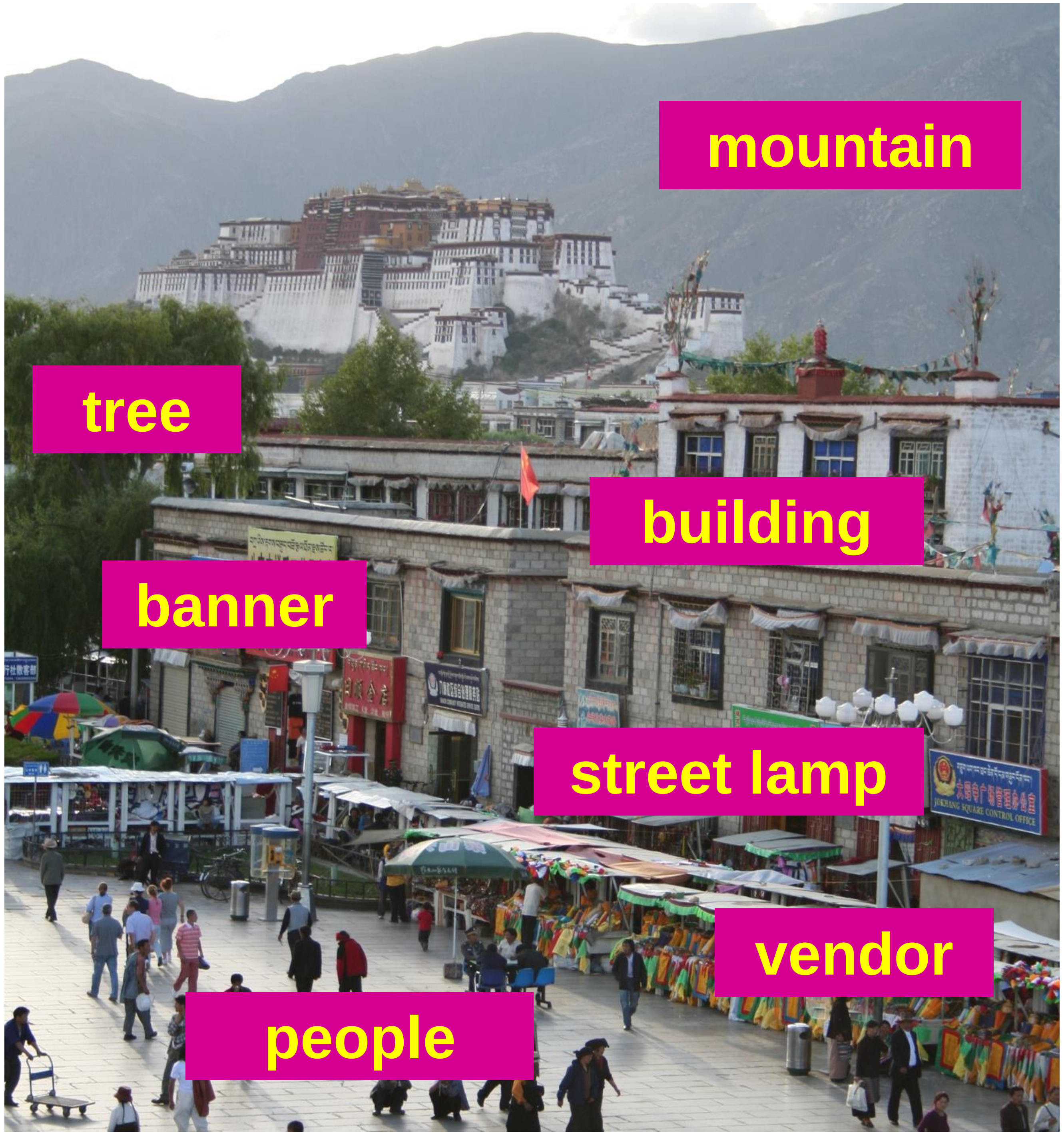
Introduction to semantic vision



16-385 Computer Vision
Spring 2018, Lecture 17

http://www.cs.cmu.edu/~16385/

Course announcements

- Homework 5 will be posted tonight.
- Yannis has extra office hours today, 4-8 pm.
- Yannis' office hours on Friday will be covered by Neeraj.
- How many of you went to Matthias Niessner's talk today?
- Talk: Angela Dai, "Understanding 3D Scans," Thursday noon, GHC 6115.

Overview of today's lecture

Leftover from last lecture: radiometric calibration.

New in this lecture:

- Introduction to semantic vision.
- Image classification.
- Bag-of-words.
- K-means clustering.
- Classification.
- K nearest neighbors.
- Naïve Bayes.
- Support vector machine.

Slide credits

Most of these slides were adapted from:

- Kris Kitani (16-385, Spring 2017).
- Noah Snavely (Cornell University).
- Fei-Fei Li (Stanford University).

Course overview

1. Image processing. ⟵ Lectures 1 – 7
   See also 18-793: Image and Video Processing

2. Geometry-based vision. ⟵ Lectures 7 – 12
   See also 16-822: Geometry-based Methods in Vision

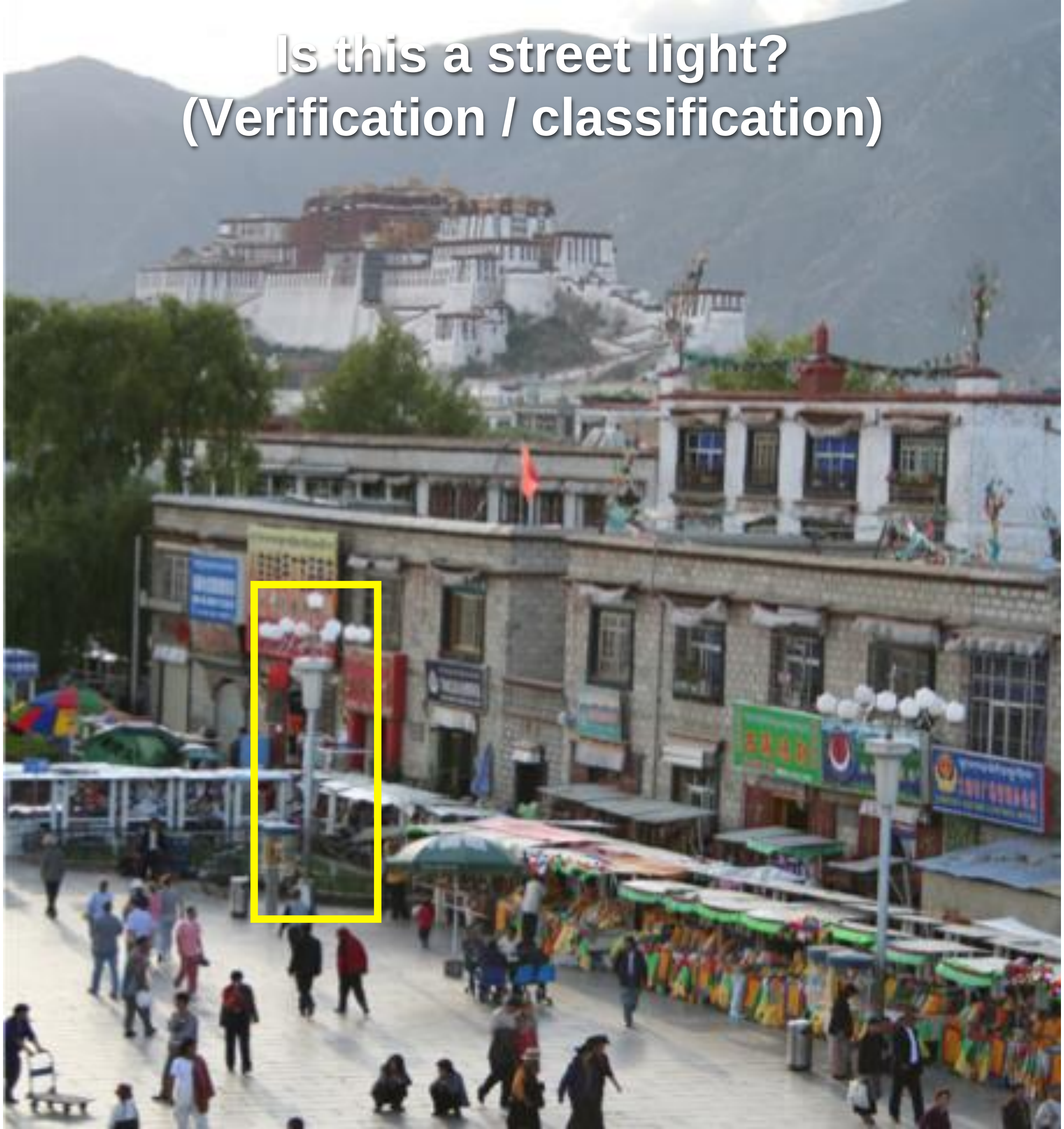
3. Physics-based vision. ⟵ Lectures 13 – 16
   See also 16-823: Physics-based Methods in Vision
   See also 15-463: Computational Photography

4. Semantic vision. ⟵ We are starting this part now

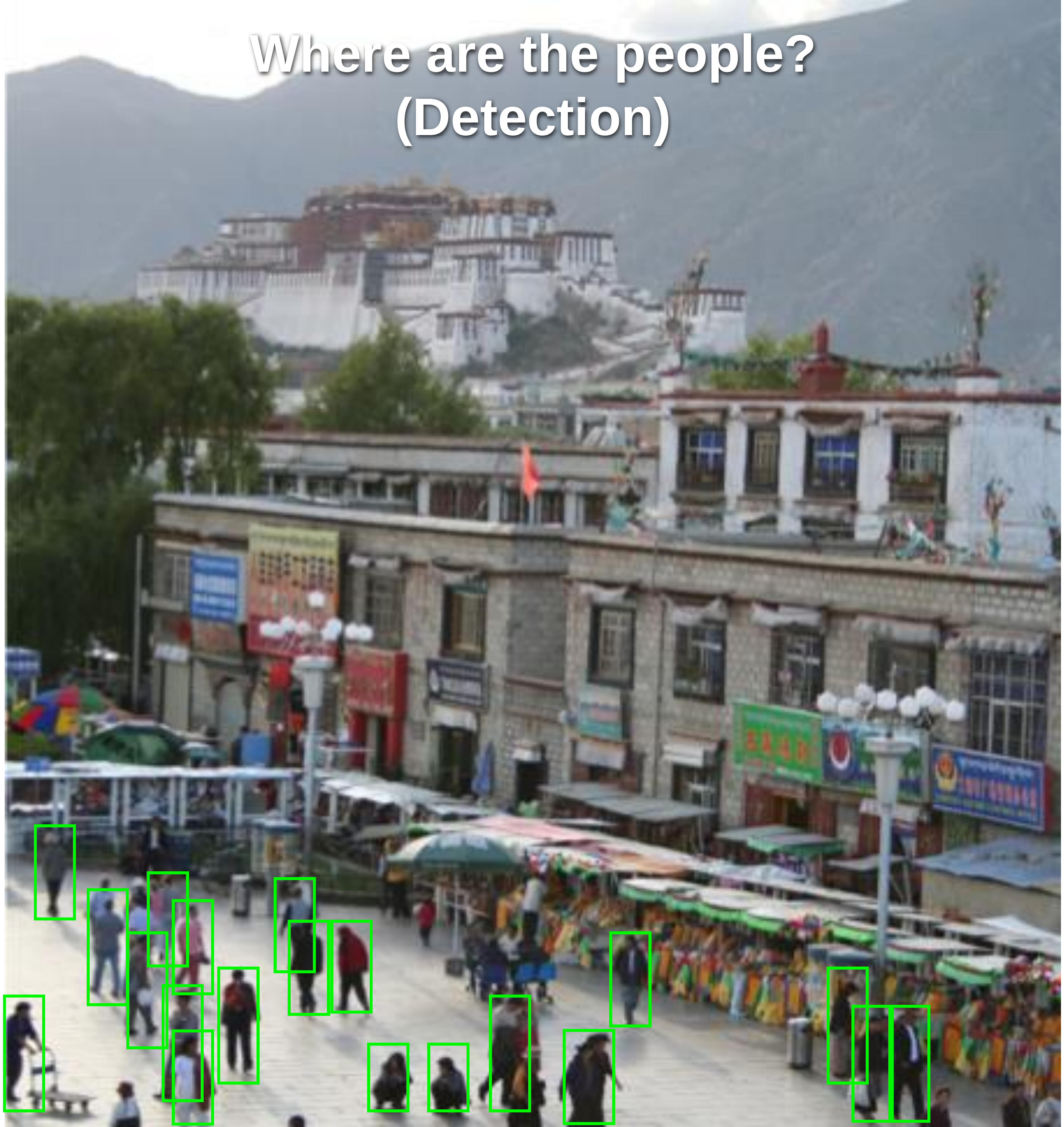
5. Dealing with motion.

What do we mean by ‘semantic vision’?

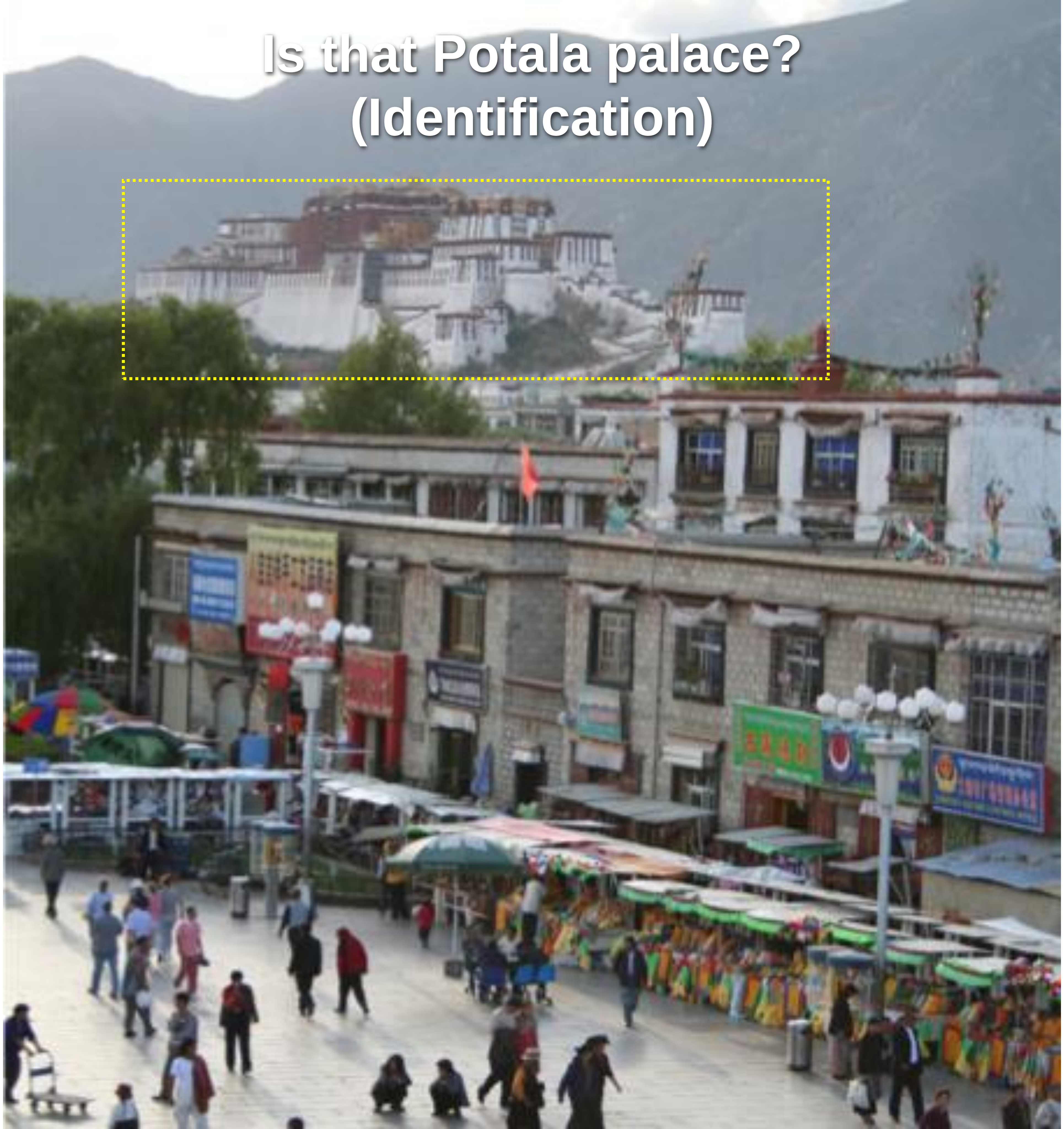
Is this a street light?
(Verification / classification)



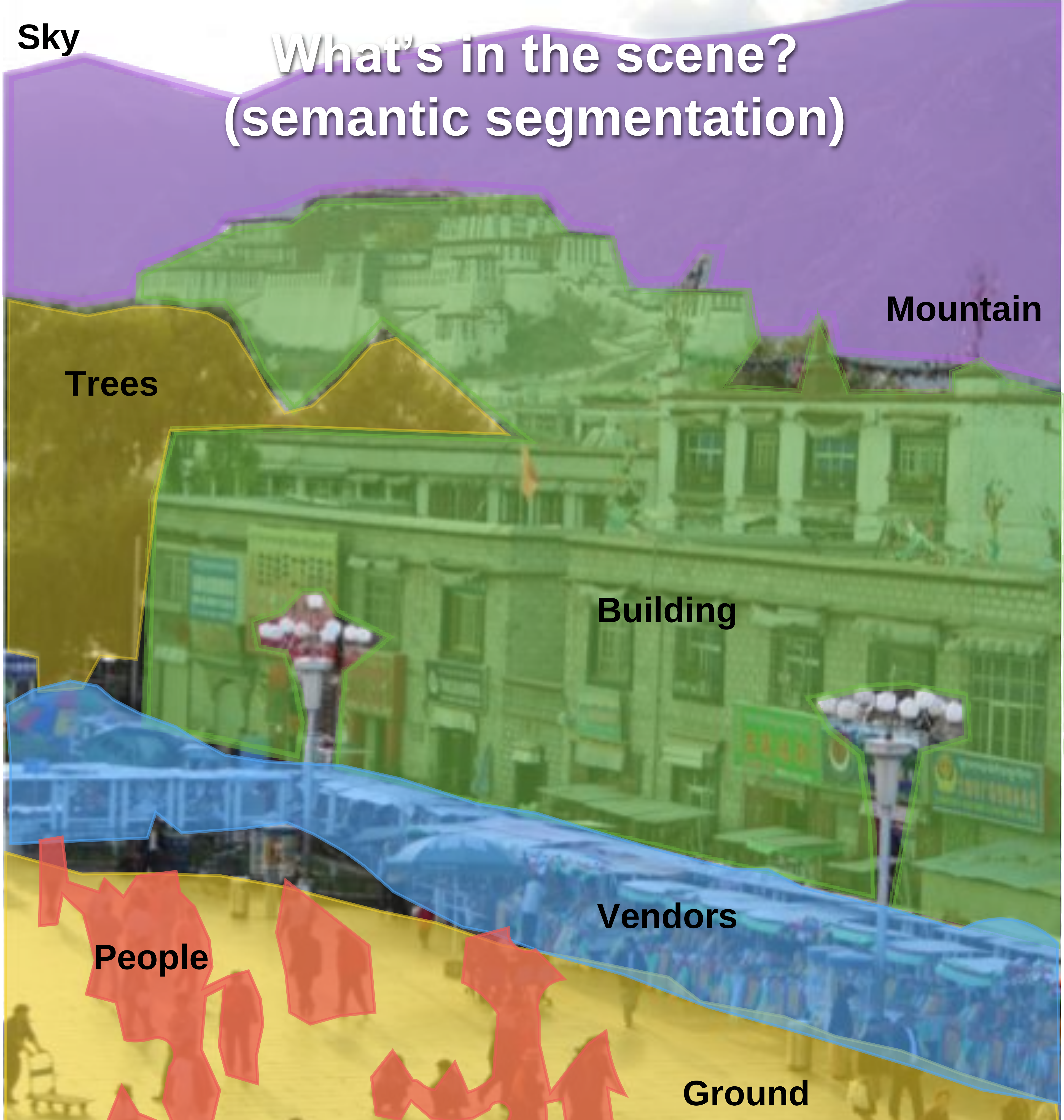
Where are the people?
(Detection)



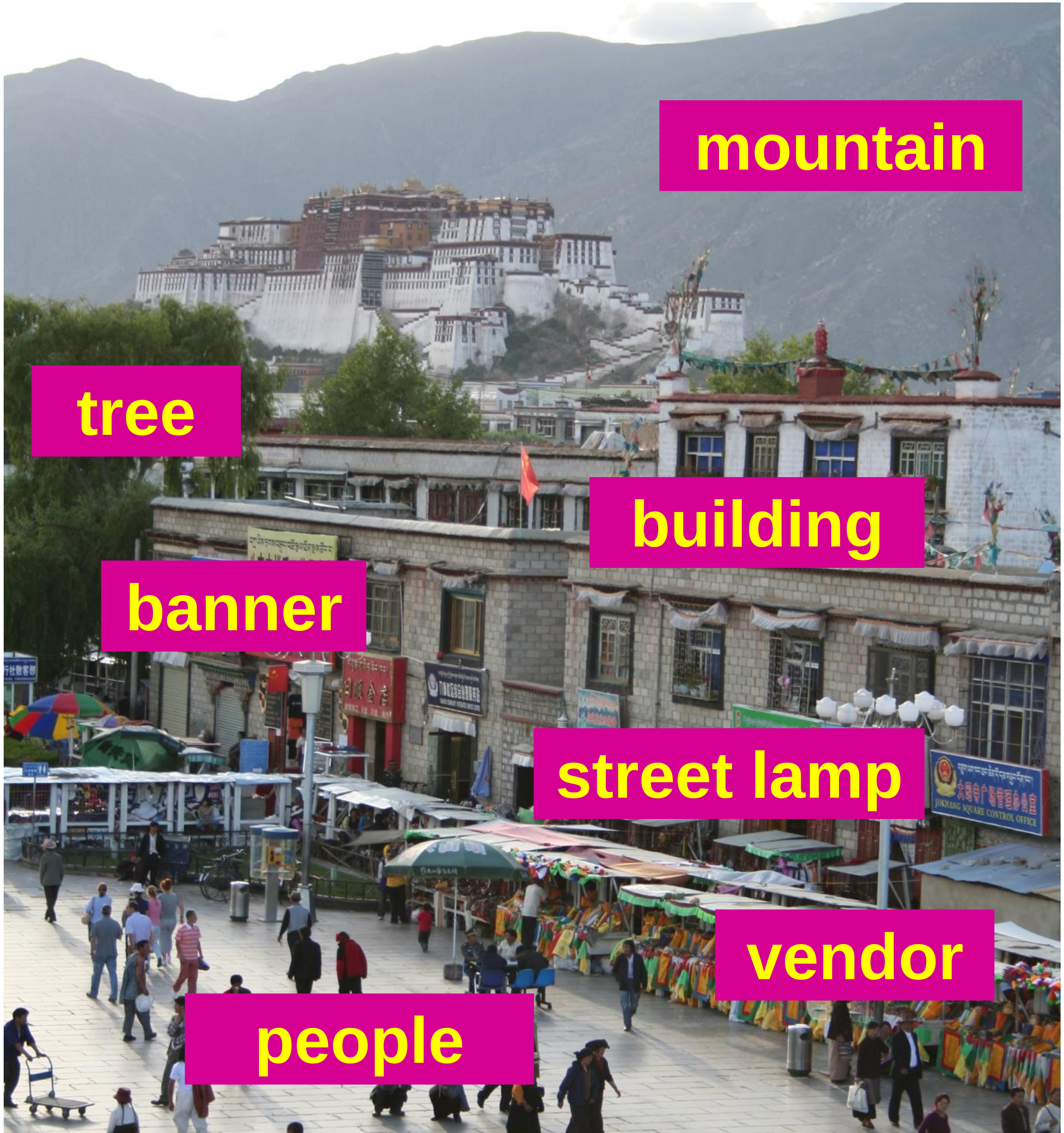
Is that Potala palace? (Identification)



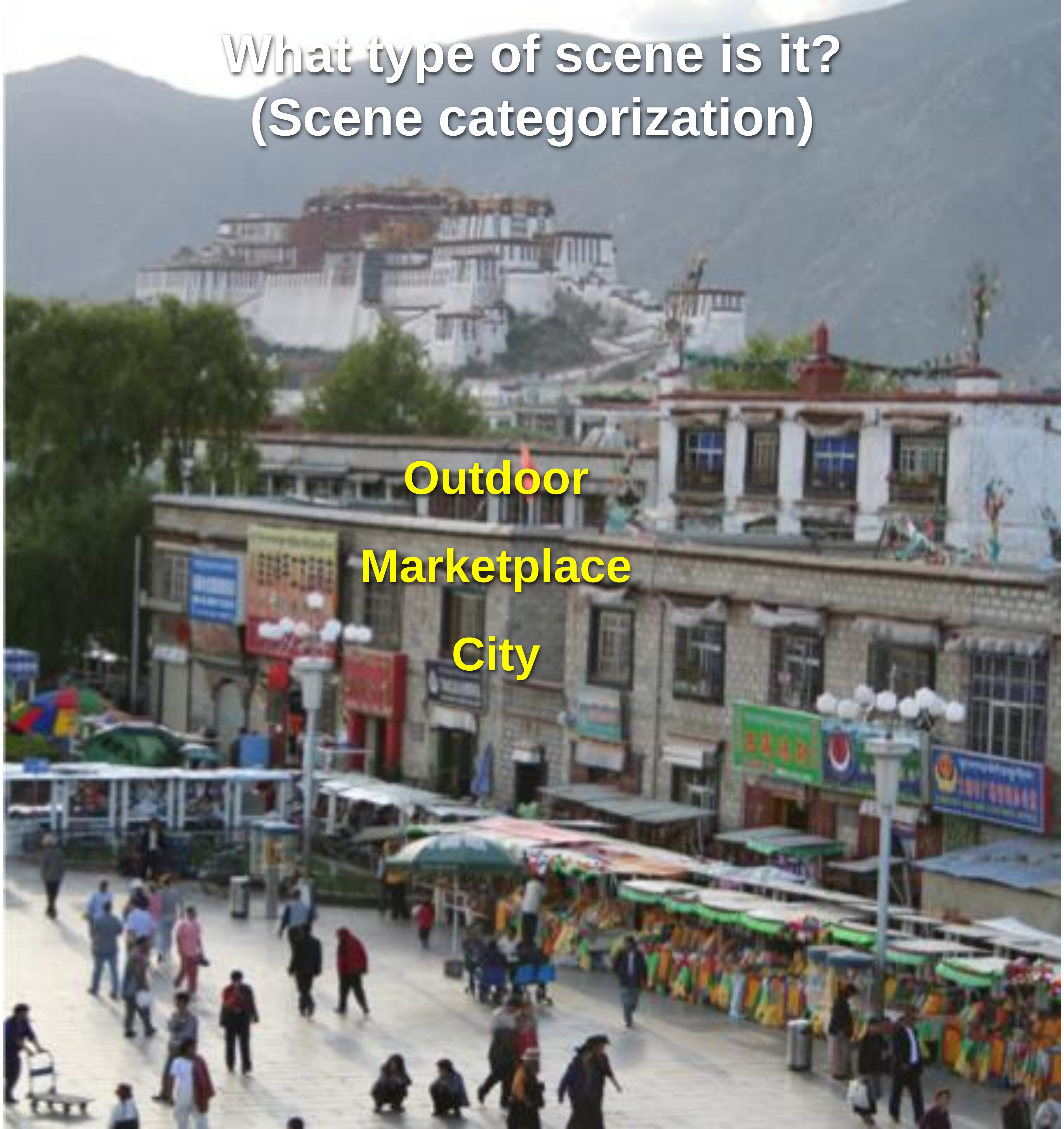
What's in the scene?
(semantic segmentation)



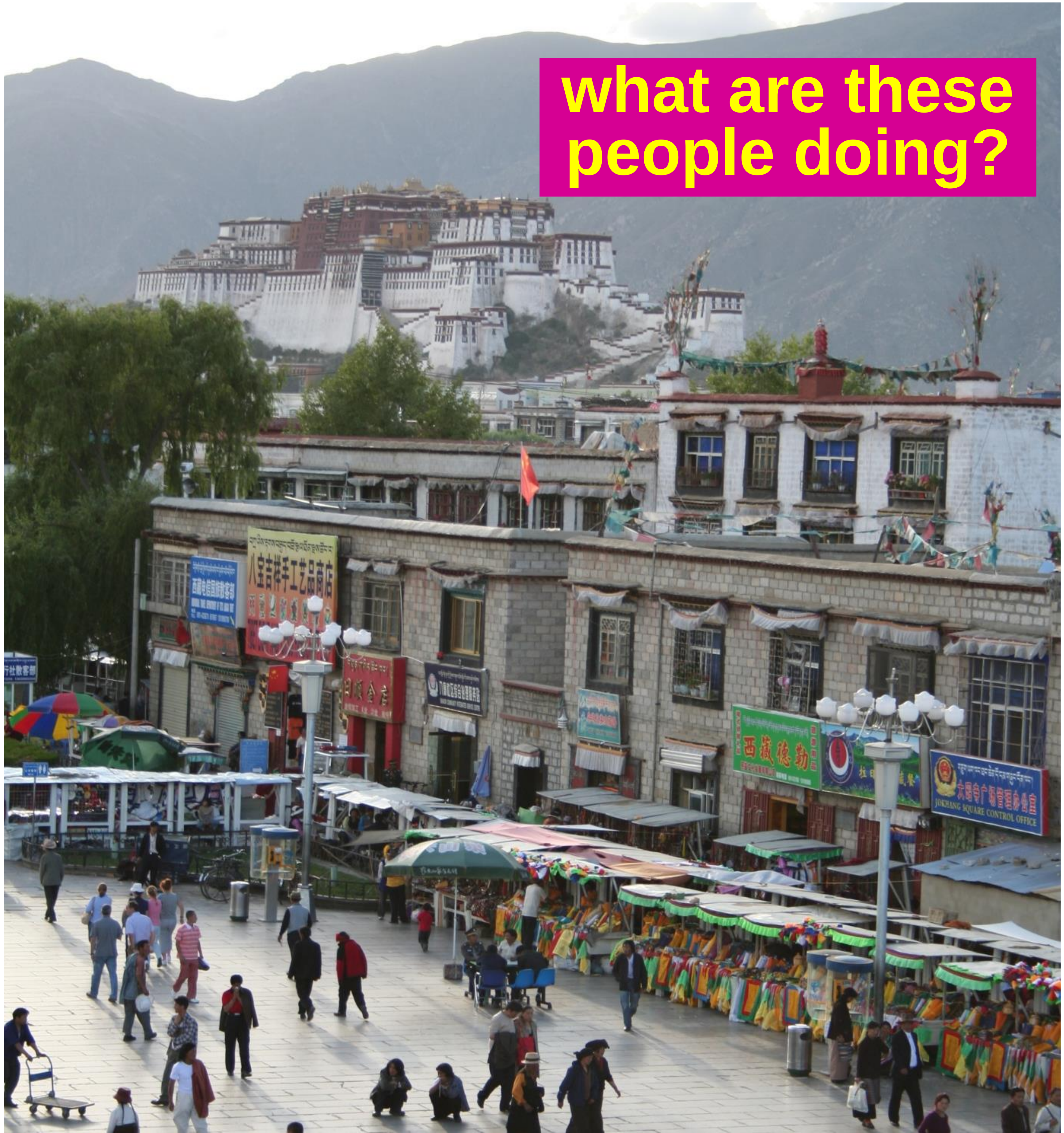
Object categorization



What type of scene is it?
(Scene categorization)

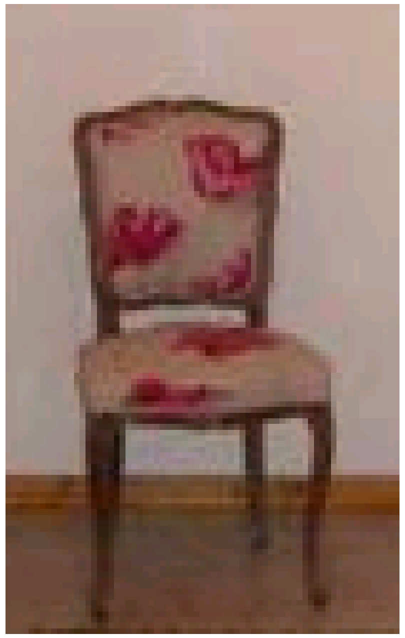


Activity / Event Recognition



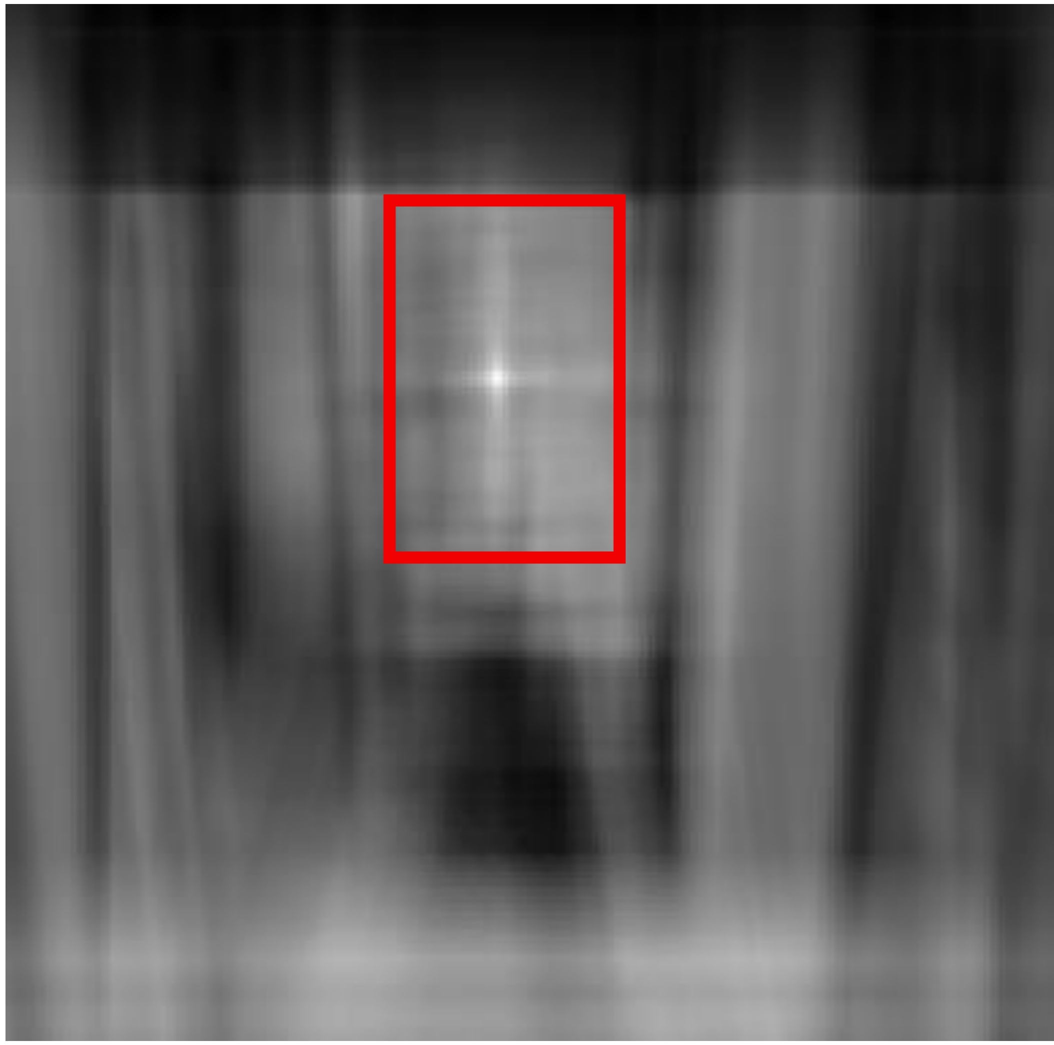
Object recognition
Is it really so hard?

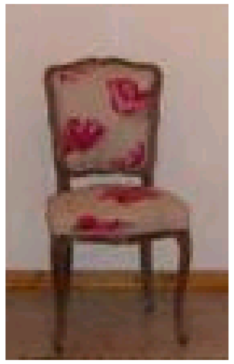
This is a chair



Find the chair in this image

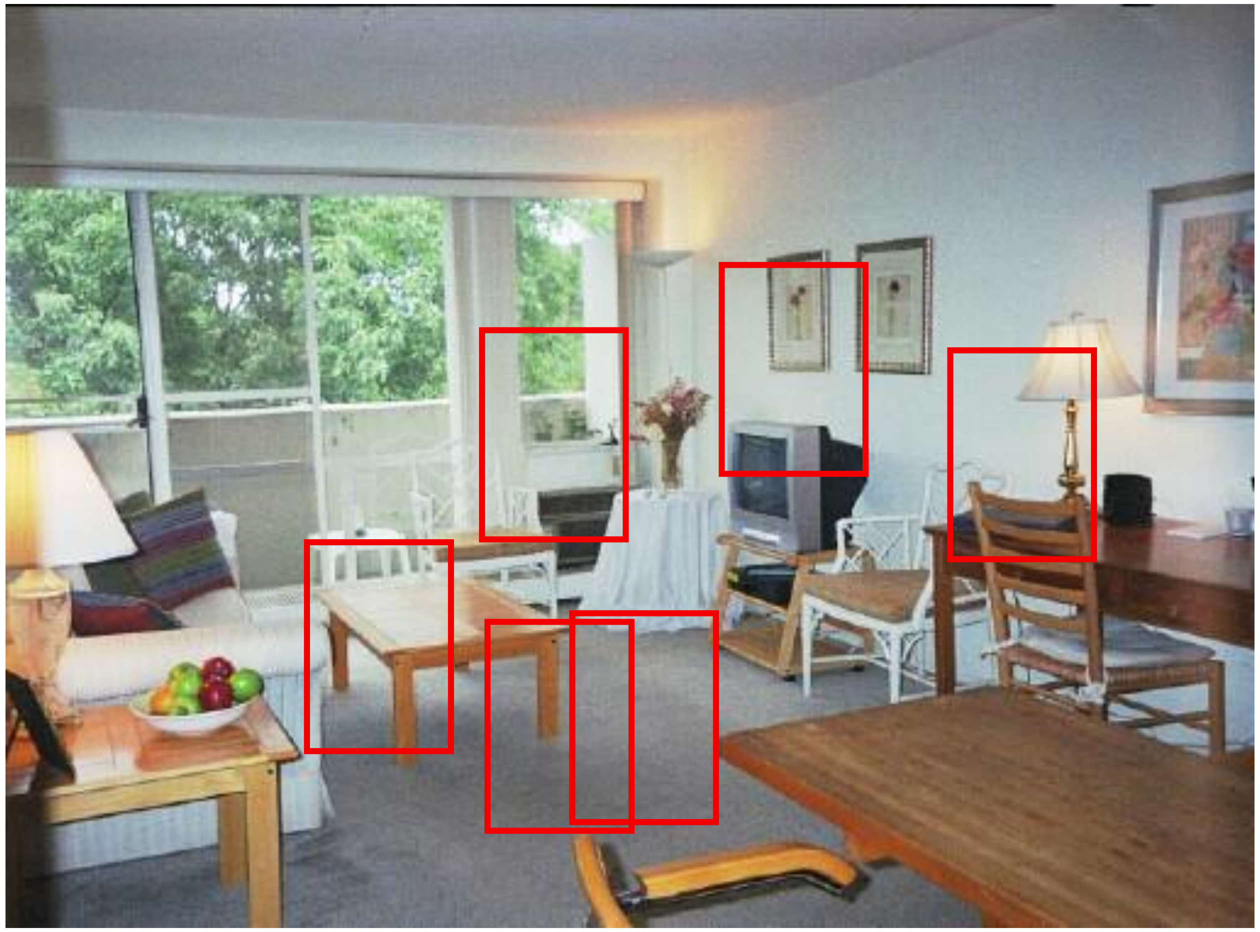
Output of normalized correlation

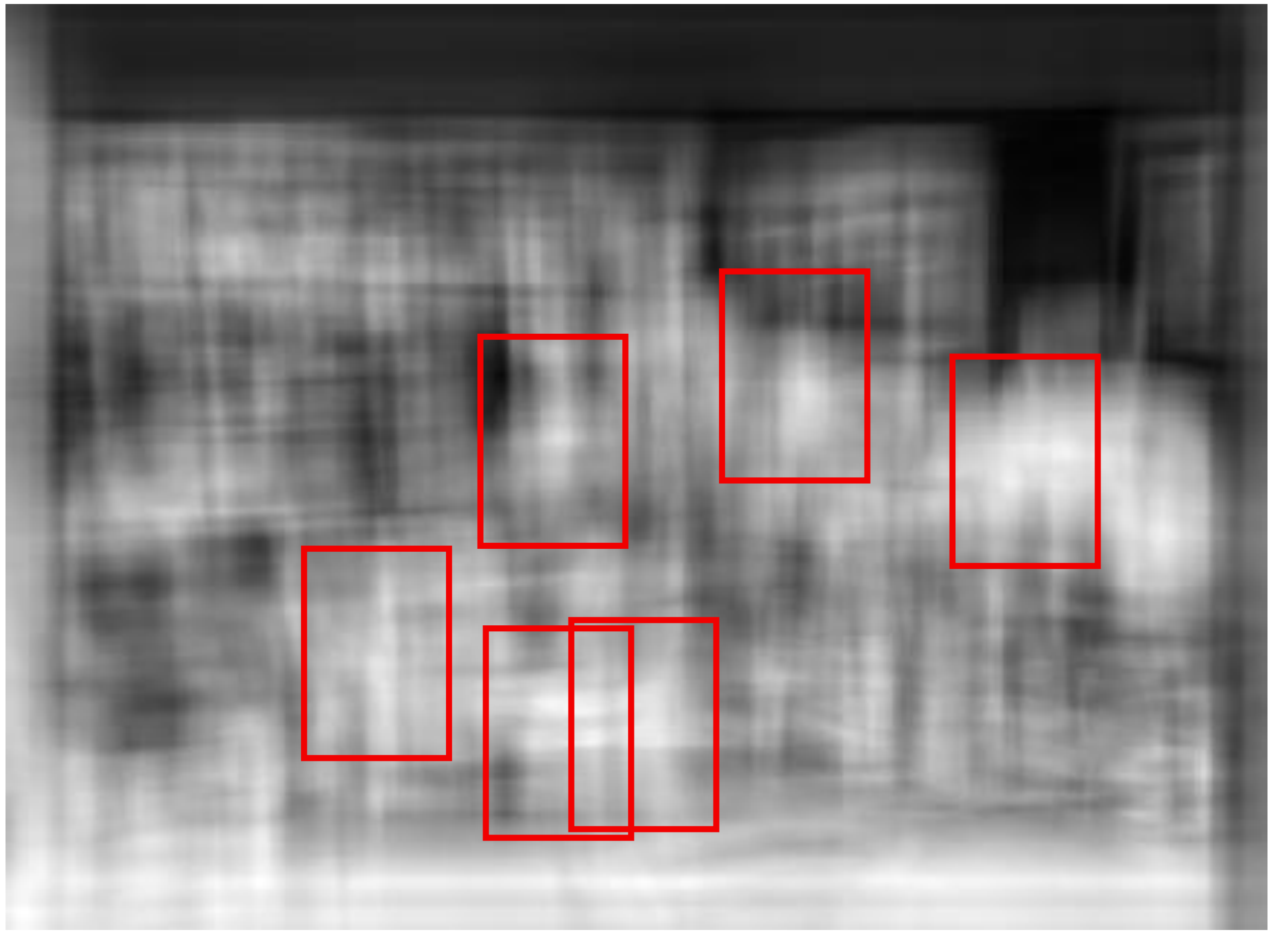




Object recognition
Is it really so hard?

Find the chair in this image





Pretty much garbage
Simple template matching is not going to make it

A "popular method is that of template matching, by point to point correlation of a model pattern with the image pattern. These techniques are inadequate for three-dimensional scene analysis for many reasons, such as occlusion, changes in viewing angle, and articulation of parts." Nivatia & Binford, 1977.

