18734: Foundations of Privacy

Introduction to Machine Learning

Giulia Fanti
Based on slides by Anupam Datta
CMU

Fall 2019

Administrative

- HW2 due on Friday, Sept 27
  - 12 pm ET/12 pm PT
- My OH this week TIME CHANGE
  - Tuesday, 5pm ET/2pm PT, CIC 2118
  - Right after Sruti's OH
  - SV: Join the Google hangout on the course website
- Wednesday lecture by Sruti

Survey Results

Things I can change

- Posting lecture slides before class
- Try to be extra explicit about why math is related to privacy problem
  - Use only privacy-related examples
- Format of math derivations
  - Document cam
  - Tablet

Things I cannot change

- Math
  - Can try to change the title to “Mathematical Foundations of Privacy” next time
- Discussion of research topics
  - Techniques are still being developed
- Video recordings
  - I don’t control the A/V situation

10-minute quiz

- On Canvas

End of Unit 1

- Privacy perspective
  - Ways to audit privacy policies
  - Insider
    - Enforcing use restrictions (MDPs)
    - Legalease + Grok
  - Outsider
    - XRay
    - Information Flow experiments
- Tools/concepts we have seen
  - Randomized sampling for group testing
  - Markov decision processes (MDPs)
  - Statistical significance tests
  - Lattices

Unit 2: Protecting Privacy and Fairness in Big Data Analytics

- Machine learning
- What are common privacy risks?
- What are common fairness risks?
- How do we fix each?

INTRO TO MACHINE LEARNING

This Lecture

- Basic concepts in classification
- Illustrated with simple classifiers
  - K-Nearest Neighbors
  - Linear Classifiers

- Note: Start exploring scikit-learn or TensorFlow/PyTorch if you are planning to use ML in your course project

Image Classification

Image Classification

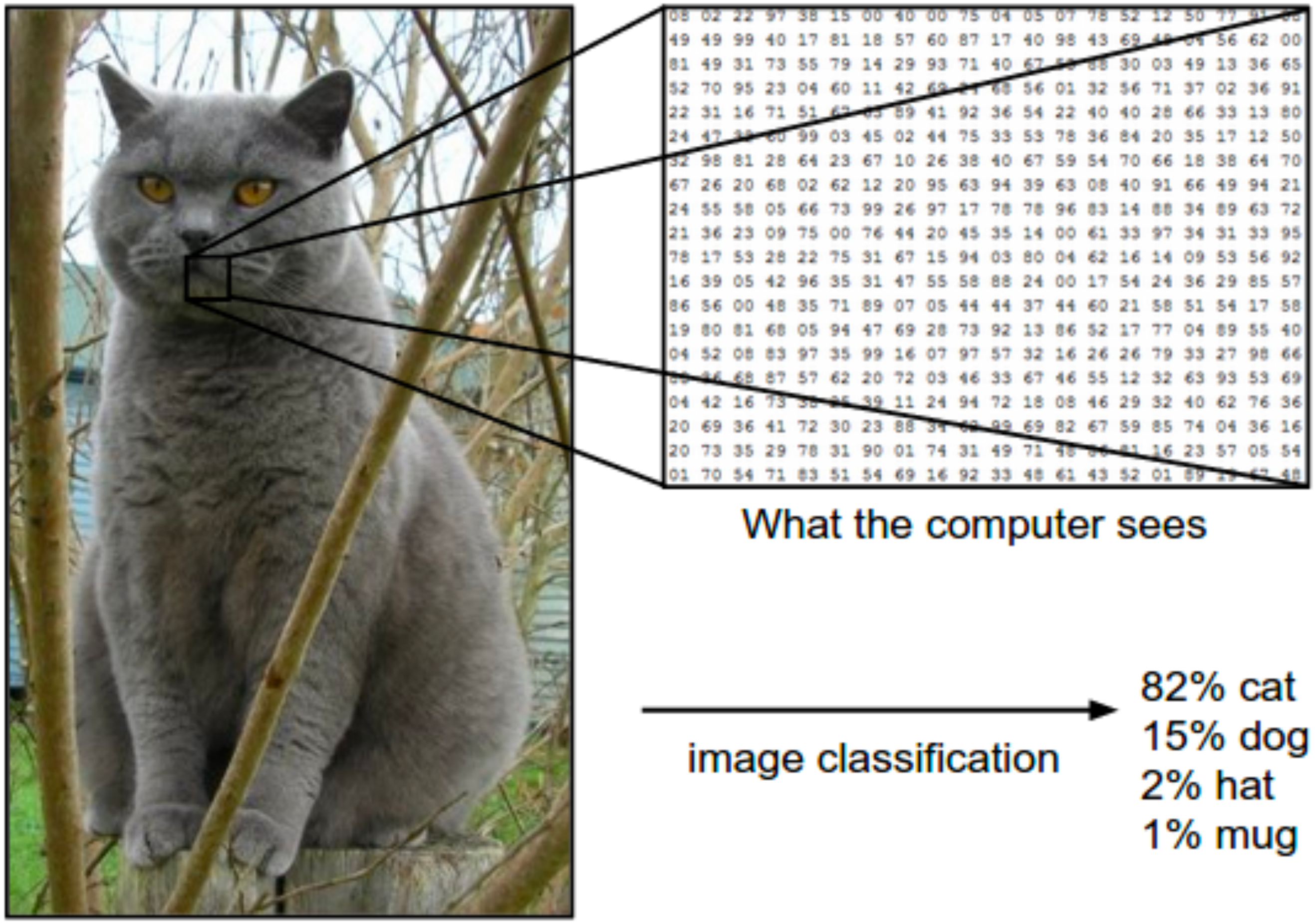
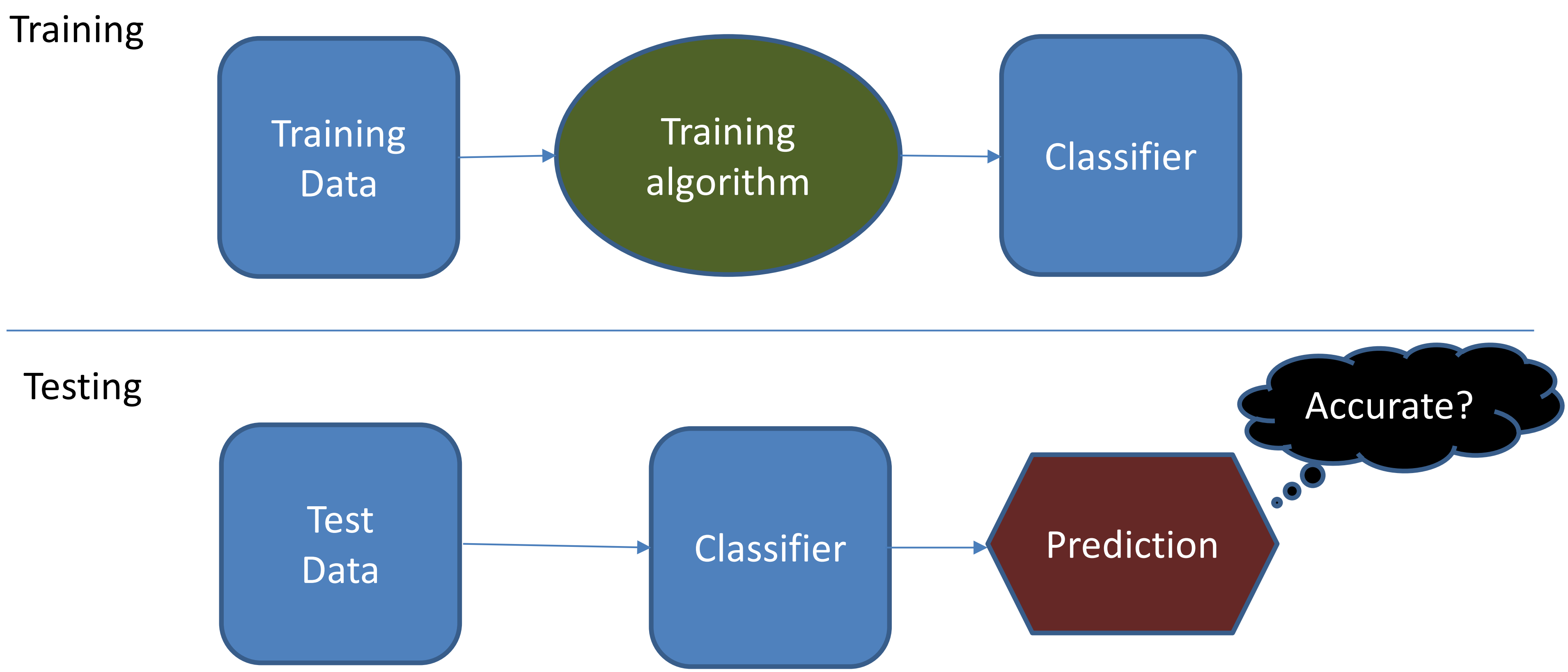


