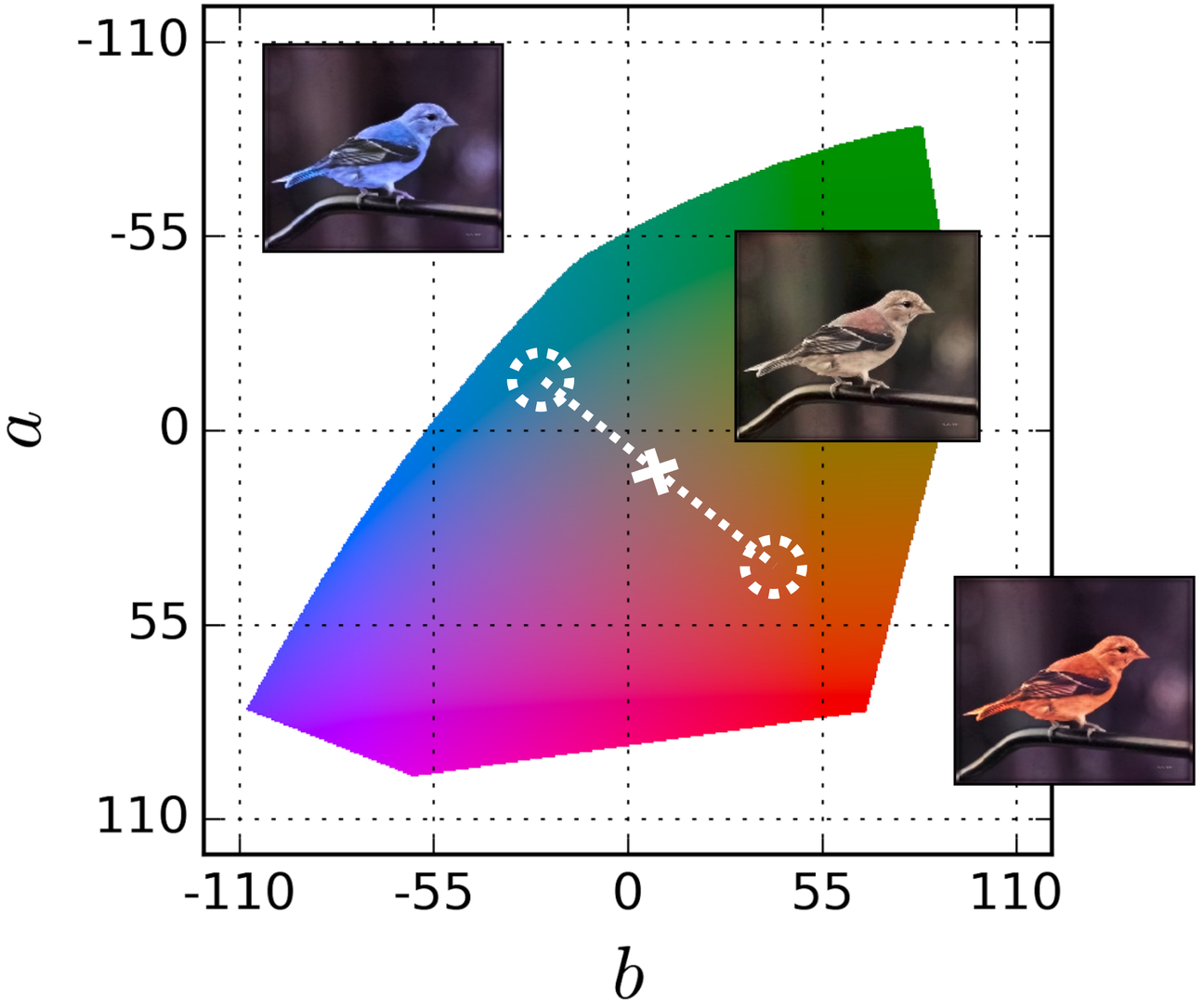
Choosing loss and representation

Input

Output

Ground truth

$$\mathcal{L}(f(\mathbf{x}), \mathbf{y}) = \|f(\mathbf{x}) - \mathbf{y}\|_2^2$$



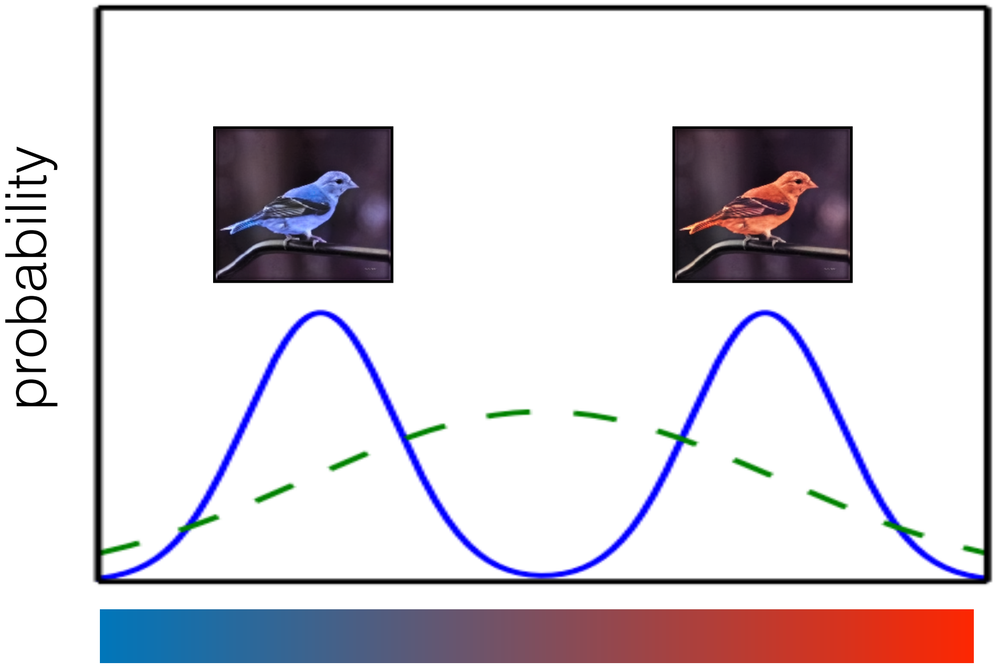
$$\mathcal{L}(f(\mathbf{x}), \mathbf{y}) = \|f(\mathbf{x}) - \mathbf{y}\|_2^2$$

Recall: least squares loss corresponds to the following modeling assumptions:

$$Y = f_\theta(X) + \epsilon \qquad \epsilon \sim \mathcal{N}(0, \sigma)$$

$$P_\theta(Y = y|X = x) \propto \exp \frac{-(y - f_\theta(x))^2}{2\sigma^2}$$

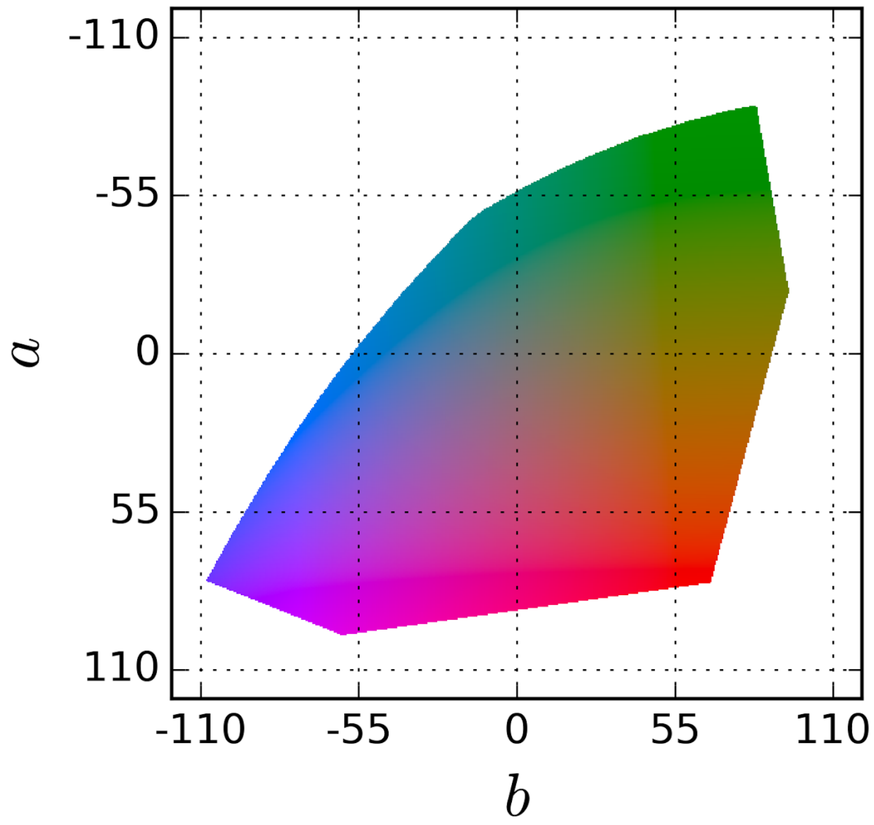
Prediction for a single pixel i,j



Best **Gaussian fit** to the **true data distribution** is to center the Gaussian on the mean of the data distribution.