And it can get a lot harder

Brady, M. J., & Kersten, D. (2003). Bootstrapped learning of novel objects. J Vis, 3(6), 413-422

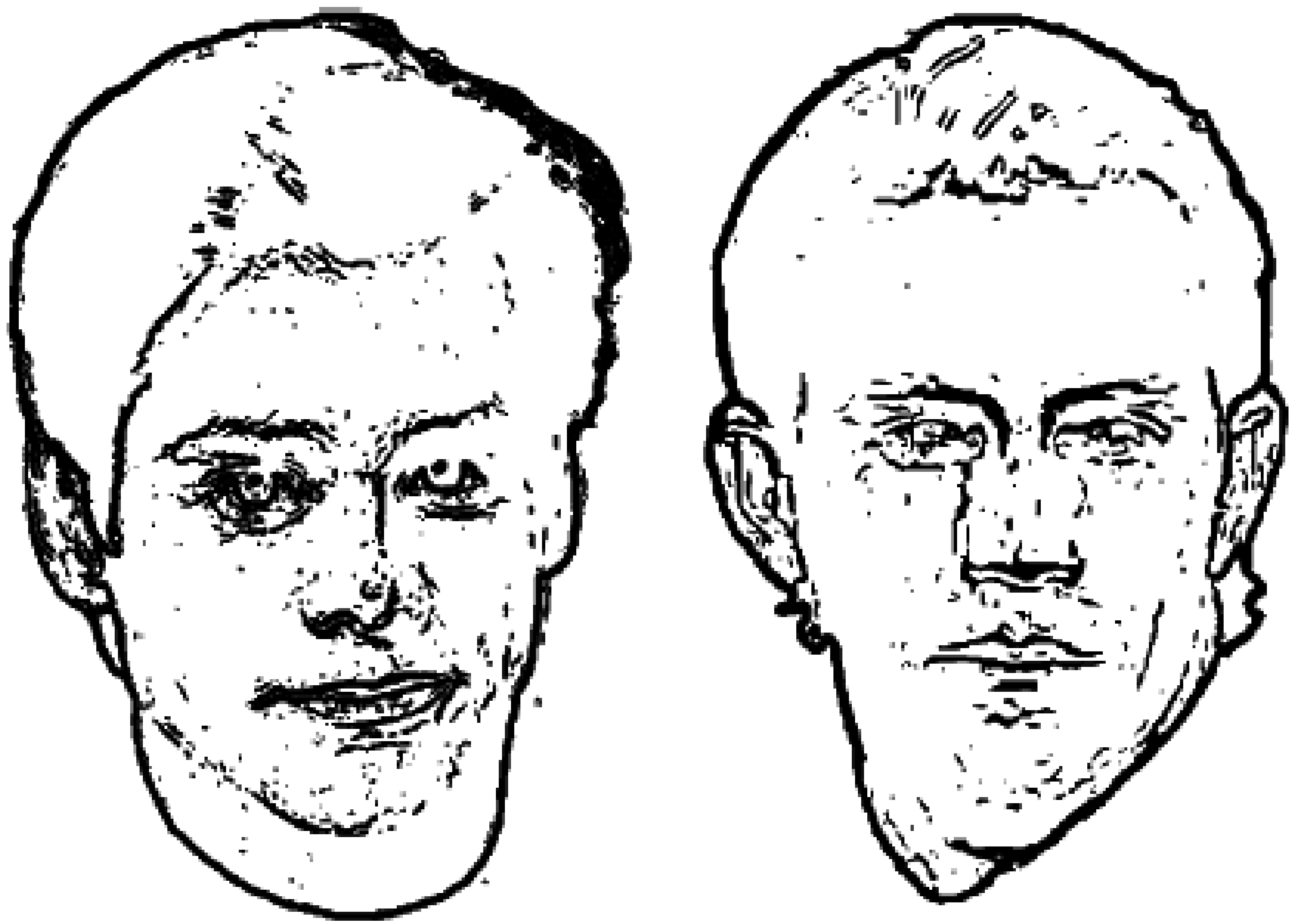
How do humans do recognition?

- We don’t completely know yet
- But we have some experimental observations.

Observation 1

- We can recognize familiar faces even in low-resolution images

Observation 2:

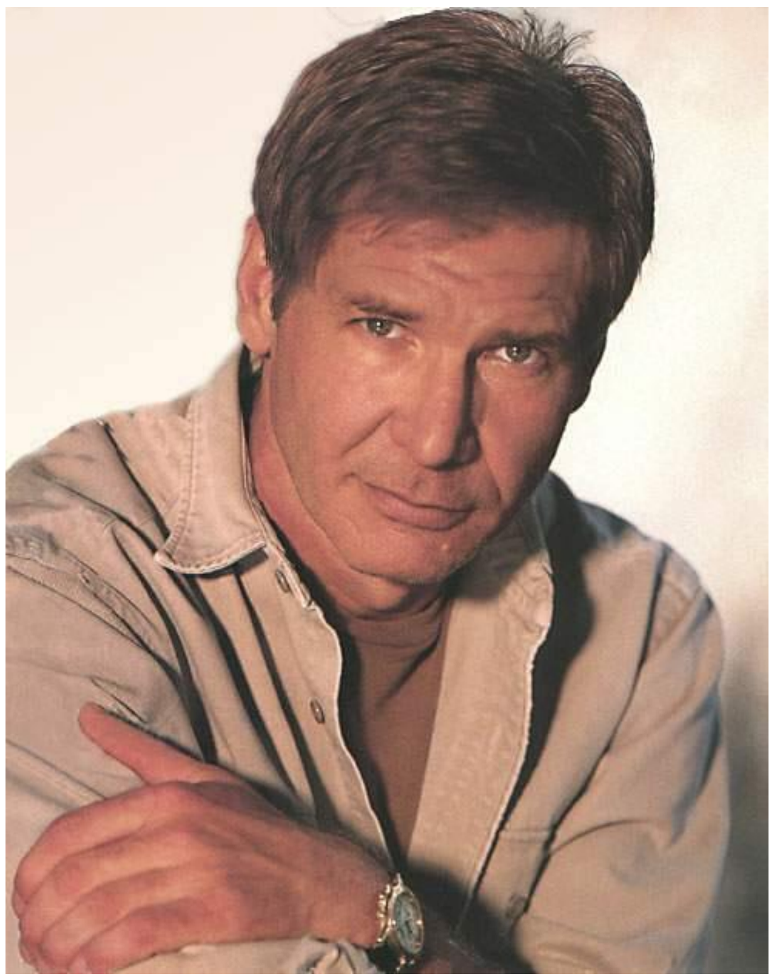


Jim Carrey

Kevin Costner

- High frequency information is not enough

What is the single most important facial features for recognition?



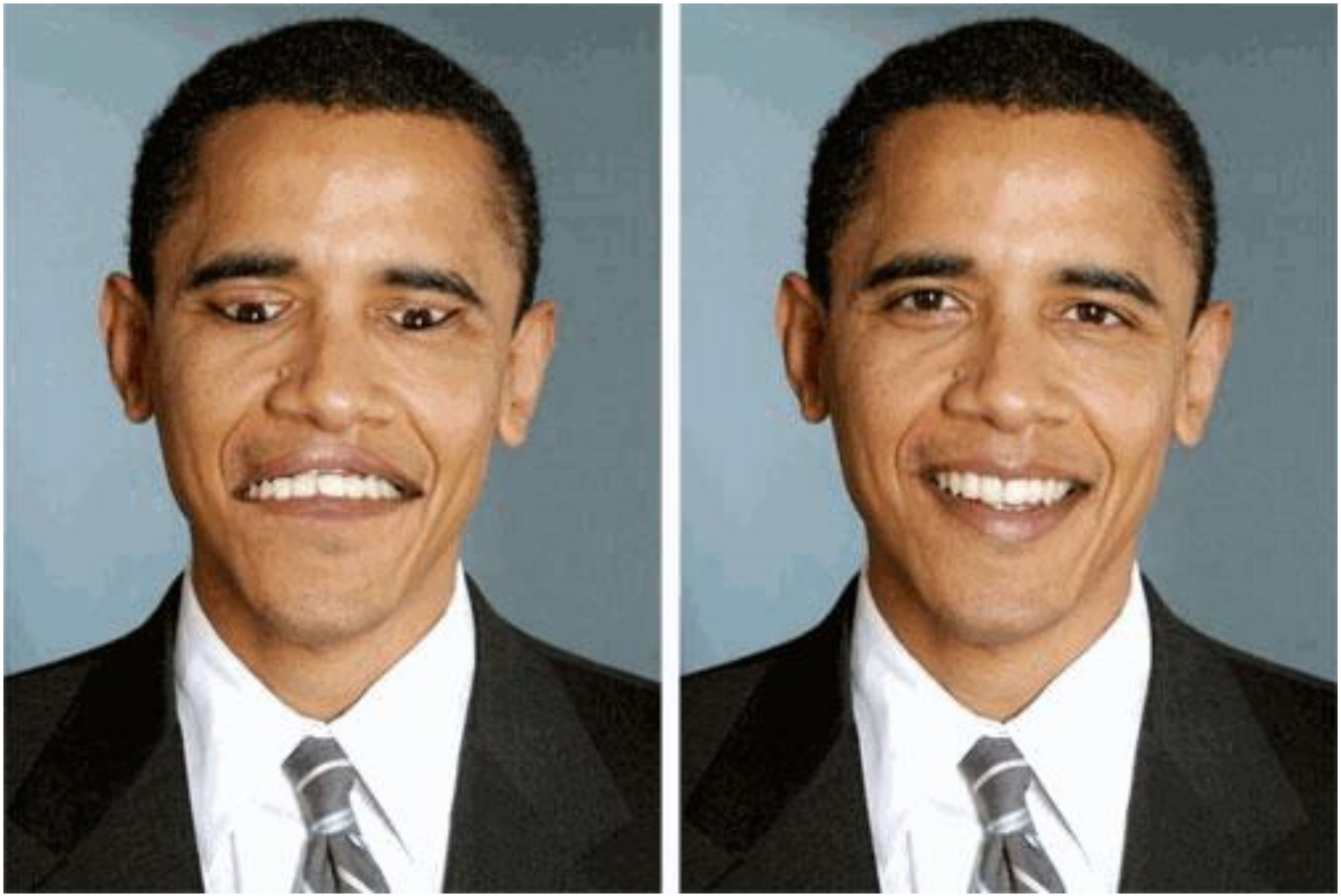
What is the single most important facial features for recognition?

Observation 4:

- Image Warping is OK

Spatial configuration matters too

Spatial configuration matters too

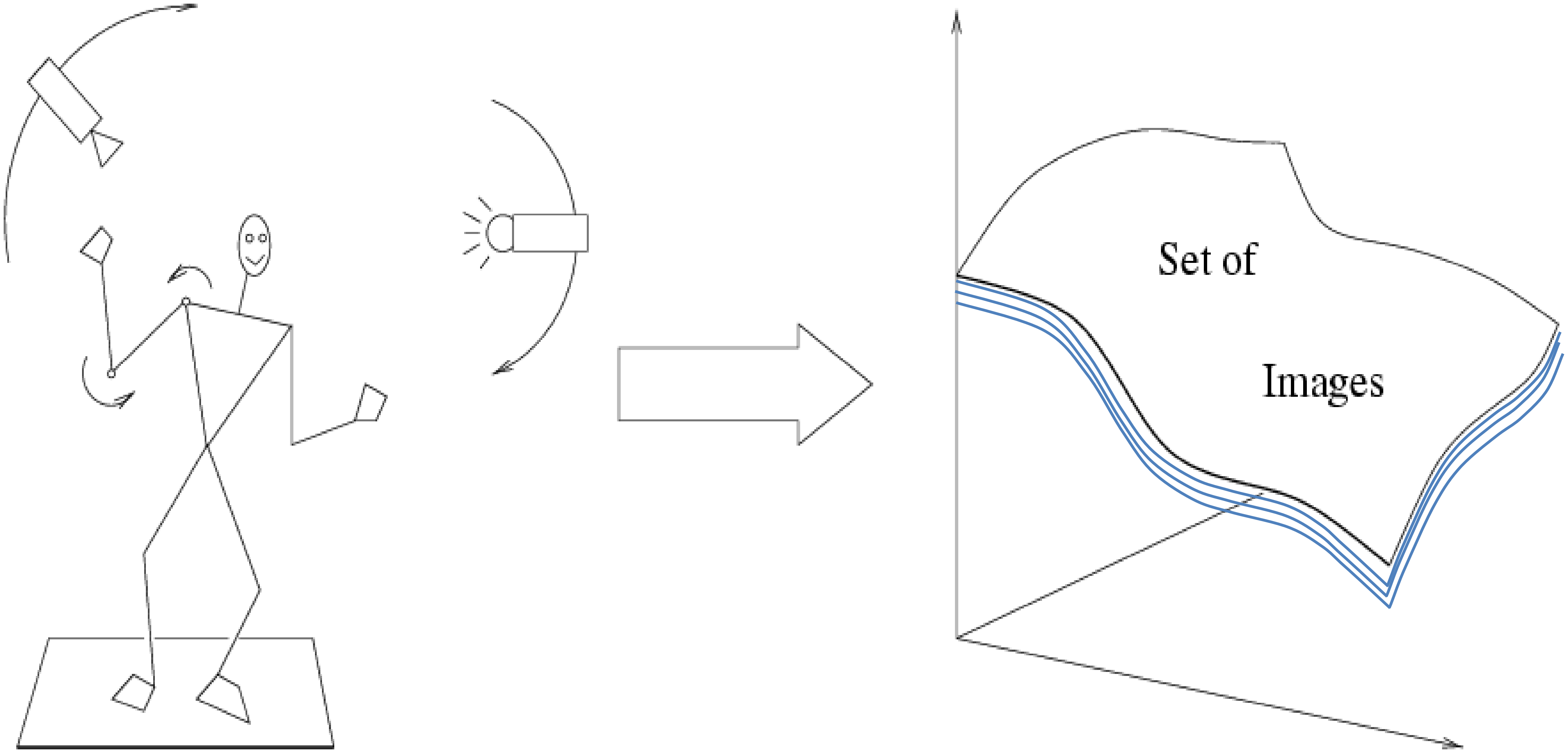


The list goes on

Face Recognition by Humans: Nineteen Results All Computer Vision Researchers Should Know About

- http://web.mit.edu/bcs/sinha/papers/19results_sinha_etal.pdf

Why is this hard?



Variability: Camera position
Illumination
Shape parameters

How many object categories are there?

Challenge: variable viewpoint

Michelangelo 1475-1564

Challenge: variable illumination

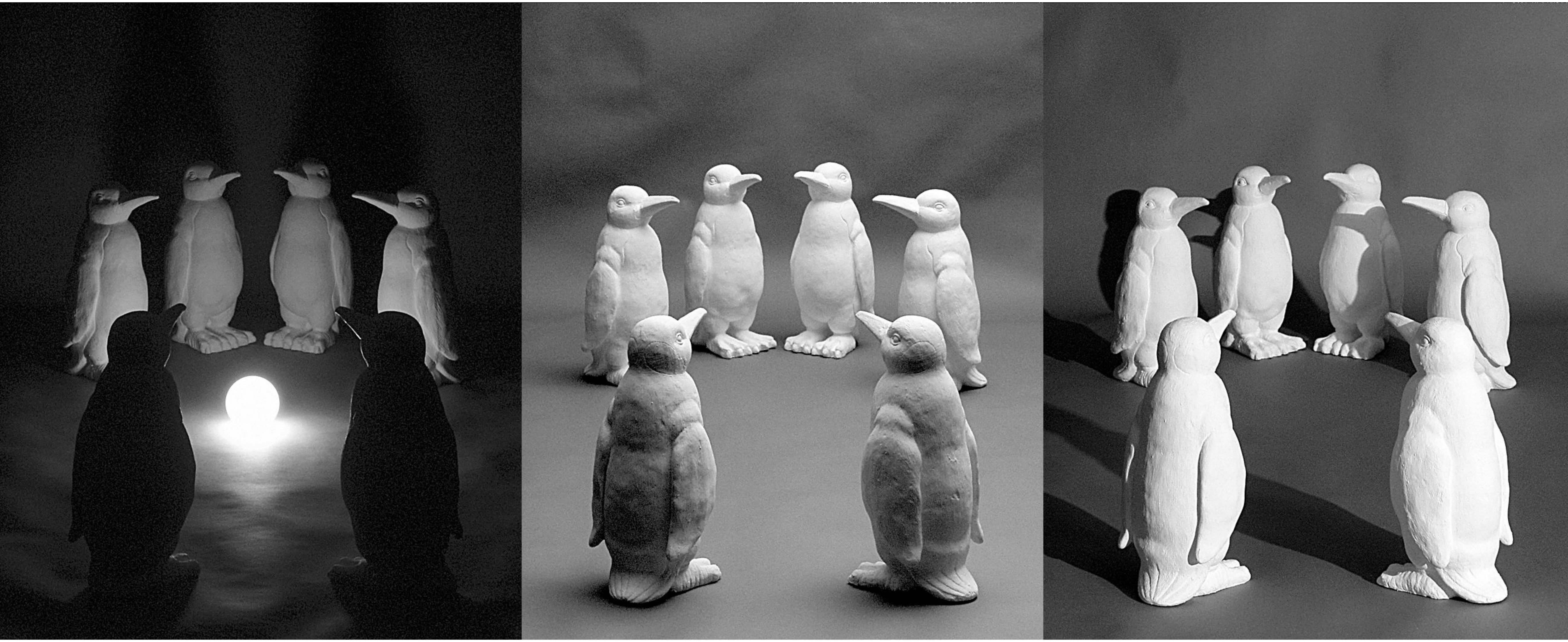


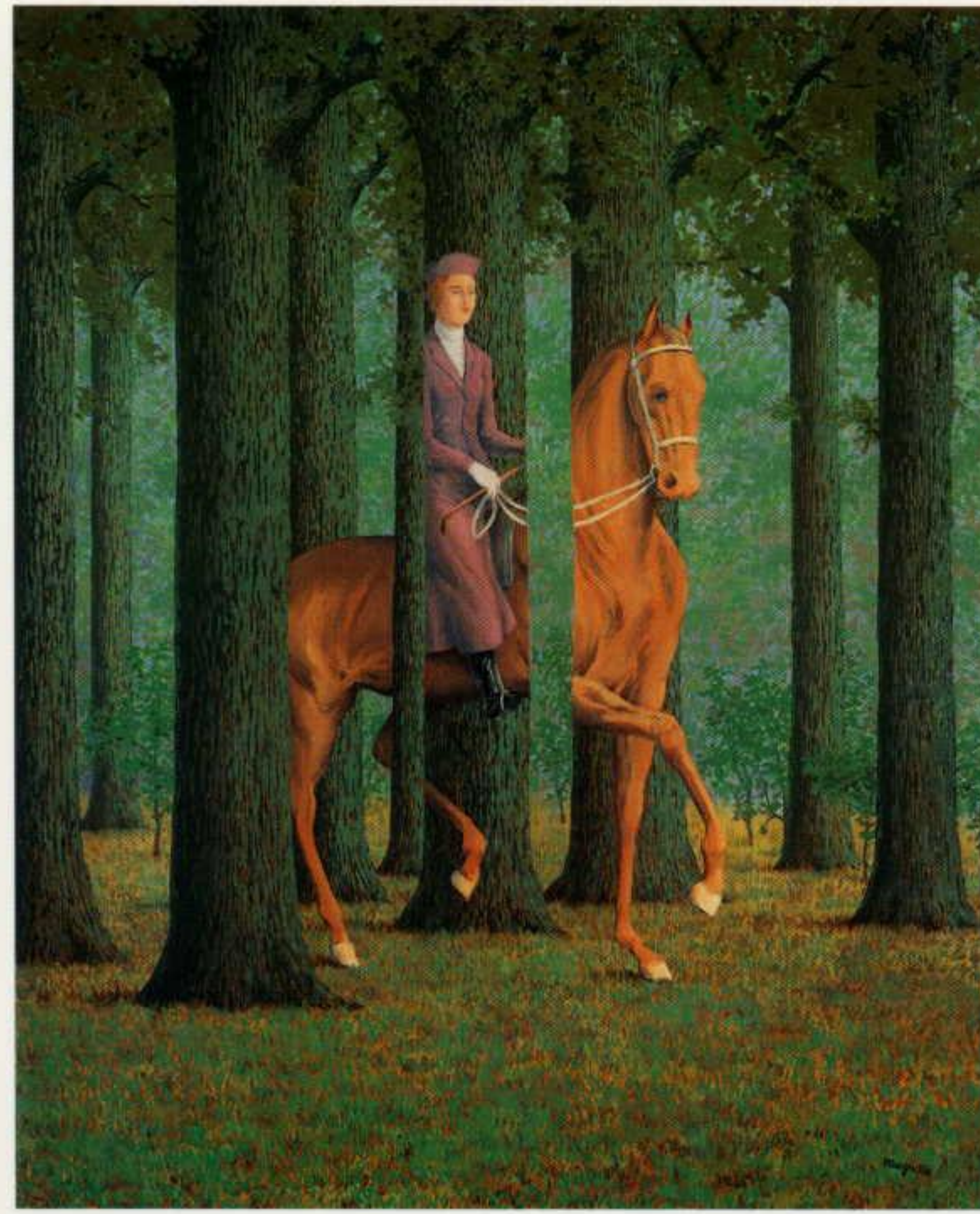
image credit: J. Koenderink

Challenge: scale

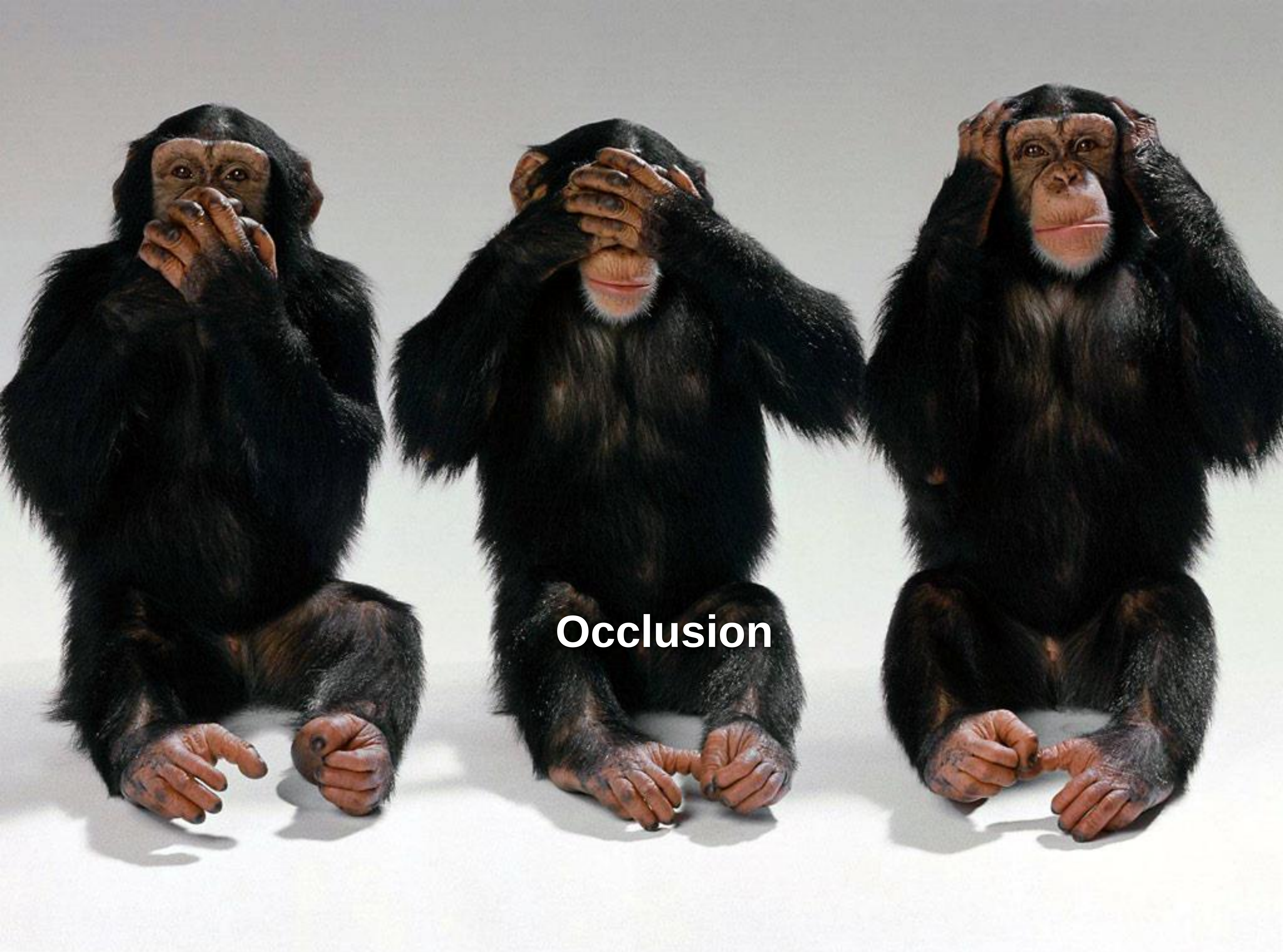
Challenge: deformation

Deformation

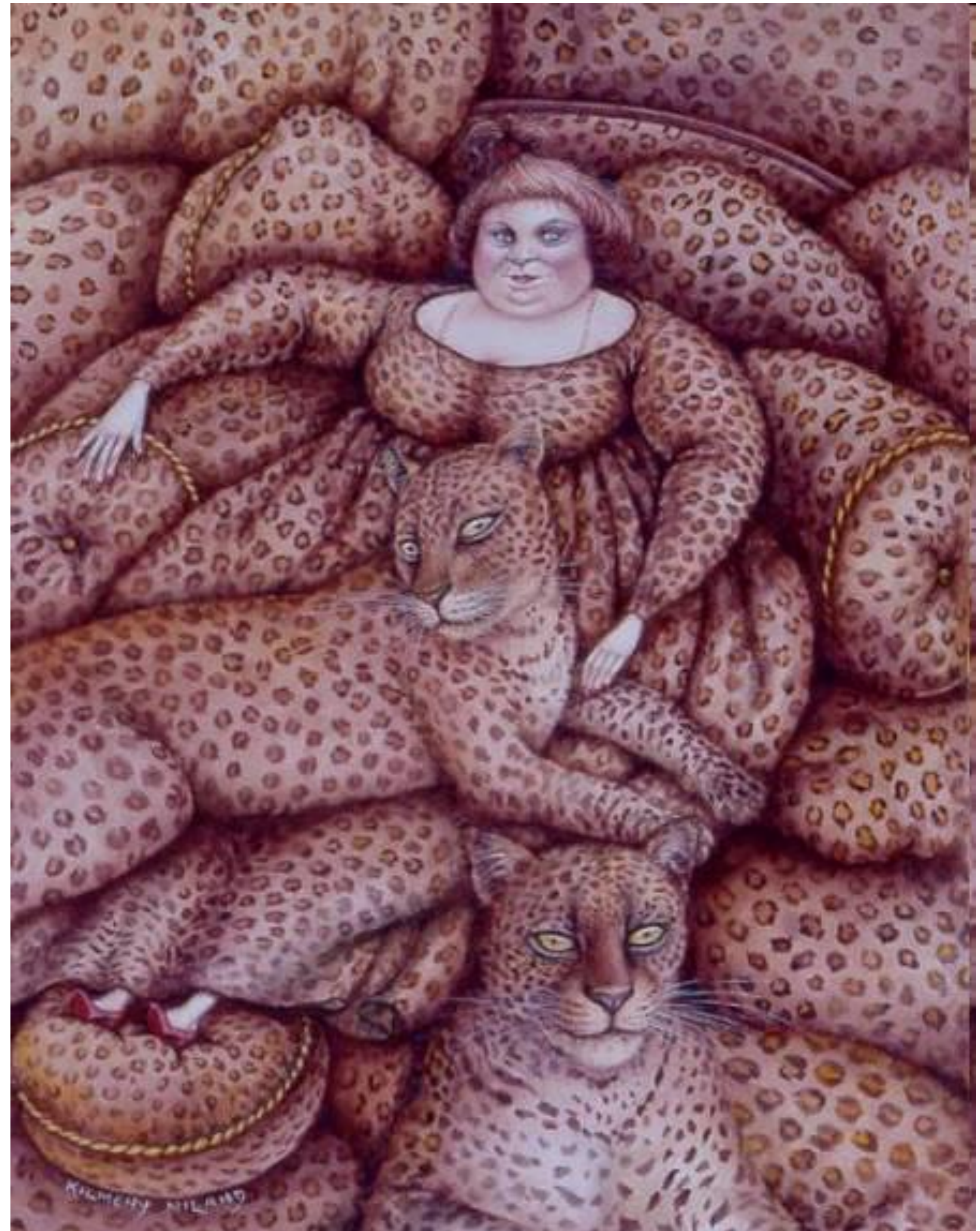
Challenge: Occlusion



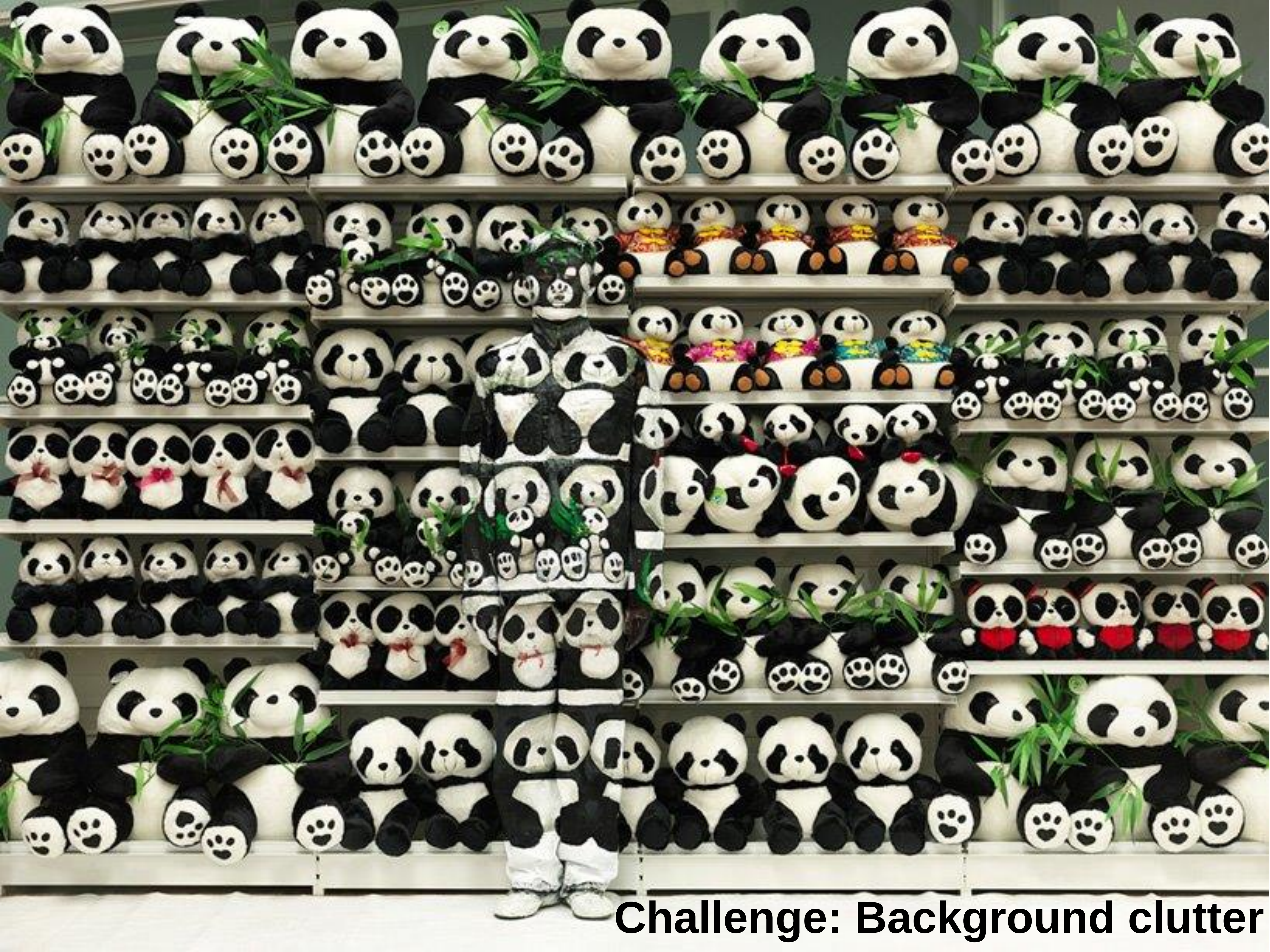
Magritte, 1957


Occlusion

Challenge: background clutter

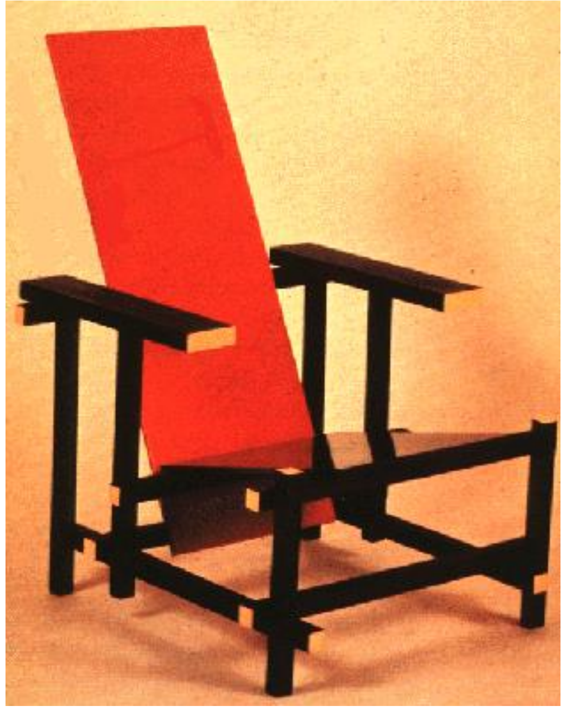


Kilmeny Niland. 1995



Challenge: Background clutter

Challenge: intra-class variations

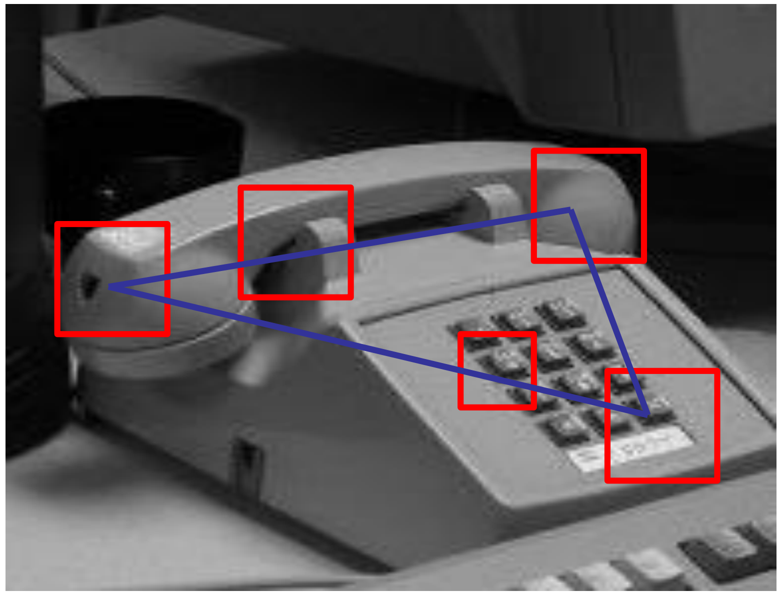


Svetlana Lazebnik

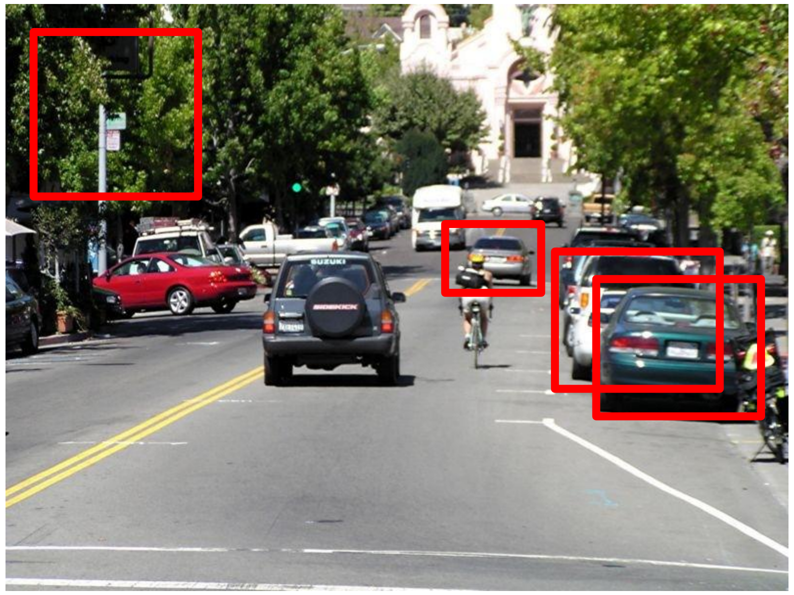
Common approaches

Common approaches: object recognition

Feature Matching



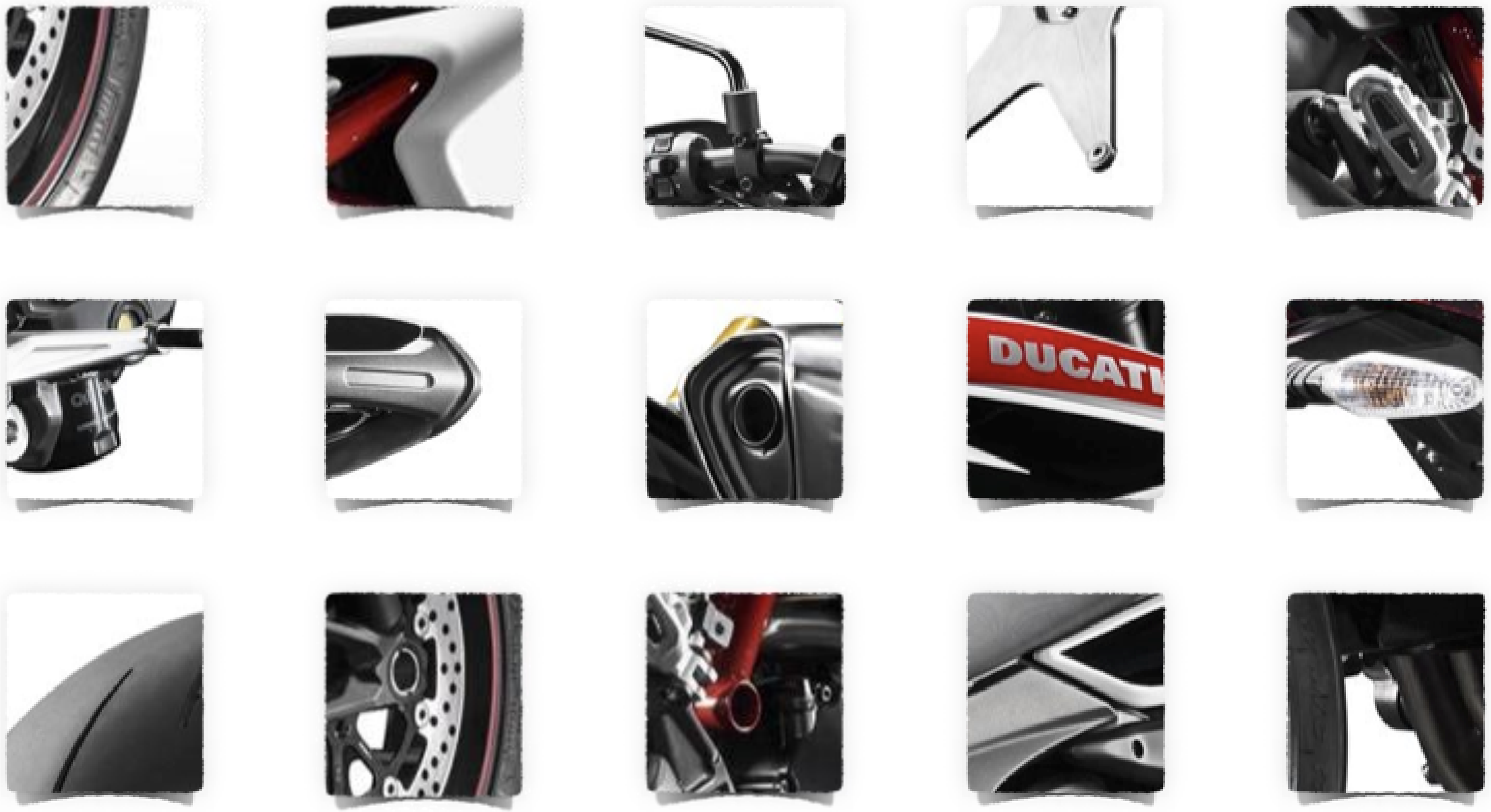
Spatial reasoning



Window classification

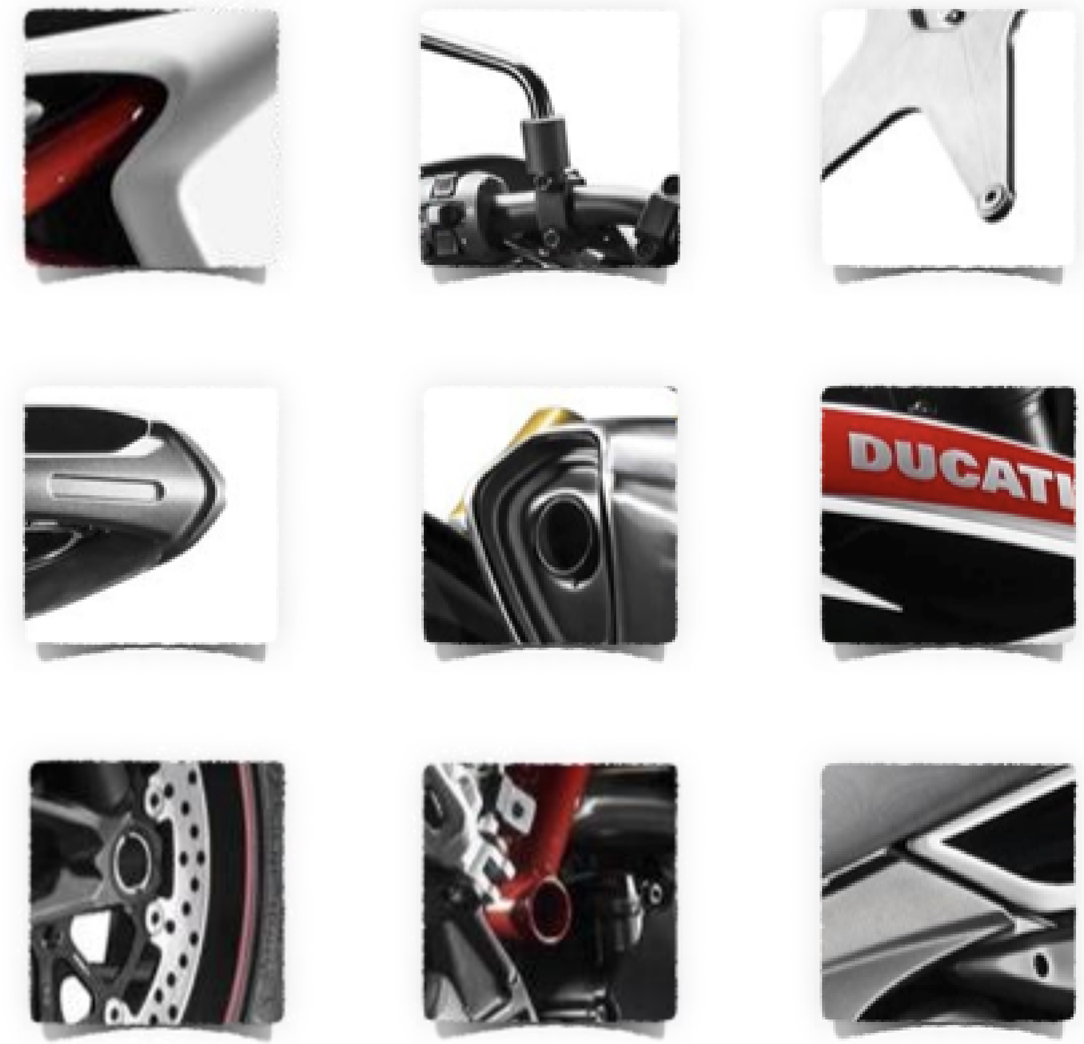
Feature matching

What object do these parts belong to?



Some local feature are very informative

An object as



a collection of local features

(bag-of-features)

- deals well with occlusion
- scale invariant
- rotation invariant

Are the positions of the parts important?

Why not use SIFT matching for everything?