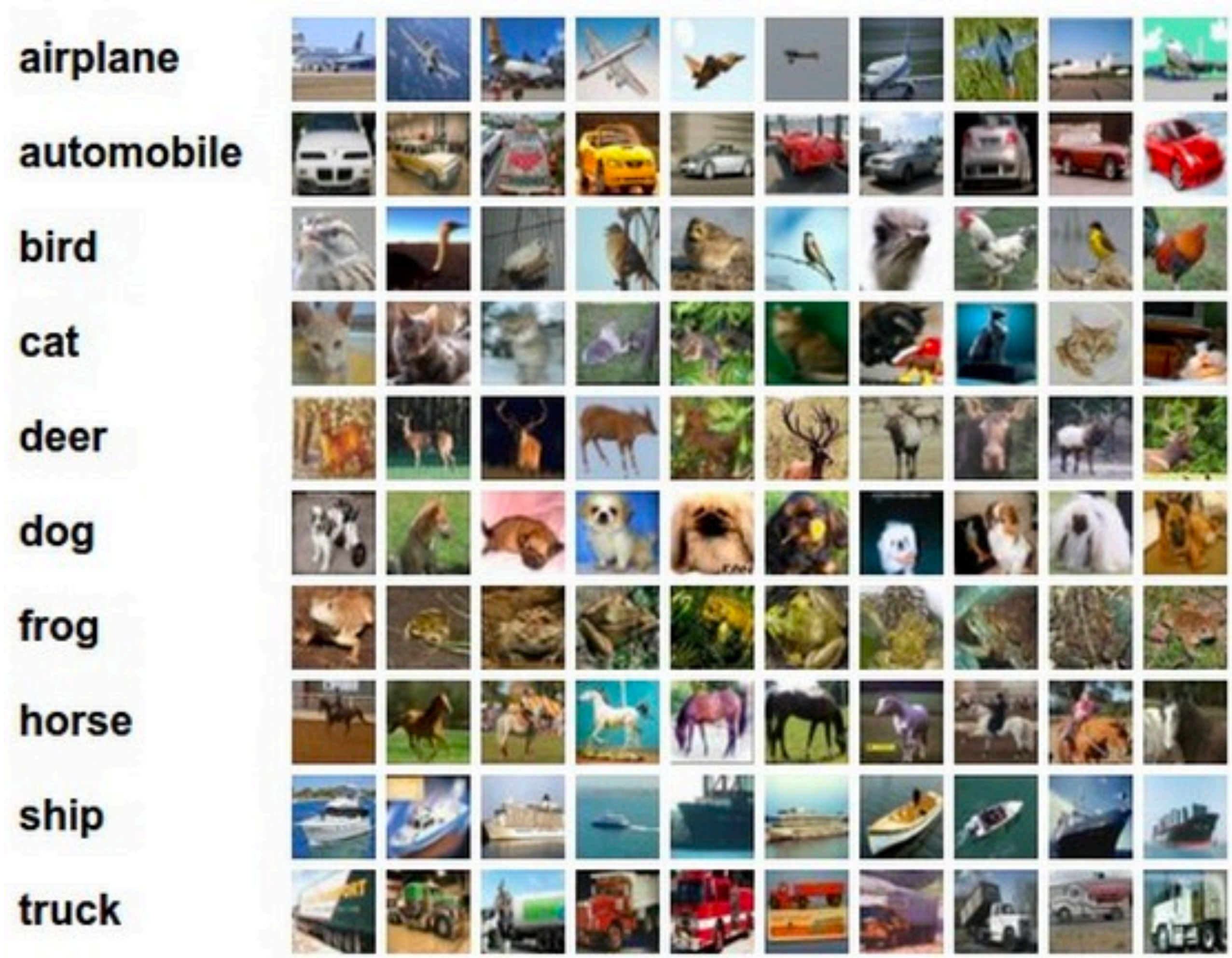
Image classification pipeline

- **Input:** A training set of *N* images, each labeled with one of *K* different classes.
- **Learning:** Use training set to learn classifier (model) that predicts what class input images belong to.
- **Evaluation:** Evaluate quality of classifier by asking it to predict labels for a new set of images that it has never seen before.

Classification pipeline



CIFAR-10 dataset



- 60,000 tiny images that are 32 pixels high and wide.
- Each image is labeled with one of 10 classes

Nearest Neighbor Classification

The top 10 nearest neighbors in the training set
according to “pixel-wise difference”.

Pixel-wise difference

test image I_1

56	32	10	18
90	23	128	133
24	26	178	200
2	0	255	220

\-

training image I_2

10	20	24	17
8	10	89	100
12	16	178	170
4	32	233	112

=

pixel-wise absolute value differences

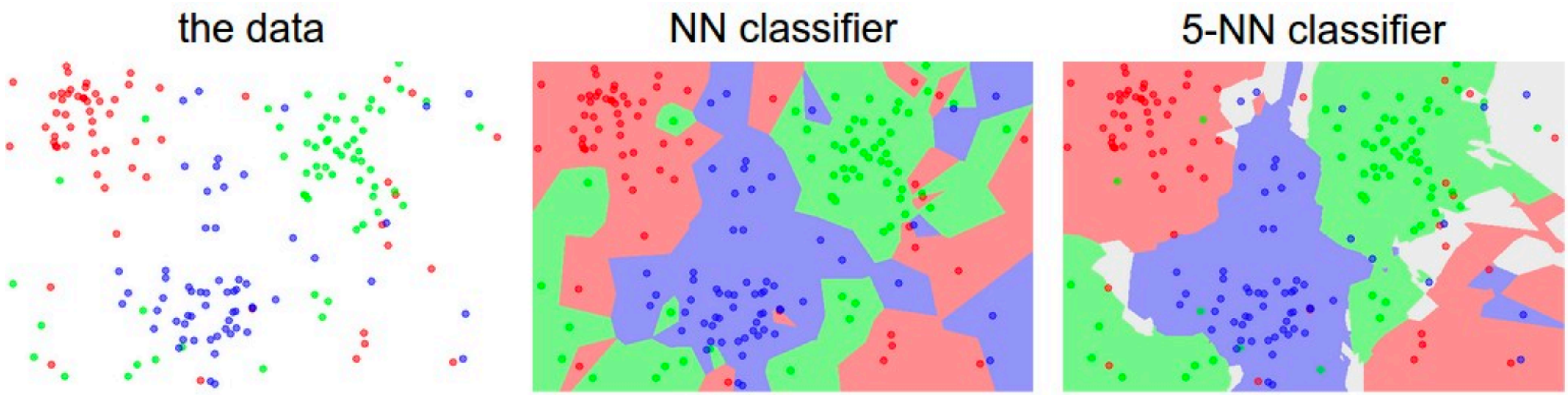
46	12	14	1
82	13	39	33
12	10	0	30
2	32	22	108

→ 456

L1 norm: $d_1(I_1, I_2) = \Sigma_p |I_1^p - I_2^p|$

L2 norm: $d_2(I_1, I_2) = \sqrt[2]{\Sigma_p (I_1^p - I_2^p)^2}$

K-Nearest Neighbor Classifier



Disadvantages of k-NN

- The classifier must *remember* all of the training data and store it for future comparisons with the test data. This is space inefficient because datasets may easily be gigabytes in size.

- Classifying a test image is expensive since it requires a comparison to all training images.

- k-NN does not work well in high dimensions

Linear Classification

Linear model

- Score function
  - Maps raw data to class scores
  - Usually parametric
- Loss function (objective function)
  - Measures how well predicted classes agree with ground truth labels
  - How good is our score function?
- Learning
  - Find parameters of score function that minimize loss function