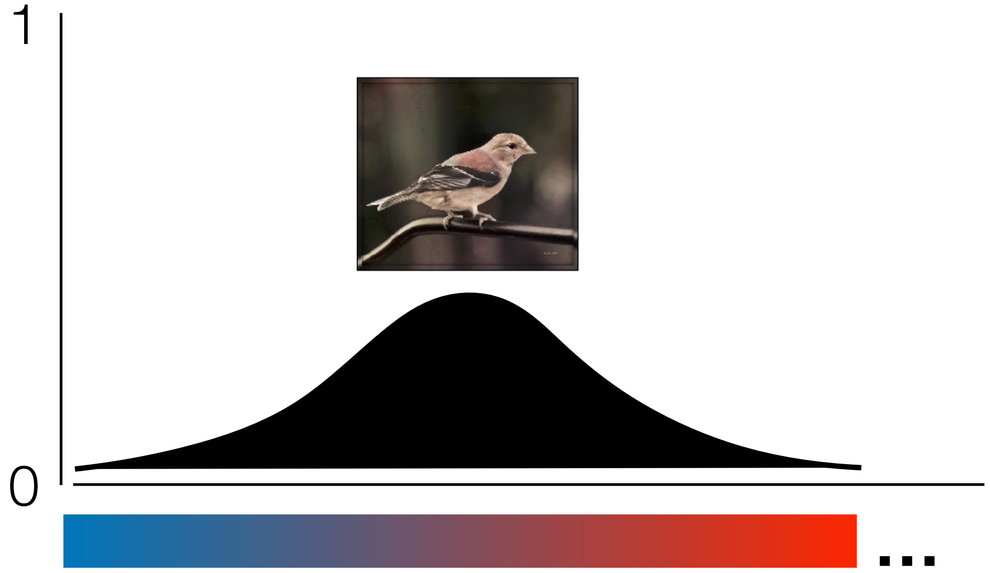
fig modified from [Goodfellow, 2016]