- Works well for object *instances*

- Not great for generic object *categories*

Pros

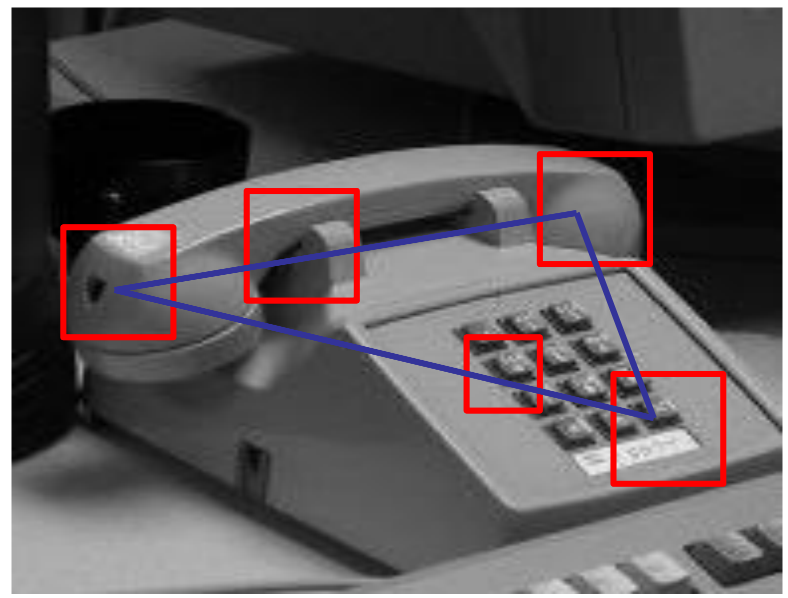
- Simple
- Efficient algorithms
- Robust to deformations

Cons

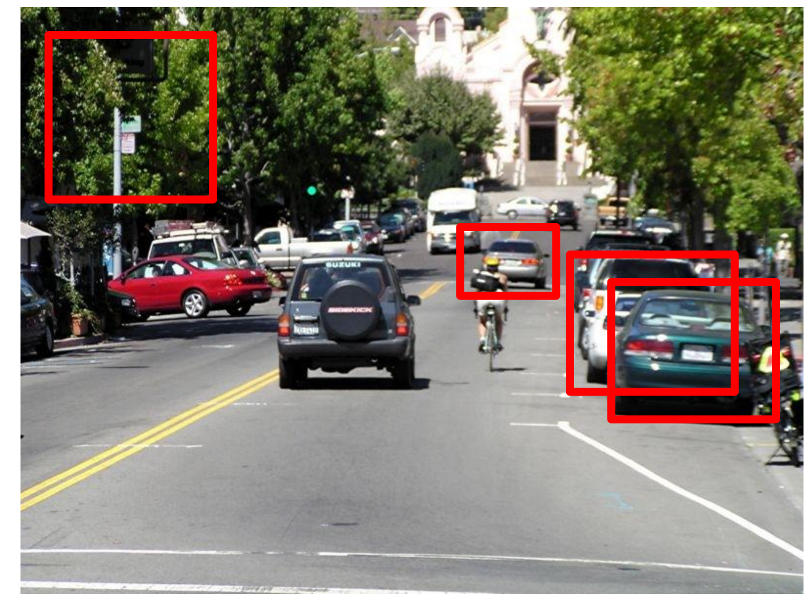
- No spatial reasoning

Common approaches: object recognition

Feature Matching



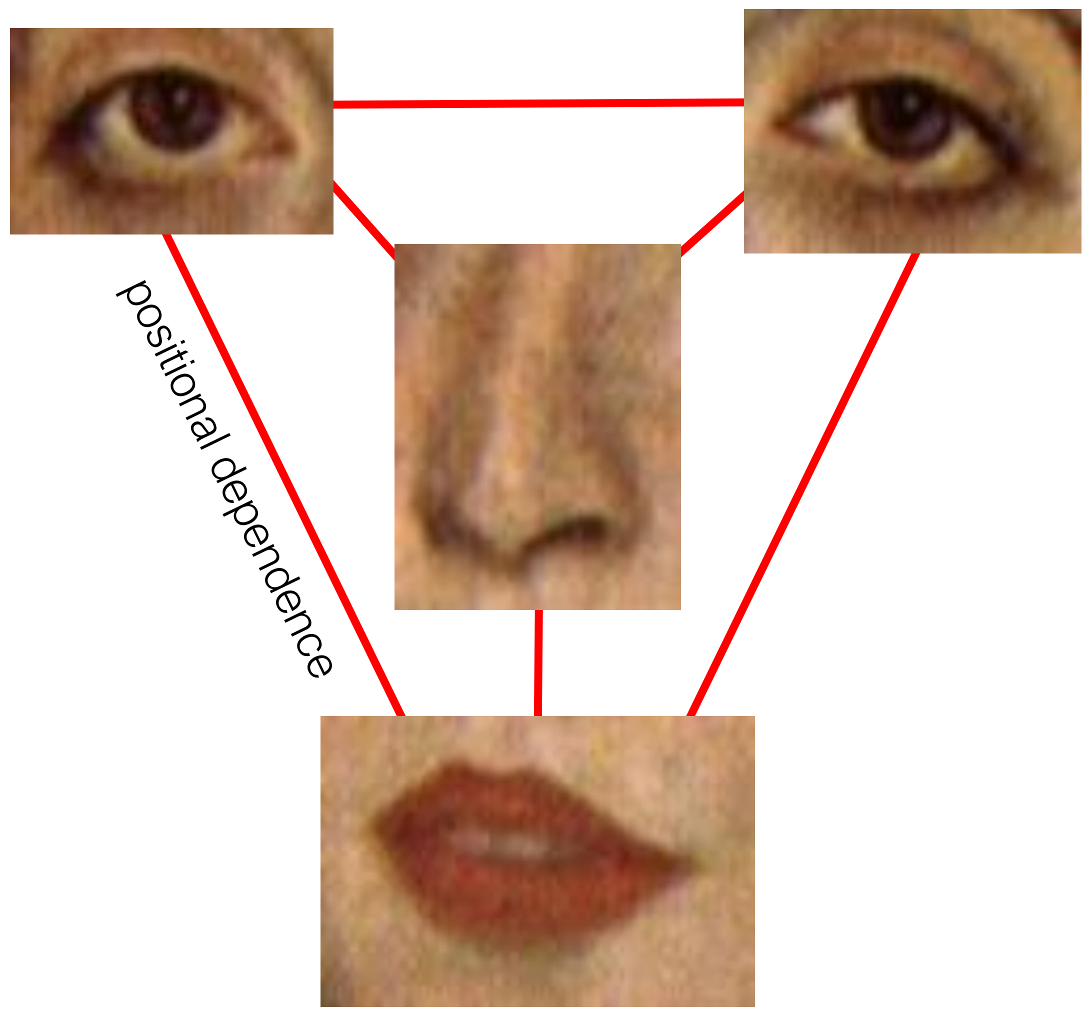
Spatial reasoning



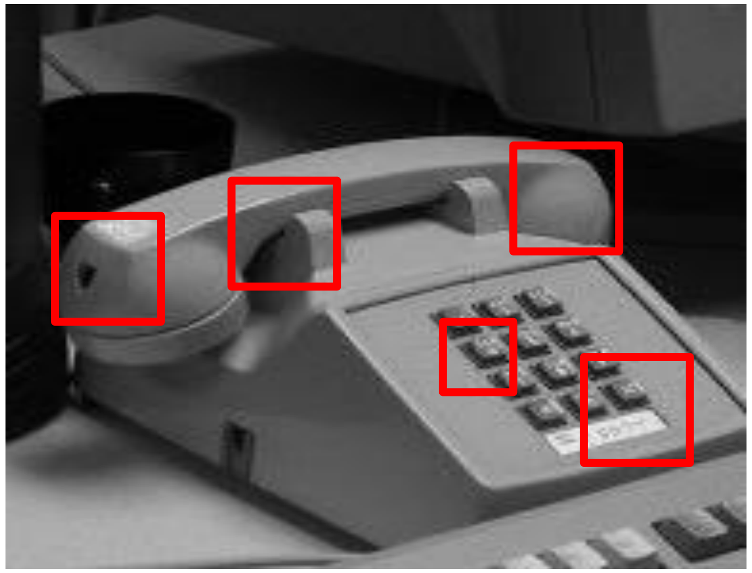
Window classification

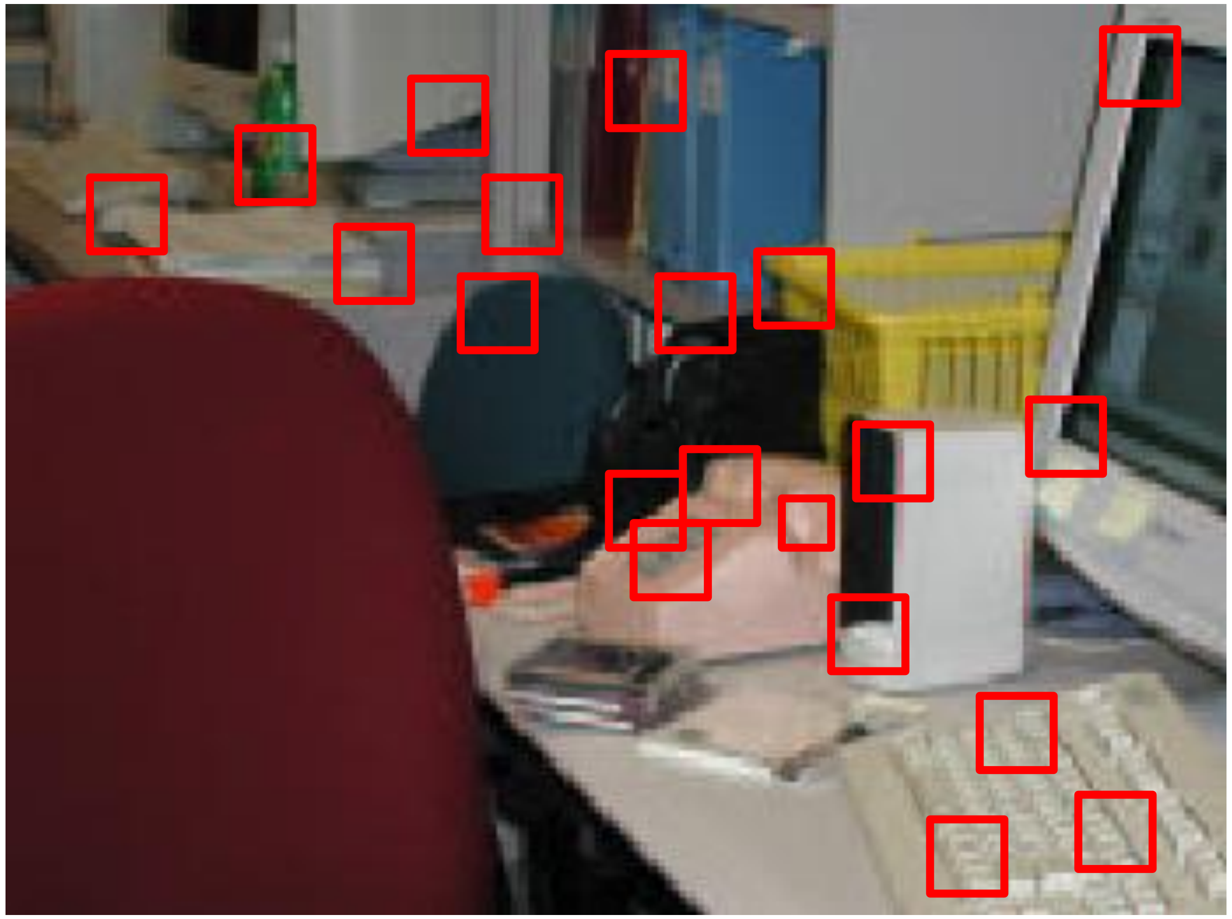
Spatial reasoning

The position of every part depends on the positions of all the other parts

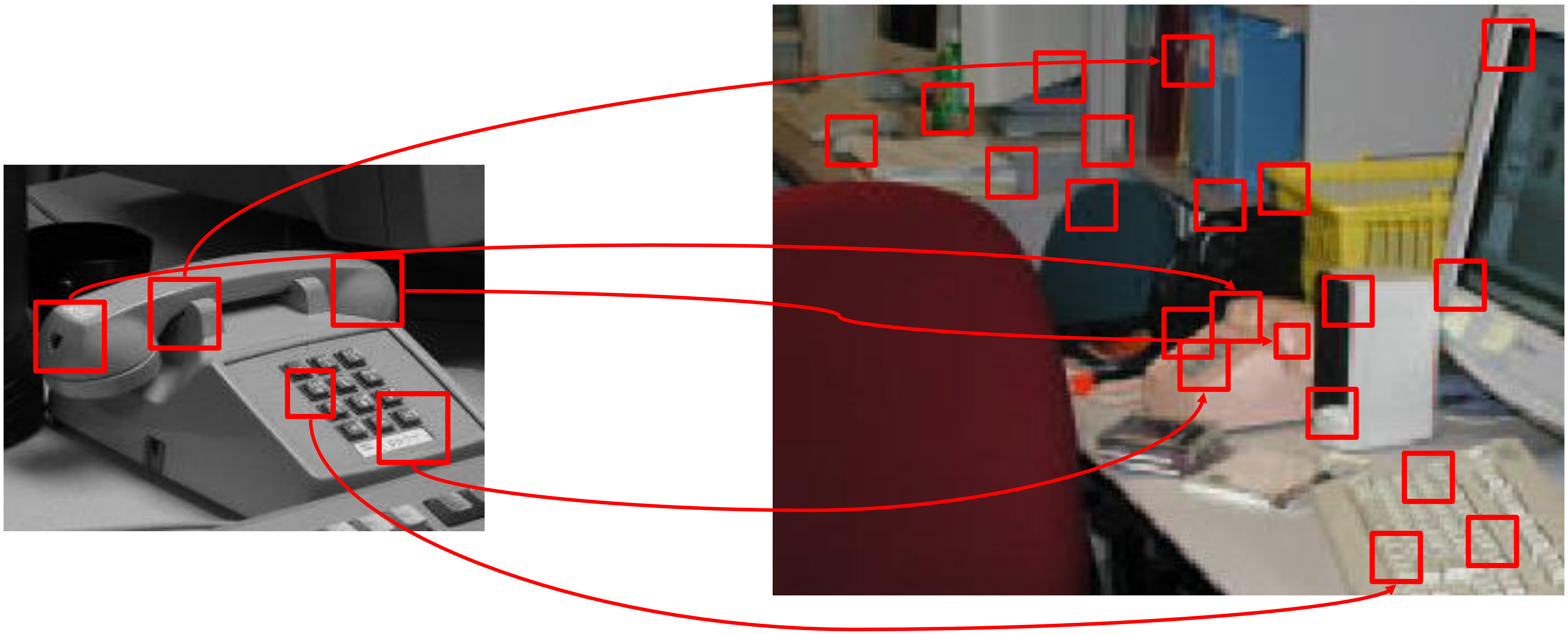


Many parts, many dependencies!





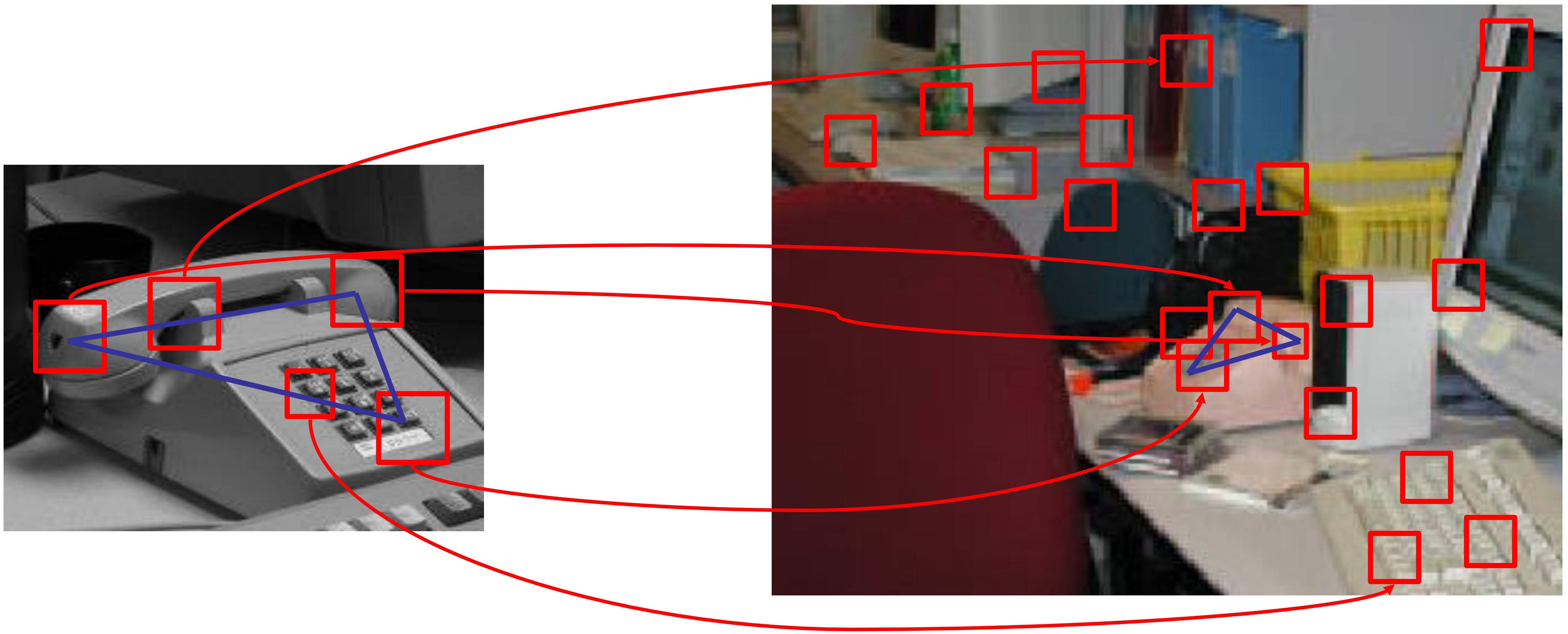
1. Extract features   2. Match features   3. Spatial verification



1. Extract features

2. Match features

3. Spatial verification



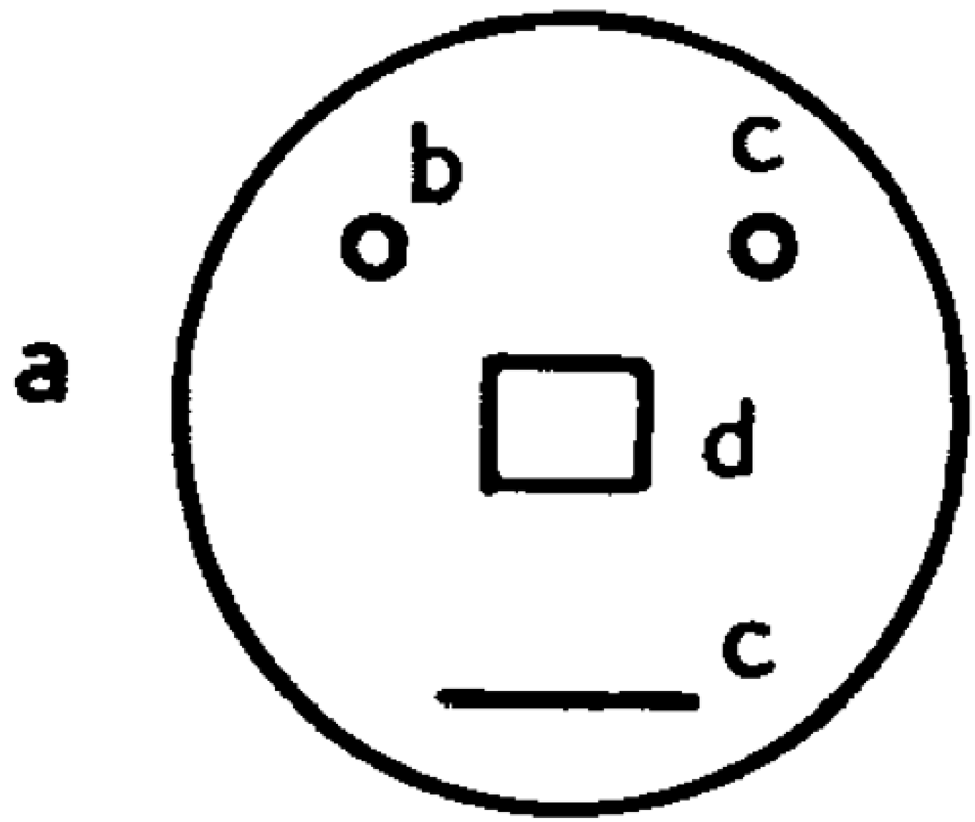
1. Extract features

2. Match features

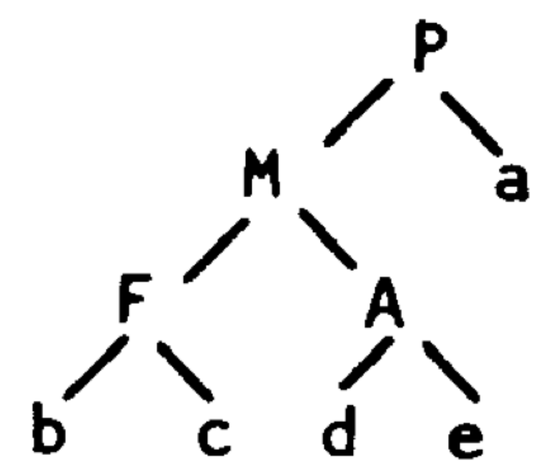
3. Spatial verification

an old idea…

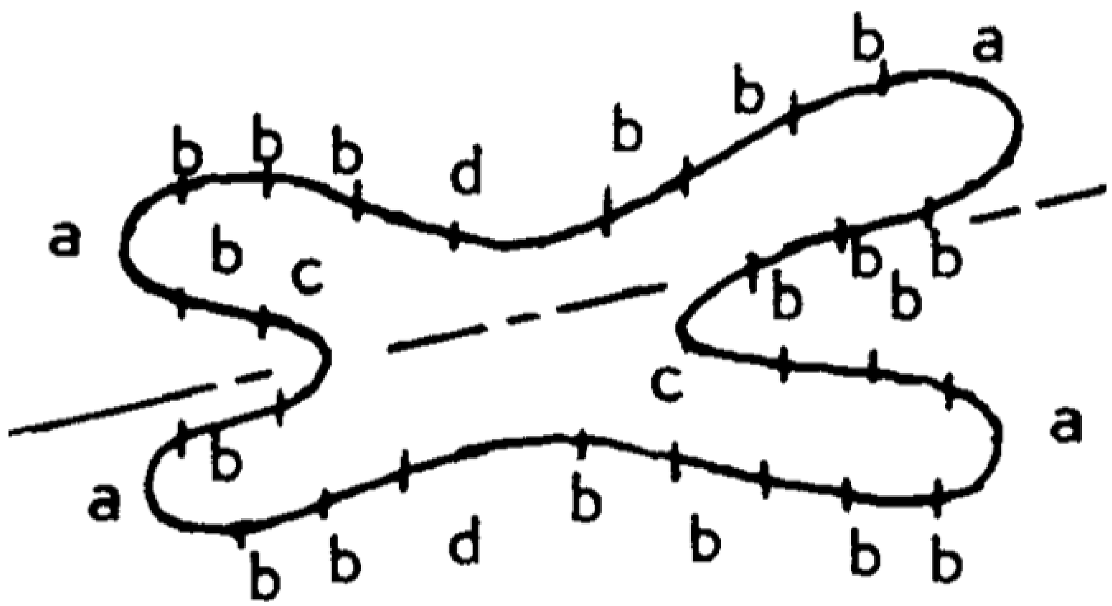
Fu and Booth. Grammatical Inference. 1975



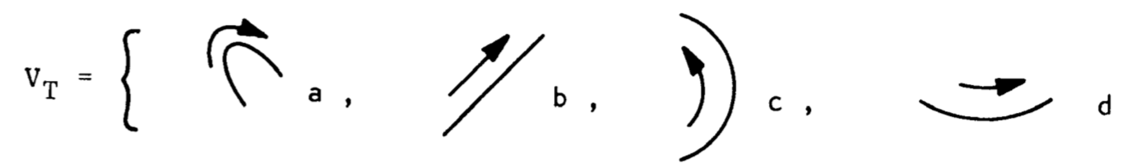
Scene



Structural (grammatical) description

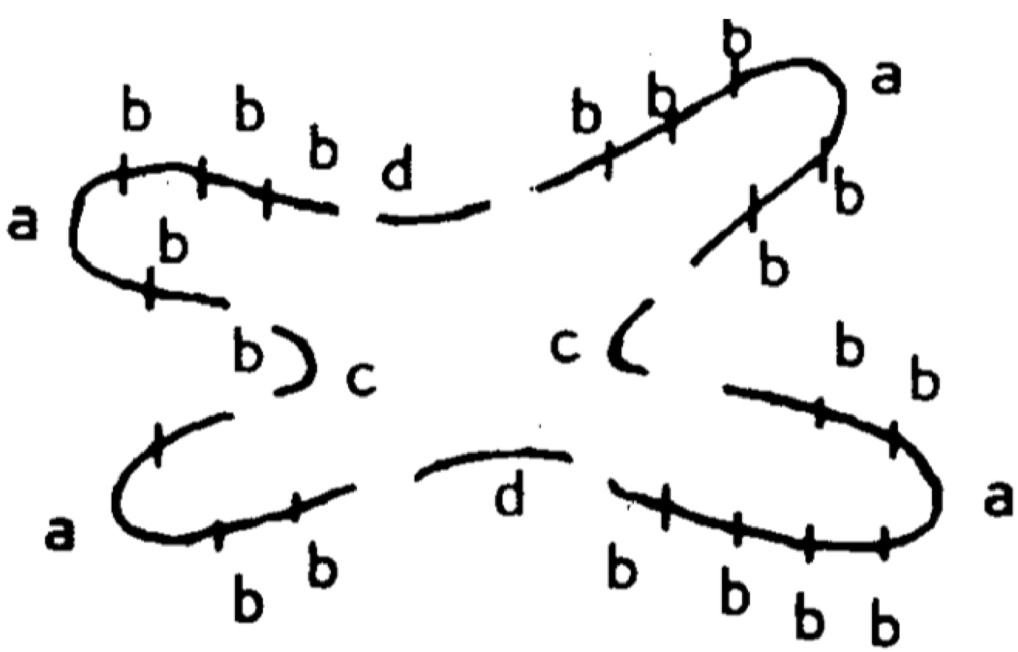


Coded Chromosome



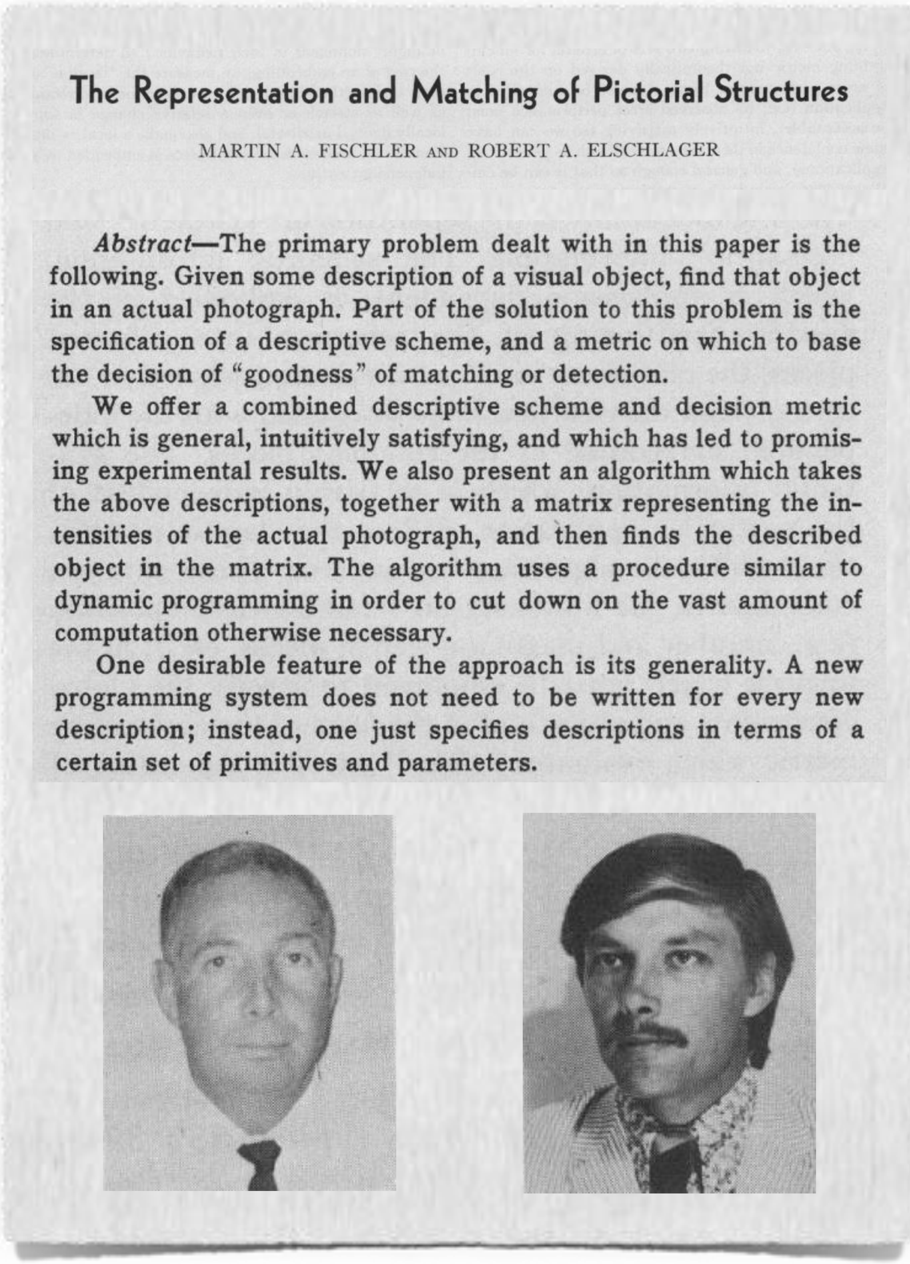
x = cdabbbdbbbabbcbbabbbbdbbabb

Substructures of Coded Chromosome



S_1 = {[b[[[a]b]b]b]; [b[b[b[a]]b]b];

[b[b[[[a]b]b]b]b]; [b[b[a]]b]}



The Representation and Matching of Pictorial Structures

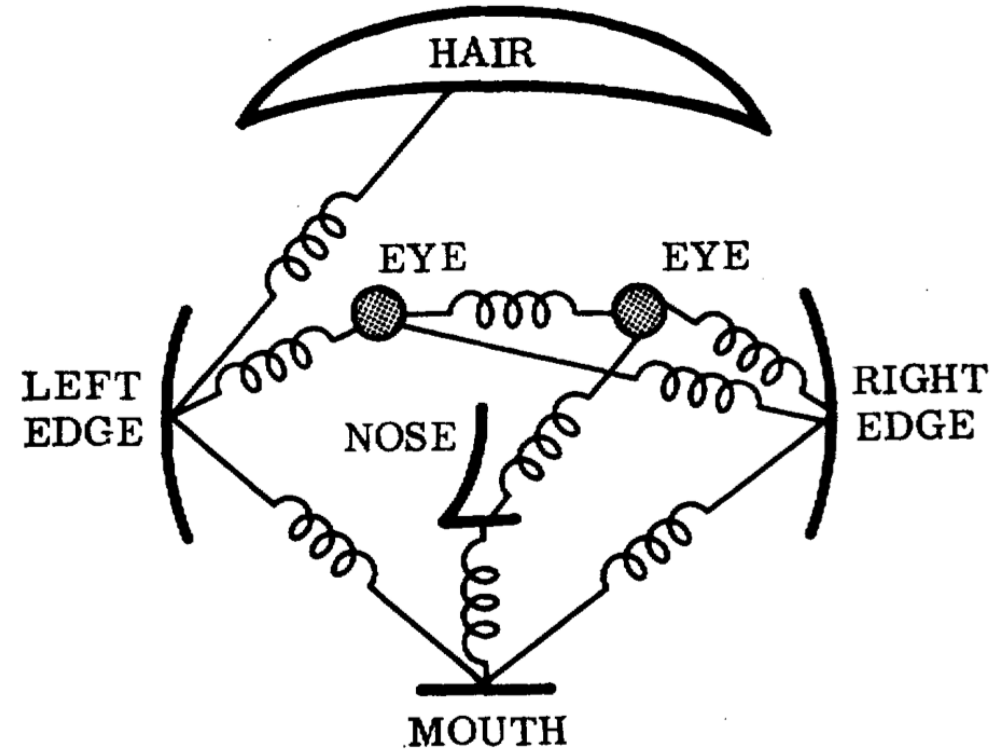
MARTIN A. FISCHLER AND ROBERT A. ELSCHLAGER

Abstract—The primary problem dealt with in this paper is the following. Given some description of a visual object, find that object in an actual photograph. Part of the solution to this problem is the specification of a descriptive scheme, and a metric on which to base the decision of "goodness" of matching or detection.

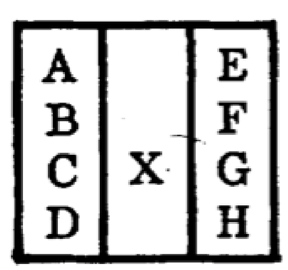
We offer a combined descriptive scheme and decision metric which is general, intuitively satisfying, and which has led to promising experimental results. We also present an algorithm which takes the above descriptions, together with a matrix representing the intensities of the actual photograph, and then finds the described object in the matrix. The algorithm uses a procedure similar to dynamic programming in order to cut down on the vast amount of computation otherwise necessary.

One desirable feature of the approach is its generality. A new programming system does not need to be written for every new description; instead, one just specifies descriptions in terms of a certain set of primitives and parameters.

1972



Description for left edge of face



VALUE(X)=(E+F+G+H)-(A+B+C+D)

Note: VALUE(X) is the value assigned to the L(EV)A corresponding to the location X as a function of the intensities of locations A through H in the sensed scene.

A more probabilistic approach…

think of locations as random variables (RV)

vector of RVs:
set of part locations

$$\boldsymbol{L} = \{\overset{\text{RV}}{L_1}, \overset{\text{RV}}{L_2}, \ldots, \overset{\text{RV}}{L_M}\}$$

A more modern probabilistic approach…

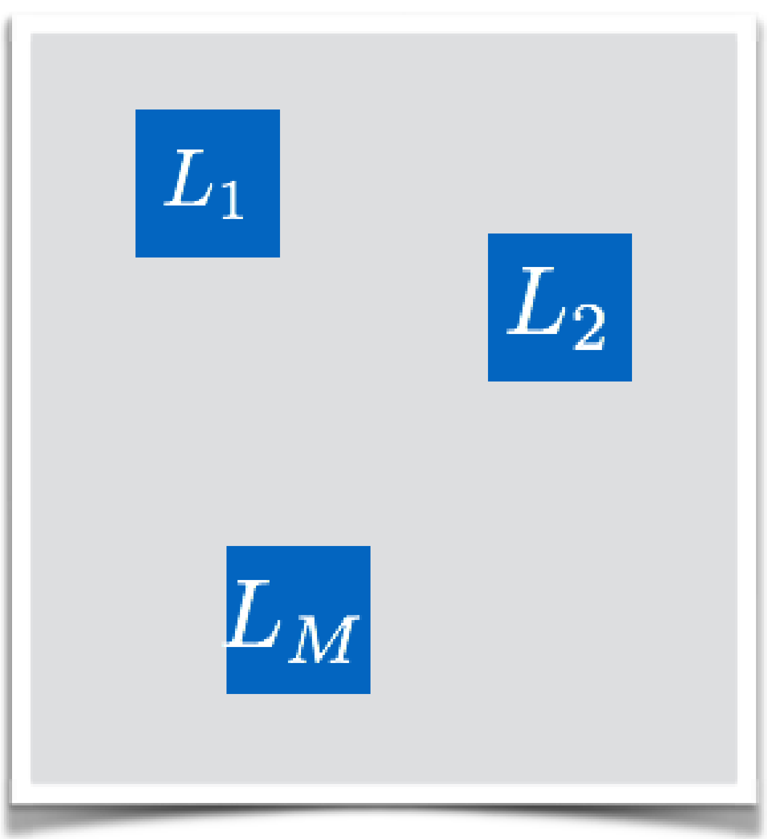
think of locations as random variables (RV)

RV RV RV

vector of RVs:
set of part locations

$$\boldsymbol{L} = \{L_1, L_2, \ldots, L_M\}$$

image (N pixels)



What are the dimensions of R.V. L?

How many possible combinations of part locations?

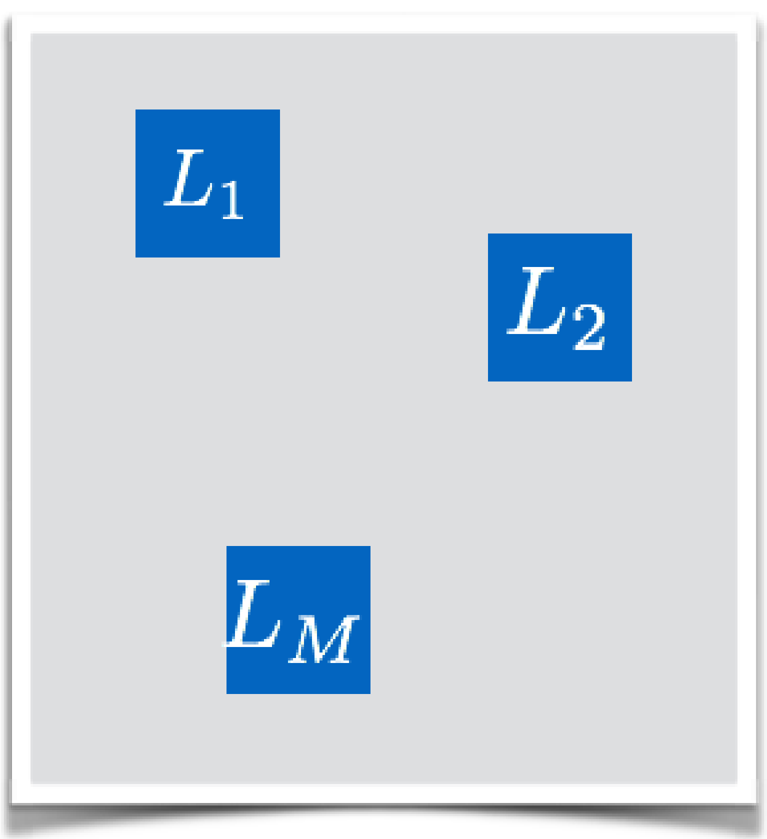
A more modern probabilistic approach…

think of locations as random variables (RV)

vector of RVs:
set of part locations

$$\boldsymbol{L} = \{\overset{\text{RV}}{L_1}, \overset{\text{RV}}{L_2}, \ldots, \overset{\text{RV}}{L_M}\}$$

image (N pixels)



What are the dimensions of R.V. L?

$$L_m = [\, x \; y \,]$$

How many possible combinations of part locations?

A more modern probabilistic approach…

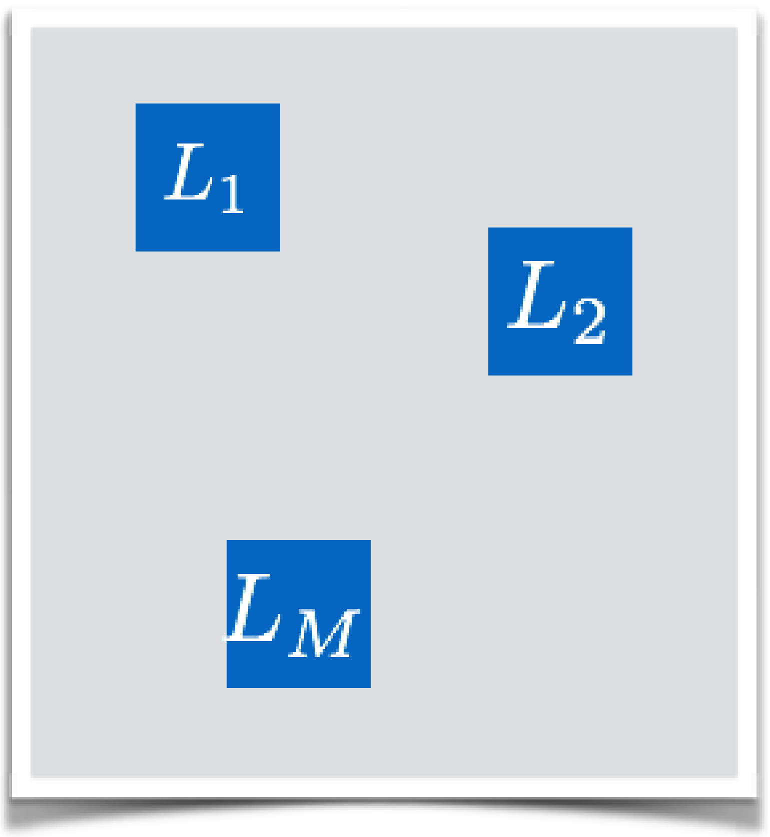
think of locations as random variables (RV)

RV RV RV

vector of RVs: set of part locations

$$\boldsymbol{L} = \{L_1, L_2, \ldots, L_M\}$$

image (N pixels)



What are the dimensions of R.V. L?

$$L_m = [\, x \; y \,]$$

How many possible combinations of part locations?

$$N^M$$

Most likely set of locations L is found by **maximizing**:

part locations

image

$$p(\boldsymbol{L}|\boldsymbol{I}) \propto p(\boldsymbol{I}|\boldsymbol{L})p(\boldsymbol{L})$$

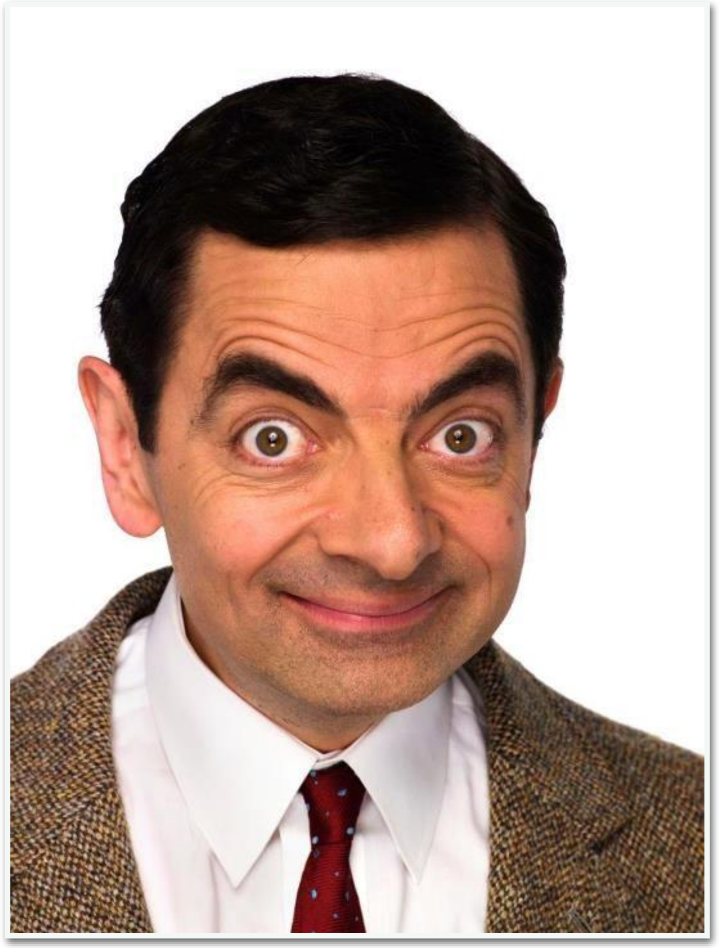
Posterior

Likelihood:
How likely it is to observe image I given that the M parts are at locations L
(scaled output of a classifier)

Prior:
spatial prior controls the geometric configuration of the parts

What kind of prior can we formulate?

Given any collection of selfie images,
where would you expect the nose to be?