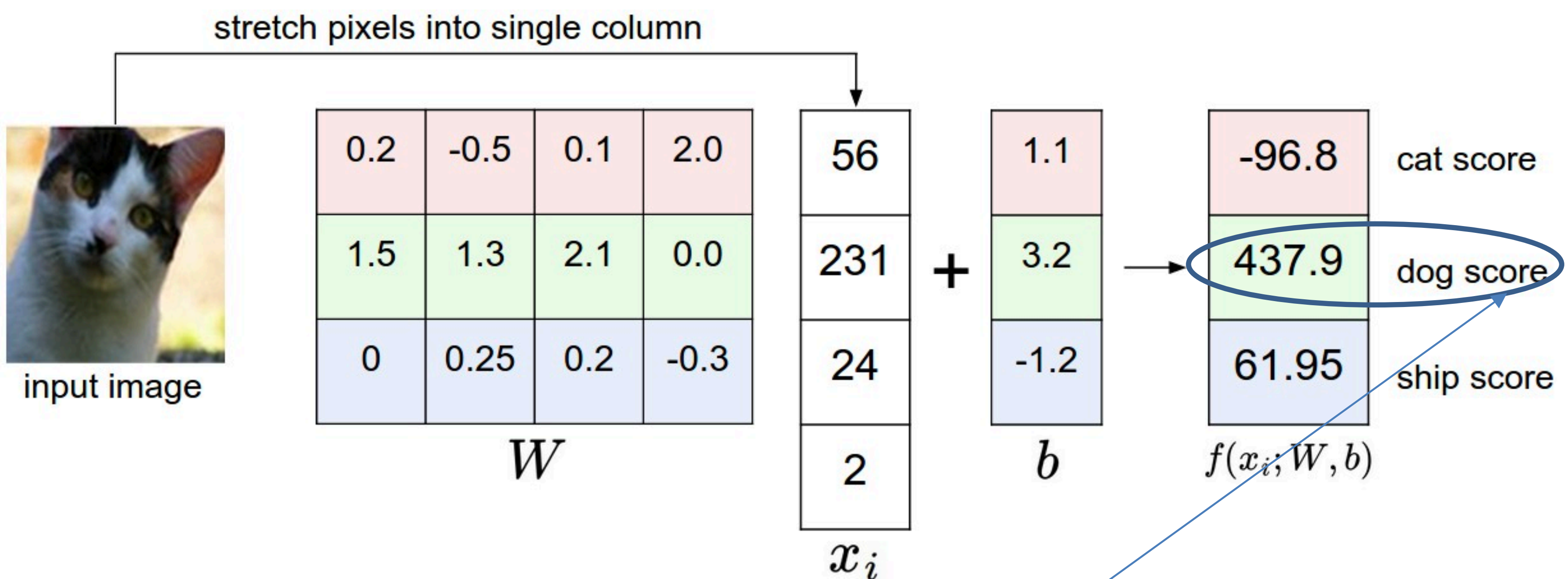
Linear score function

$$f(x_i, W, b) = Wx_i + b$$

- $x_i \in R^n$ input image
- $W \in R^{m \times n}$ weights
- $b \in R^m$ bias

Learning goal:
Learn weights and bias that minimize loss

Using score function



Predict class with highest score

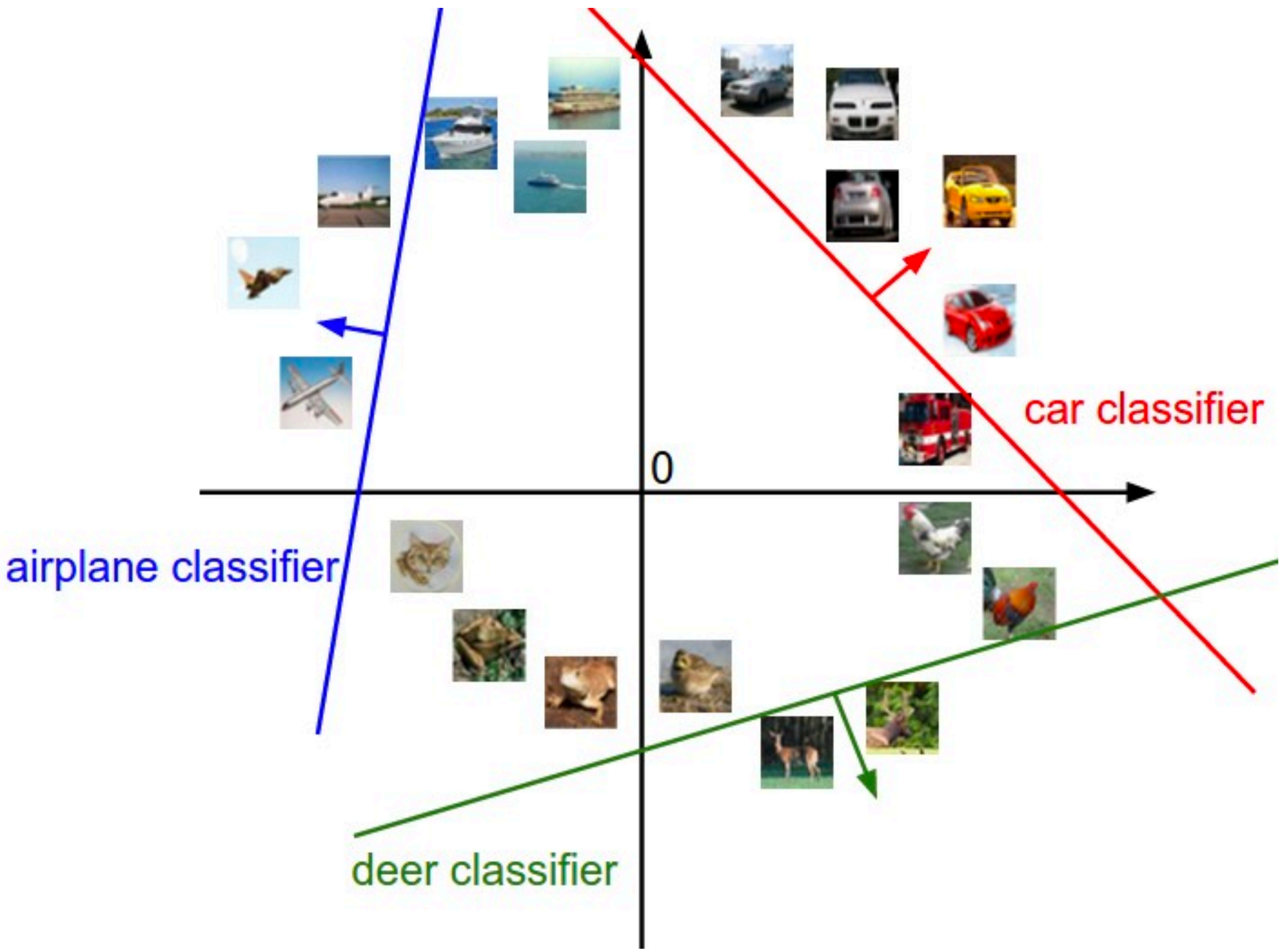
Addresses disadvantages of k-NN

- The classifier does not need to remember all of the training data and store it for future comparisons with the test data. It only needs the weights and bias.

- Classifying a test image is inexpensive since it just involves matrix multiplication. It does not require a comparison to all training images.

- Does this solve the curse of dimensionality?