$$\mathbf{y} \in \mathbb{R}^{H \times W \times 2}$$



f

Prediction for a single pixel i,j

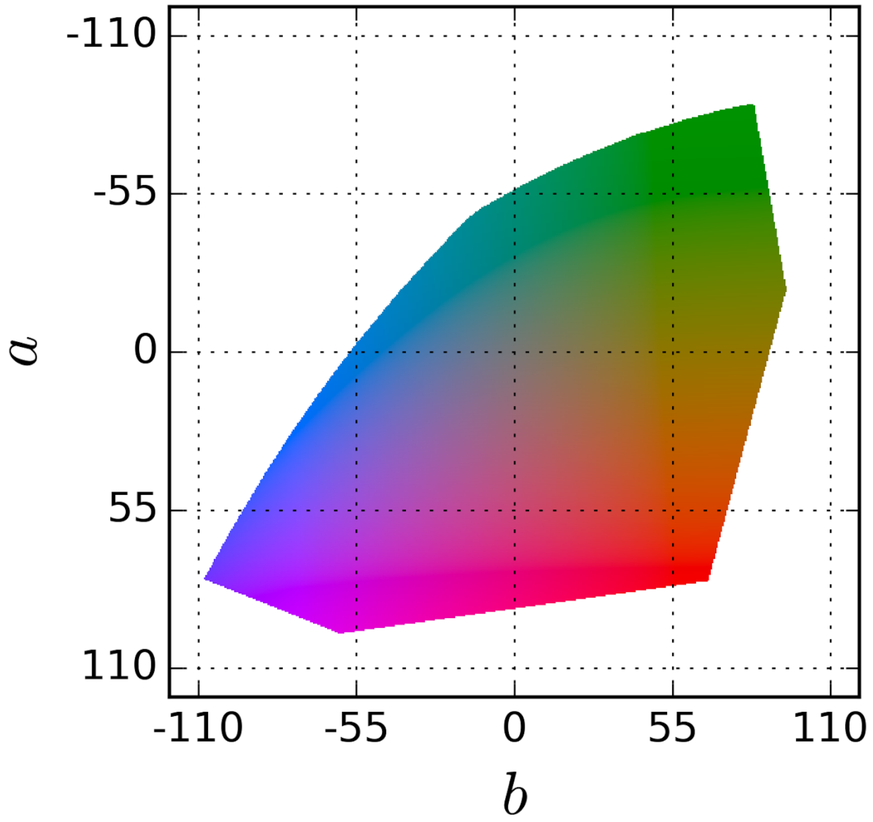


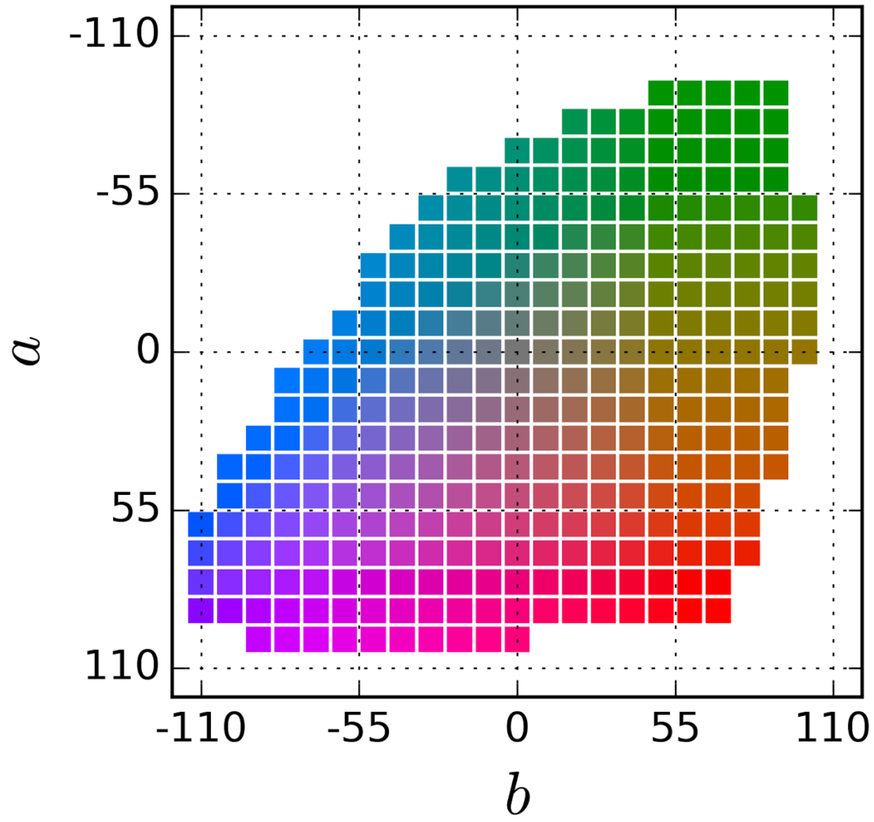
$$\mathcal{L}(f(\mathbf{x}), \mathbf{y}) = \|f(\mathbf{x}) - \mathbf{y}\|_2^2$$

one-hot representation of K discrete classes

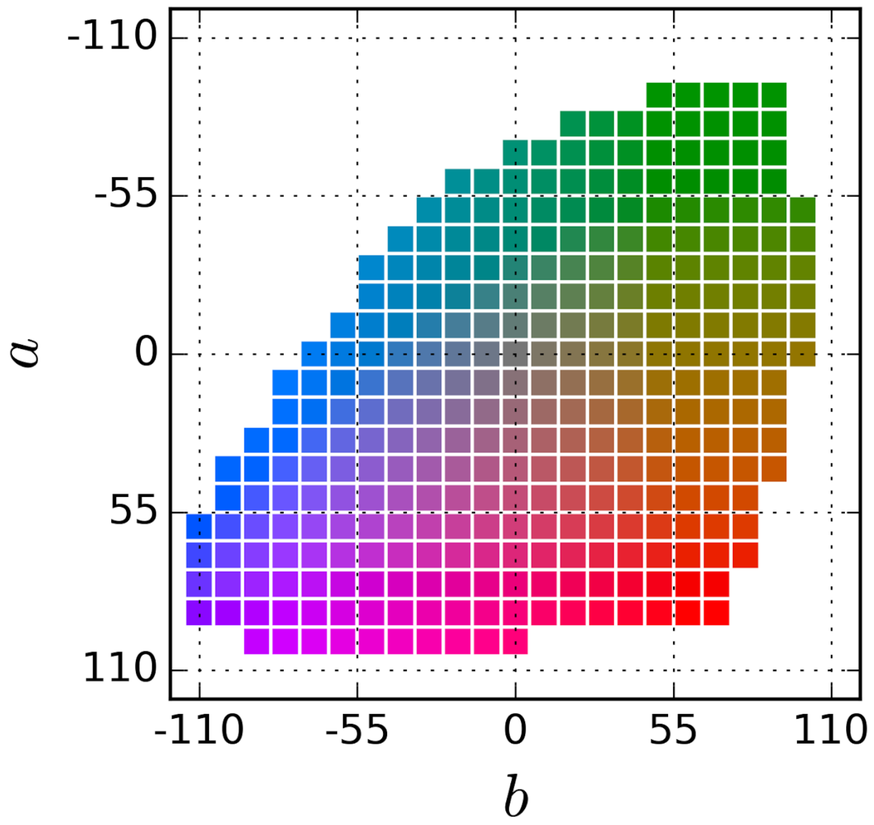
$\mathbf{y} \in \mathbb{R}^{H \times W \times 2}$