*What would be an appropriate **prior**?*

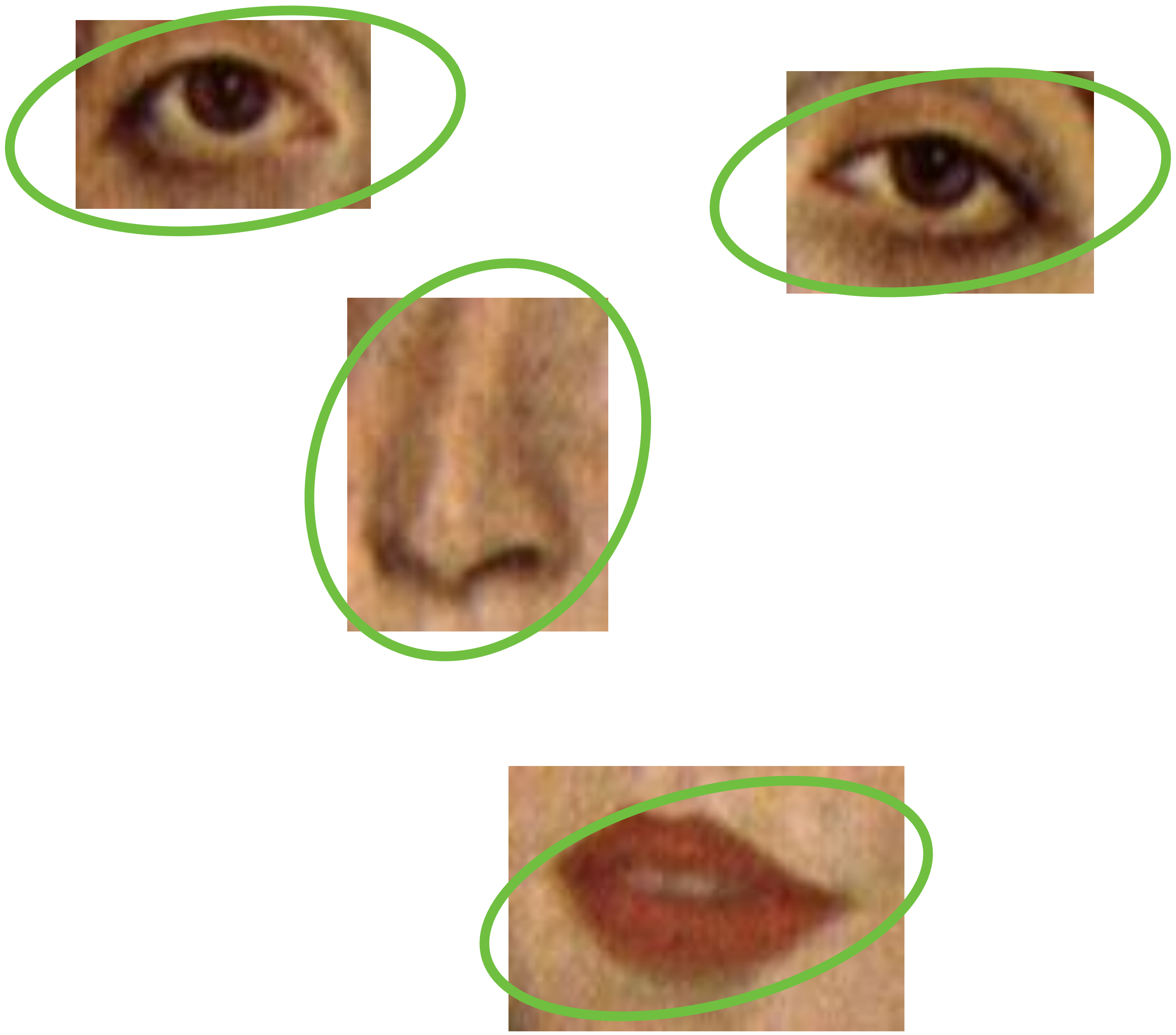
$$P(L_{\text{nose}}) = ?$$

A simple factorized model

$$p(\boldsymbol{L}) = \prod_m p(L_m)$$

Break up the joint probability into smaller (independent) terms

Independent locations



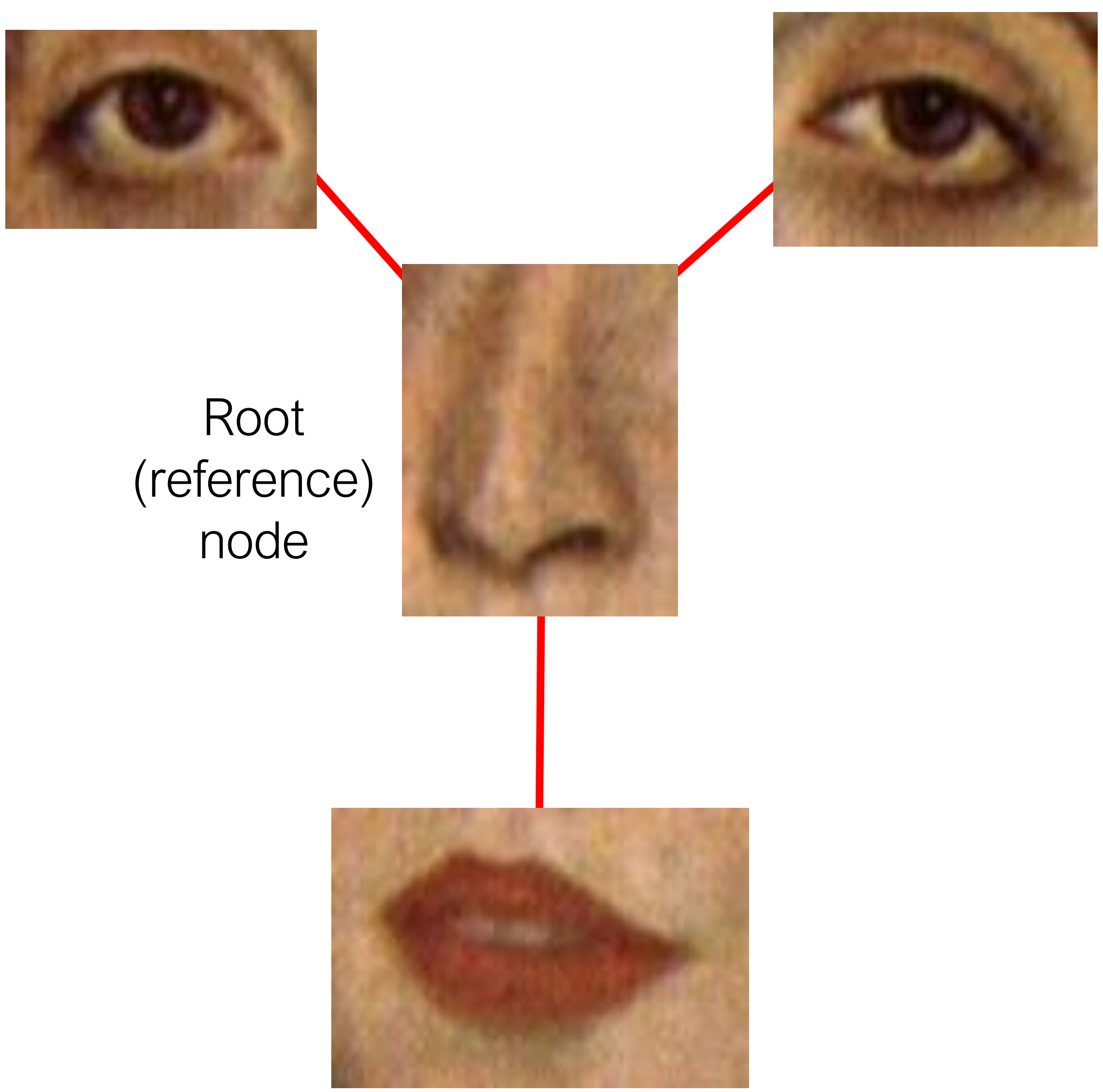
$$p(\boldsymbol{L}) = \prod_m p(L_m)$$

Each feature is allowed to move independently

Does not model the **relative** location of parts at all

Tree structure

(star model)

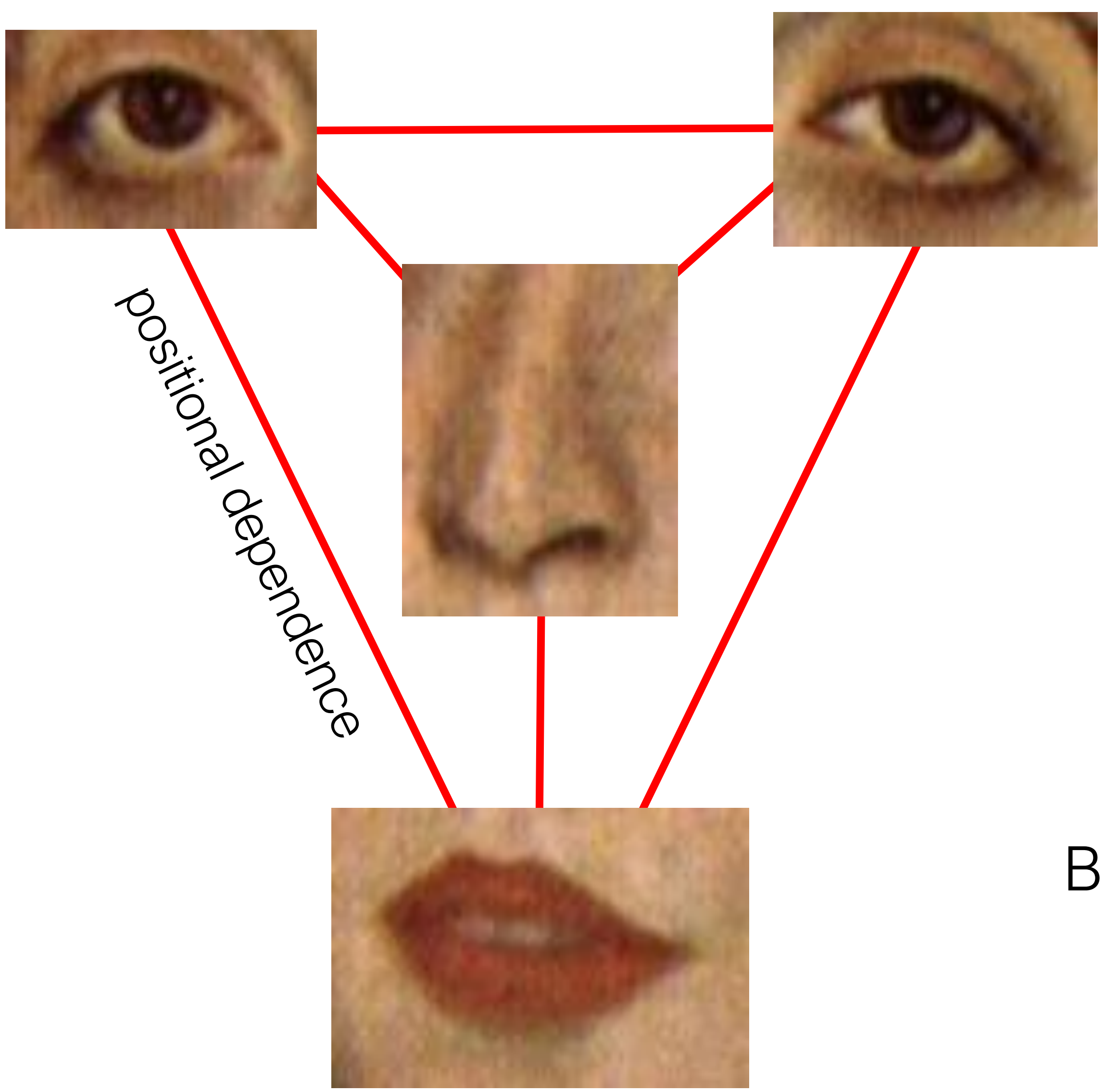


$$p(\boldsymbol{L}) = p(L_{\text{root}}) \prod_{m=1}^{M-1} p(L_m | L_{\text{root}})$$

Represent the location of all the parts relative to a single reference part

Assumes that one reference part is defined (who will decide this?)

Fully connected
(constellation model)



$$p(L) = p(l_1, \ldots, l_N)$$

Explicitly represents the
joint distribution of locations

<u>Good model:</u>
Models relative location of parts
BUT Intractable for moderate number of parts

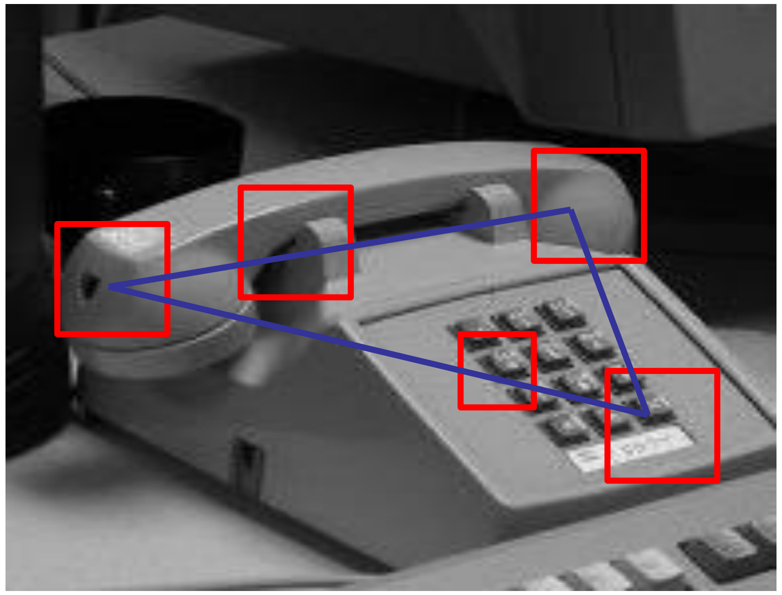
Pros

- Retains spatial constraints
- Robust to deformations

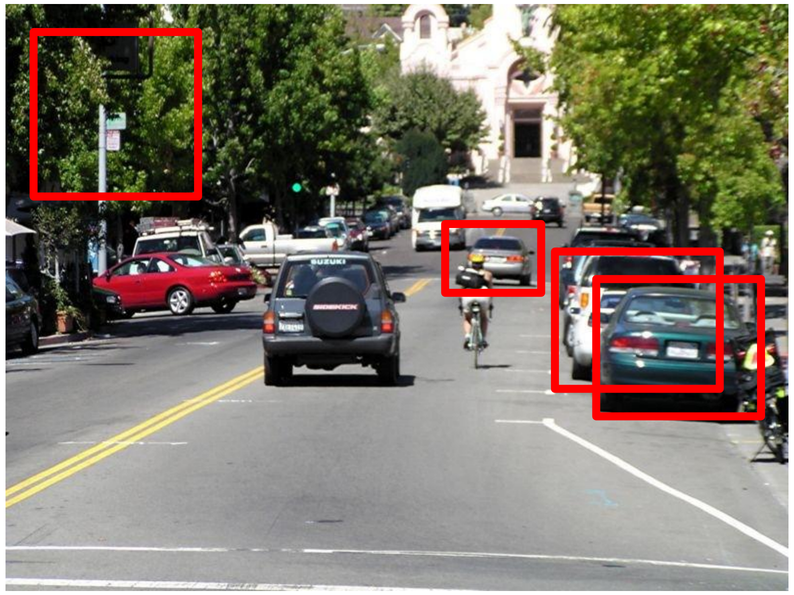
Cons

- Computationally expensive
- Generalization to **large** inter-class variation (e.g., modeling chairs)

Feature Matching



Spatial reasoning

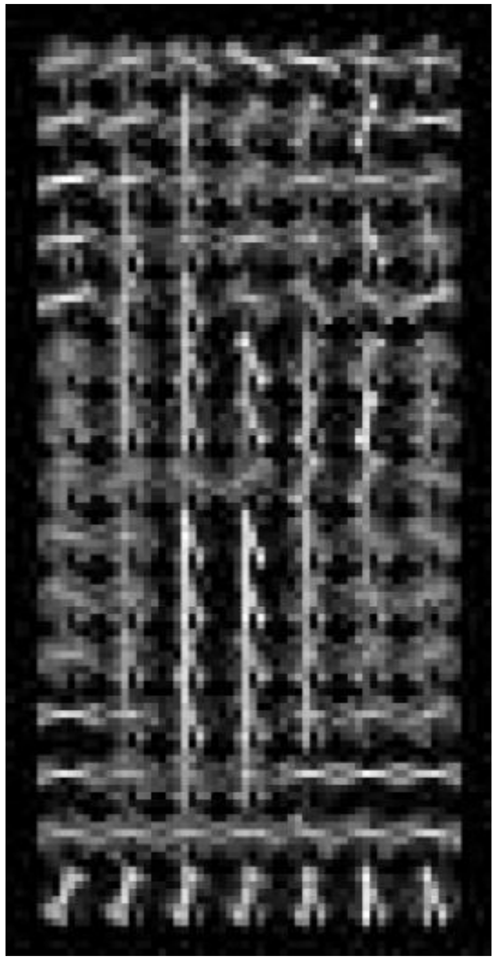


Window classification

Window-based

Template Matching

1. get image window



2. extract features

3. classify

When does this work and when does it fail?

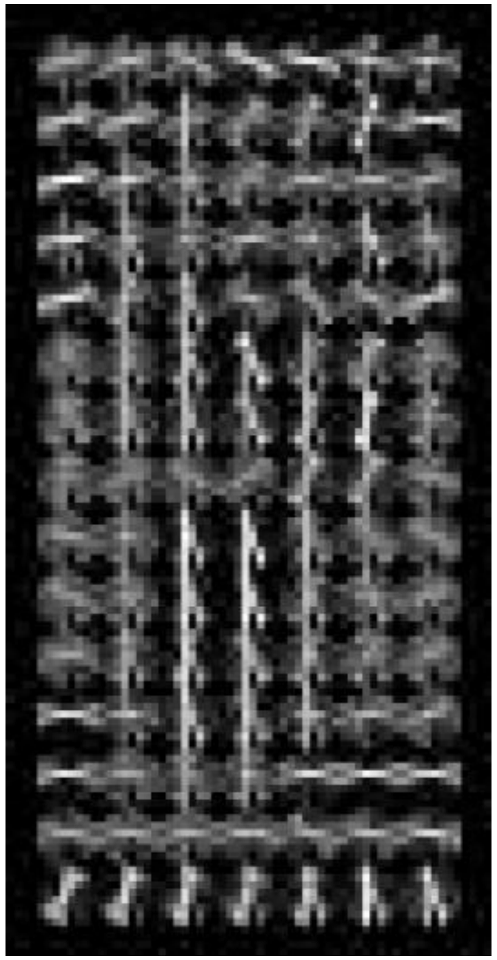
How many templates do you need?

Per-exemplar

find the ‘nearest’ exemplar, inherit its label

Template Matching

1. get image window
(or region proposals)



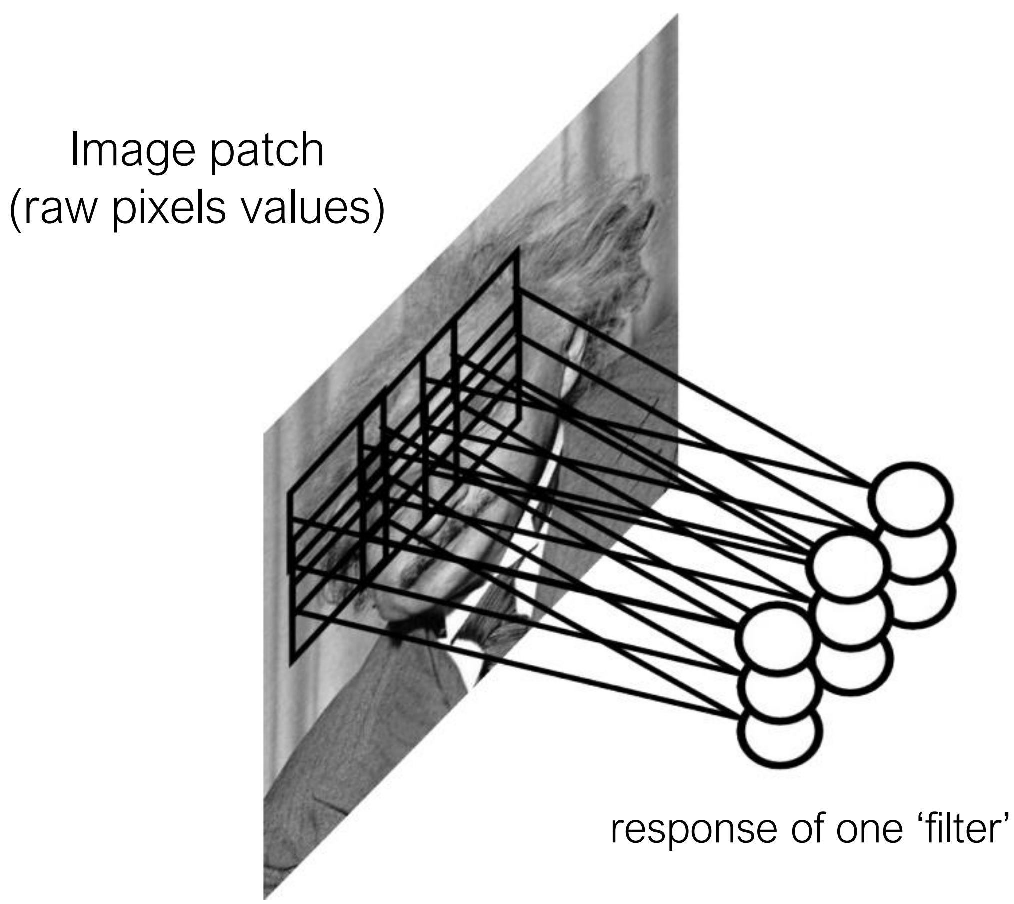
2. extract features

3. compare to template

Do this part with one big classifier
'end to end learning'

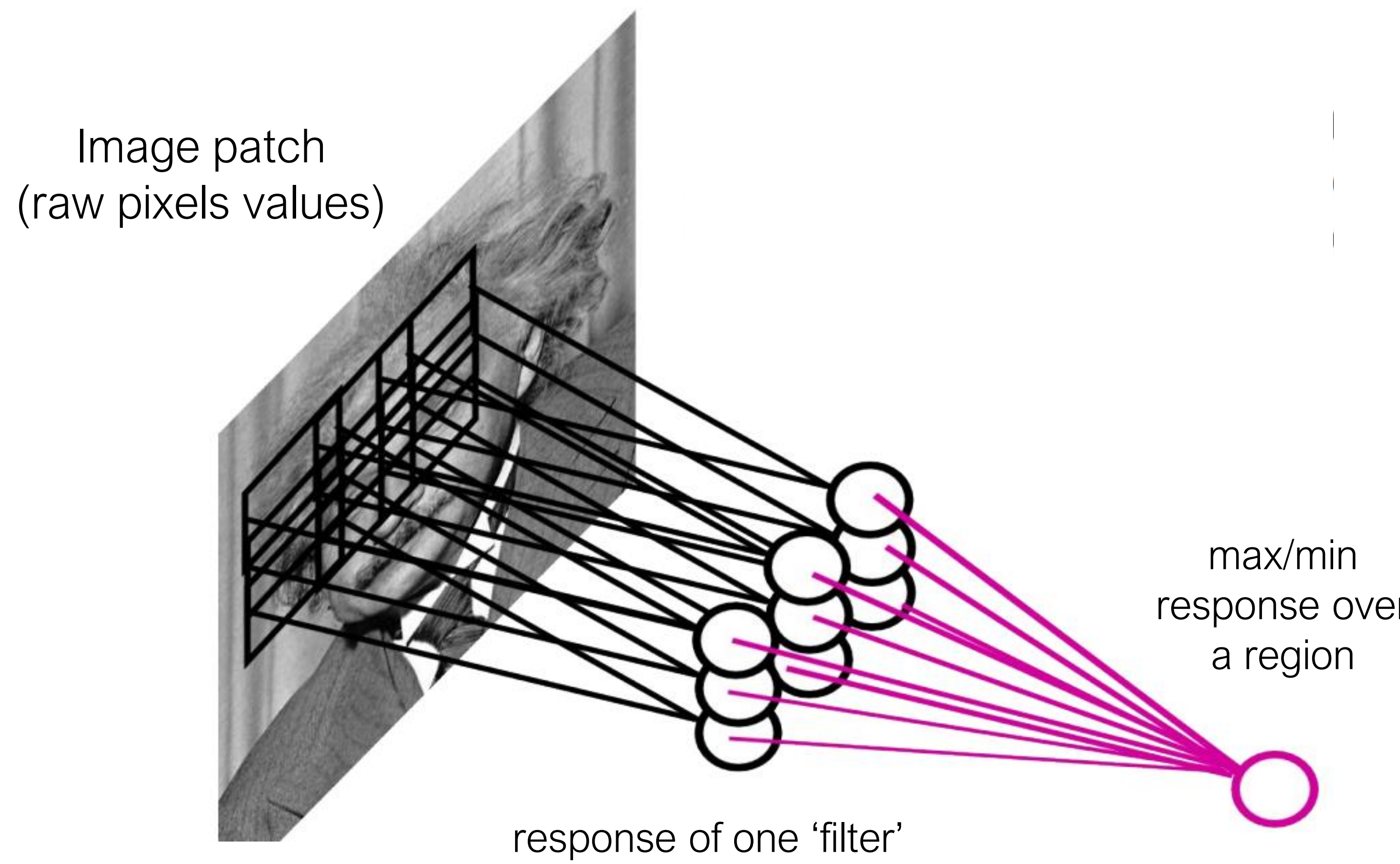
Convolutional Neural Networks

Convolution

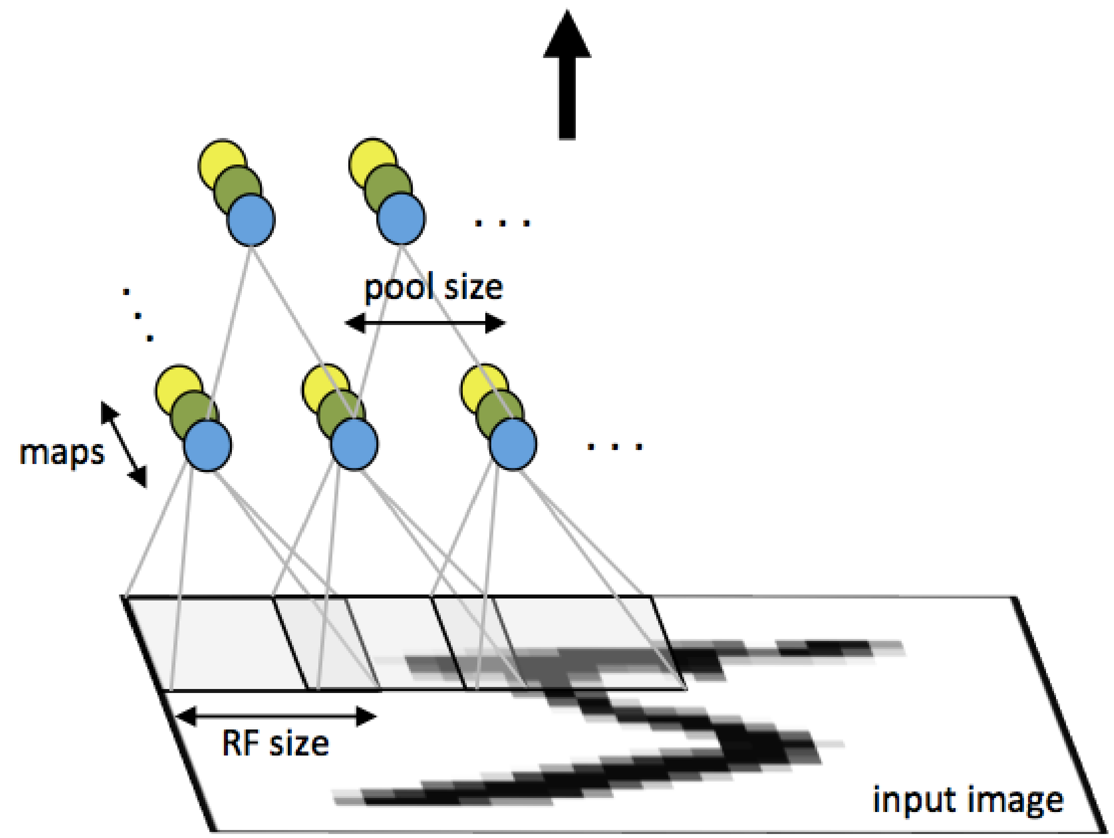


A 96 x 96 image convolved with 400 filters (features) of size 8 x 8 generates about 3 million values (89^2x400)

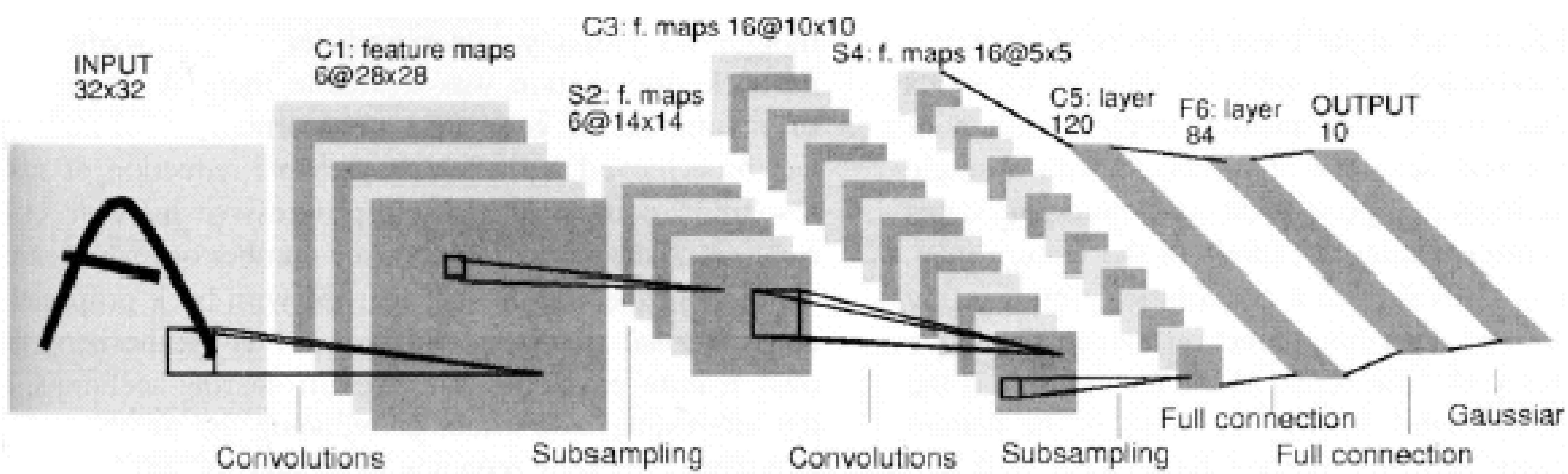
Pooling



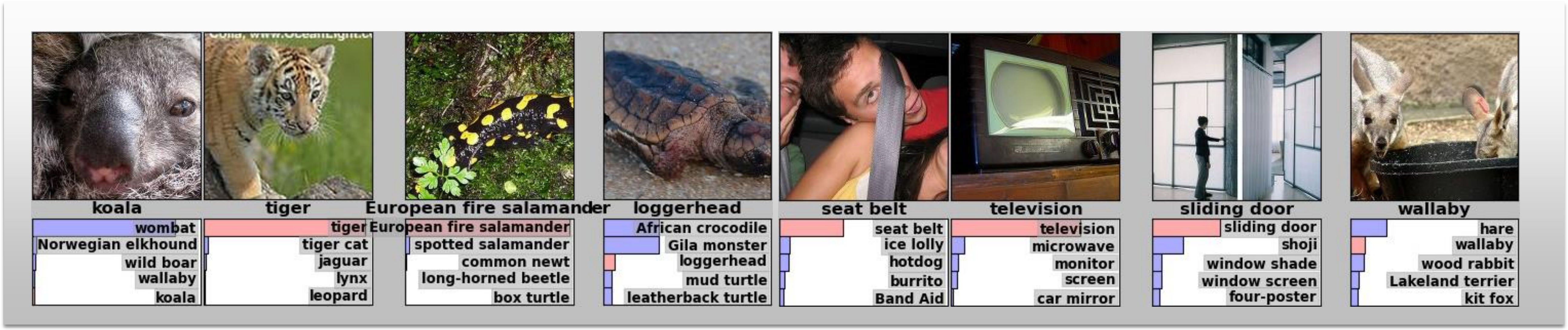
Pooling aggregates statistics and lowers the dimension of convolution

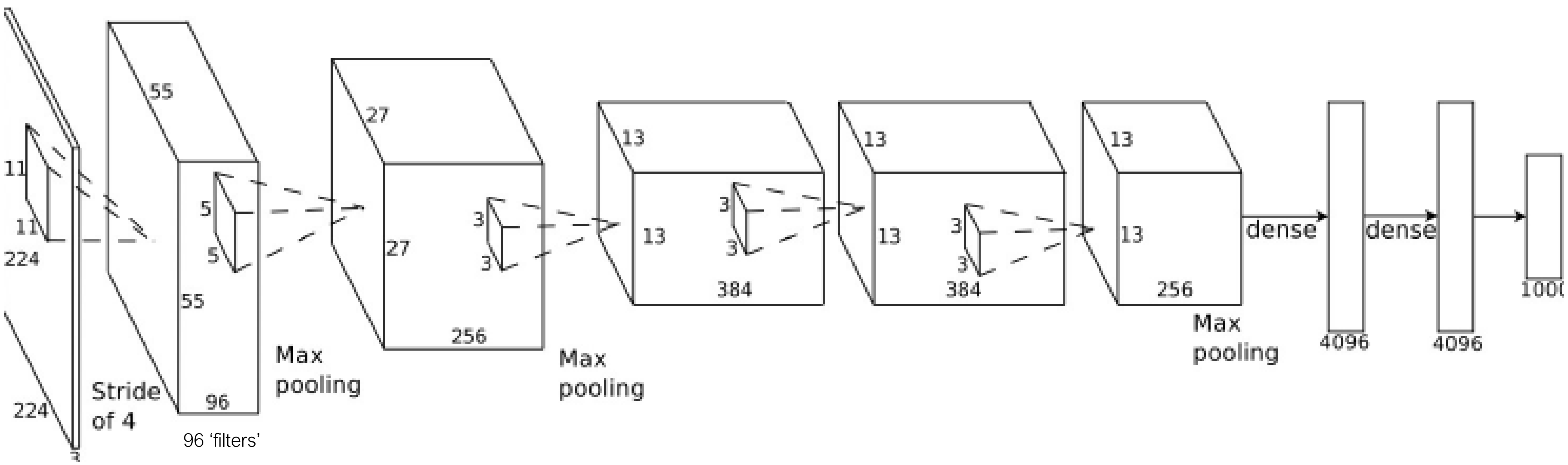


pool size
maps
RF size
input image



INPUT
32x32
C1: feature maps
6@28x28
S2: f. maps
6@14x14
C3: f. maps 16@10x10
S4: f. maps 16@5x5
C5: layer
120
F6: layer
84
OUTPUT
10
Convolutions
Subsampling
Convolutions
Subsampling
Full connection
Full connection
Gaussian





224/4=56

630 million connections
60 millions parameters to learn

Krizhevsky, A., Sutskever, I. and Hinton, G. E.
ImageNet Classification with Deep Convolutional Neural Networks, NIPS 2012.

Pros

- Retains spatial constraints
- Efficient test time performance

Cons

- Many many possible windows to evaluate
- Requires large amounts of data
- Sometimes (very) slow to train

History of ideas in recognition

- 1960s – early 1990s: the geometric era
- 1990s: appearance-based models
- Mid-1990s: sliding window approaches
- Late 1990s: local features
- Early 2000s: parts-and-shape models
- Mid-2000s: bags of features
- Present trends: data-driven methods, **deep learning**

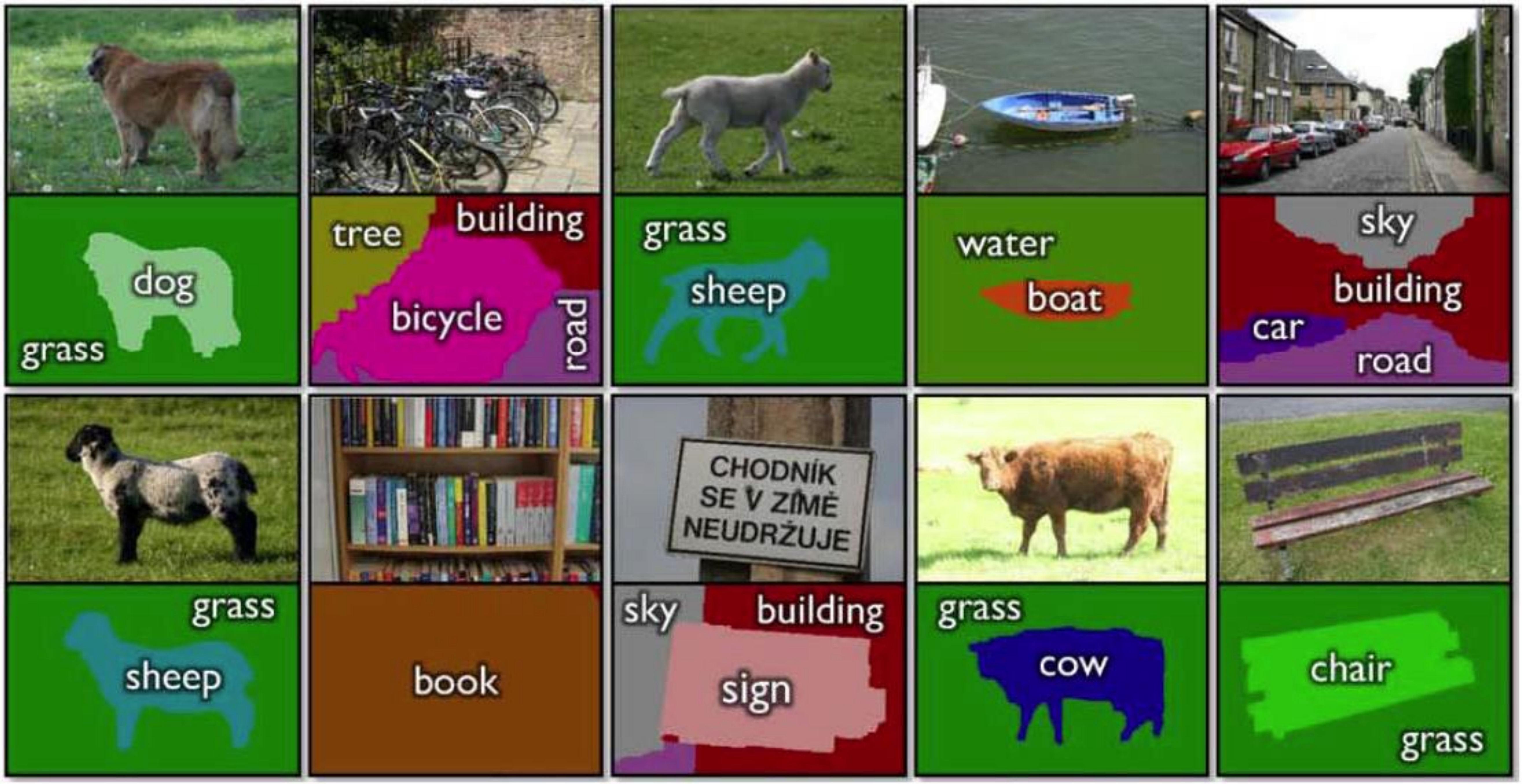
What Matters in Recognition?

- Learning Techniques
  - E.g. choice of classifier or inference method
- Representation
  - Low level: SIFT, HoG, GIST, edges
  - Mid level: Bag of words, sliding window, deformable model
  - High level: Contextual dependence
  - Deep features
- Data
  - More is always better
  - Annotation is the hard part

Types of Recognition

- Instance recognition
  - Recognizing a known object but in a new viewpoint, with clutter and occlusion
  - Location/Landmark Recognition
    - Recognize Paris, Rome, ... in photographs
    - Ideas from information retrieval
- Category recognition
  - Harder problem, even for humans
  - Bag of words, part-based, recognition and segmentation

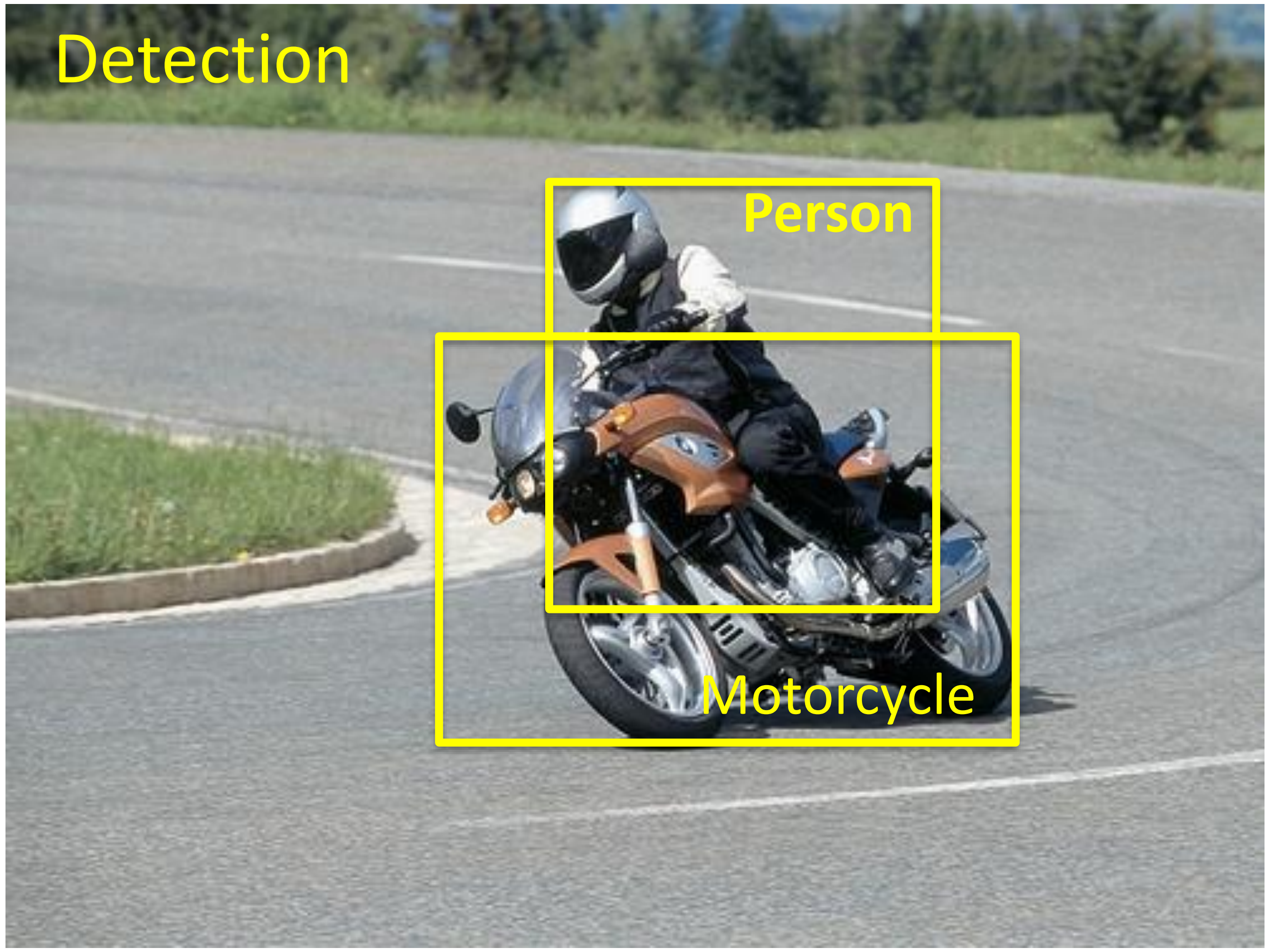
Simultaneous recognition and detection

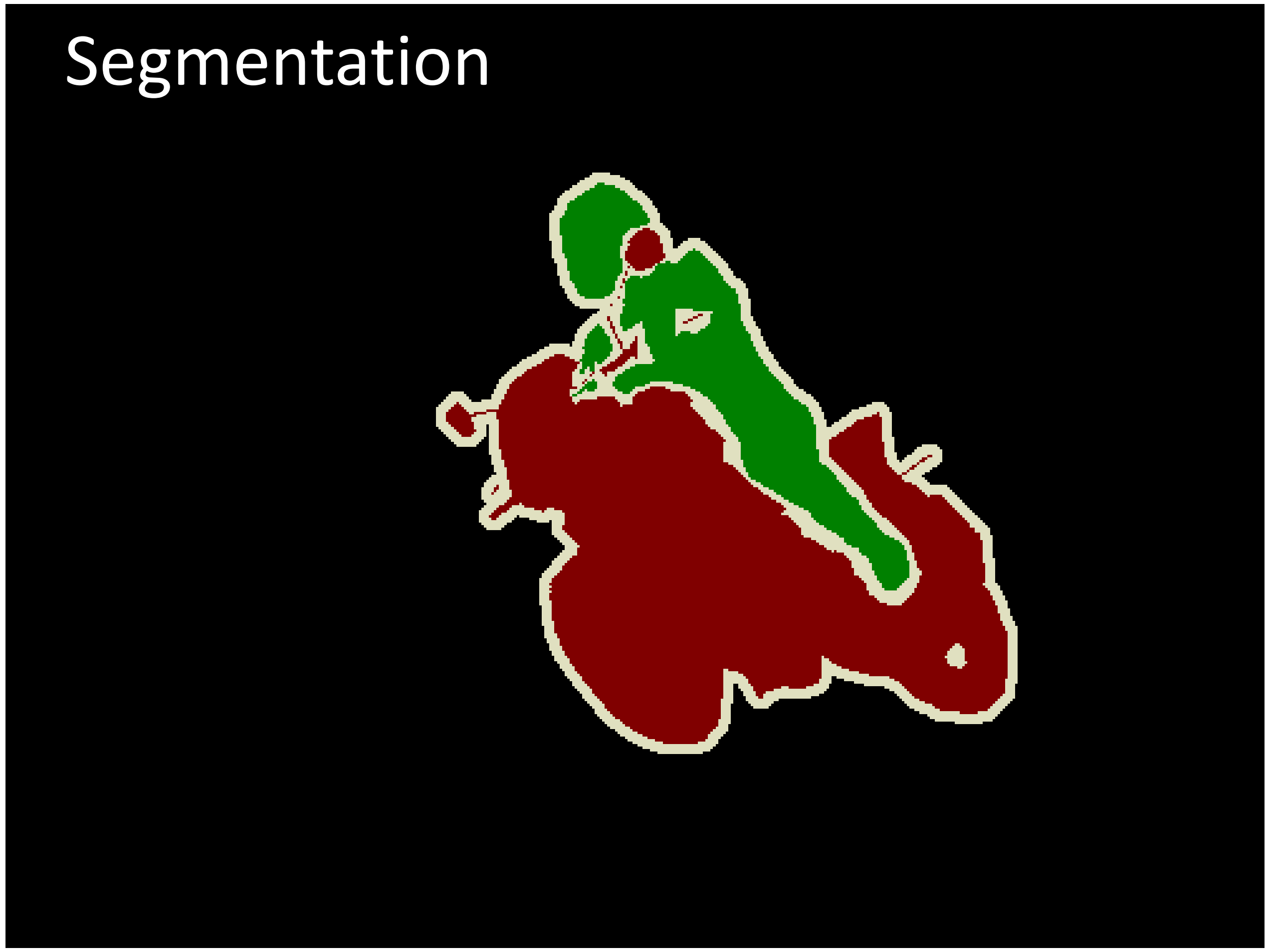


PASCAL VOC 2005-2012

20 object classes **22,591 images**

Classification: person, motorcycle



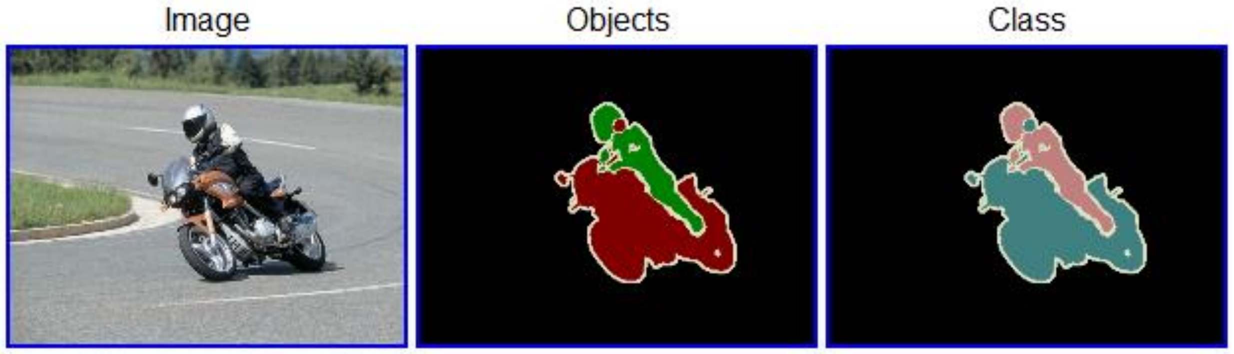


Action: riding bicycle

Everingham, Van Gool, Williams, Winn and Zisserman.
The PASCAL Visual Object Classes (VOC) Challenge. IJCV 2010.

The PASCAL Visual Object Classes Challenge 2009 (VOC2009)

- 20 object categories (aeroplane to TV/monitor)
- Three (+2) challenges:
  - – Classification challenge (is there an X in this image?)
  - – Detection challenge (draw a box around every X)
  - – Segmentation challenge (which class is each pixel?)