Linear classifiers as hyperplanes

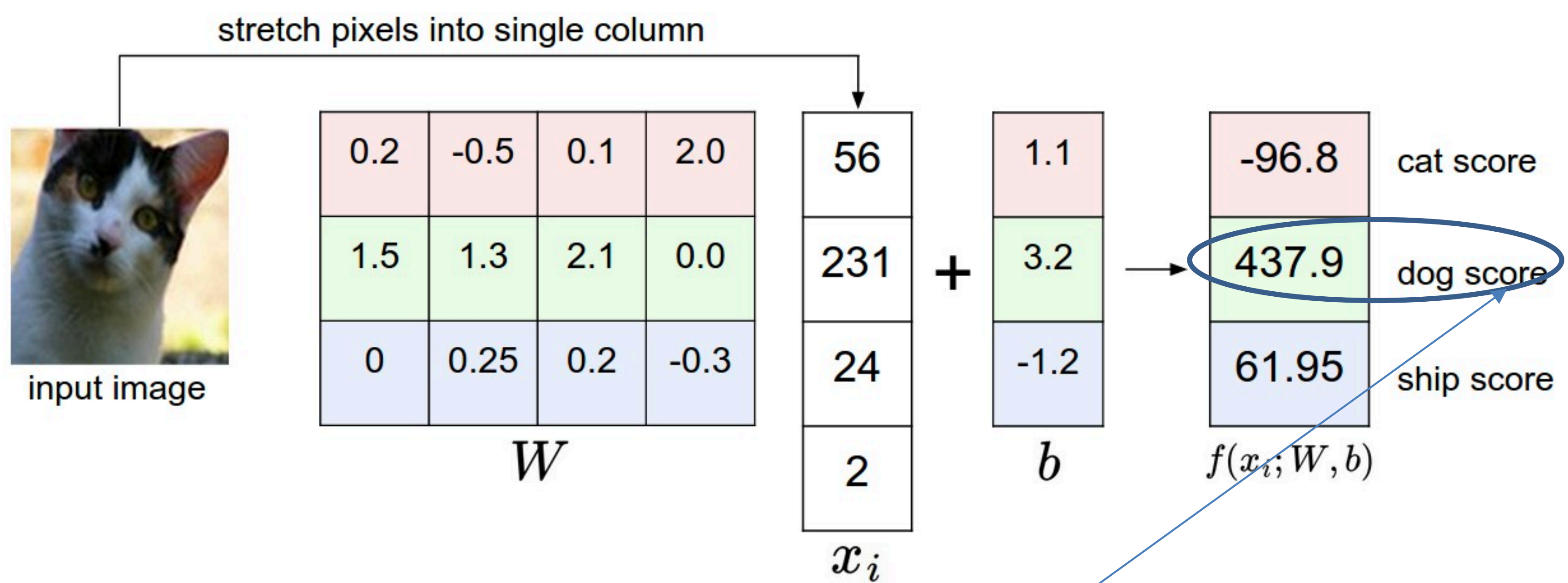


Linear classifiers as template matching

- Each row of the weight matrix is a template for a class

- The score of each class for an image is obtained by comparing each template with the image using an *inner product* (or *dot product*) one by one to find the one that “fits” best.

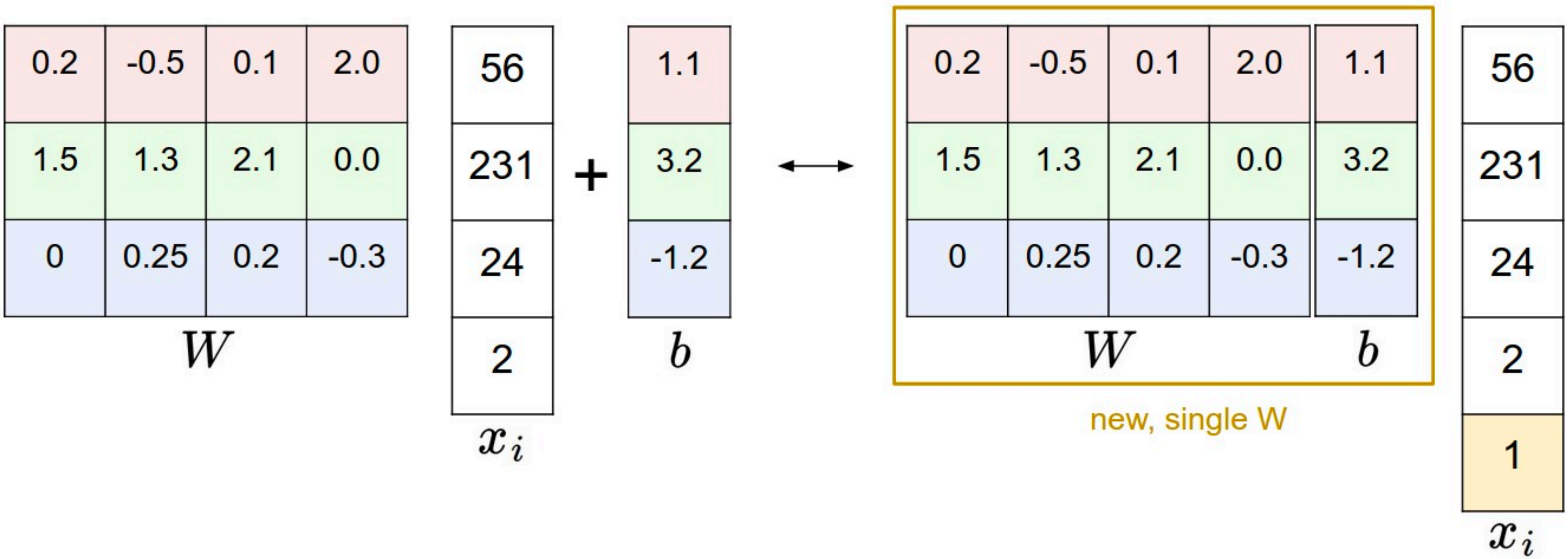
Template matching example



Predict class with highest score
(i.e., best template match)