$\mathbf{y} \in \mathbb{R}^{H \times W \times K}$



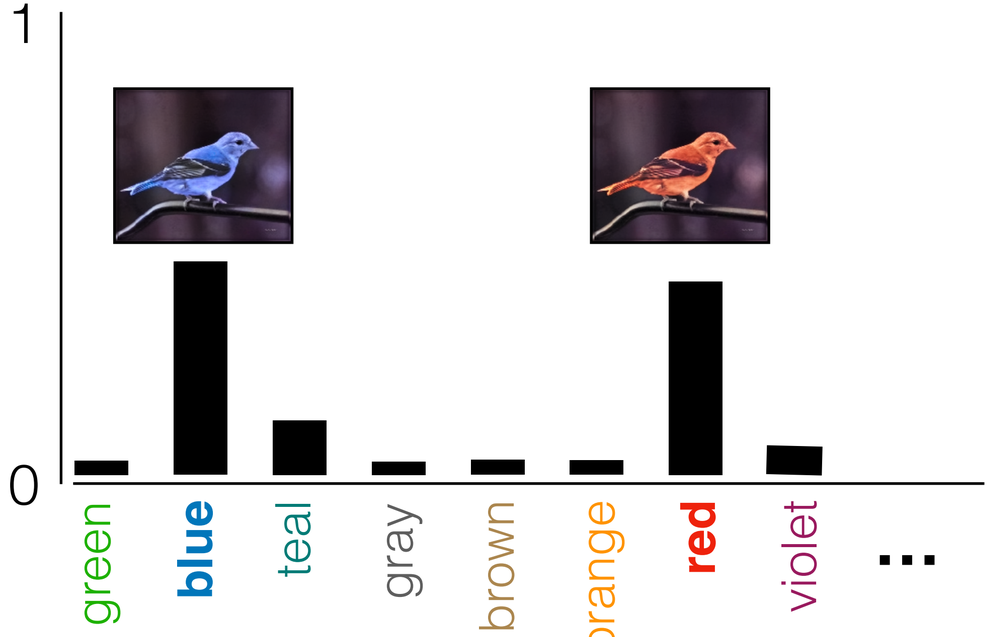


$$\mathbf{y} \in \mathbb{R}^{H \times W \times K}$$

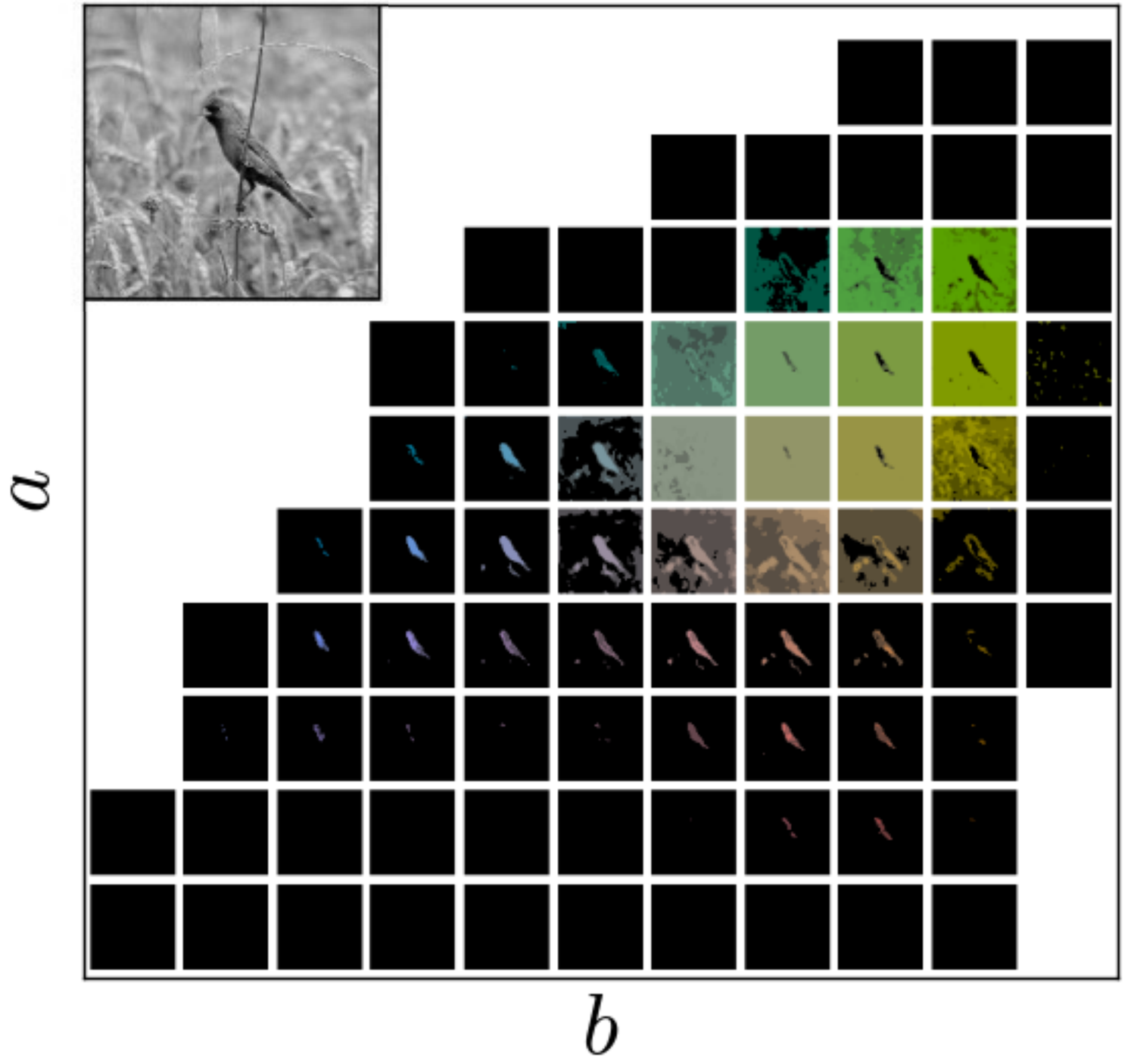


f

Prediction for a single pixel i,j



$$\mathcal{L}(\mathbf{y}, f_\theta(\mathbf{x})) = H(\mathbf{y}, \texttt{softmax}(f_\theta(\mathbf{x})))$$