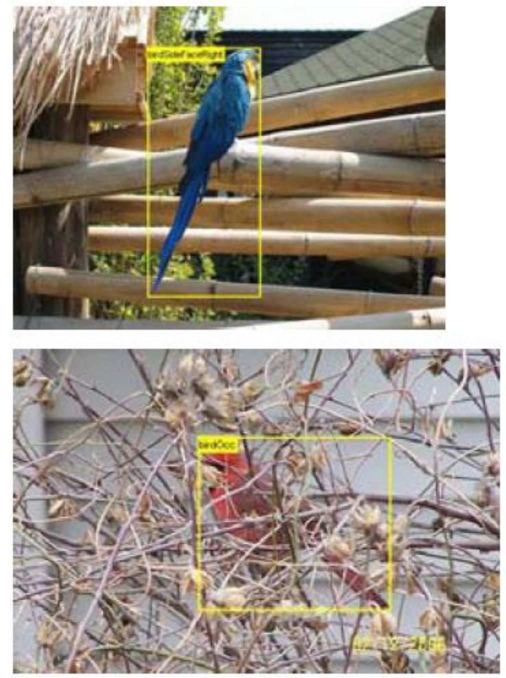


Examples

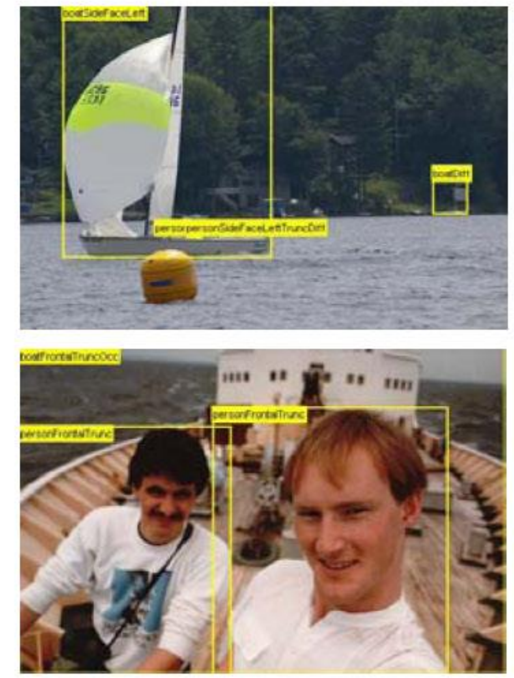
Aeroplane

Bicycle

Bird



Boat



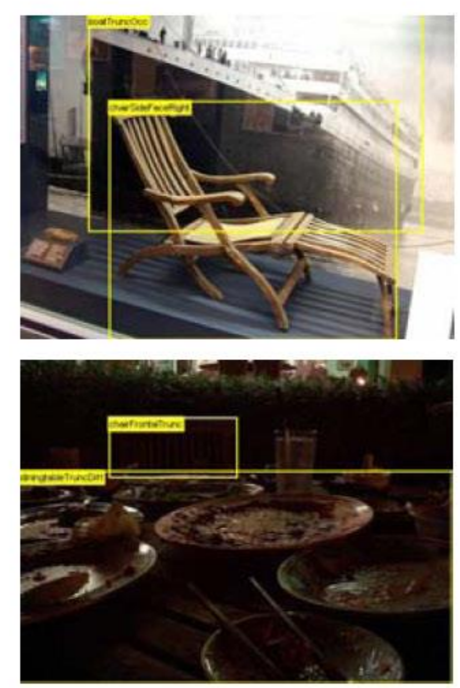
Bottle

Bus

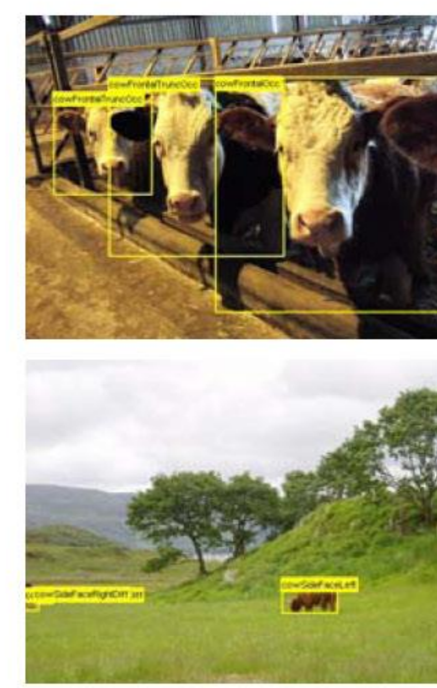
Car

Cat

Chair



Cow

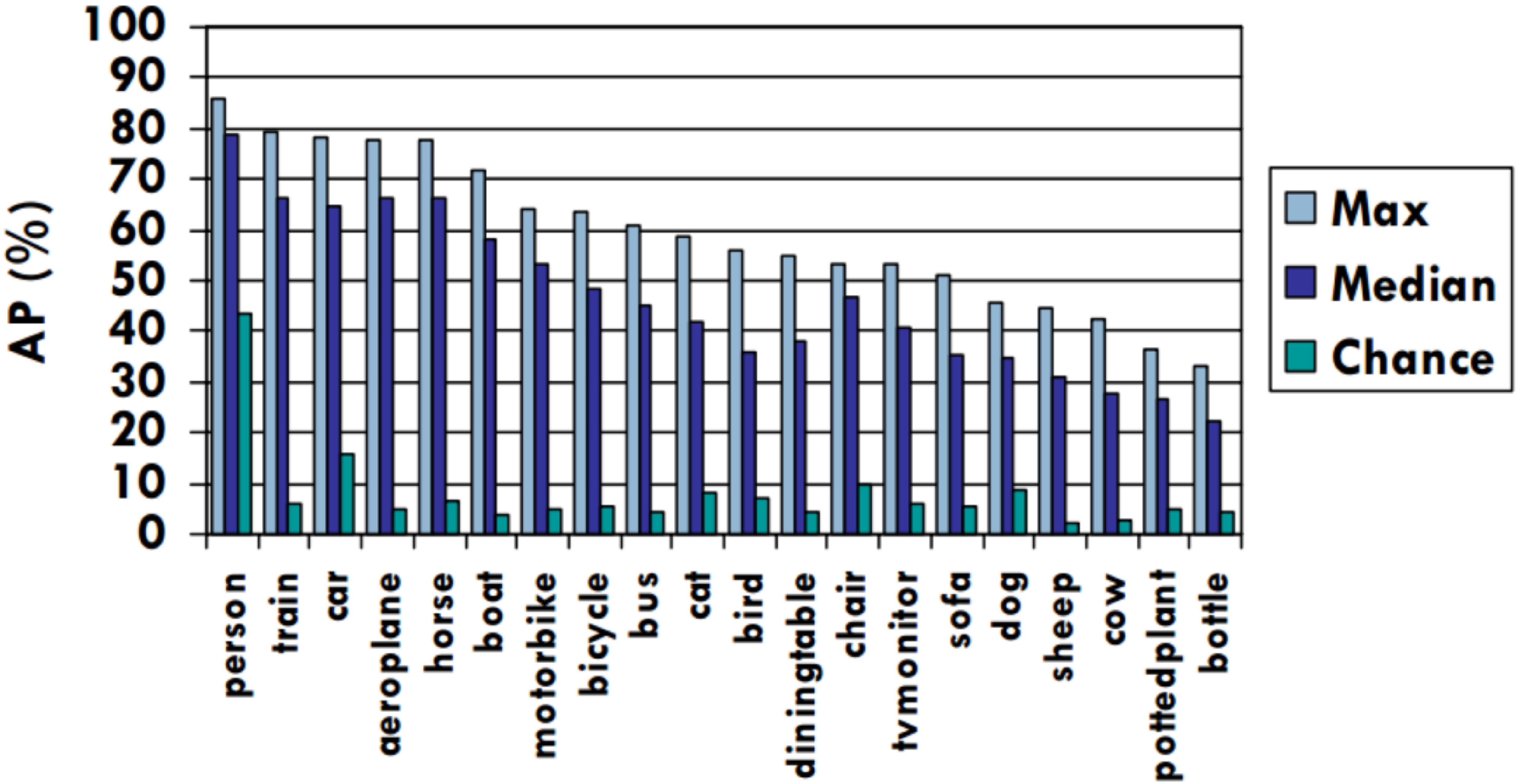


Classification Challenge

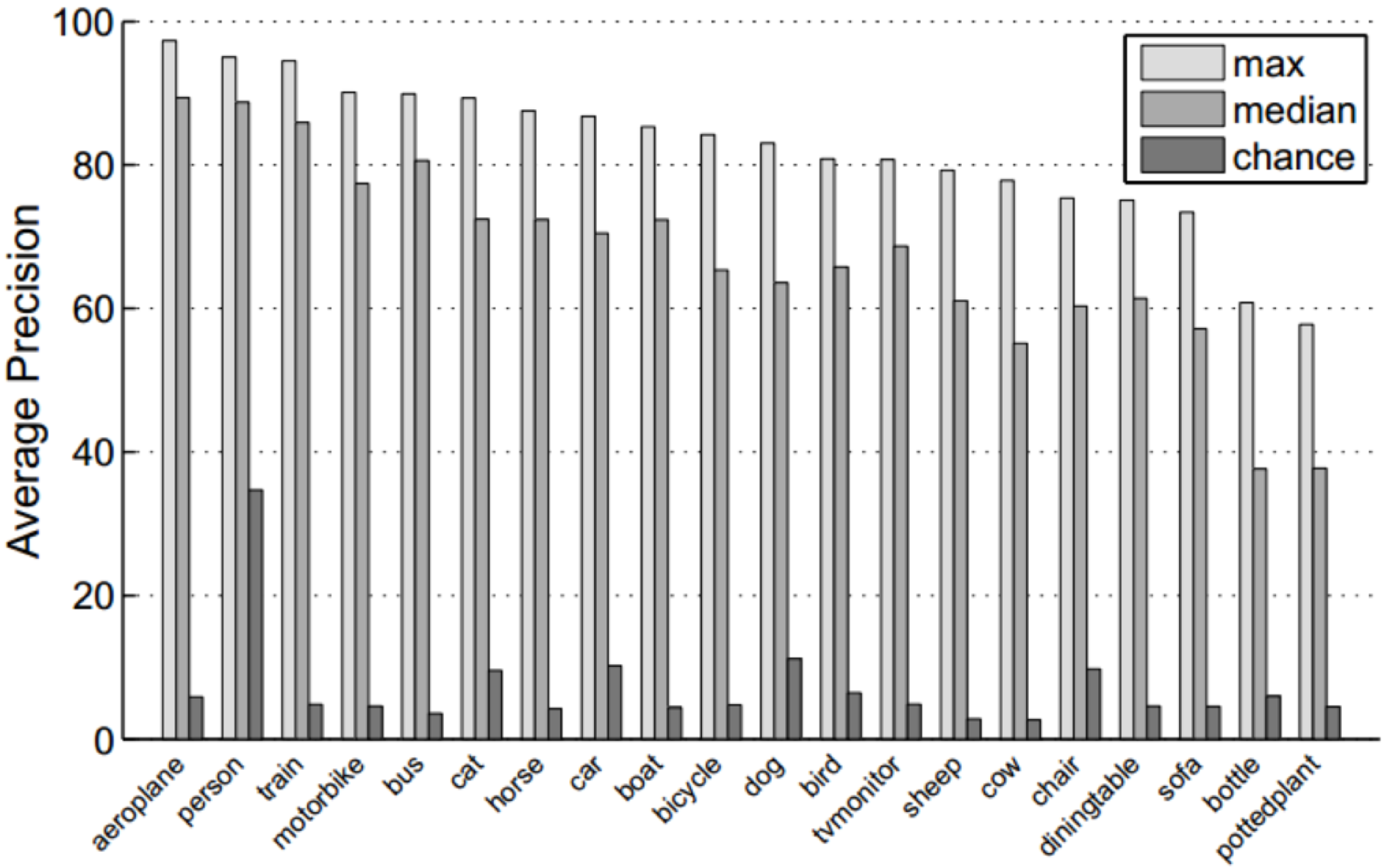
- Predict whether at least one object of a given class is present in an image

is there a cat?

Pascal VOC 2007 Average Precision



Pascal VOC 2012 Average Precision



Detection Challenge

- Predict the bounding boxes of all objects of a given class in an image (if any)

True Positives - Person

UoCTTI_LSVM-MDPM

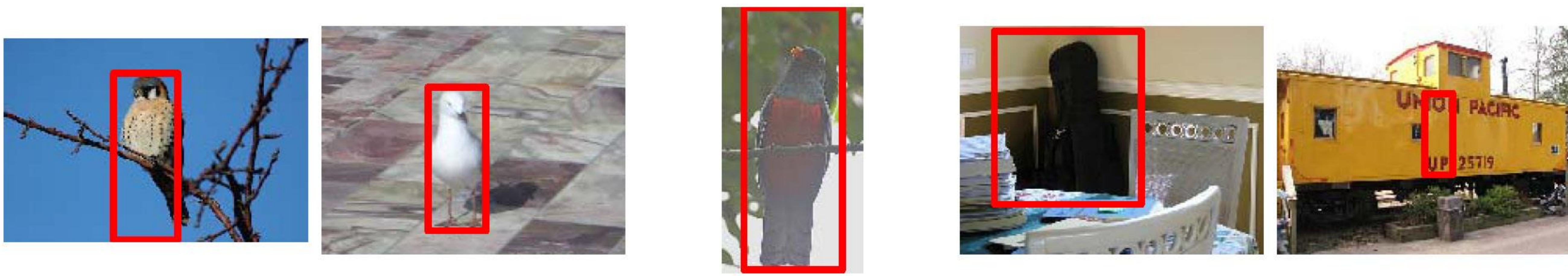
MIZZOU_DEF-HOG-LBP

NECUIUC_CLS-DTCT

False Positives - Person

UoCTTI_LSVM-MDPM

MIZZOU_DEF-HOG-LBP



NECUIUC_CLS-DTCT

“Near Misses” - Person

UoCTTI_LSVM-MDPM

MIZZOU_DEF-HOG-LBP

NECUIUC_CLS-DTCT

True Positives - Bicycle

UoCTTI_LSVM-MDPM

OXFORD_MKL

NECUIUC_CLS-DTCT

False Positives - Bicycle

UoCTTI_LSVM-MDPM

OXFORD_MKL

NECUIUC_CLS-DTCT

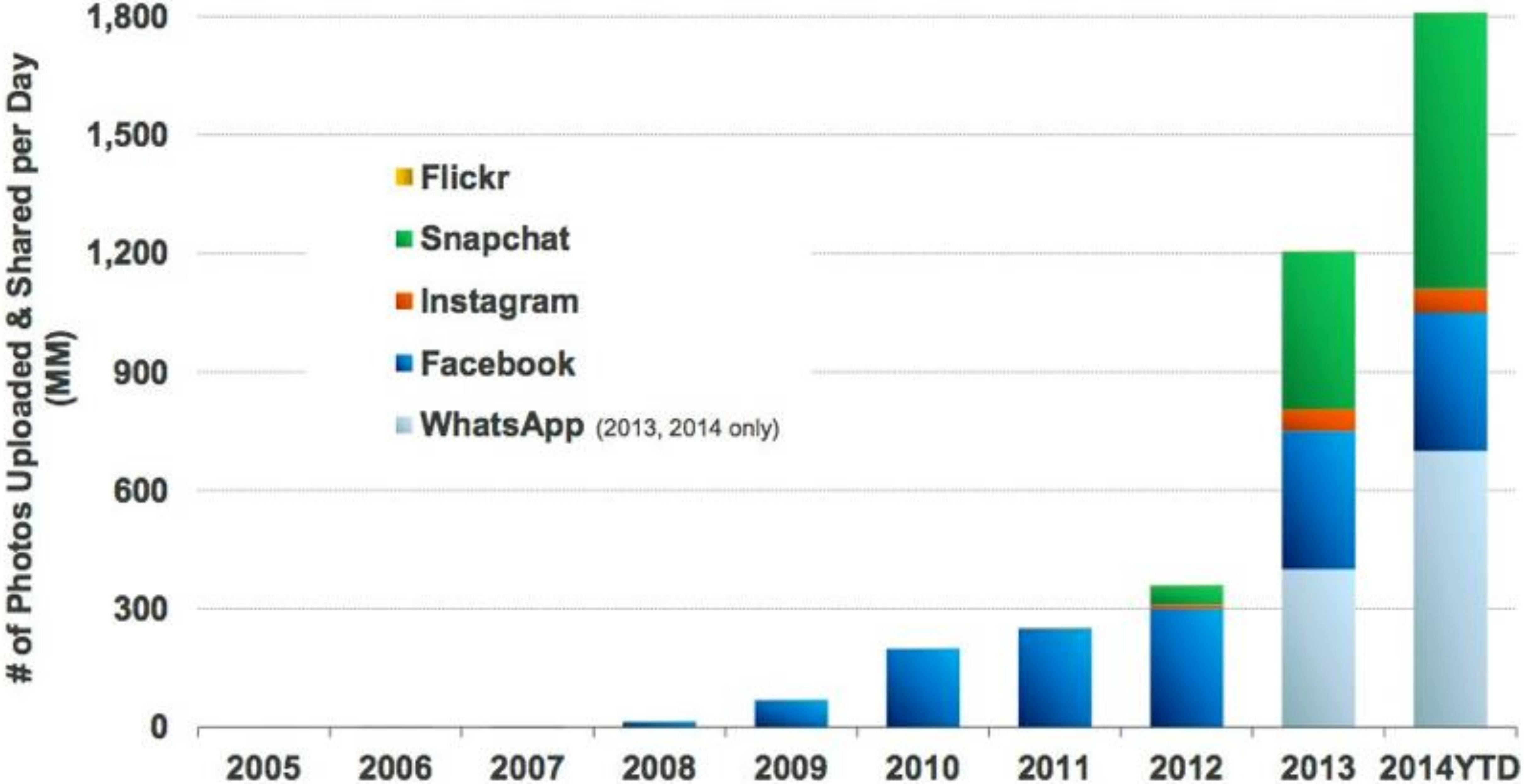
Where to from here?

- Scene Understanding
  - Big data – lots of images
  - Crowd-sourcing – lots of people
  - Deep Learning – lots of compute

24 Hrs in Photos

http://www.kesselskramer.com/exhibitions/24-hrs-of-photos installation by Erik Kessels

Daily Number of Photos Uploaded & Shared on Select Platforms, 2005 – 2014YTD



@KPCB Source: KPCB estimates based on publicly disclosed company data, 2014 YTD data per latest as of 5/14.

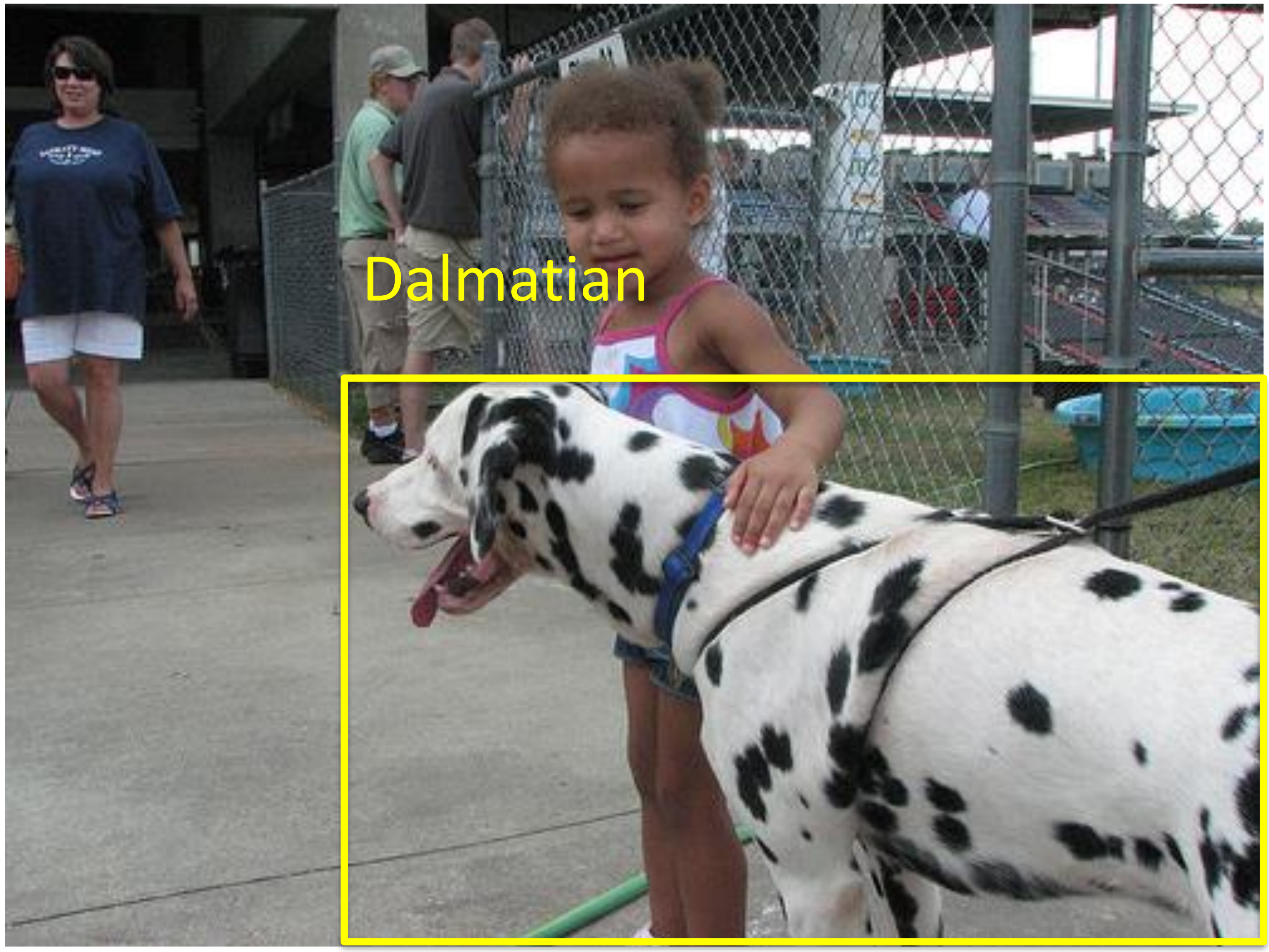
Data Sets

- ImageNet
  - Huge, Crowdsourced, Hierarchical, *Iconic* objects
- PASCAL VOC
  - *Not* Crowdsourced, bounding boxes, 20 categories
- SUN Scene Database, Places
  - *Not* Crowdsourced, 397 (or 720) scene categories
- LabelMe (Overlaps with SUN)
  - Sort of Crowdsourced, Segmentations, Open ended
- SUN *Attribute* database (Overlaps with SUN)
  - Crowdsourced, 102 attributes for every scene
- OpenSurfaces
  - Crowdsourced, materials
- Microsoft COCO
  - Crowdsourced, large-scale objects

IMAGENET Large Scale Visual Recognition Challenge (ILSVRC) 2010-2012

~~20 object classes 22,591 images~~

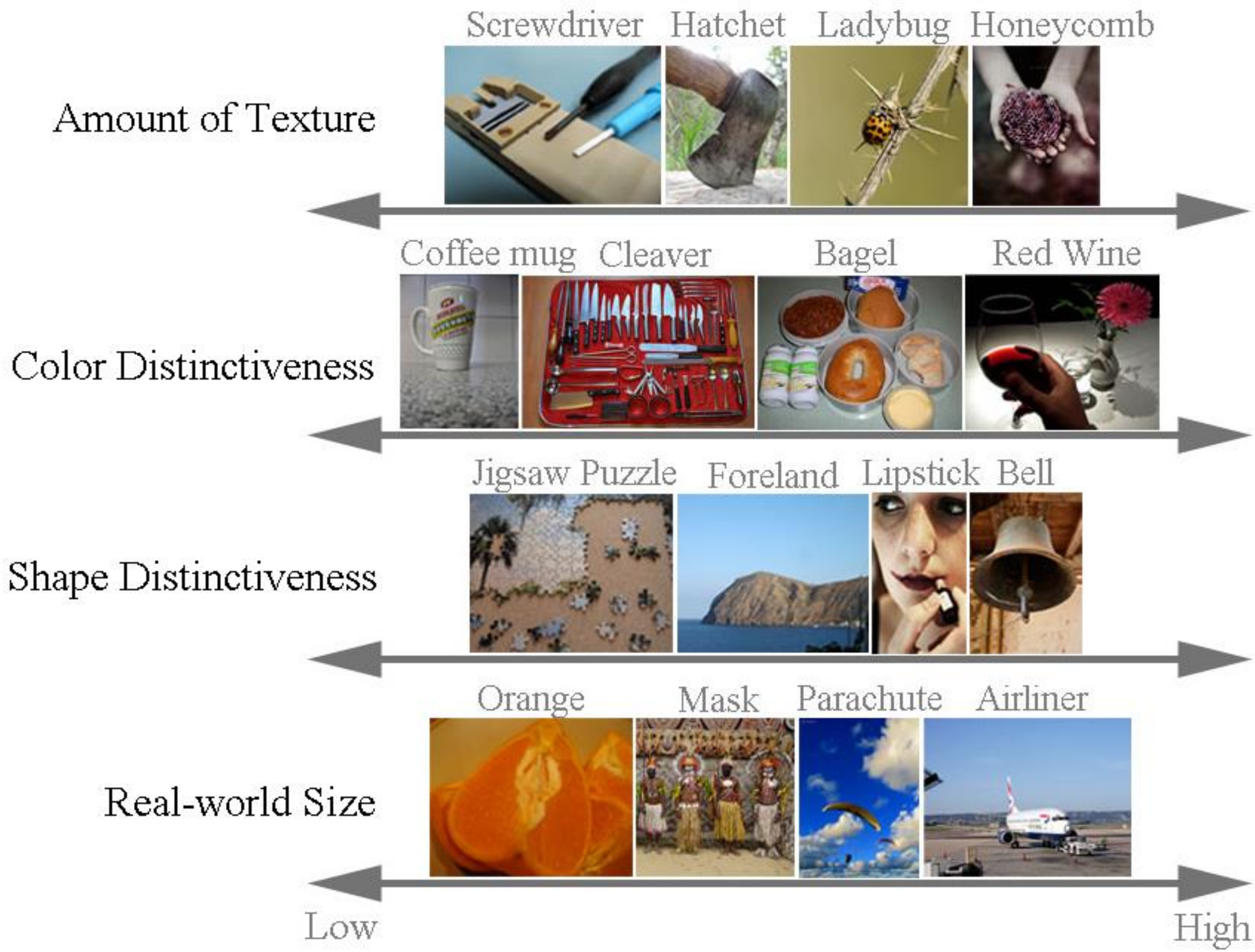
1000 object classes **1,431,167 images**



http://image-net.org/challenges/LSVRC/{2010,2011,2012}

Variety of object classes in ILSVRC

Variety of object classes in ILSVRC




7 APR 2007

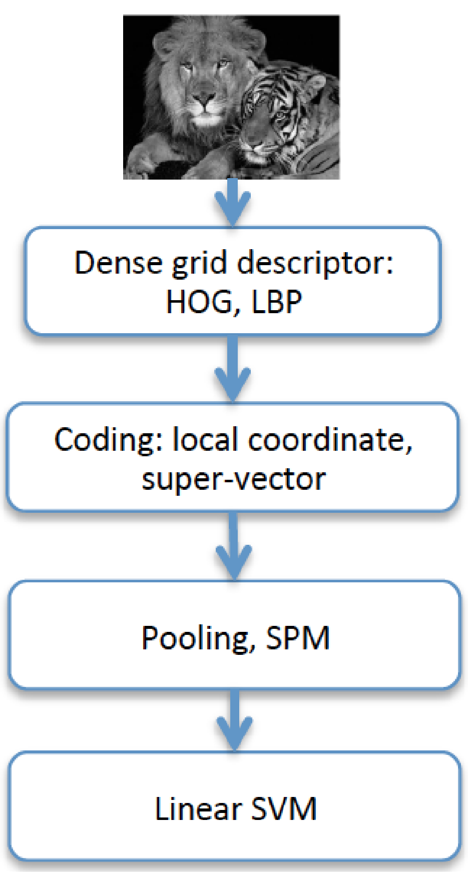
Deep Learning or CNNs

- Since 2012, huge impact..., best results
- Can soak up all the data for better prediction

IMAGENET Large Scale Visual Recognition Challenge

Year 2010

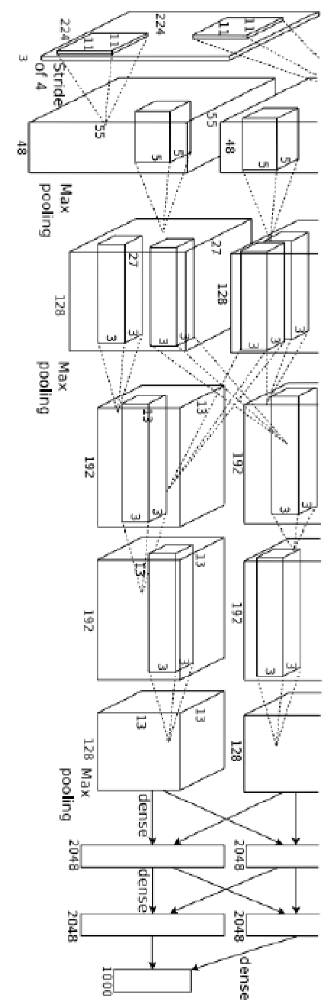
NEC-UIUC



[Lin CVPR 2011]

Year 2012

SuperVision



[Krizhevsky NIPS 2012]

Year 2014

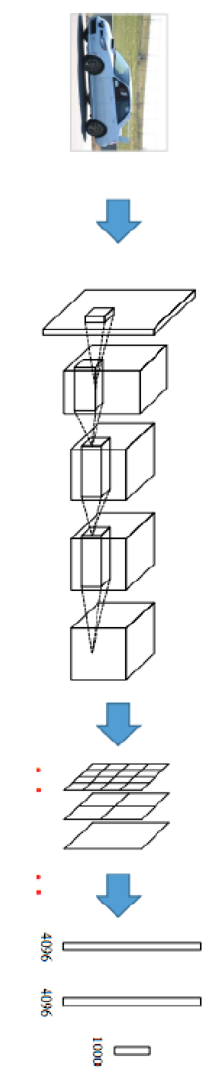
GoogLeNet

[Szegedy arxiv 2014]

VGG

[Simonyan arxiv 2014]

MSRA



[He arxiv 2014]

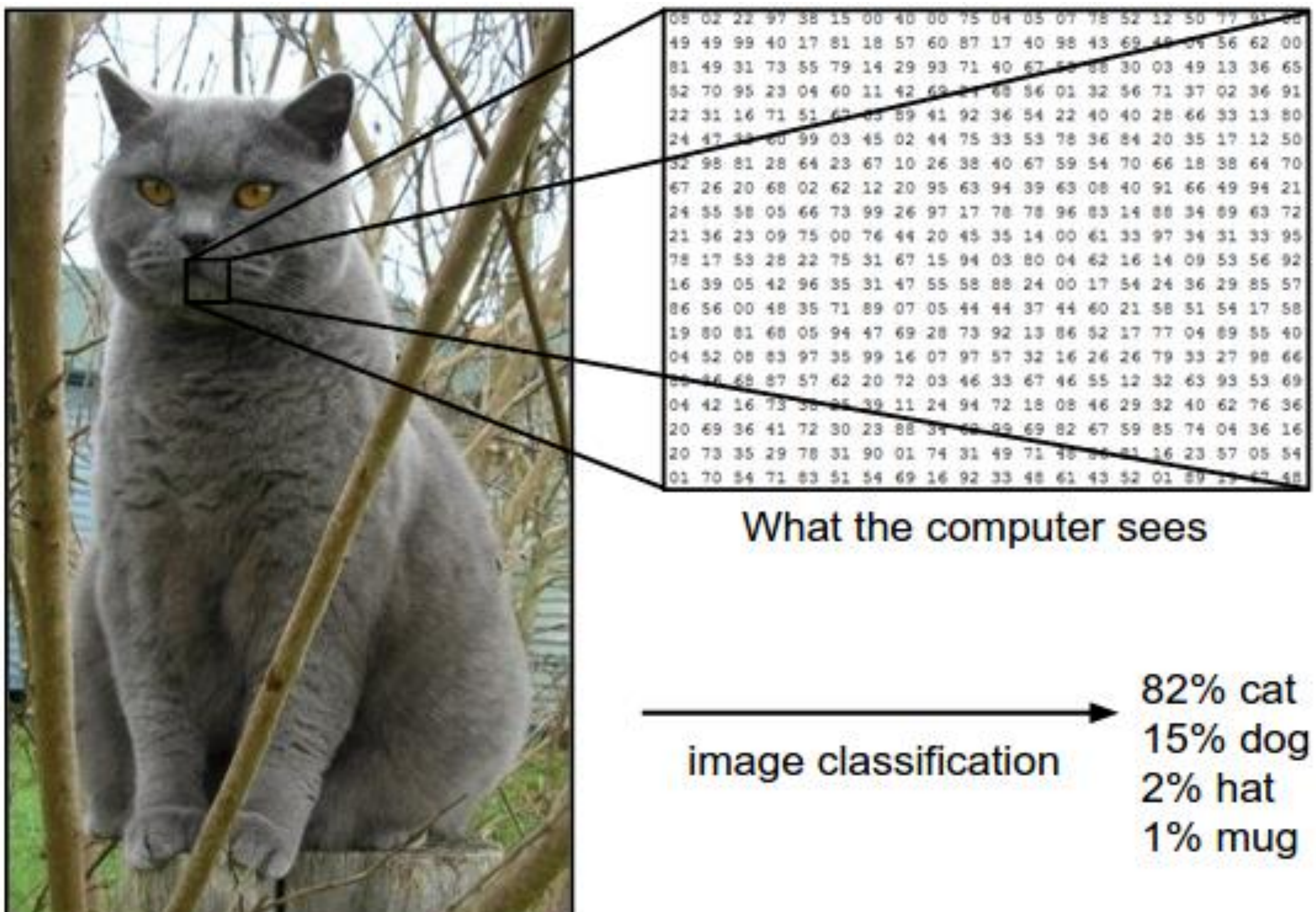
Image classification

Image Classification

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

→ cat

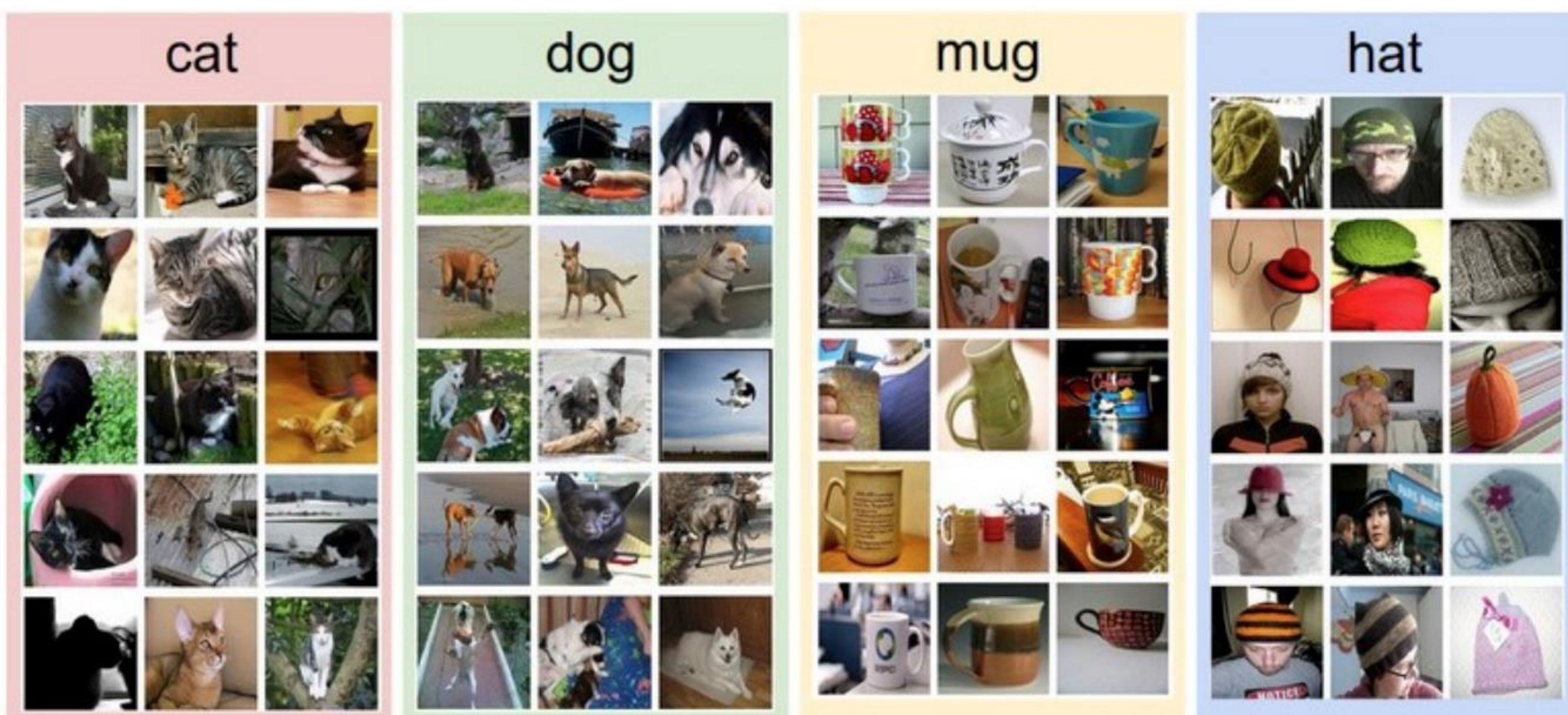
Image Classification: Problem



Data-driven approach

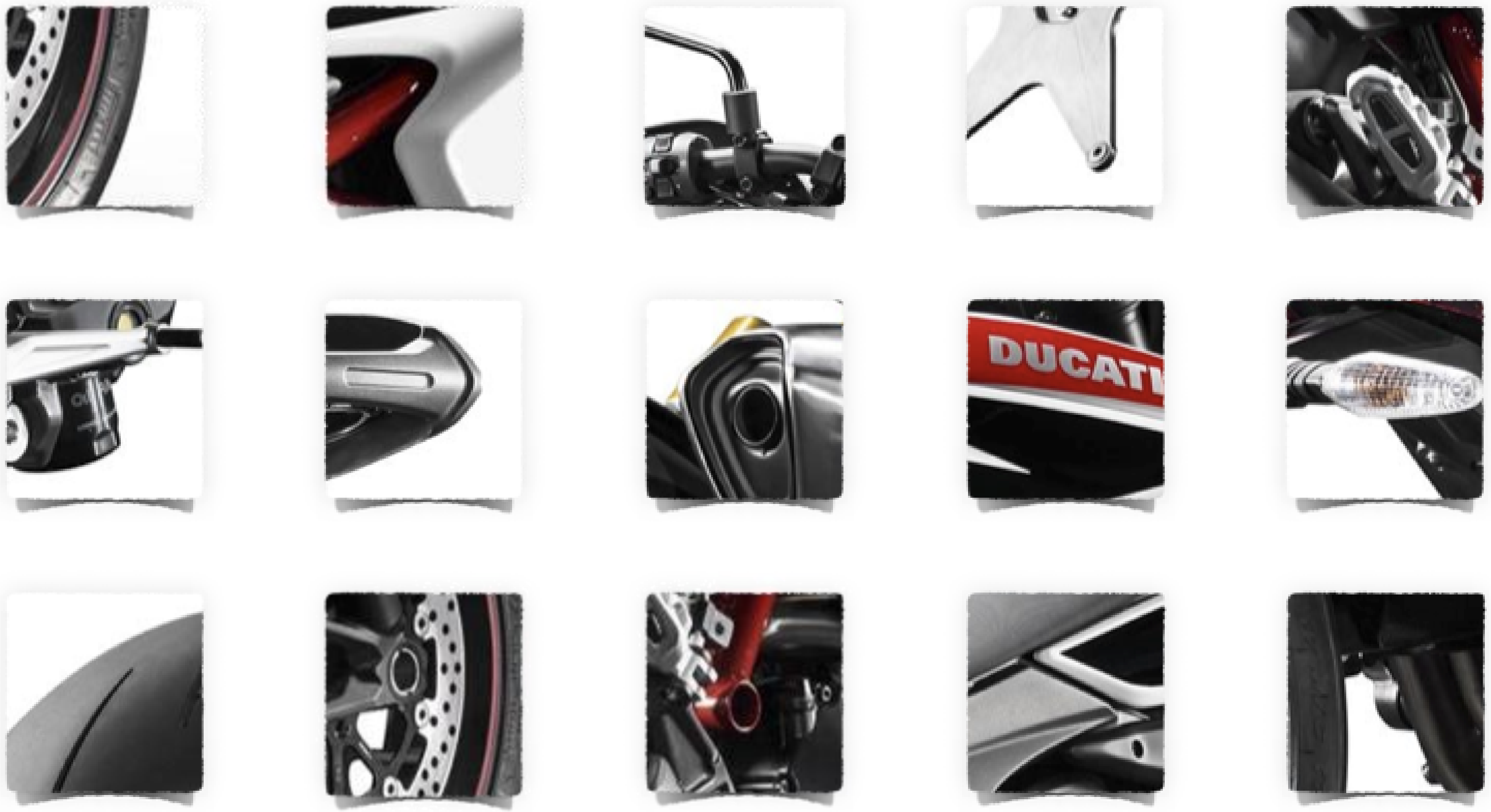
- Collect a database of images with labels
- Use ML to train an image classifier
- Evaluate the classifier on test images

Example training set



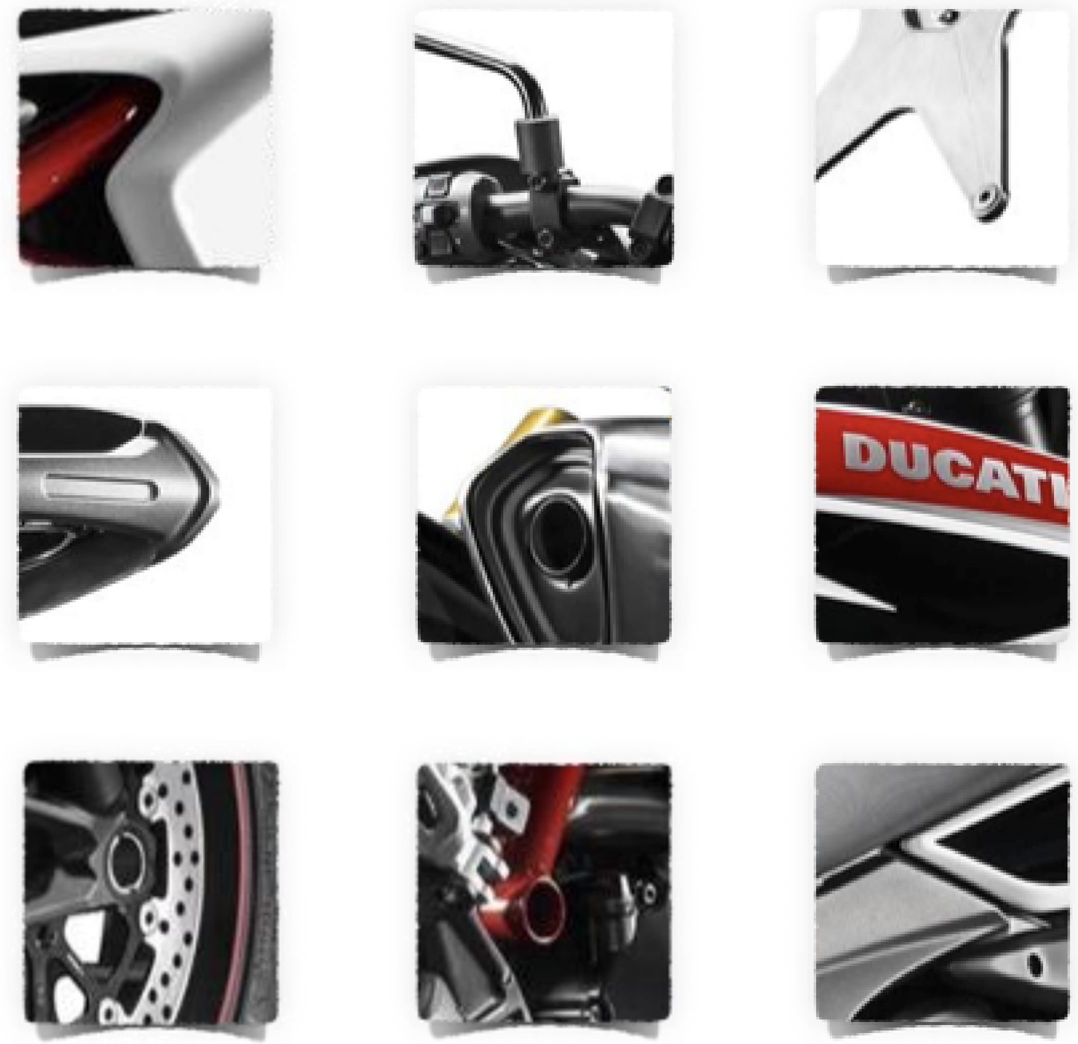
Bag of words

What object do these parts belong to?