Bias trick

$$f(x_i, W) = W x_i$$

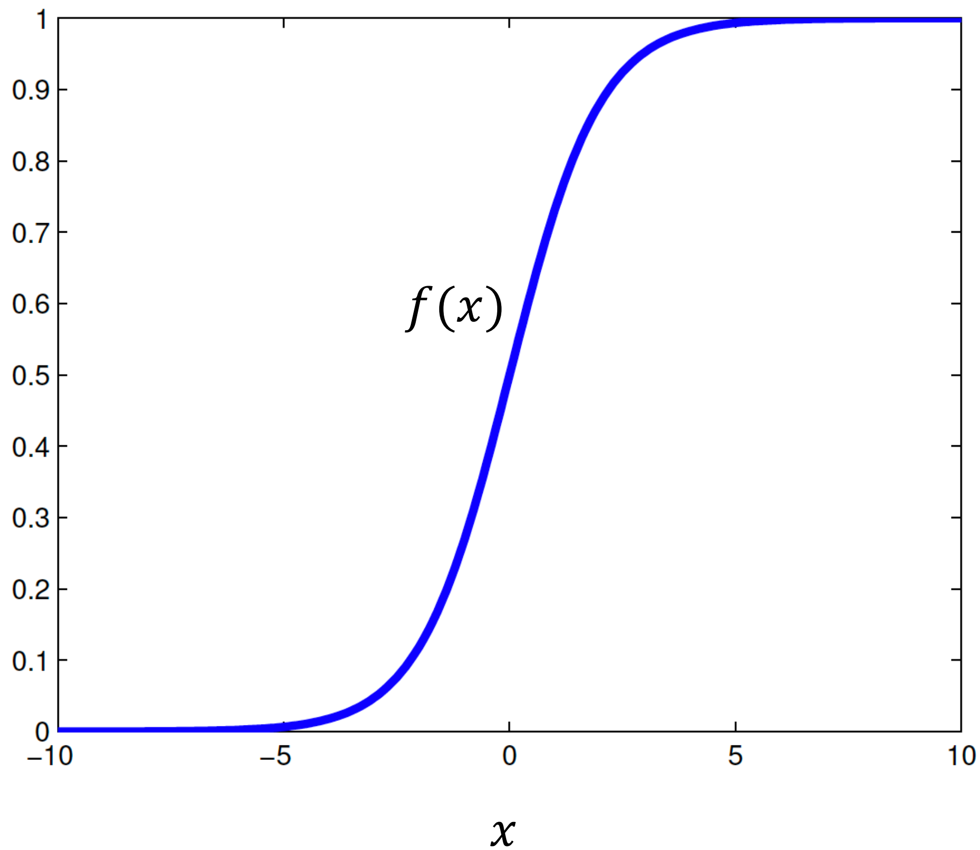


$\widetilde{W} = [W\ b]$

Linear model

- Score function
  - Maps raw data to class scores
- Loss function
  - Measures how well predicted classes agree with ground truth labels
- Learning
  - Find parameters of score function that minimize loss function