a
b

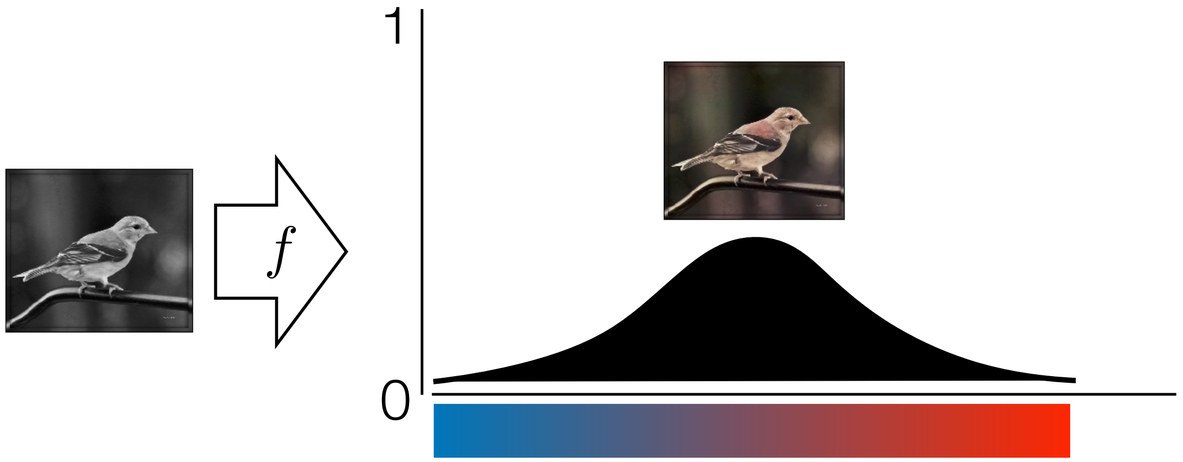
Input

Zhang et al. 2016

Ground truth

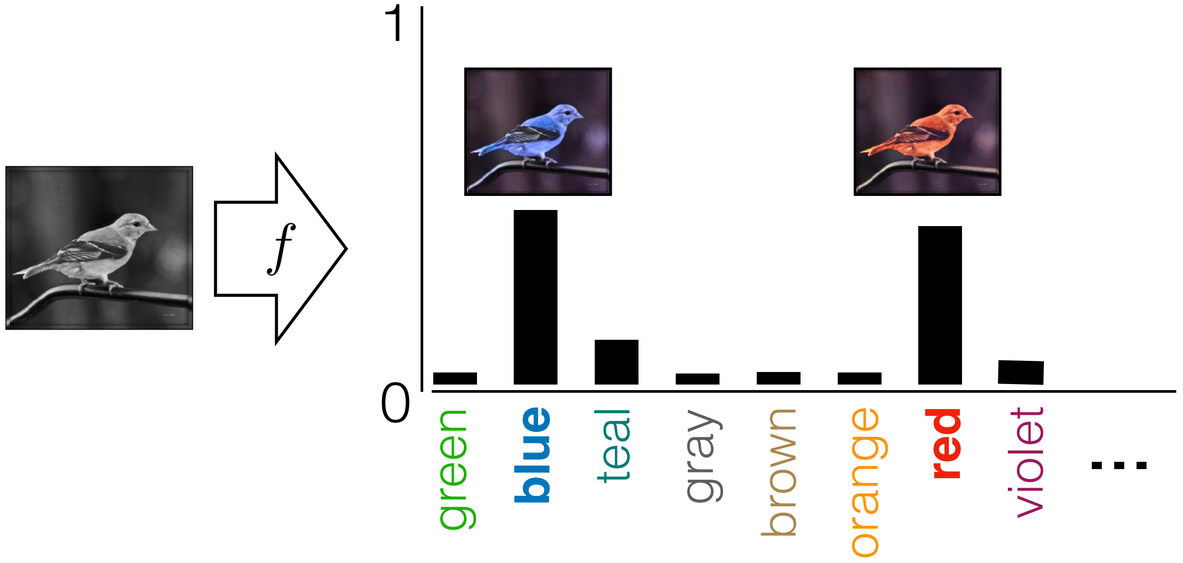
$$\mathcal{L}(\mathbf{x}, \mathbf{y}) = H(\mathbf{y}, \texttt{softmax}(f_\theta(\mathbf{x})))$$

“Regression”



- Continuous-valued prediction
- (Usually) models unimodal distribution

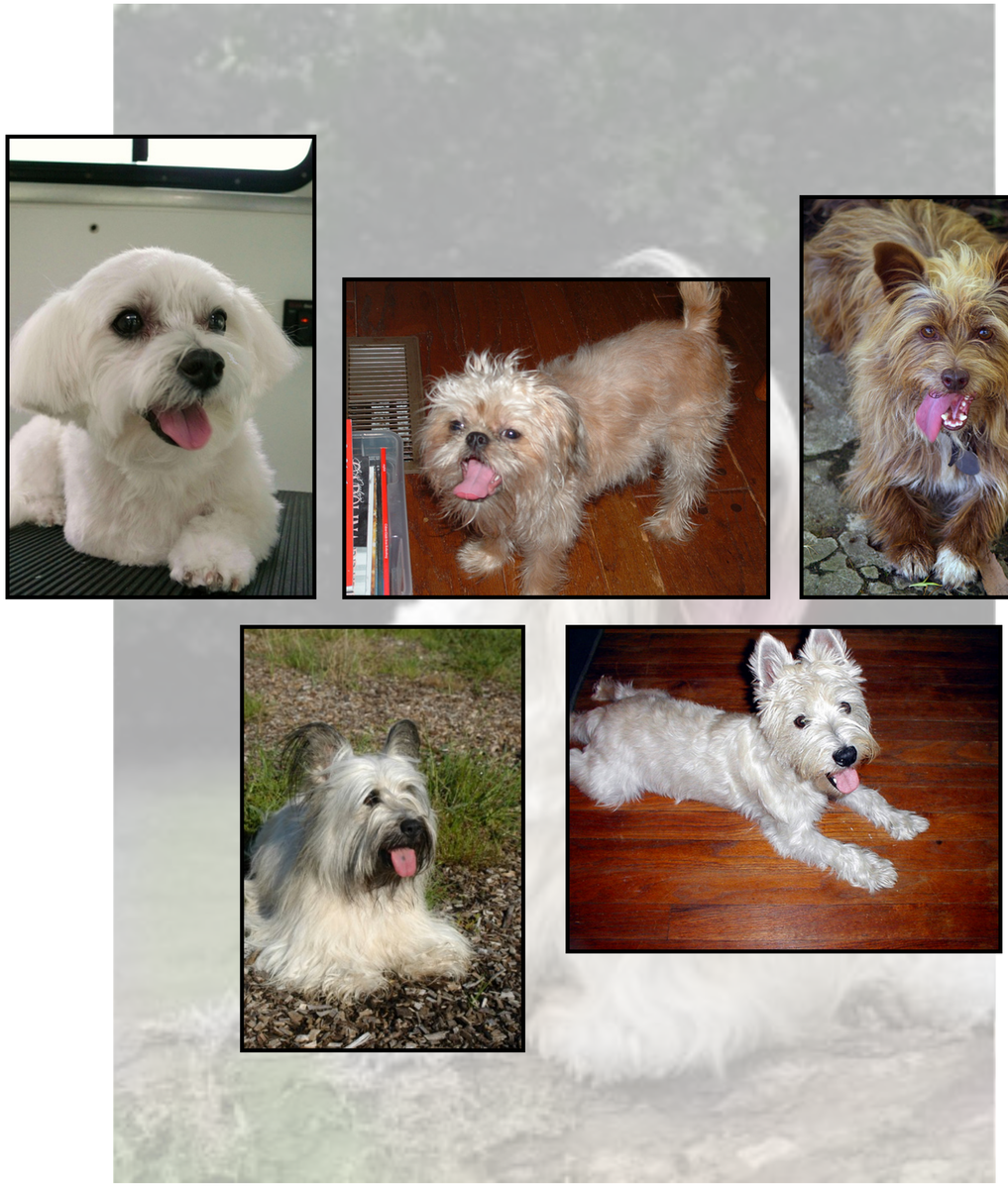
“Classification”