Some local feature are very informative

An object as

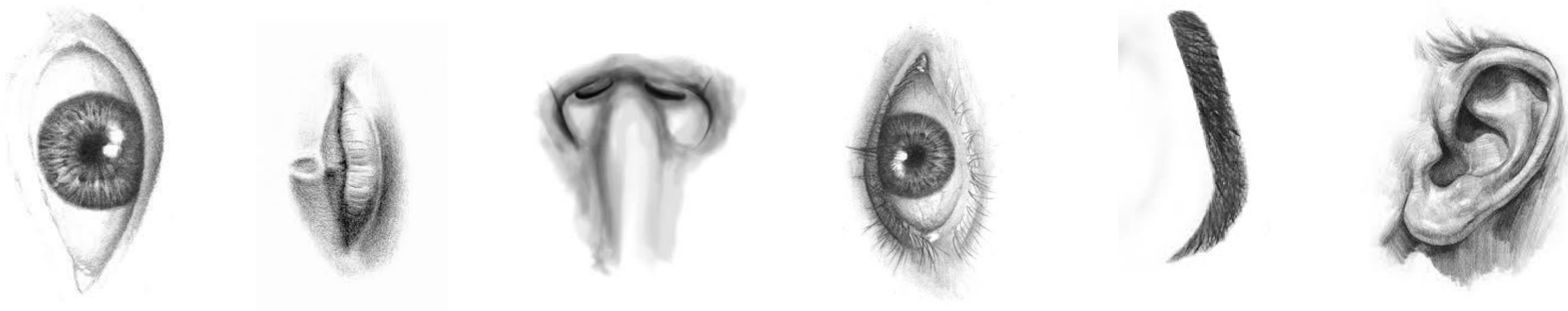


a collection of local features

(bag-of-features)

- deals well with occlusion
- scale invariant
- rotation invariant

(not so) crazy assumption



spatial information of local features
can be ignored for object recognition (i.e., verification)

CalTech6 dataset

class	bag of features Zhang et al. (2005)	bag of features Willamowski et al. (2004)	Parts-and-shape model Fergus et al. (2003)
airplanes	**98.8**	97.1	90.2
cars (rear)	98.3	**98.6**	90.3
cars (side)	**95.0**	87.3	88.5
faces	**100**	99.3	96.4
motorbikes	**98.5**	98.0	92.5
spotted cats	**97.0**	—	90.0

Works pretty well for image-level classification

Csurka et al. (2004), Willamowski et al. (2005), Grauman & Darrell (2005), Sivic et al. (2003, 2005)

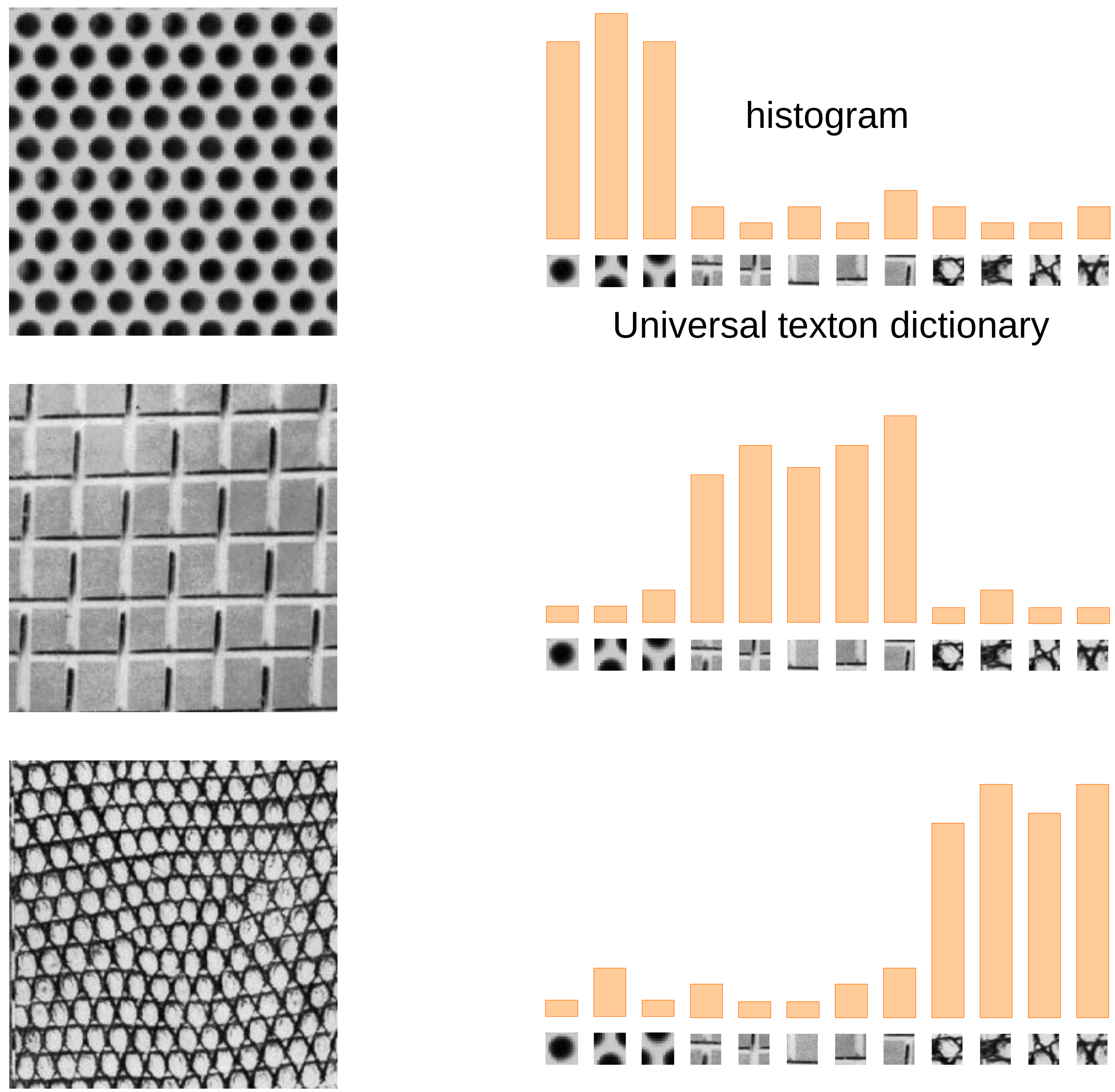
Bag-of-features

represent a data item (document, texture, image)
as a histogram over features

an old idea

(e.g., texture recognition and information retrieval)

Texture recognition



Vector Space Model

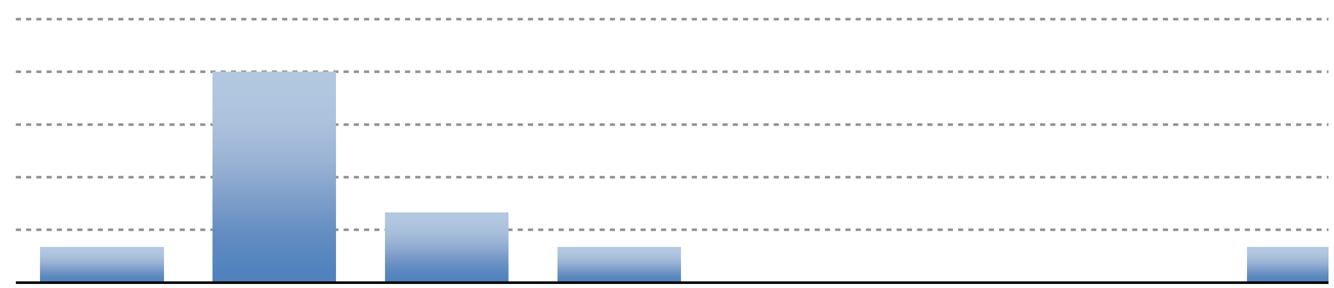
G. Salton. ‘Mathematics and Information Retrieval’ Journal of Documentation, 1979

The Newsp

Sunday, December 22, 2013

DARPA Selects Carnegie Me

The Tartan Rescue Team from Carnegie Mellon University's Robotics Engineering National Robotics Engineering Center ranked third among teams competing in the Defense Advanced Research Projects Agency (DARPA) Robotics Challenge Trials this weekend in Homestead, Fla., and was selected by the agency as one of eight teams eligible for DARPA funding to prepare for next December's finals. The team's four-limbed CMU Highly Intelligent Mobile Platform, or CHIMP, robot scored 18 out of a possible 32 points during the two-day trials. It demonstrated its ability to perform such tasks as removing debris, cutting a hole through a wall and closing a series of valves.



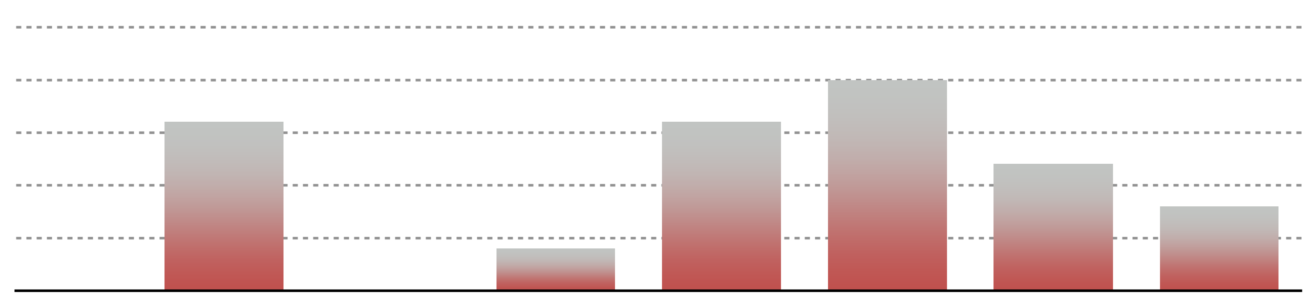
1	6	2	1	0	0	0	1
Tartan	robot	CHIMP	CMU	bio	soft	ankle	sensor

Tartan Tim

Monday, January 20, 2014

Bio-Inspired Robotic Device

PITTSBURGH—A soft, wearable device that mimics the muscles, tendons and ligaments of the lower leg could aid in the rehabilitation of patients with ankle-foot disorders such as drop foot, said Yong-Lae Park, an assistant professor of robotics at Carnegie Mellon University. Park, working with collaborators at Harvard University, the University of Southern California, MIT and BioSensics, developed an active orthotic device using soft plastics and composite materials, instead of a rigid exoskeleton. The soft materials, combined with pneumatic artificial muscles (PAMs), lightweight sensors and advanced control software, made it possible for the robotic device to achieve natural motions in the ankle.



0	4	0	1	4	5	3	2
Tartan	robot	CHIMP	CMU	bio	soft	ankle	sensor

A document (datapoint) is a vector of counts over each word (feature)

$$\boldsymbol{v}_d = [n(w_{1,d}) \;\; n(w_{2,d}) \;\; \cdots \;\; n(w_{T,d})]$$

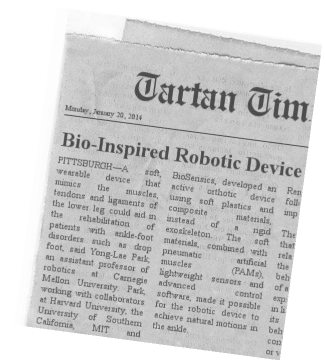
$n(\cdot)$ counts the number of occurrences

just a histogram over words

What is the similarity between two documents?

The Newspa

DARPA Selects Carnegie Me



Tartan Tim

Bio-Inspired Robotic Device

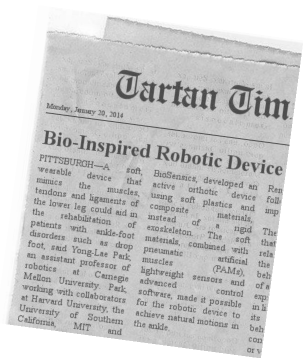
A document (datapoint) is a vector of counts over each word (feature)

$$\boldsymbol{v}_d = [n(w_{1,d}) \quad n(w_{2,d}) \quad \cdots \quad n(w_{T,d})]$$

just a histogram over words

$n(\cdot)$ counts the number of occurrences

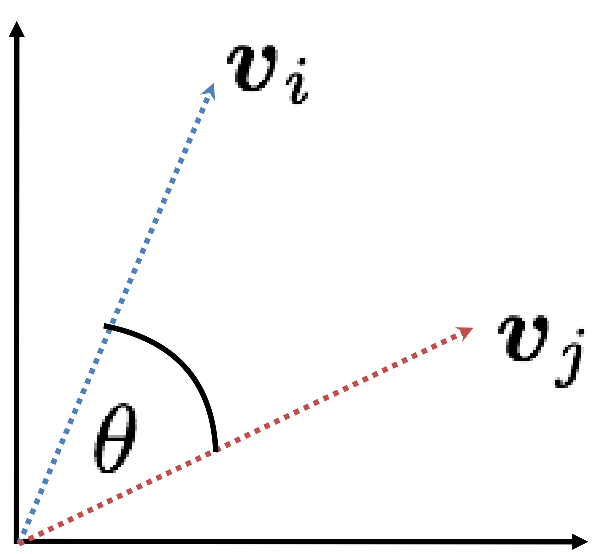
What is the similarity between two documents?

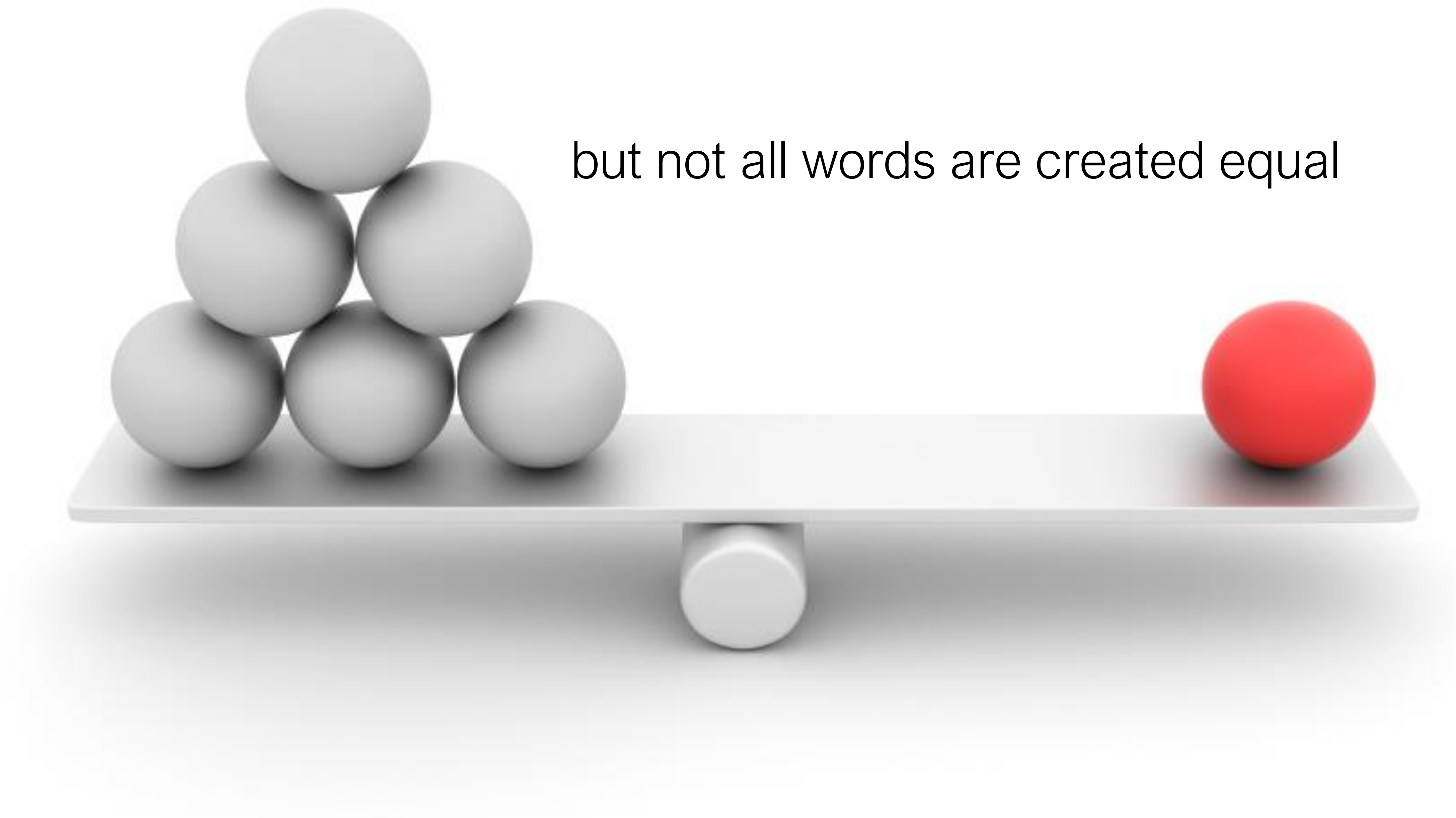


Use any distance you want but the cosine distance is fast.

$$d(\boldsymbol{v}_i, \boldsymbol{v}_j) = \cos\theta$$

$$= \frac{\boldsymbol{v}_i \cdot \boldsymbol{v}_j}{\|\boldsymbol{v}_i\|\|\boldsymbol{v}_j\|}$$





but not all words are created equal

TF-IDF

Term **F**requency **I**nverse **D**ocument **F**requency

$$\boldsymbol{v}_d = [n(w_{1,d}) \quad n(w_{2,d}) \quad \cdots \quad n(w_{T,d})]$$

weigh each word by a heuristic

$$\boldsymbol{v}_d = [n(w_{1,d})\alpha_1 \quad n(w_{2,d})\alpha_2 \quad \cdots \quad n(w_{T,d})\alpha_T]$$

$$n(w_{i,d})\alpha_i = \underbrace{n(w_{i,d})}_{\text{term frequency}} \log \underbrace{\left\{ \frac{D}{\sum_{d'} \mathbf{1}[w_i \in d']} \right\}}_{\text{inverse document frequency}}$$

(down-weights **common** terms)

Standard BOW pipeline

(for image classification)

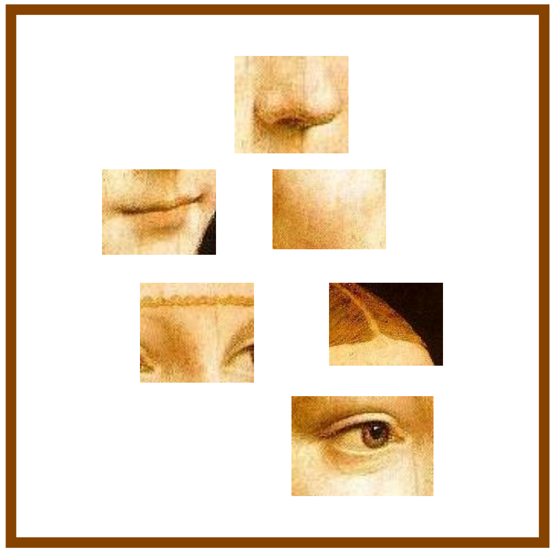
Dictionary Learning:
Learn Visual Words using clustering

Encode:
build Bags-of-Words (BOW) vectors
for each image

Classify:
Train and test data using BOWs

Dictionary Learning:
Learn Visual Words using clustering

1. extract features (e.g., SIFT) from images

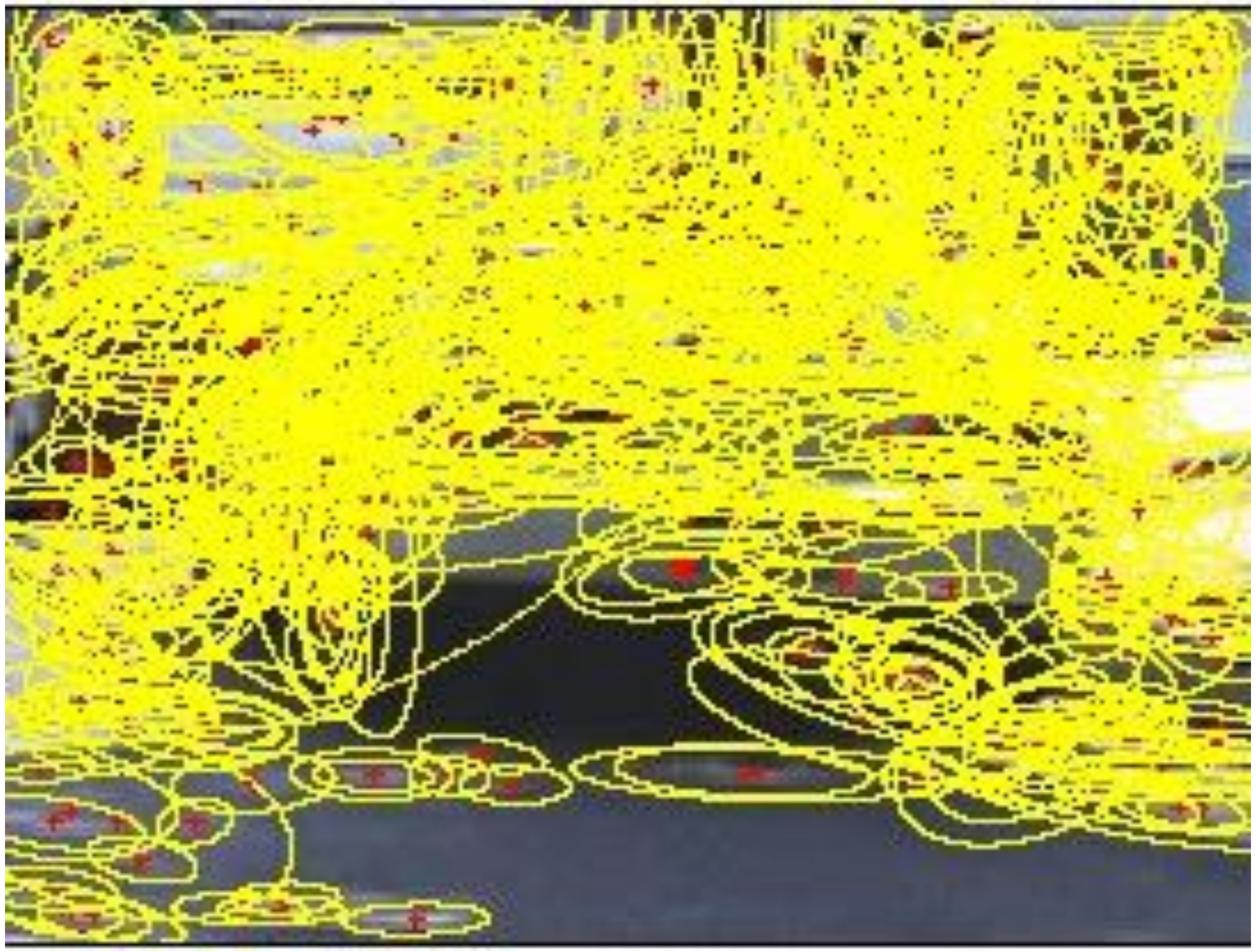


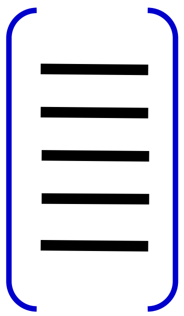
Dictionary Learning:
Learn Visual Words using clustering

2. Learn visual dictionary (e.g., K-means clustering)

What kinds of features can we extract?

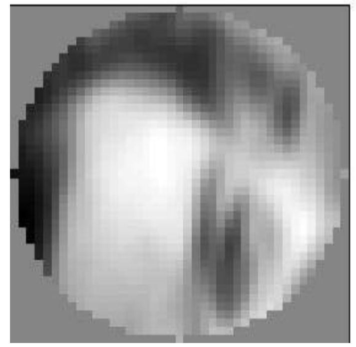
- Regular grid
  - Vogel & Schiele, 2003
  - Fei-Fei & Perona, 2005
- Interest point detector
  - Csurka et al. 2004
  - Fei-Fei & Perona, 2005
  - Sivic et al. 2005
- Other methods
  - Random sampling (Vidal-Naquet & Ullman, 2002)
  - Segmentation-based patches (Barnard et al. 2003)



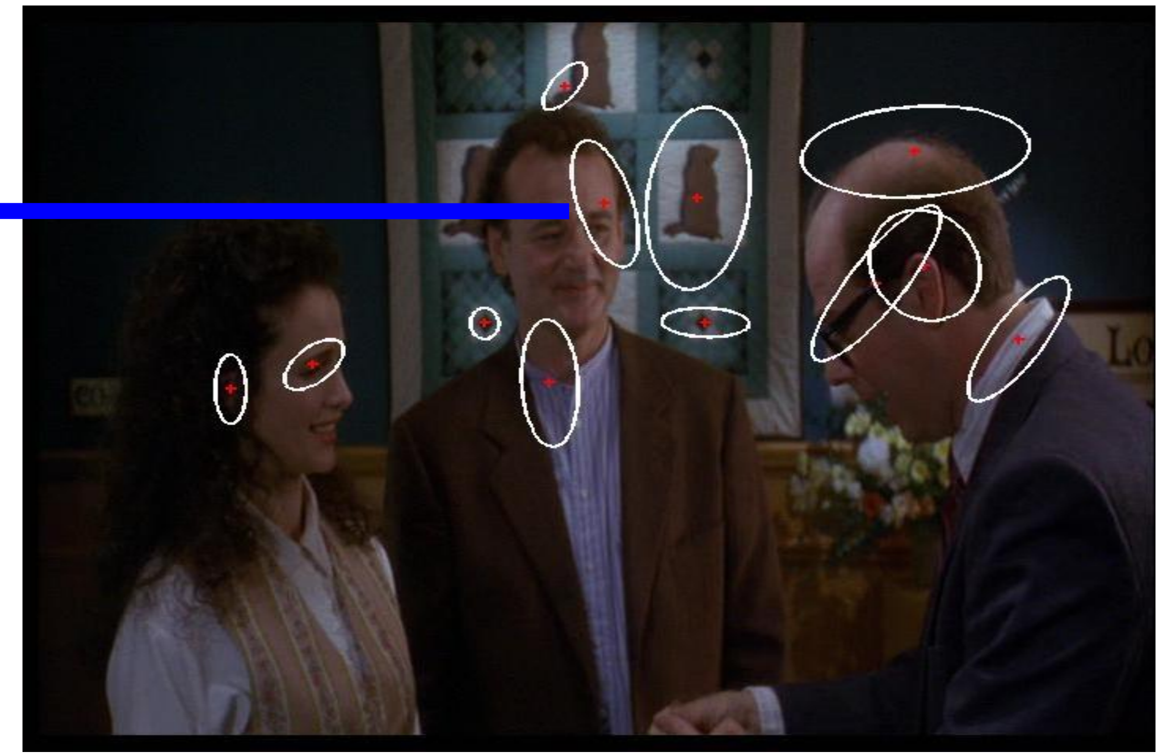


Compute SIFT descriptor

[Lowe’99]



Normalize patch

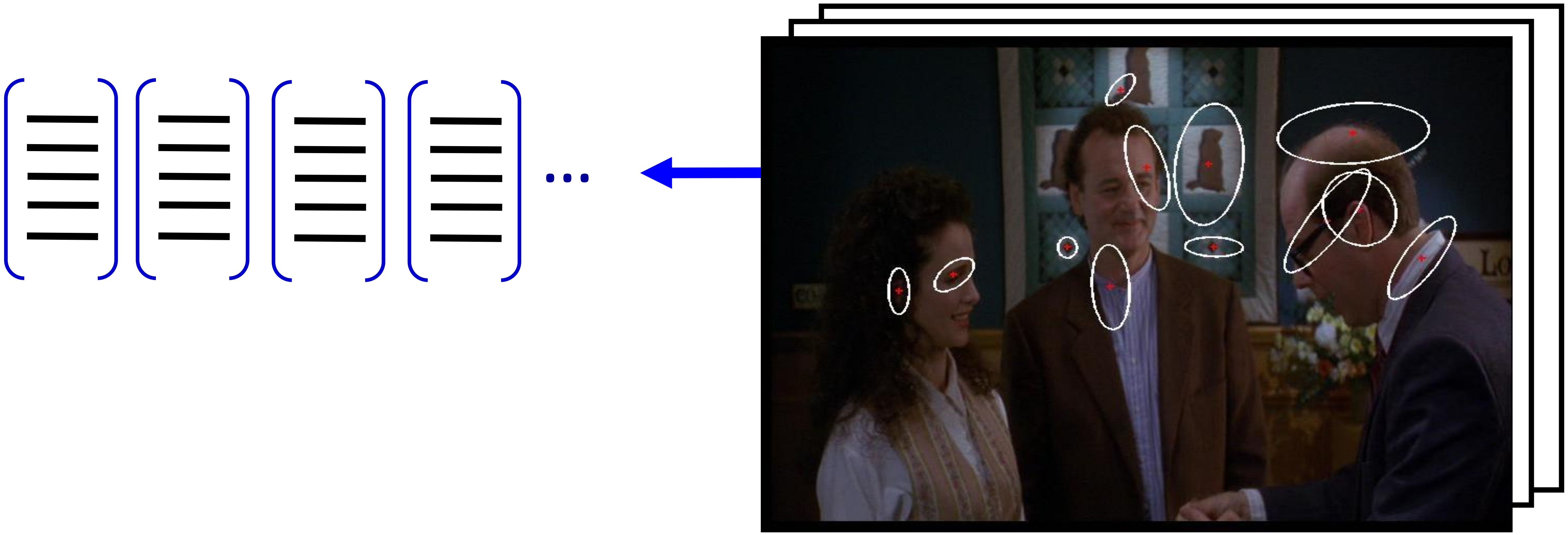


Detect patches

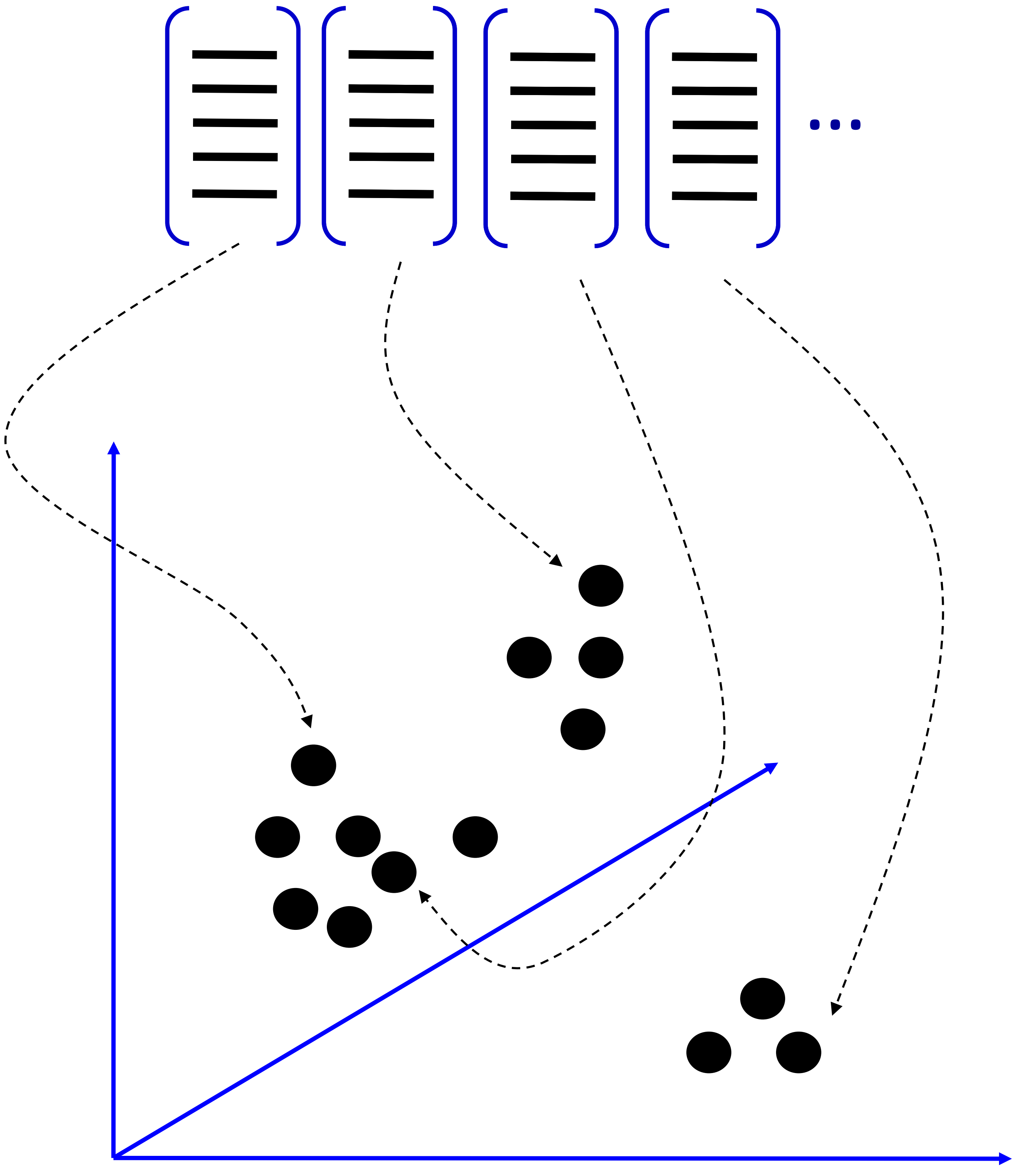
[Mikojaczyk and Schmid ’02]

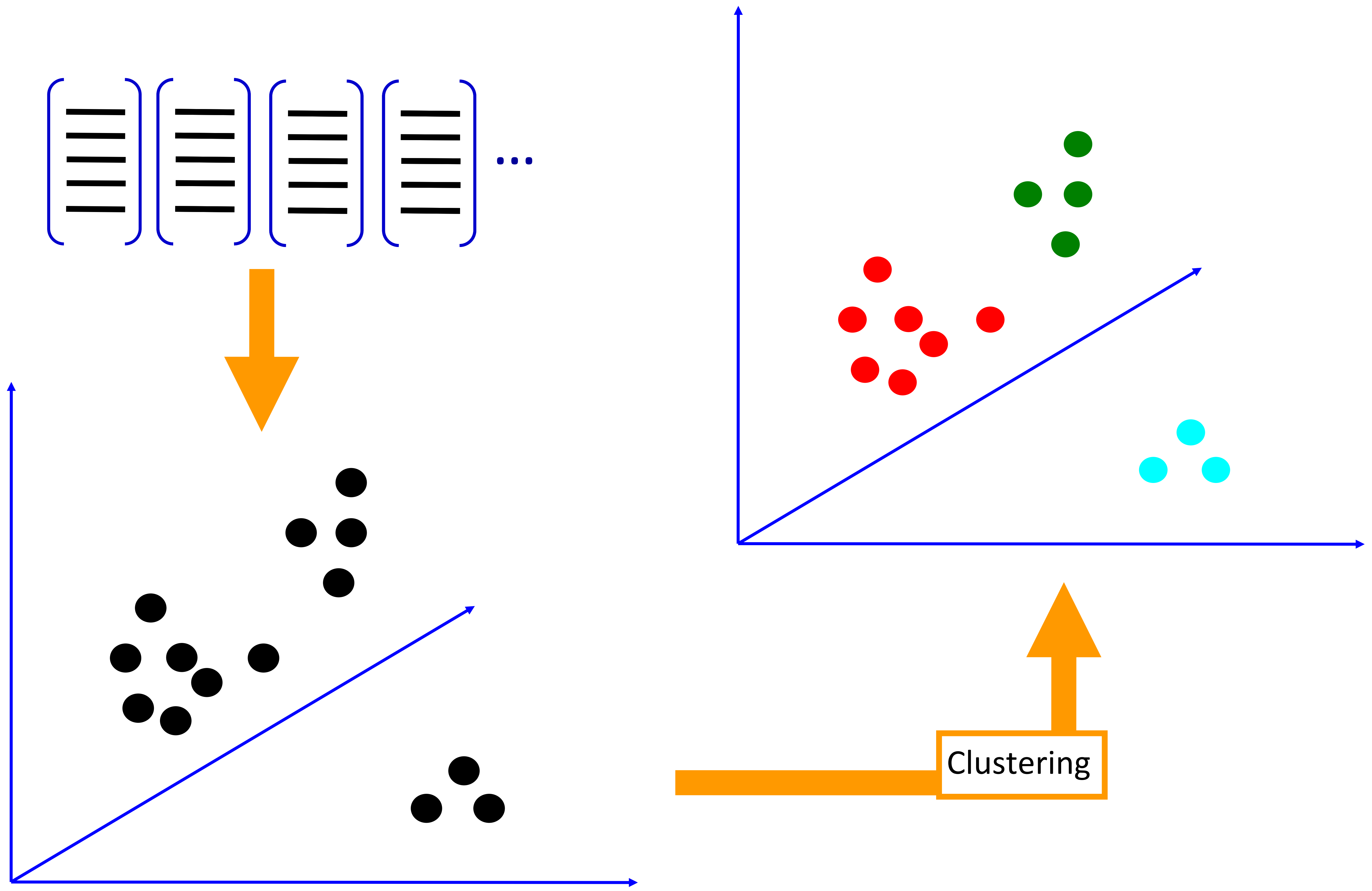
[Mata, Chum, Urban & Pajdla, ’02]

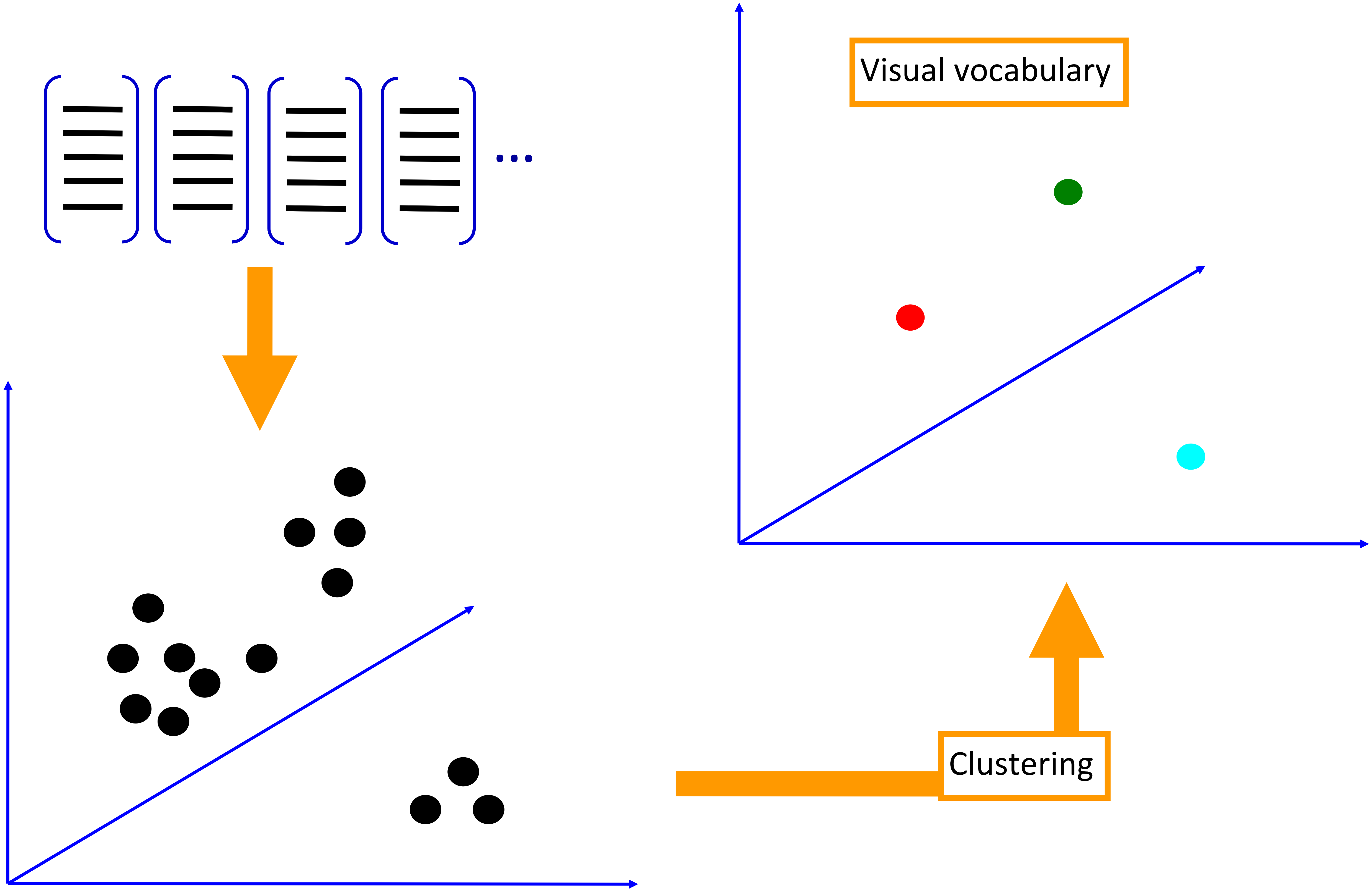
[Sivic & Zisserman, ’03]



How do we learn the dictionary?

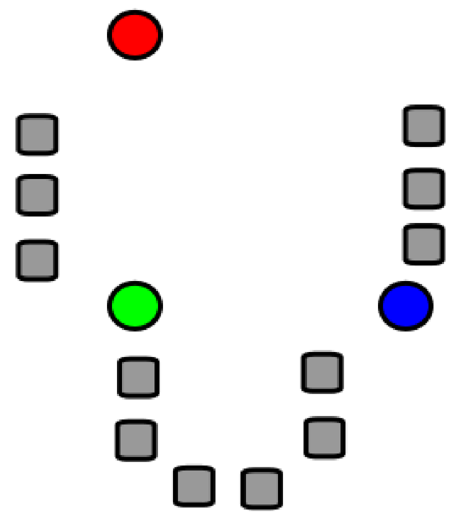



Clustering

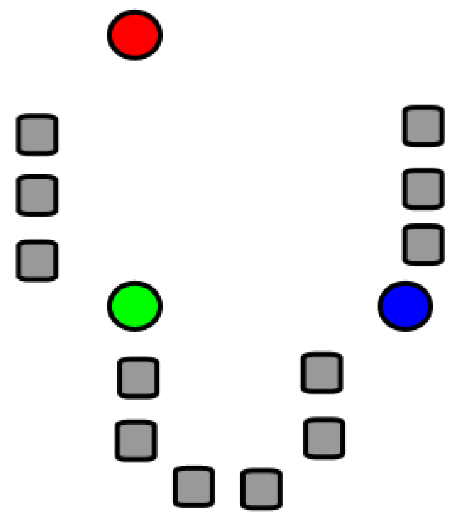


Visual vocabulary
Clustering

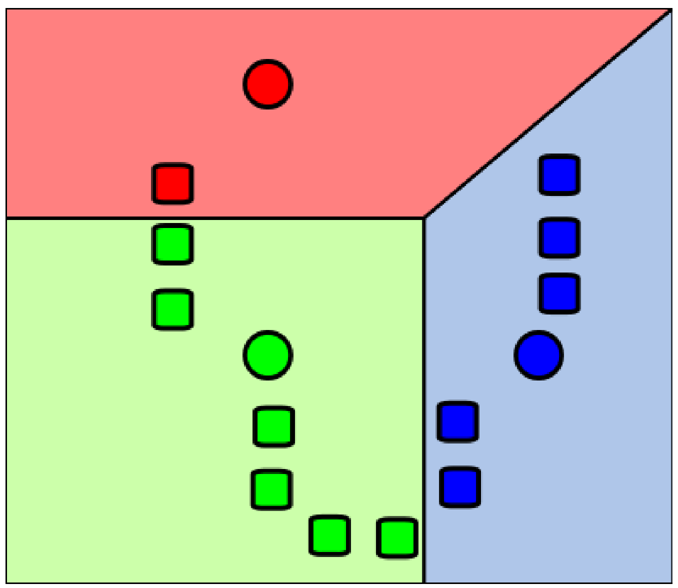
K-means clustering



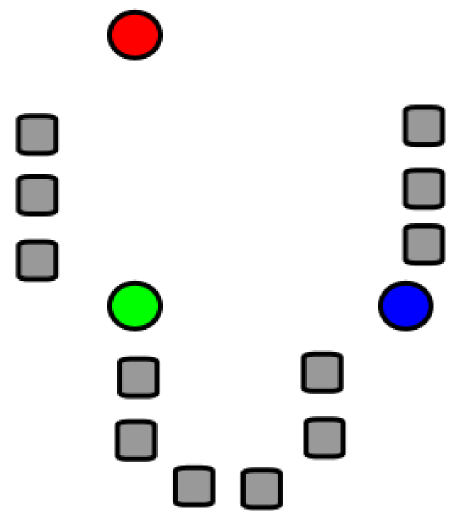
1. Select initial
centroids at random



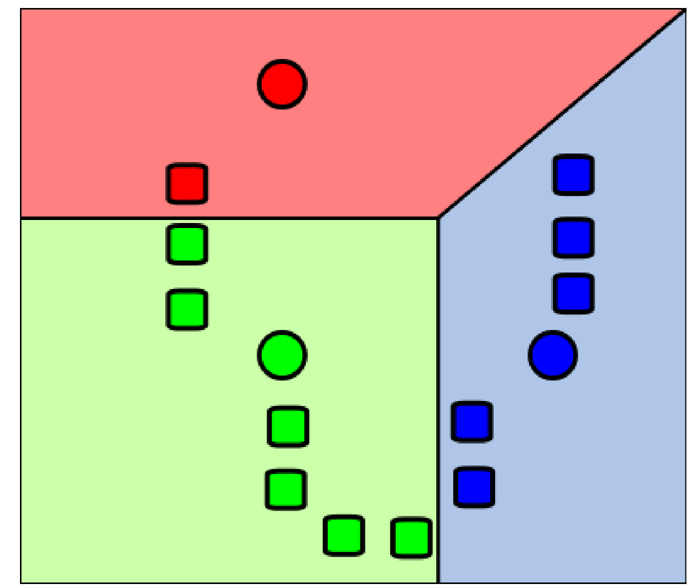
1. Select initial centroids at random



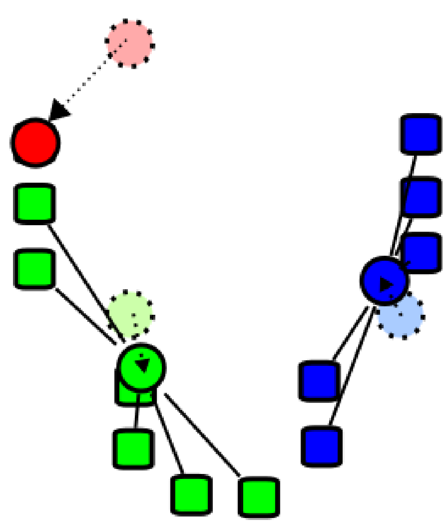
2. Assign each object to the cluster with the nearest centroid.