Logistic function

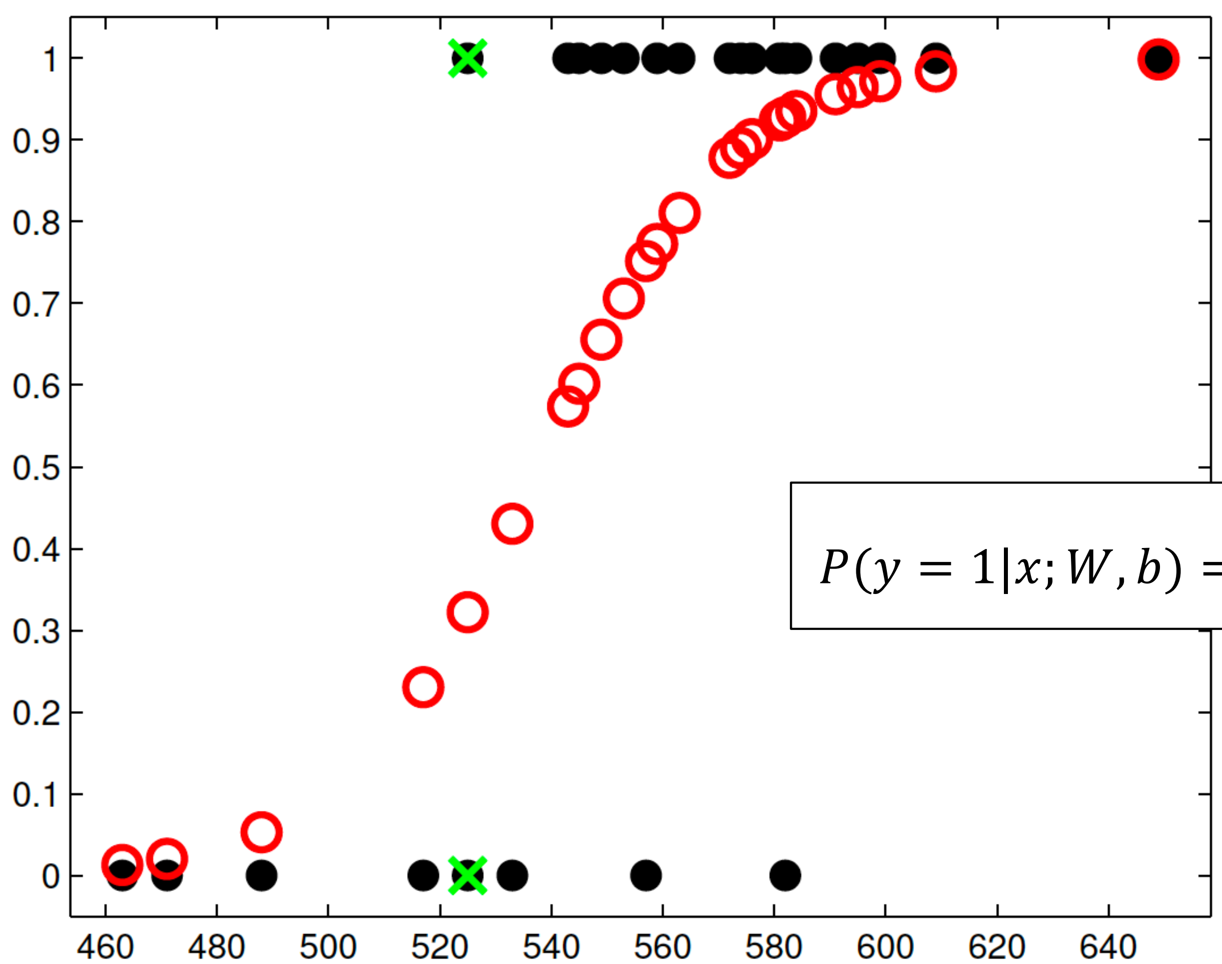


$$f(x) = \frac{e^x}{e^x + 1}$$

Figure 1.19(a) from Murphy

Logistic regression example

Pass/fail y



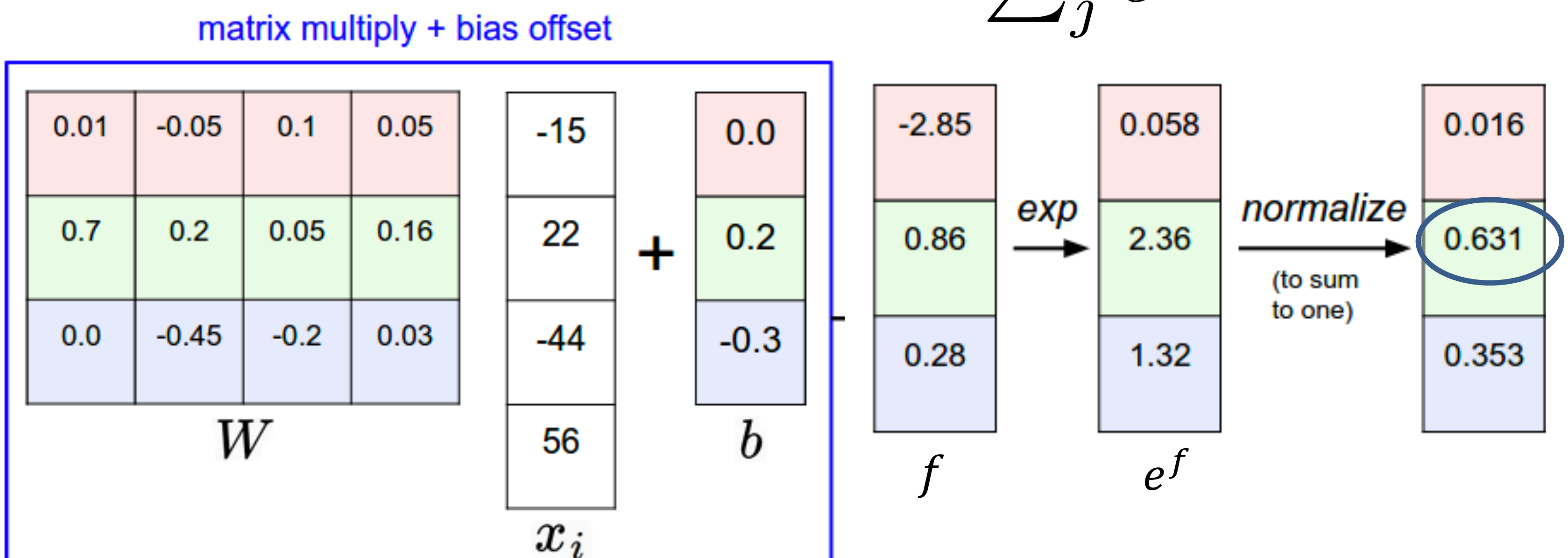
SAT scores x

$$P(y = 1|x; W, b) = \frac{e^{Wx+b}}{e^{Wx+b} + 1}$$

Figure 1.19(b) from Murphy

Softmax classifier (multiclass logistic regression)