- Discrete-valued prediction
- Models multimodal distribution

Instructive failure

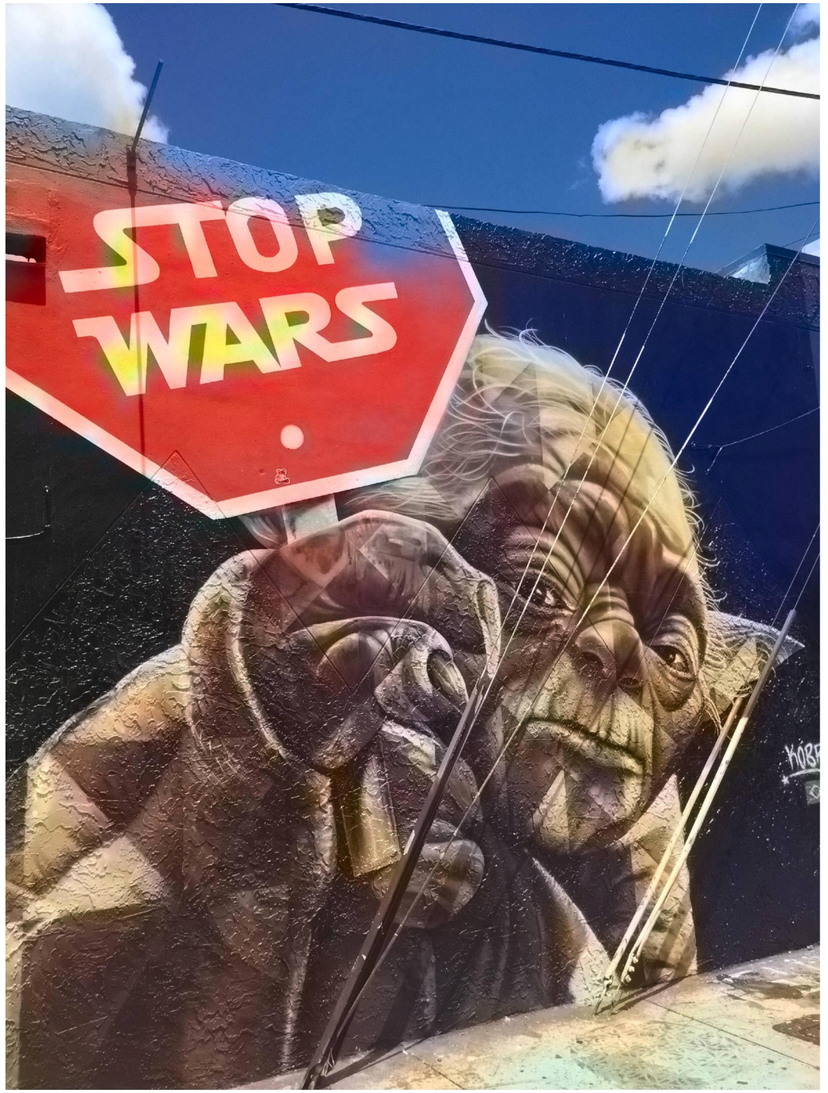
Instructive failure



[from Reddit /u/SherySantucci]

[Recolorized by Reddit ColorizeBot]

Photo taken by Reddit /u/ Timteroo, Mural from street artist Eduardo Kobra



Recolorized by
Reddit
ColorizeBot

Image colorization in a nutshell

Data

$\mathbf{x}$ $\mathbf{y}$

$\mathbf{x} \in \mathbb{R}^{H \times W \times 1}$

$\mathbf{y} \in \mathbb{R}^{H \times W \times K}$

$\rightarrow$

Learner

Objective

$\mathcal{L}(\mathbf{y}, f_\theta(\mathbf{x})) = H(\mathbf{y}, \texttt{softmax}(f_\theta(\mathbf{x})))$

Hypothesis space

Convolutional neural net

Optimizer

Stochastic gradient descent

$\rightarrow$ f

Next up:

1. overfitting, regularization, generalization
2. neural nets!