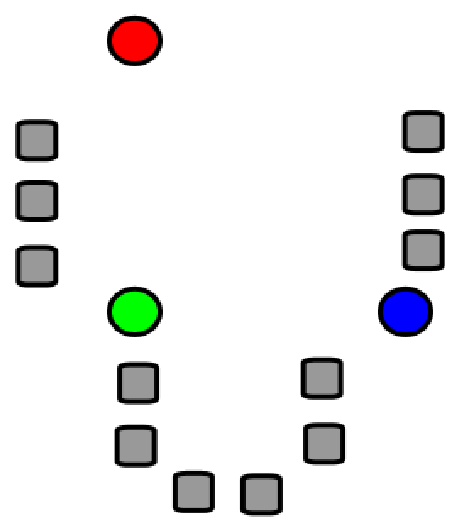
1. Select initial centroids at random



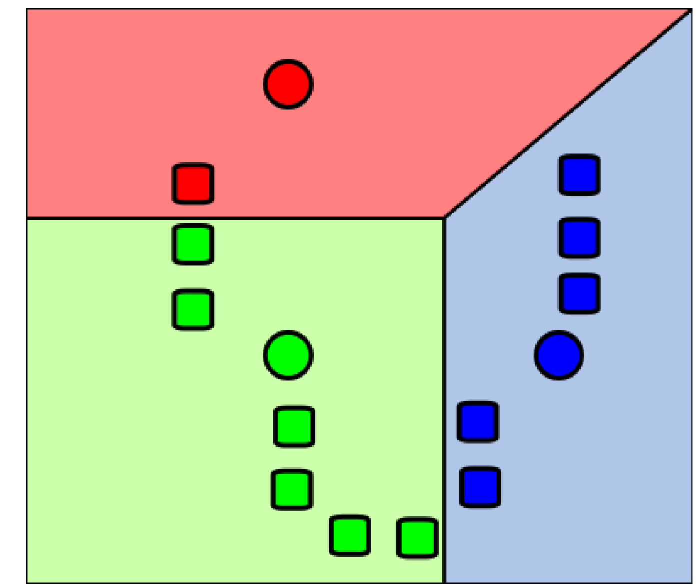
2. Assign each object to the cluster with the nearest centroid.



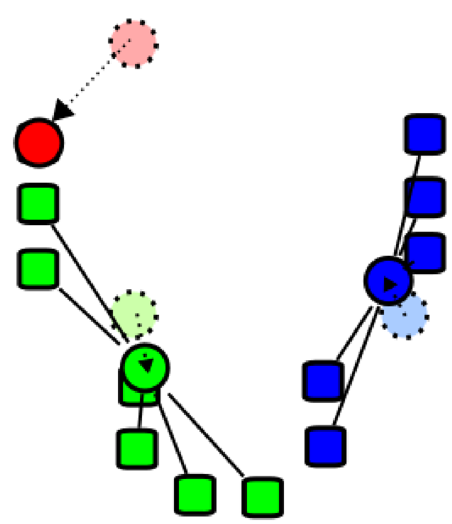
3. Compute each centroid as the mean of the objects assigned to it (go to 2)



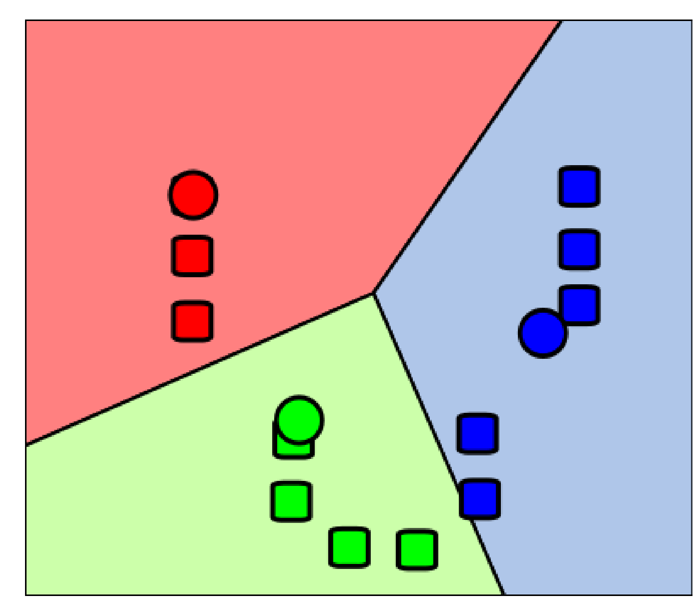
1. Select initial centroids at random



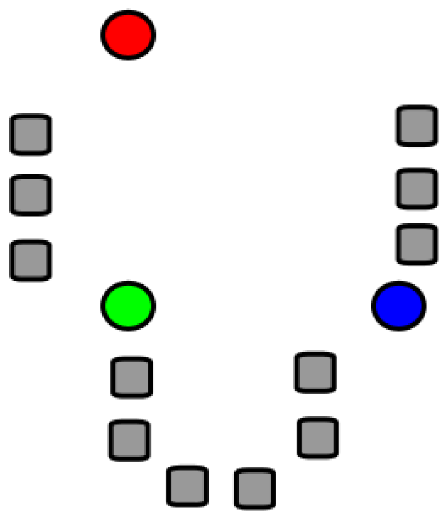
2. Assign each object to the cluster with the nearest centroid.



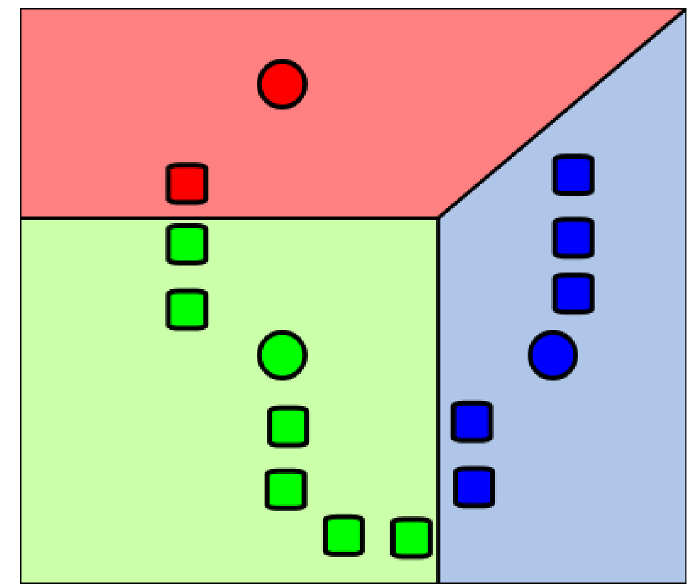
3. Compute each centroid as the mean of the objects assigned to it (go to 2)



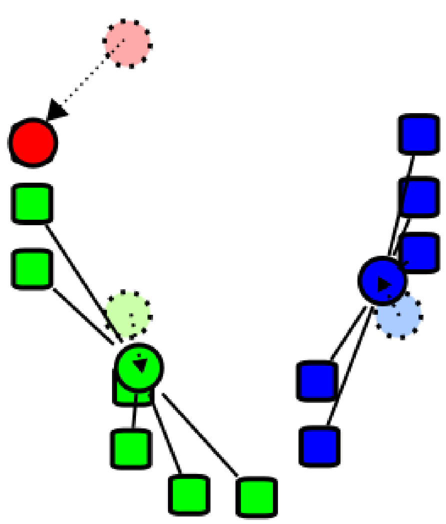
2. Assign each object to the cluster with the nearest centroid.



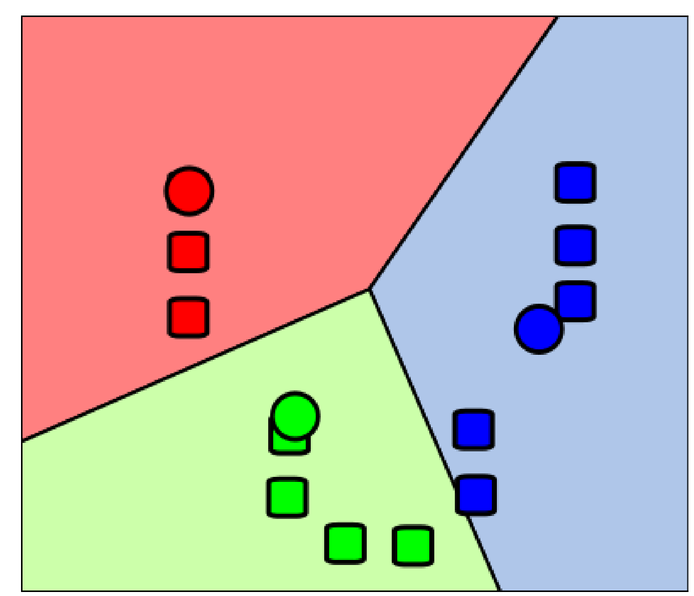
1. Select initial centroids at random



2. Assign each object to the cluster with the nearest centroid.



3. Compute each centroid as the mean of the objects assigned to it (go to 2)



2. Assign each object to the cluster with the nearest centroid.

Repeat previous 2 steps until no change

K-means Clustering

Given k:

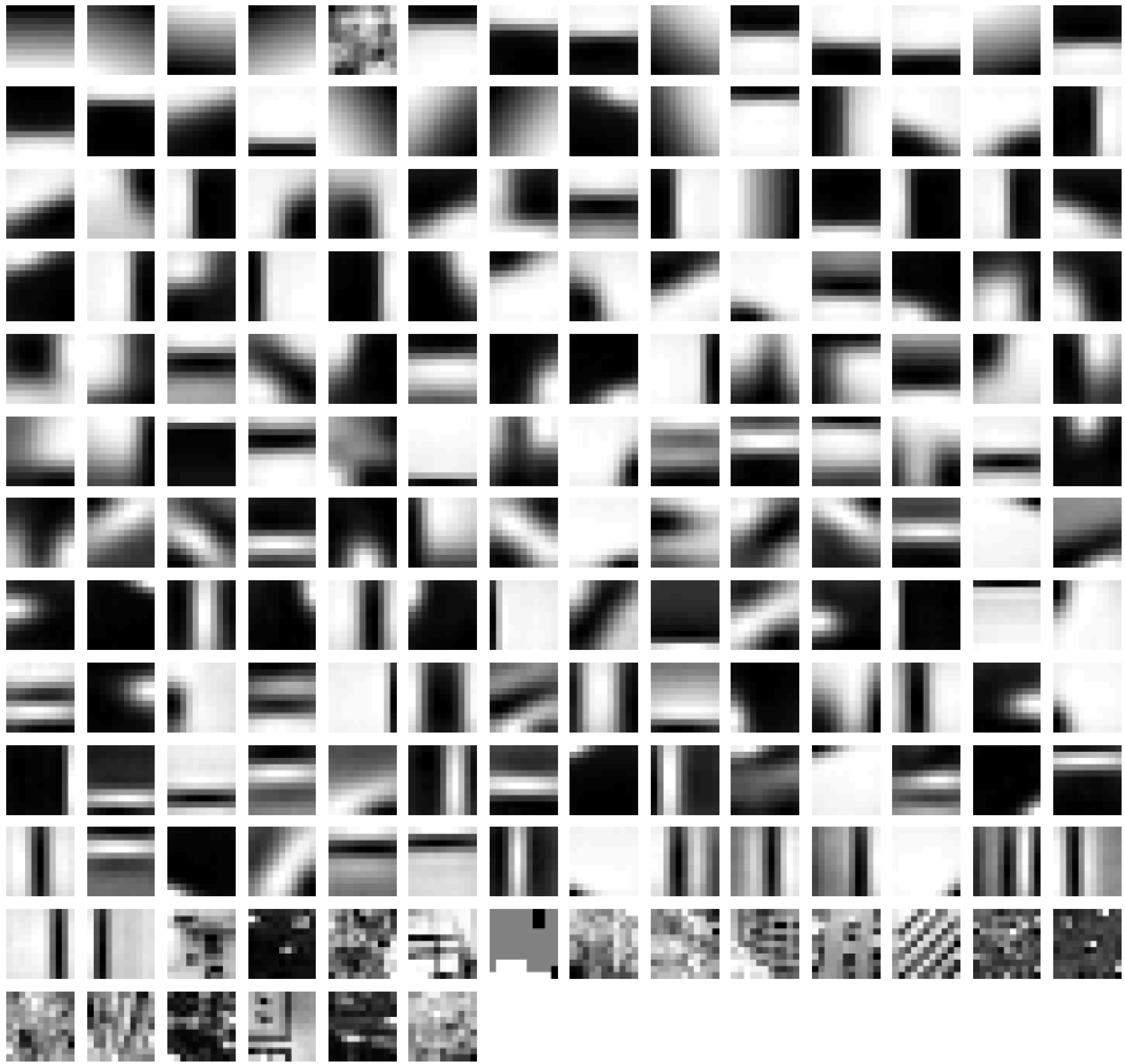
1. Select initial centroids at random.
2. Assign each object to the cluster with the nearest centroid.
3. Compute each centroid as the mean of the objects assigned to it.
4. Repeat previous 2 steps until no change.

*From what **data** should I learn the dictionary?*

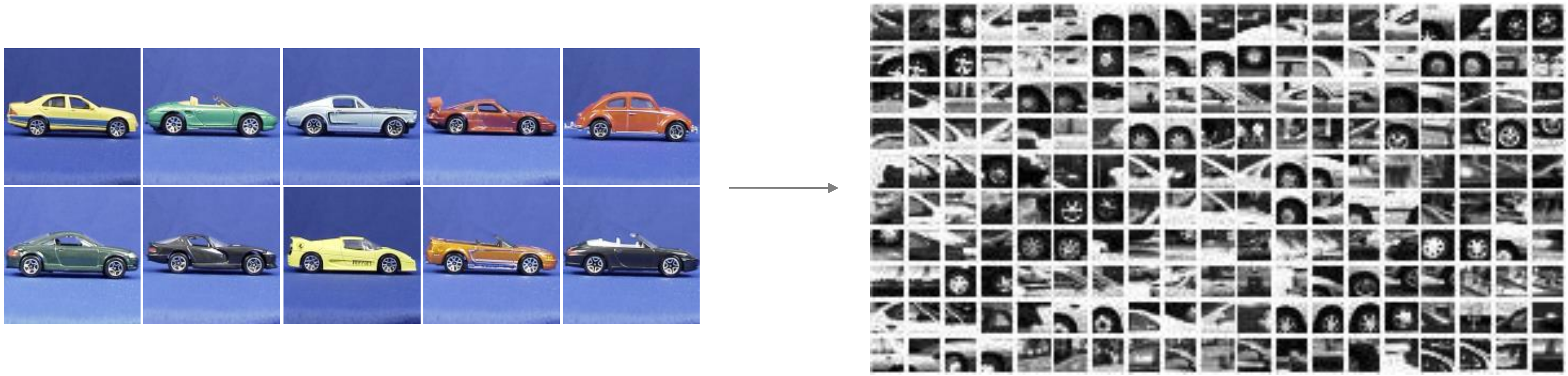
*From what **data** should I learn the dictionary?*

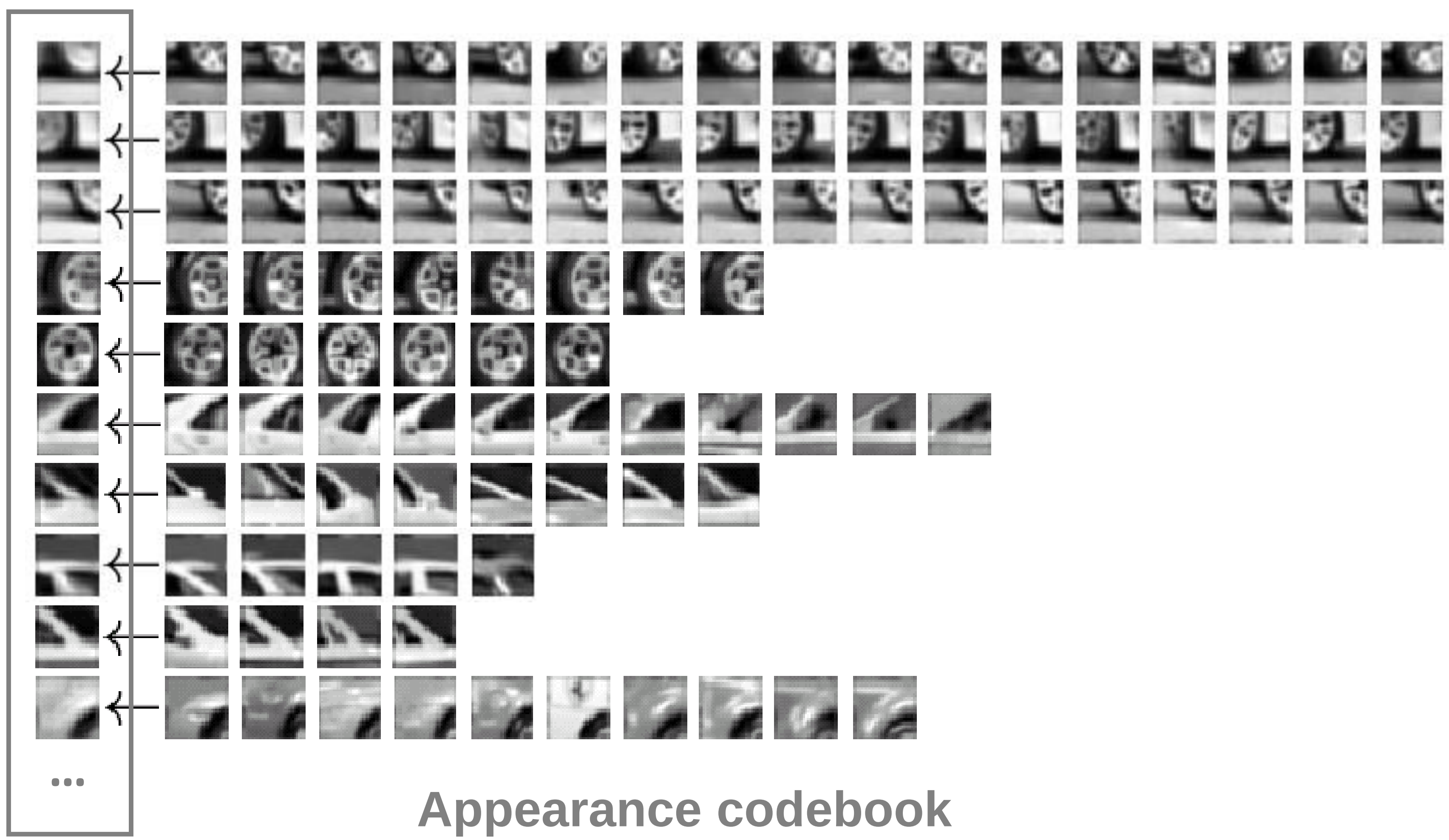
- Dictionary can be learned on separate training set
- Provided the training set is sufficiently representative, the dictionary will be "universal"

Example visual dictionary



Example dictionary

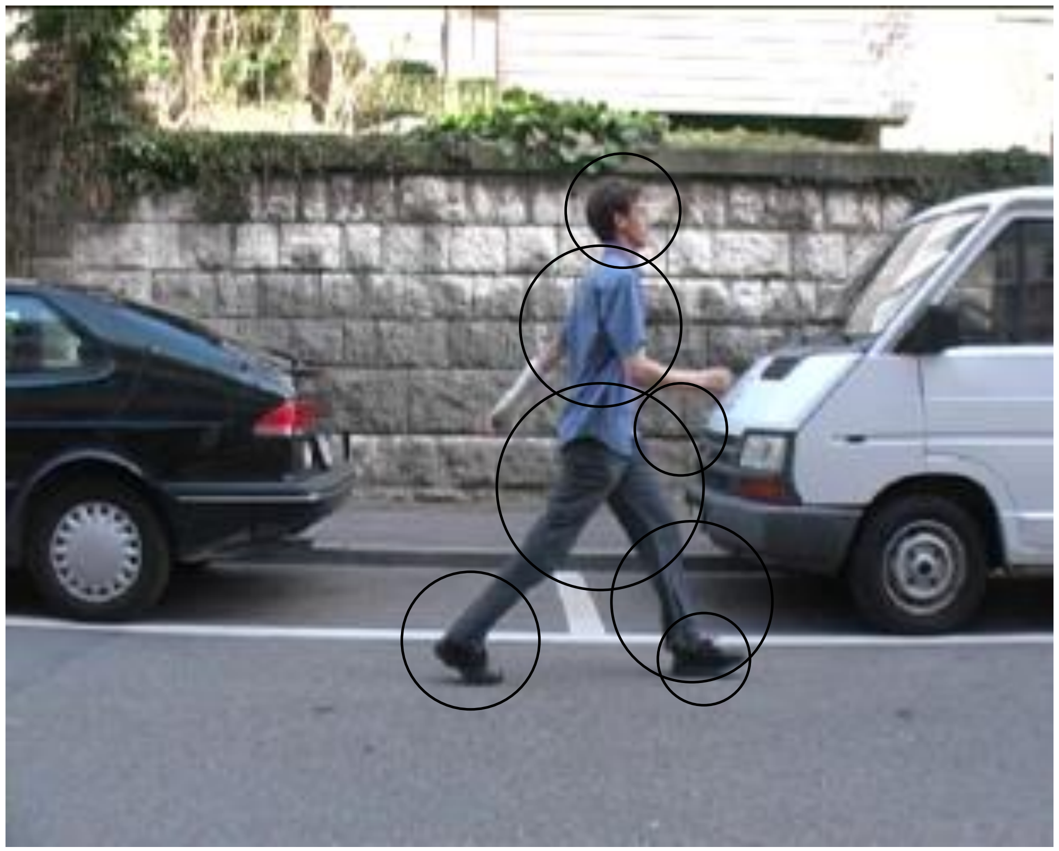


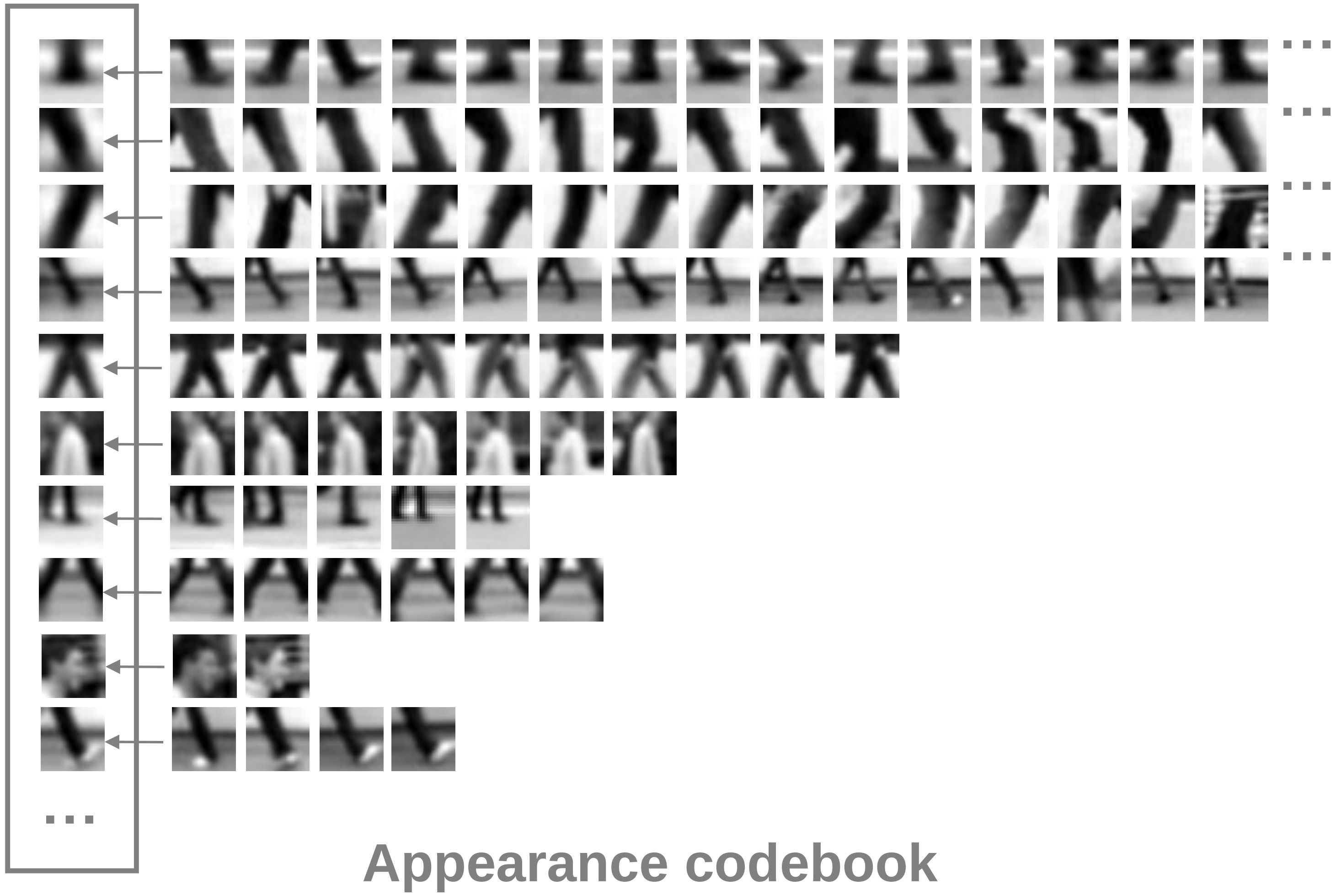


Appearance codebook

Source: B. Leibe

Another dictionary





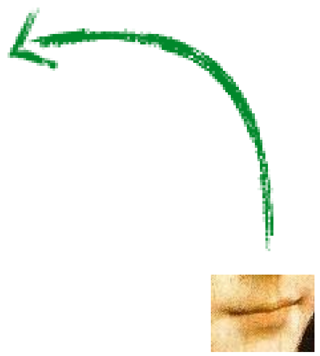
Appearance codebook

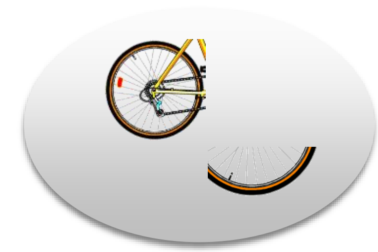
Source: B. Leibe

Dictionary Learning:
Learn Visual Words using clustering

Encode:
build Bags-of-Words (BOW) vectors
for each image

Classify:
Train and test data using BOWs

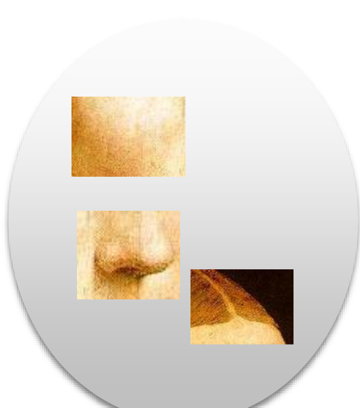




1. Quantization: image features gets associated to a visual word (nearest cluster center)

Encode:

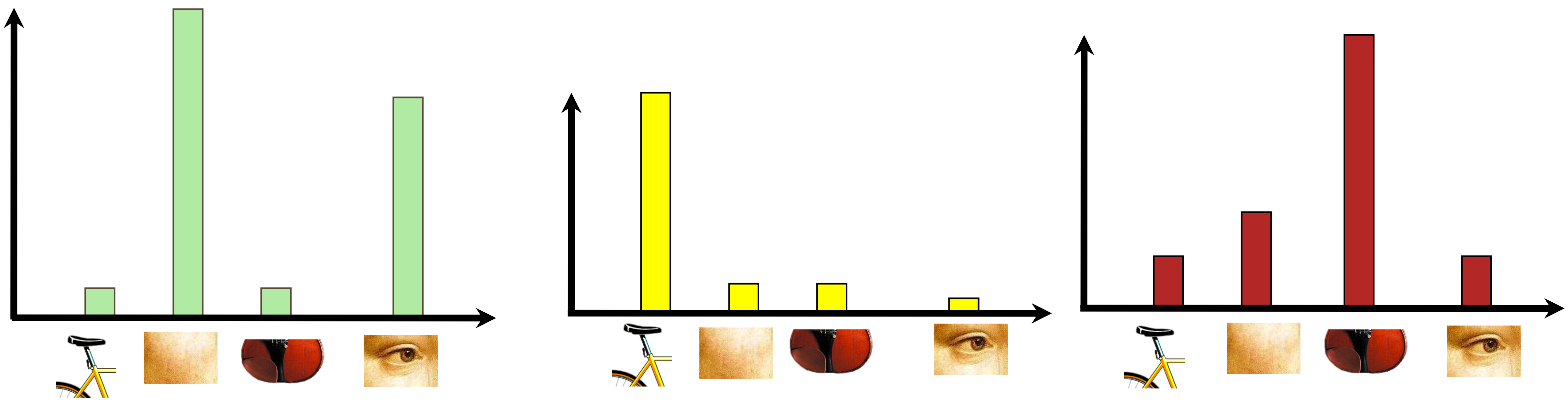
build Bags-of-Words (BOW) vectors for each image

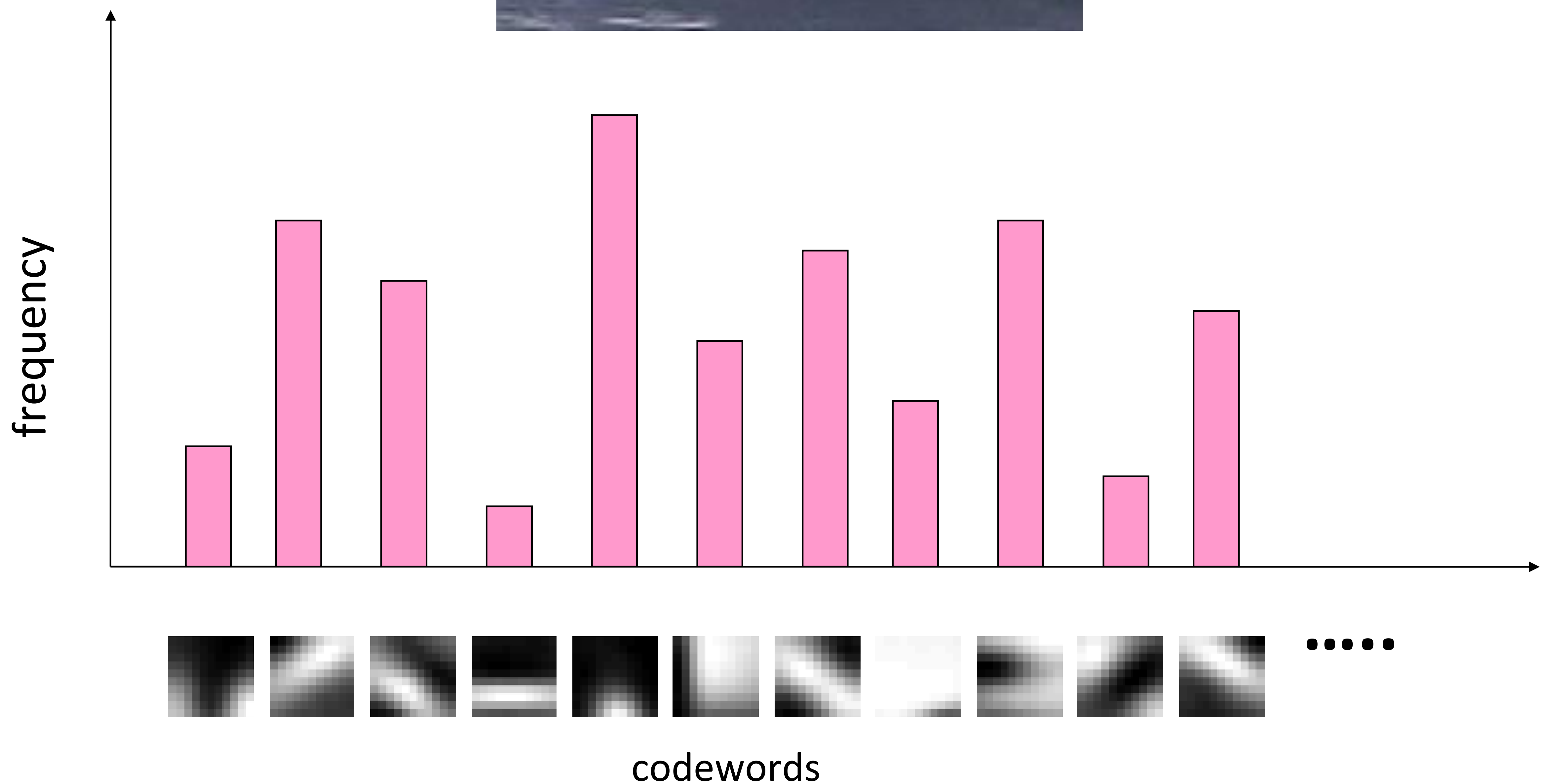


Encode:

build Bags-of-Words (BOW) vectors
for each image

2. Histogram: count the number of visual word occurrences

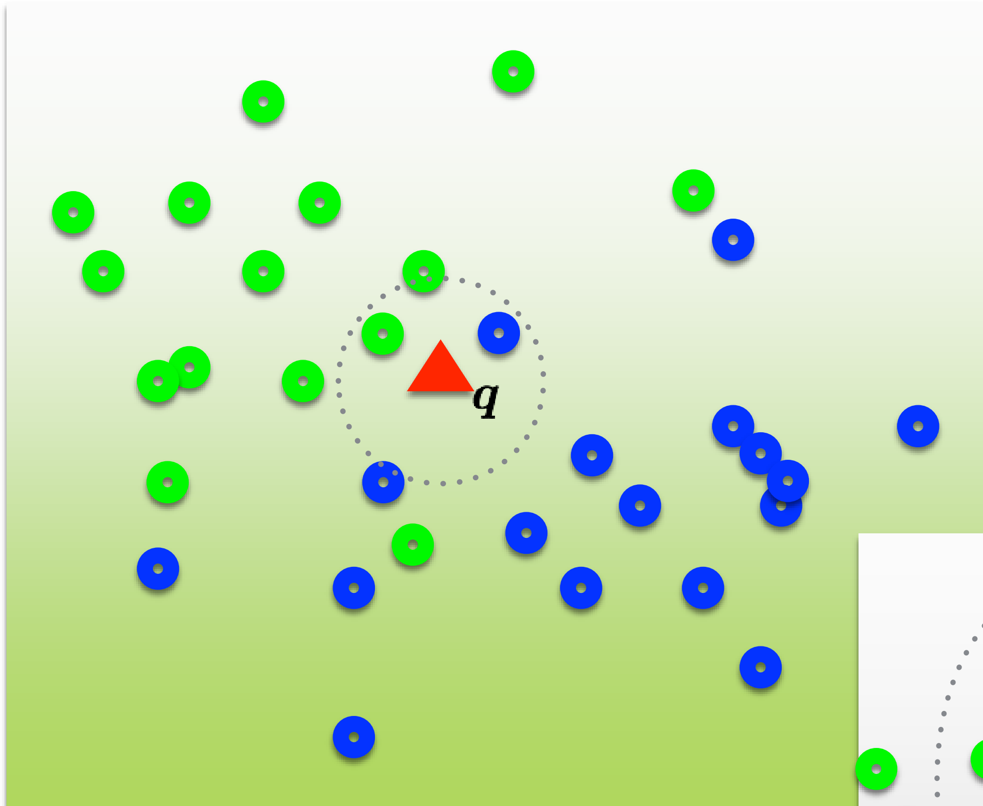



frequency
codewords

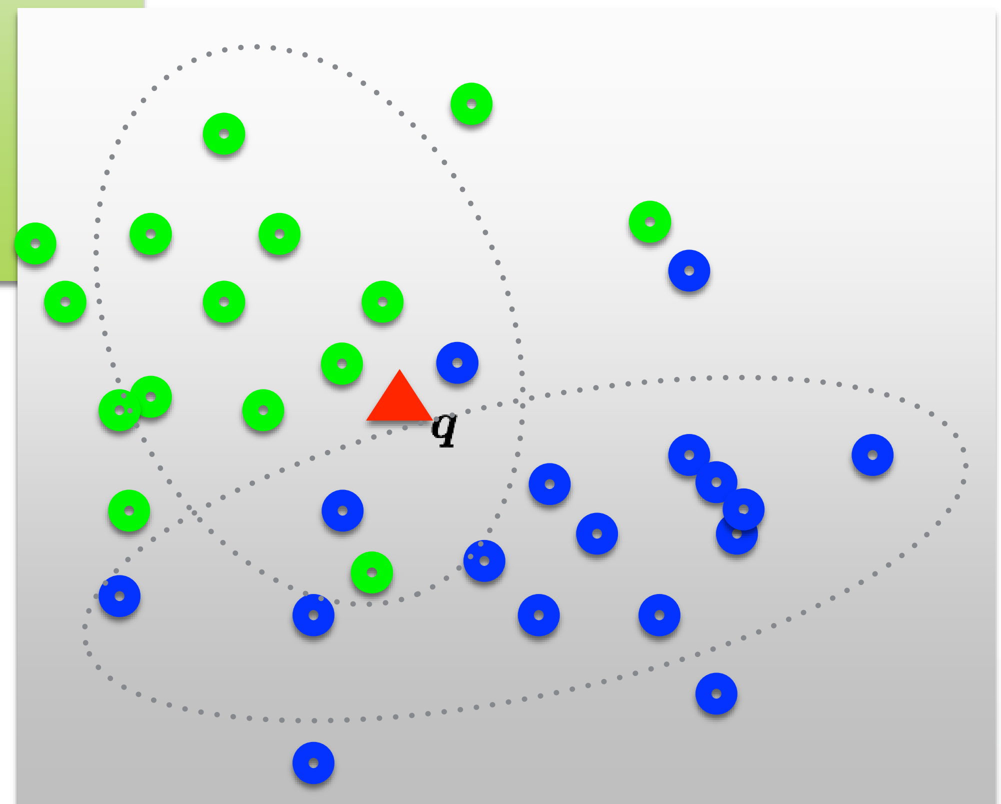
Dictionary Learning:
Learn Visual Words using clustering