$$P(y_i \mid x_i; W) = \frac{e^{f_{y_i}}}{\sum_j e^{f_j}}$$

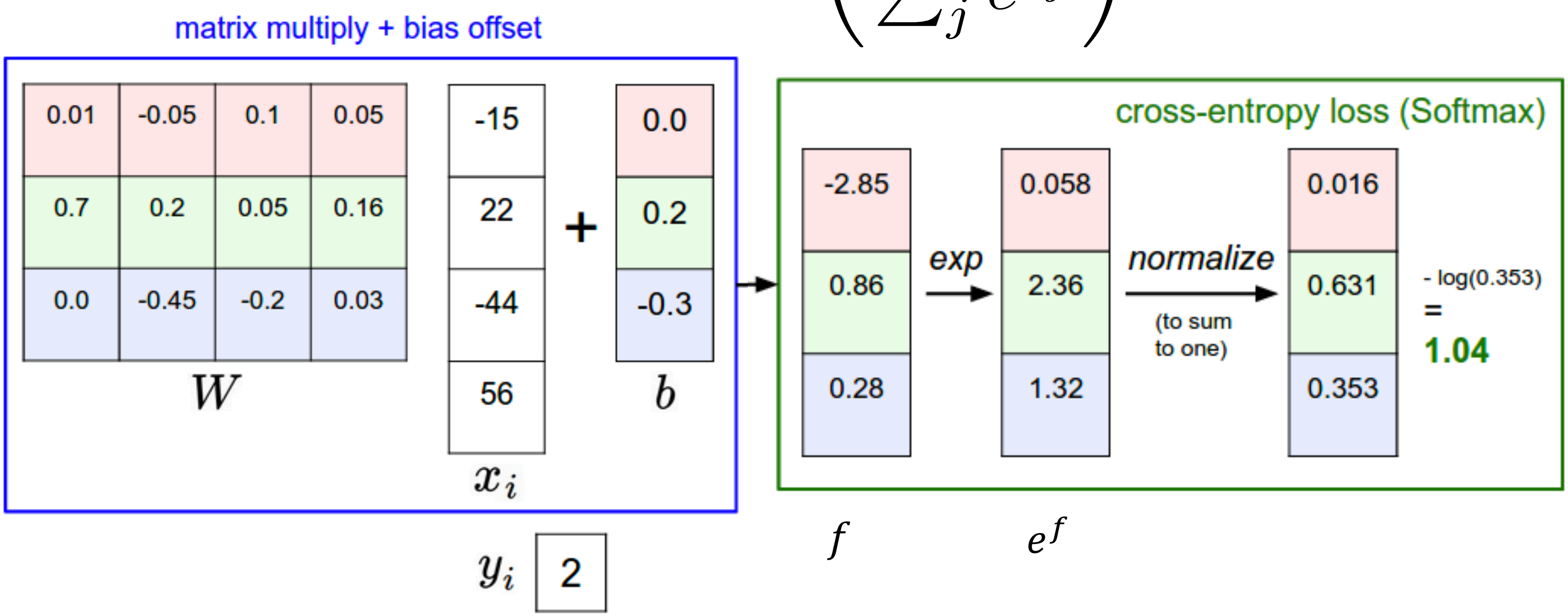


$f(x_i; W, b) = W x_i + b$ $\qquad y_i$ 2

Pick class with highest probability

Cross-entropy loss

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



Full loss for the dataset is the mean of L_i over all training examples plus a regularization term

Interpreting cross-entropy loss

The cross-entropy objective *wants* the predicted distribution to have all of its mass on the correct answer.

Information-theoretic motivation for cross-entropy loss

Entropy of a distribution p

$$H(p) = -\sum_x p(x) \log p(x) = E[-\log p(x)]$$

Q: What is the entropy of distribution with pmf

$$p = [0\ 0\ 1\ 0\ 0]$$

A: 0

Q: What is the entropy of distribution with pmf

$$p = [¼\ ¼\ ¼\ ¼]$$

A: 2

Information-theoretic motivation for cross-entropy loss

Cross-entropy between a true distribution p and an estimated distribution q

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

Q: What is the cross-entropy $H(p,p)$*?*

A: $H(p)$

Q: What if $p = [0\ 0\ 0\ 1\ 0\ 0]$*?*

A: 0

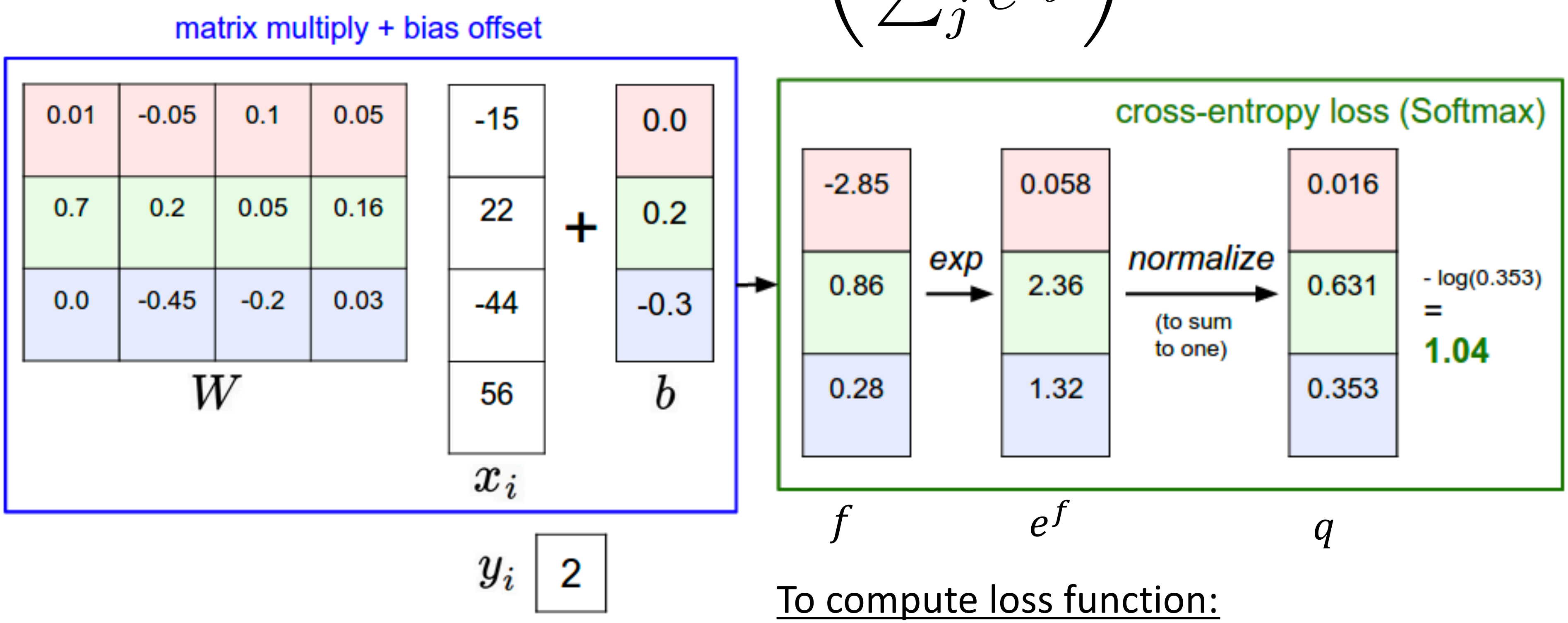
Information theory motivation for cross-entropy loss

The Softmax classifier is minimizing the cross-entropy between the estimated class probabilities ($q = e^{f_{y_i}} / \sum_j e^{f_j}$) and

the "true" distribution, which in this interpretation is the distribution where all probability mass is on the correct class ($p = [0, \ldots 1, \ldots, 0]$ contains a single 1 in the y_i position)

Cross-entropy loss

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



$(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$

<u>To compute loss function:</u>

For each sample: (x_i, dog)

1) Compute true distribution $p = [0,1,0]$
2) Compute estimated distribution q
3) Compute cross-entropy $H(p, q) \approx 1.04$

Average over all n samples

What have we done here?

- Seen how to take a score function and integrate it into a loss function
- Seen very important loss function called cross-entropy loss
  - Very widely used in neural networks
- Loss function tells us how good/bad our score function is
- What's missing?