Encode:
build Bags-of-Words (BOW) vectors
for each image

Classify:
Train and test data using BOWs

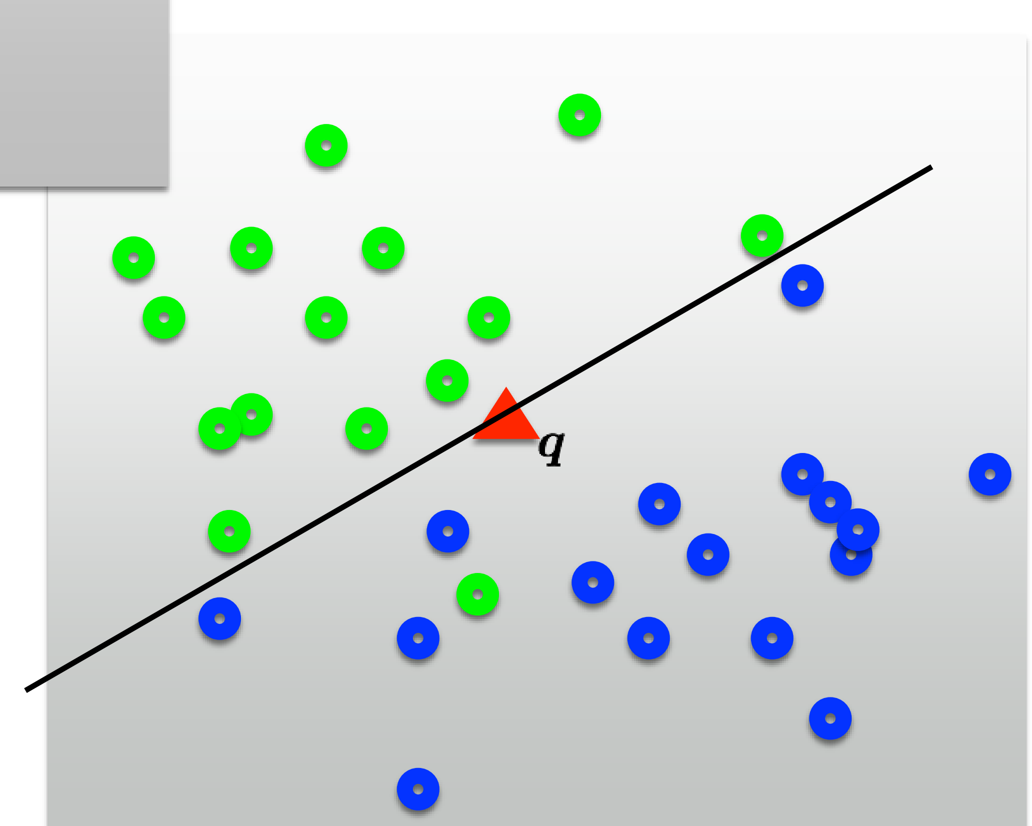


K nearest neighbors



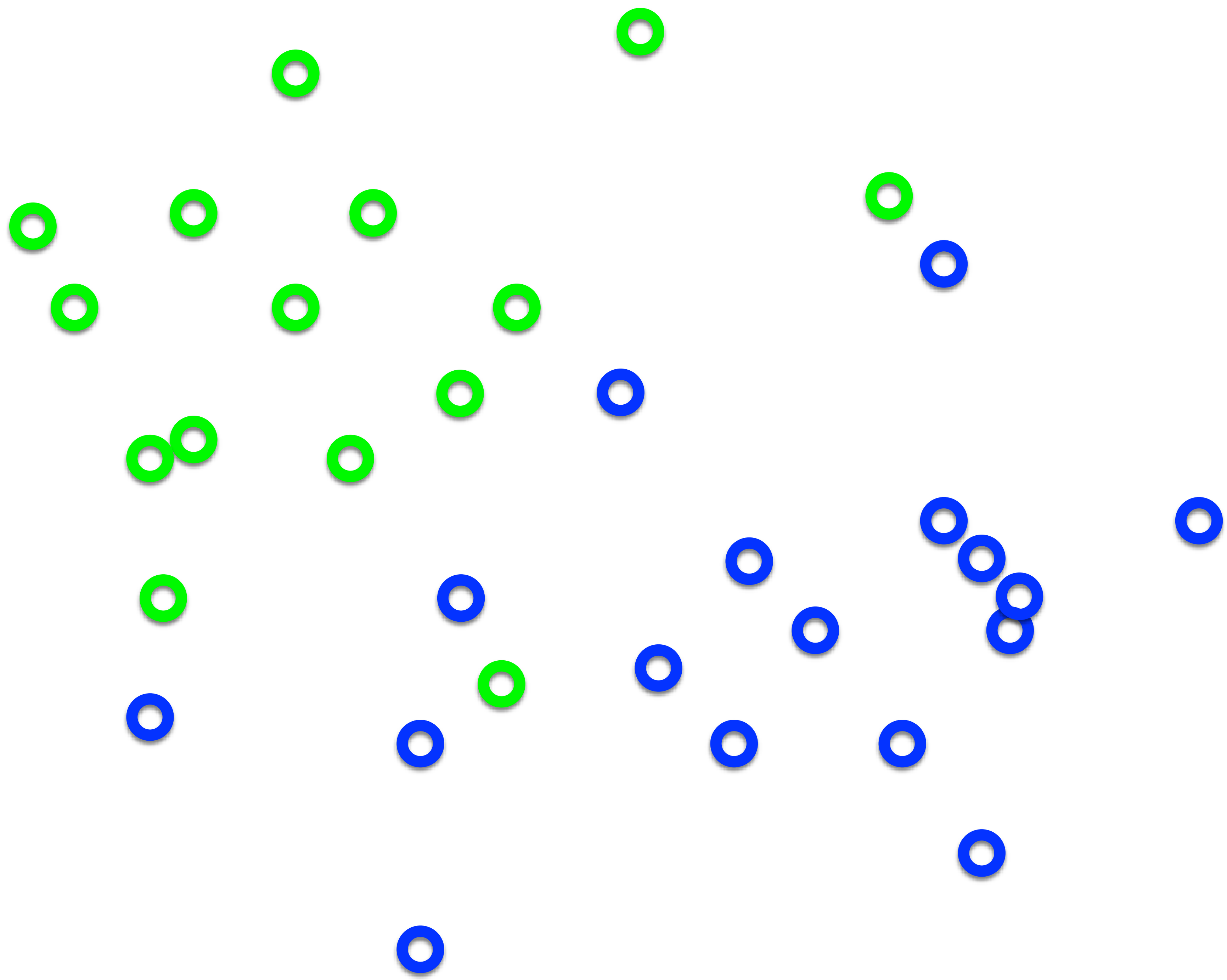
Naïve Bayes

Support Vector Machine

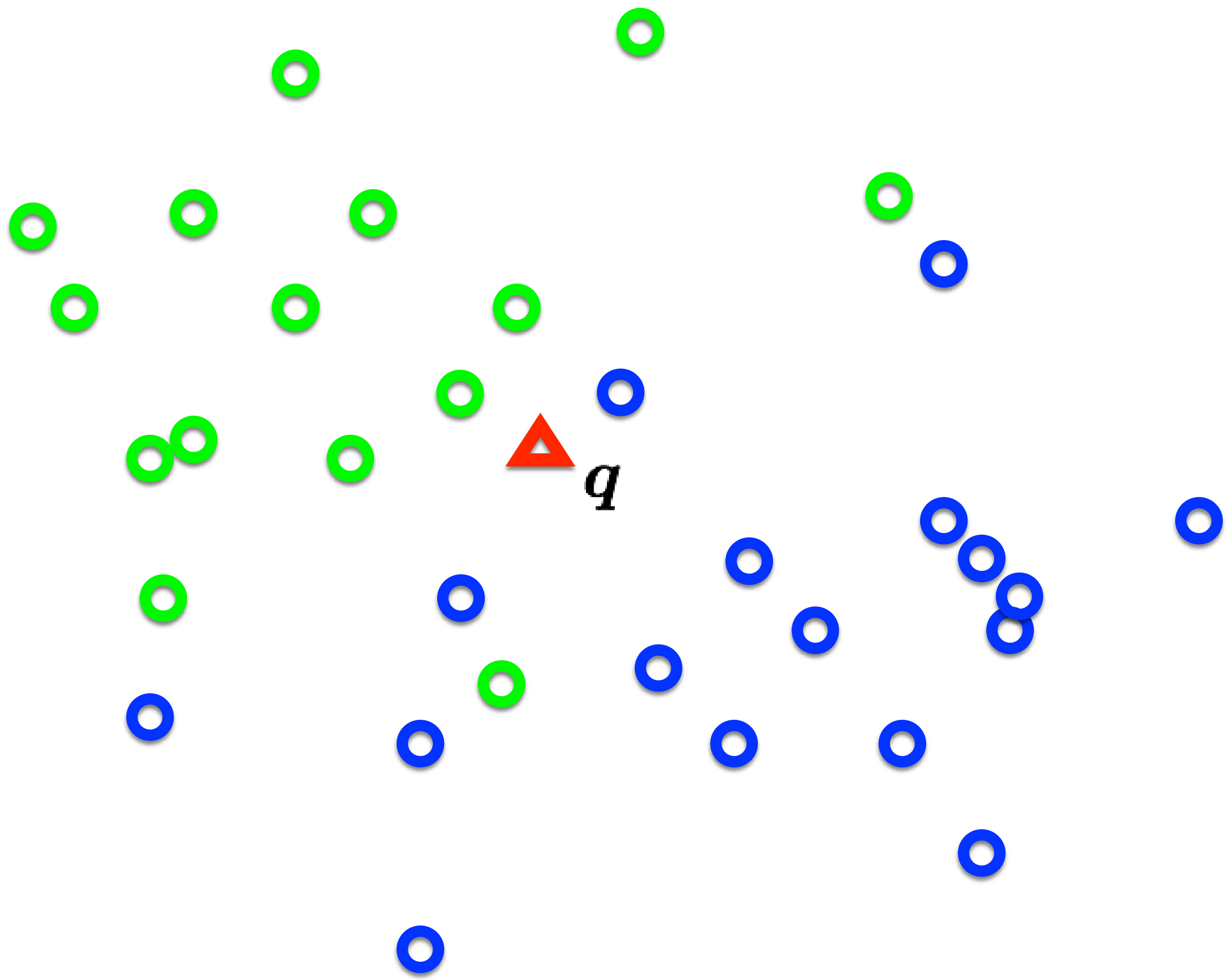


K nearest neighbors

Distribution of data from two classes

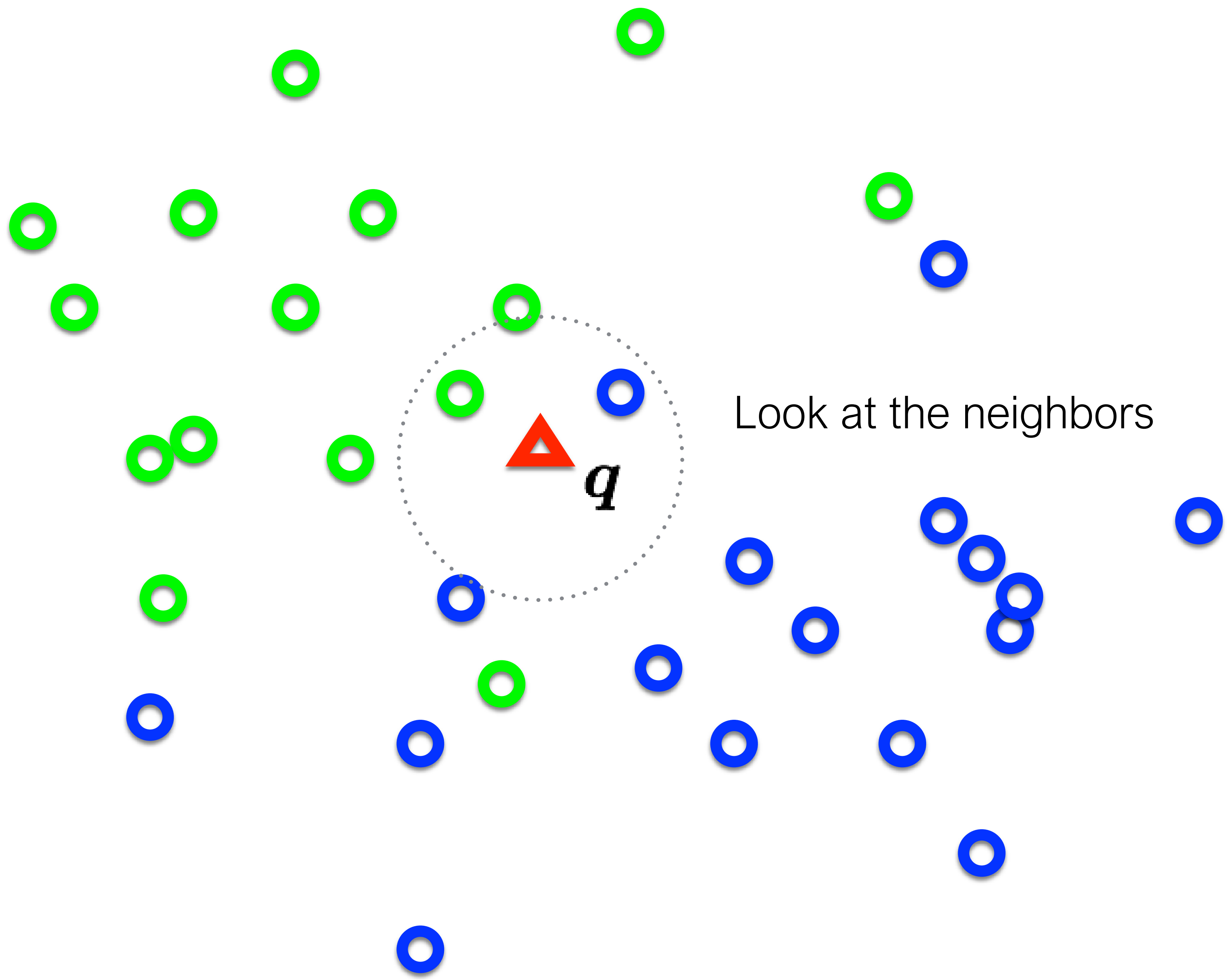


Distribution of data from two classes

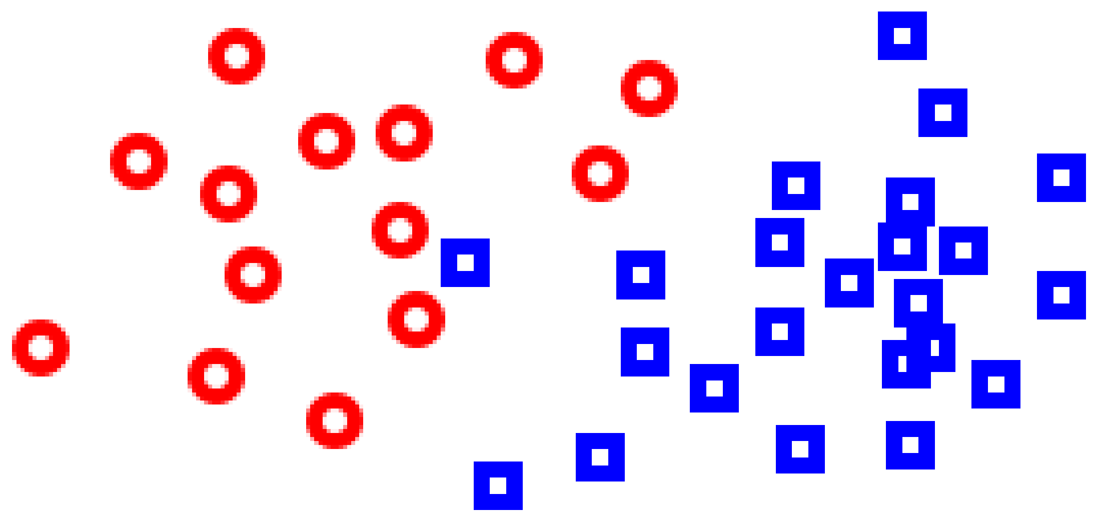


Which class does q belong too?

Distribution of data from two classes



K-Nearest Neighbor (KNN) Classifier

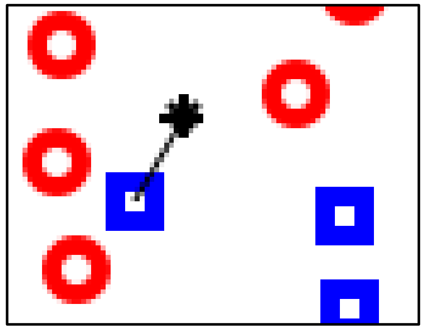


Non-parametric pattern classification approach

Consider a two class problem where each sample consists of two measurements (x,y).

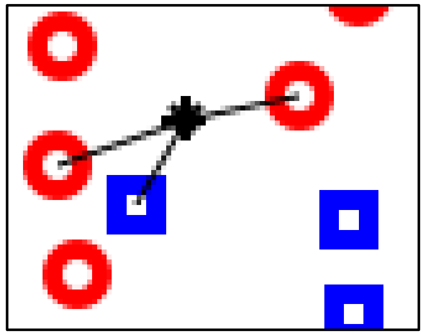
For a given query point q, assign the class of the nearest neighbor

k = 1



Compute the k nearest neighbors and assign the class by majority vote.

k = 3



Nearest Neighbor is competitive

MNIST Digit Recognition

- Handwritten digits
- 28x28 pixel images: d = 784
- 60,000 training samples
- 10,000 test samples

Yann LeCunn

	Test Error Rate (%)
Linear classifier (1-layer NN)	12.0
K-nearest-neighbors, Euclidean	5.0
K-nearest-neighbors, Euclidean, deskewed	2.4
K-NN, Tangent Distance, 16x16	1.1
K-NN, shape context matching	0.67
1000 RBF + linear classifier	3.6

What is the best distance metric between data points?

- Typically Euclidean distance
- Locality sensitive distance metrics
- Important to normalize.
  Dimensions have different scales

How many K?

- Typically k=1 is good
- Cross-validation (try different k!)

Distance metrics

$$D(\boldsymbol{x}, \boldsymbol{y}) = \sqrt{(x_1 - y_1)^2 + \cdots + (x_N - y_N)^2}$$

Euclidean

$$D(\boldsymbol{x}, \boldsymbol{y}) = \frac{\boldsymbol{x} \cdot \boldsymbol{y}}{\|\boldsymbol{x}\| \|\boldsymbol{y}\|} = \frac{x_1 y_1 + \cdots + x_N y_N}{\sqrt{\sum_n x_n^2} \sqrt{\sum_n y_n^2}}$$

Cosine

$$D(\boldsymbol{x}, \boldsymbol{y}) = \frac{1}{2} \sum_n \frac{(x_n - y_n)^2}{(x_n + y_n)}$$

Chi-squared

Choice of distance metric

- Hyperparameter

L1 (Manhattan) distance

$$d_1(I_1, I_2) = \sum_p |I_1^p - I_2^p|$$

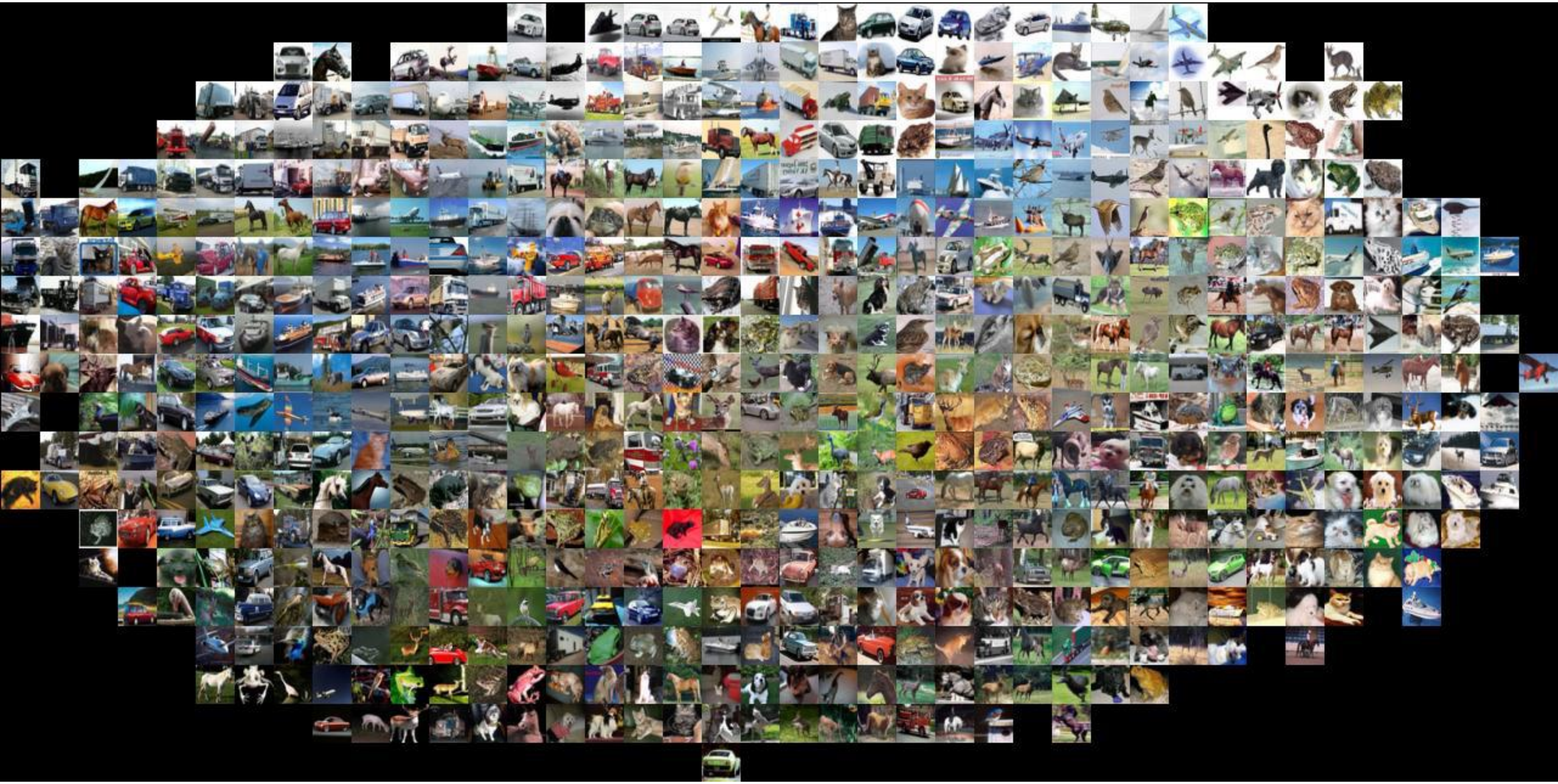
L2 (Euclidean) distance

$$d_2(I_1, I_2) = \sqrt{\sum_p \left(I_1^p - I_2^p\right)^2}$$

- Two most commonly used special cases of p-norm

$$||x||_p = \left(|x_1|^p + \cdots + |x_n|^p\right)^{\frac{1}{p}} \qquad p \geq 1, x \in \mathbb{R}^n$$

Visualization: L2 distance



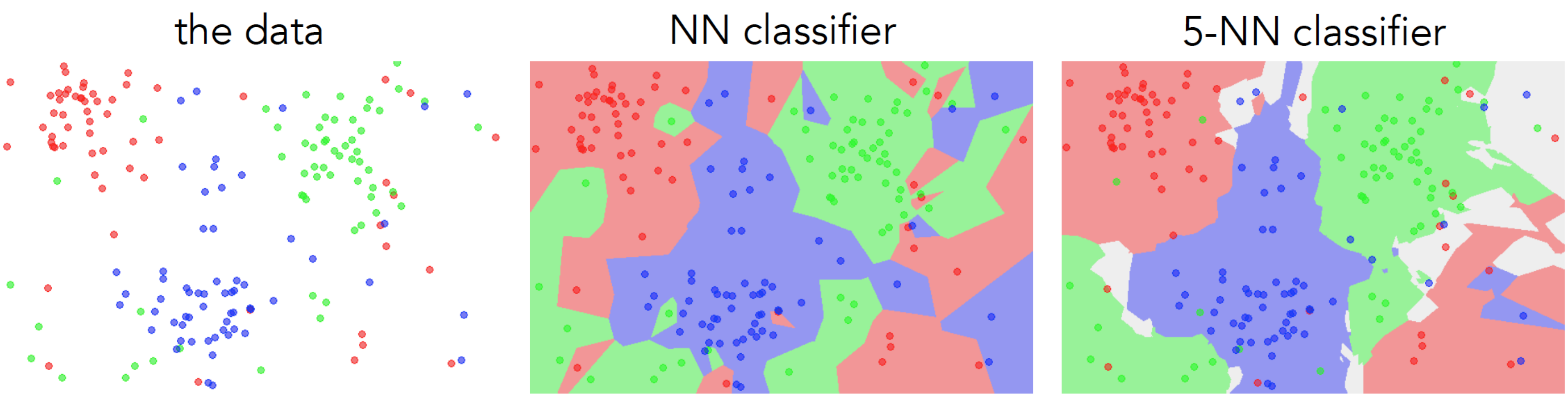
CIFAR-10 and NN results

Example dataset: **CIFAR-10**
10 labels
50,000 training images
10,000 test images.

For every test image (first column), examples of nearest neighbors in rows

k-nearest neighbor

- Find the k closest points from training data
- Labels of the k points “vote” to classify



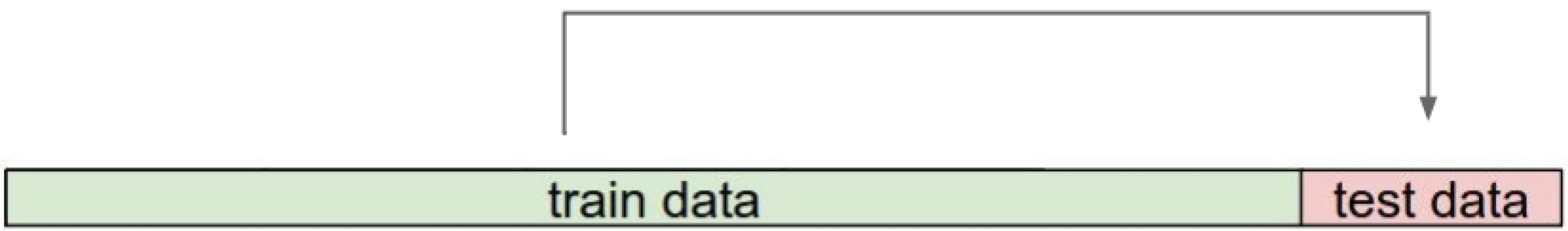
Hyperparameters

- What is the best distance to use?
- What is the best value of k to use?

- i.e., how do we set the hyperparameters?

- Very problem-dependent
- Must try them all and see what works best

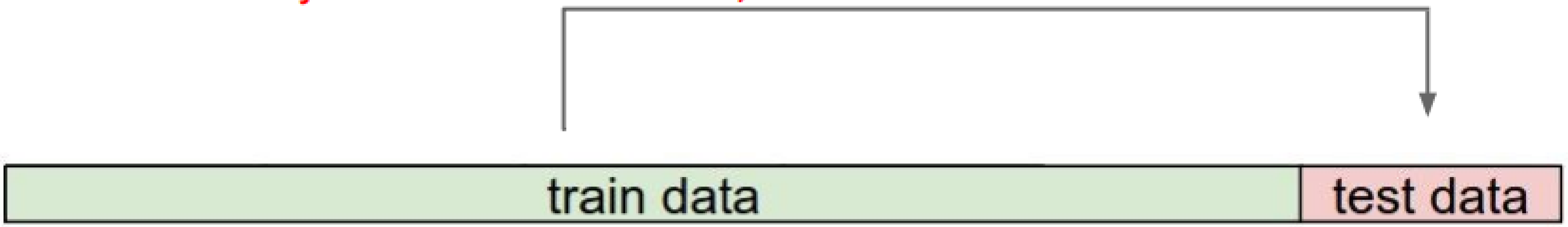
Try out what hyperparameters work best on test set.



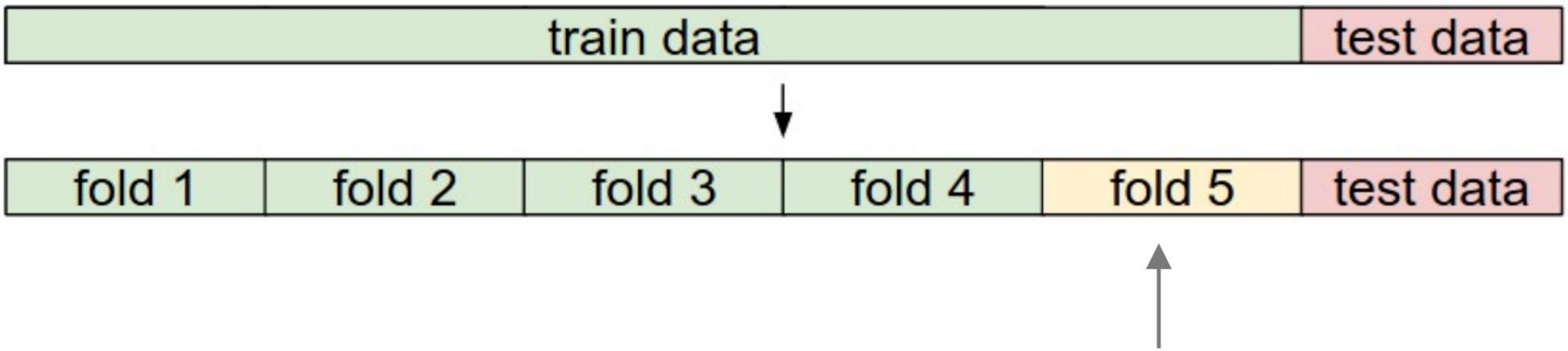
Trying out what hyperparameters work best on test set:

Very bad idea. The test set is a proxy for the generalization performance!

Use only **VERY SPARINGLY,** at the end.



Validation

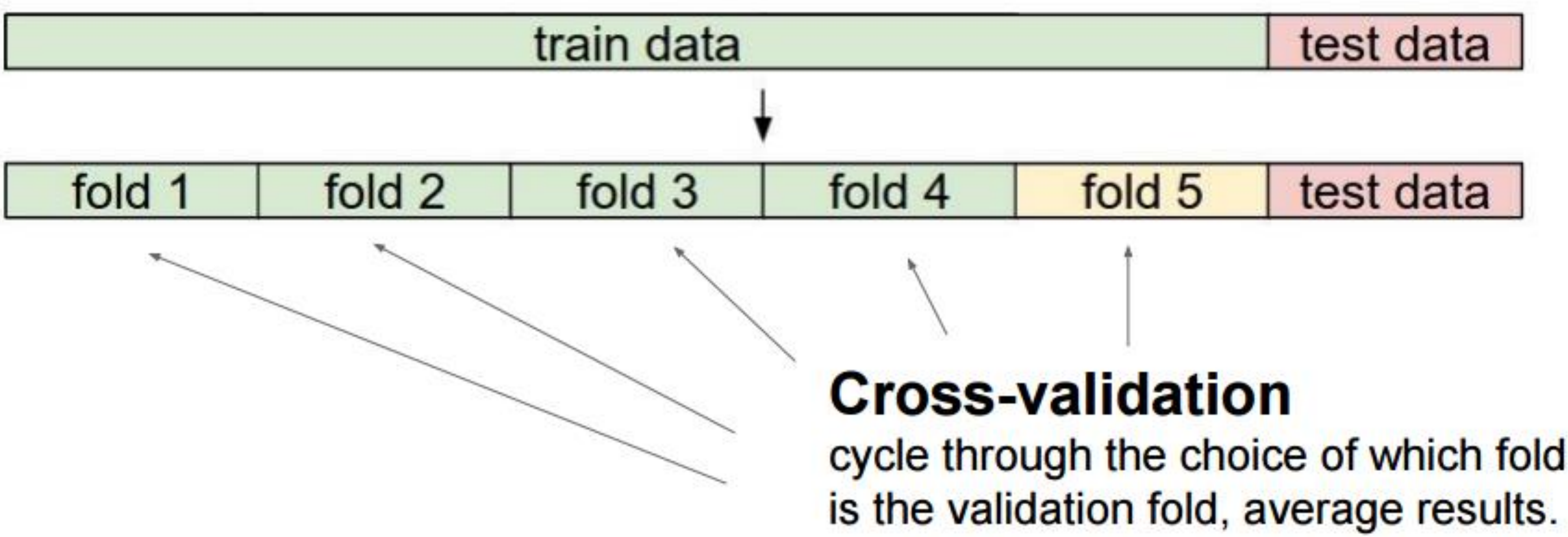


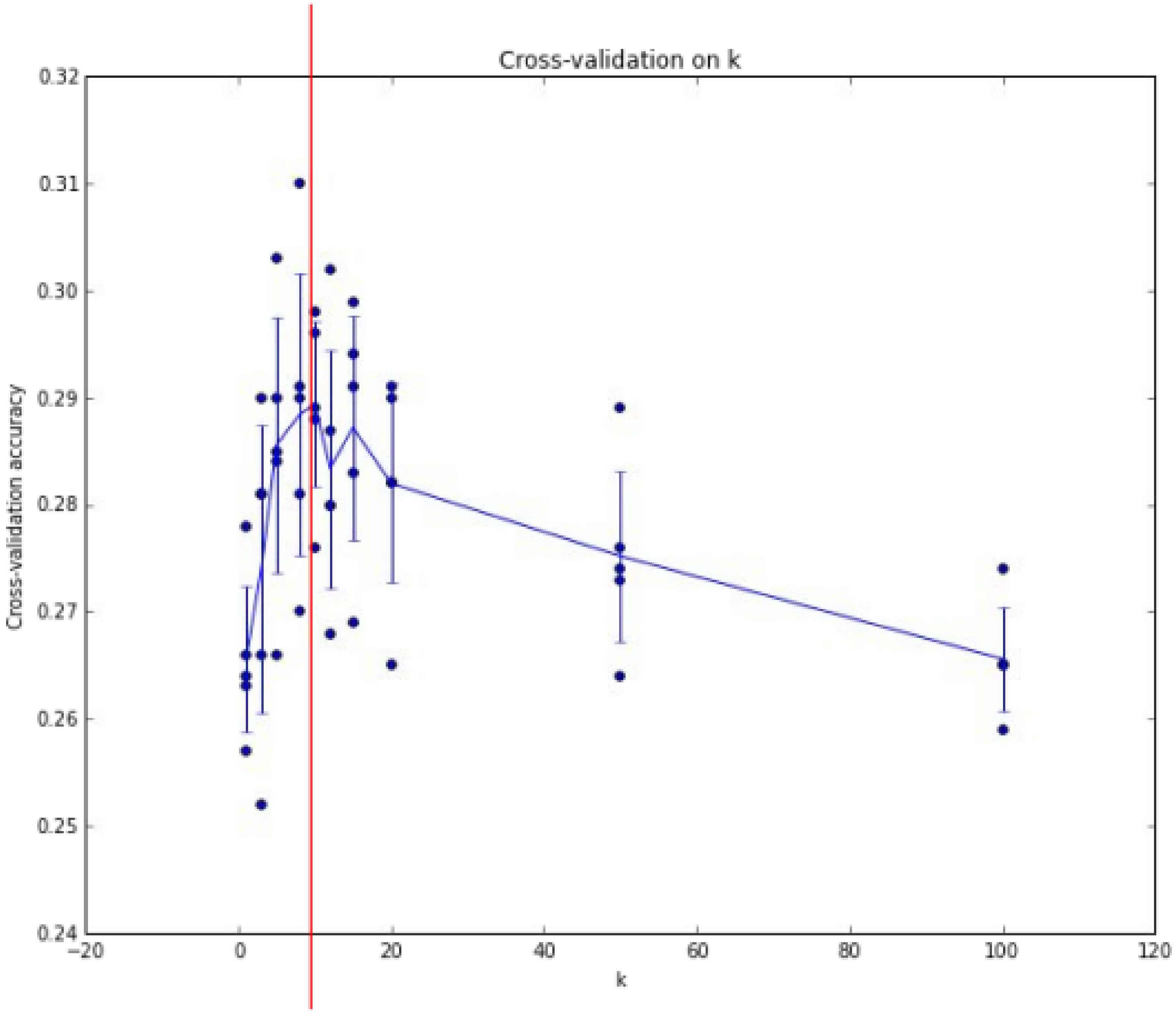
Validation data

use to tune hyperparameters

evaluate on test set ONCE at the end

Cross-validation





Example of
5-fold cross-validation
for the value of **k.**

Each point: single
outcome.

The line goes
through the mean, bars
indicated standard
deviation

(Seems that k ~= 7 works best
for this data)

How to pick hyperparameters?

- Methodology
  - Train and test
  - Train, validate, test

- Train for original model
- Validate to find hyperparameters
- Test to understand generalizability

Pros

– simple yet effective

Cons

– search is expensive (can be sped-up)

– storage requirements

– difficulties with high-dimensional data

kNN -- Complexity and Storage

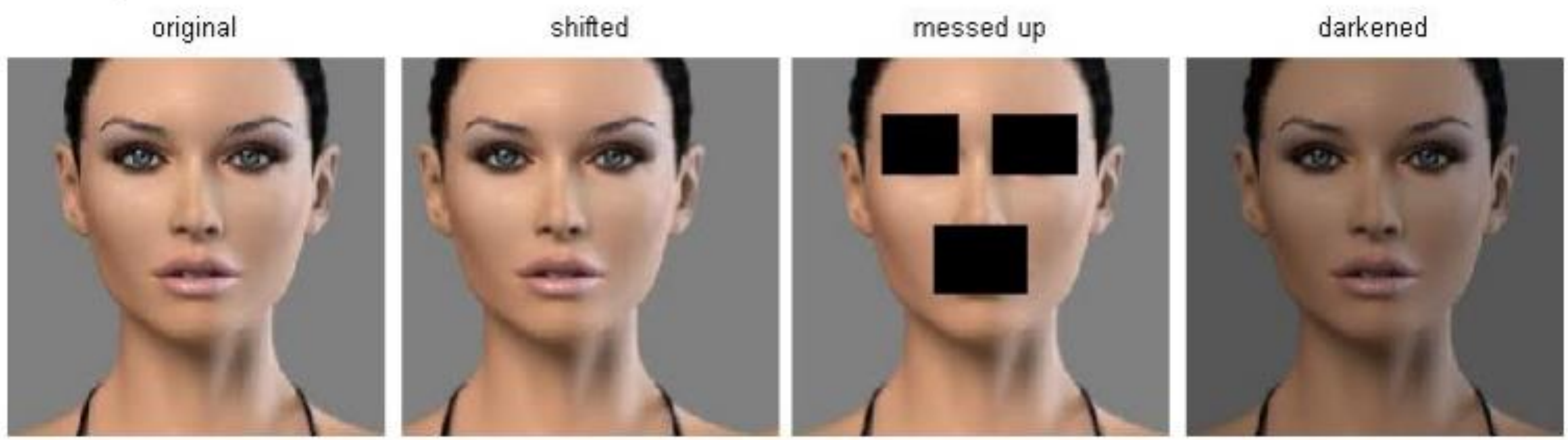
- N training images, M test images

- Training: O(1)
- Testing: O(MN)

- Hmm…
  – Normally need the opposite
  – Slow training (ok), fast testing (necessary)

k-Nearest Neighbor on images **never used.**

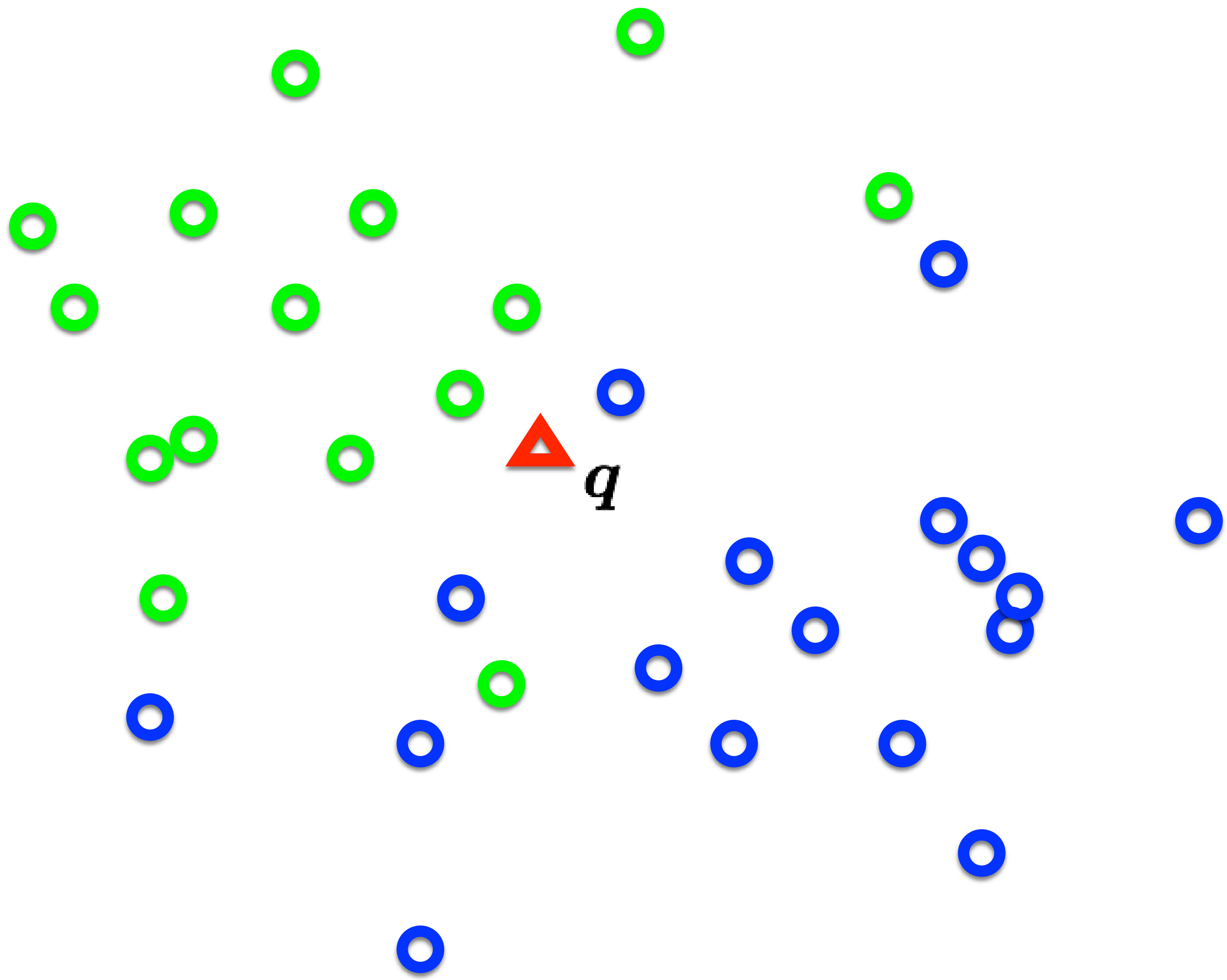
- terrible performance at test time
- distance metrics on level of whole images can be very unintuitive



(all 3 images have same L2 distance to the one on the left)

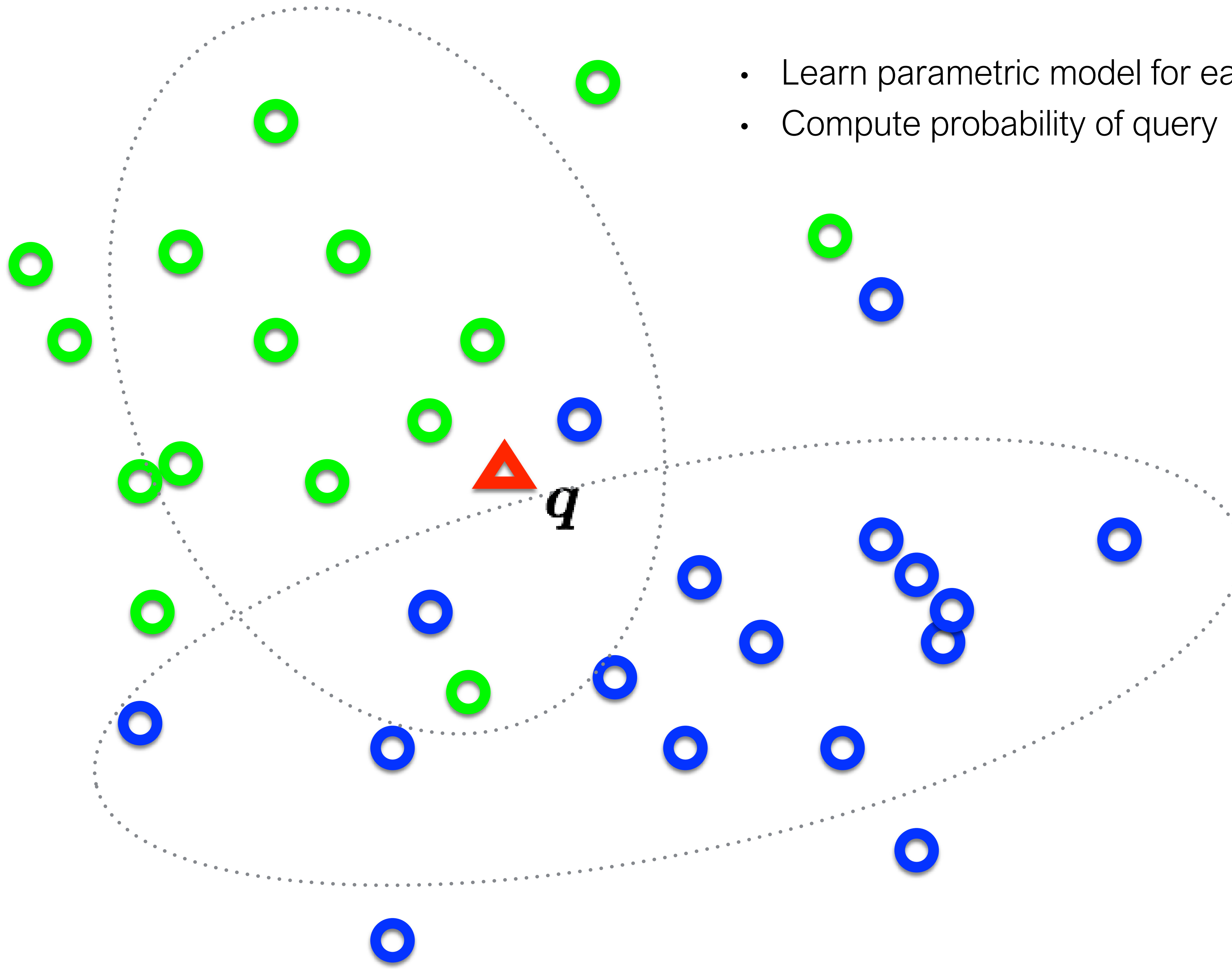
Naïve Bayes

Distribution of data from two classes



Which class does q belong too?

Distribution of data from two classes



- Learn parametric model for each class
- Compute probability of query

This is called the posterior.

the probability of a class z given the observed features X

$$p(z|\boldsymbol{X})$$

For classification, z is a discrete random variable (e.g., car, person, building)

X is a set of observed features (e.g., features from a single image)

(it’s a function that returns a single probability value)

This is called the posterior:

the probability of a class z given the observed features X

$$p(\boldsymbol{z}|\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N)$$

For classification, z is a discrete random variable
(e.g., car, person, building)

Each x is an observed feature
(e.g., visual words)

(it's a function that returns a single probability value)

Recall:

The posterior can be decomposed according to

Bayes' Rule

likelihood prior

$$p(A|B) = \frac{p(B|A)p(A)}{p(B)}$$

posterior

In our context…

$$p(\boldsymbol{z}|\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N) = \frac{p(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N|\boldsymbol{z})p(\boldsymbol{z})}{p(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N)}$$

The naive Bayes' classifier is solving this optimization

$$\hat{z} = \arg\max_{z \in \boldsymbol{z}} p(z|\boldsymbol{X})$$

MAP (maximum a posteriori) estimate

$$\hat{z} = \arg\max_{z \in \boldsymbol{z}} \frac{p(\boldsymbol{X}|z)p(z)}{p(\boldsymbol{X})}$$

Bayes' Rule

$$\hat{z} = \arg\max_{z \in \boldsymbol{z}} p(\boldsymbol{X}|z)p(z)$$

Remove constants

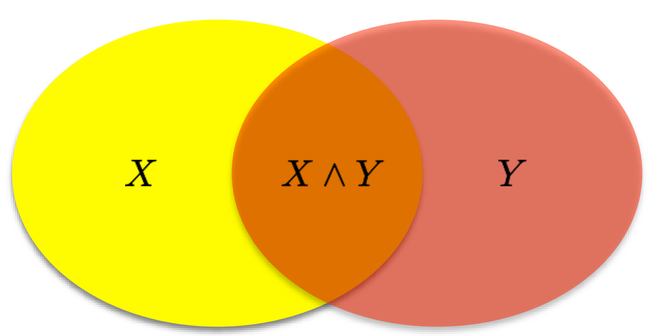
To optimize this…we need to compute this

Compute the likelihood…

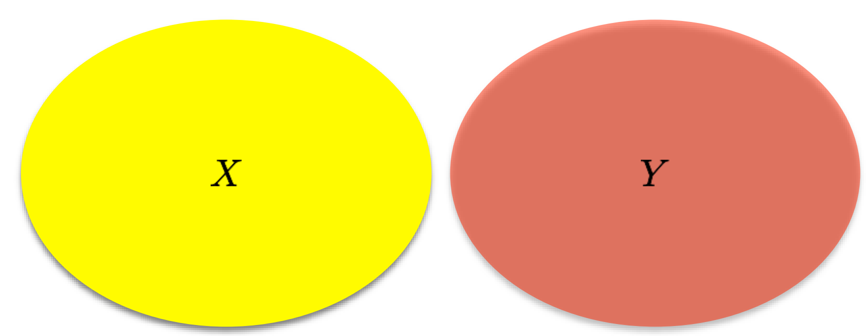
A naive Bayes' classifier assumes all features are
conditionally independent

$$\begin{aligned} p(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N | \boldsymbol{z}) &= p(\boldsymbol{x}_1 | \boldsymbol{z}) p(\boldsymbol{x}_2, \ldots, \boldsymbol{x}_N | \boldsymbol{z}) \\ &= p(\boldsymbol{x}_1 | \boldsymbol{z}) p(\boldsymbol{x}_2 | \boldsymbol{z}) p(\boldsymbol{x}_3, \ldots, \boldsymbol{x}_N | \boldsymbol{z}) \\ &= p(\boldsymbol{x}_1 | \boldsymbol{z}) p(\boldsymbol{x}_2 | \boldsymbol{z}) \cdots p(\boldsymbol{x}_N | \boldsymbol{z}) \end{aligned}$$

Recall:



$$p(x, y) = p(x|y)p(y)$$



$$p(x, y) = p(x)p(y)$$

To compute the MAP estimate

Given (1) a set of known parameters

$$p(\boldsymbol{z}) \quad p(\boldsymbol{x}|\boldsymbol{z})$$

(2) observations

$$\{x_1, x_2, \ldots, x_N\}$$

Compute which z has the largest probability

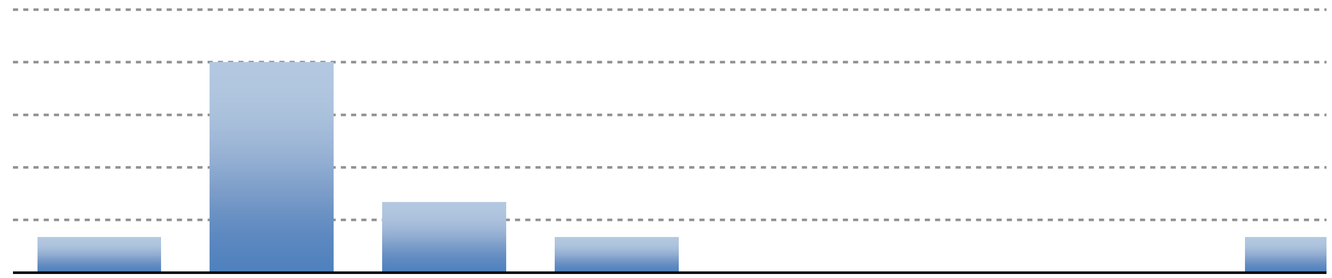
$$\hat{z} = \arg\max_{z \in \boldsymbol{z}} p(z) \prod_n p(x_n|z)$$

The Newspa

Sunday, December 22, 2013

DARPA Selects Carnegie Me

The Tartan Rescue Team from Carnegie Mellon University's National Robotics Engineering Center ranked third among teams competing in the Defense Advanced Research Projects Agency (DARPA) Robotics Challenge Trials this weekend in Homestead, Fla., and was selected by the agency as one of eight teams eligible for DARPA funding to prepare for next December's finals. The team's four-limbed CMU Highly Intelligent Mobile Platform, or CHIMP, robot scored 18 out of a possible 32 points during the two-day trials. It demonstrated its ability to perform such tasks as removing debris, cutting a hole through a wall and closing a series of valves.



count	1	6	2	1	0	0	0	1
word	Tartan	robot	CHIMP	CMU	bio	soft	ankle	sensor
p(x|z)	0.09	0.55	0.18	0.09	0.0	0.0	0.0	0.09

$$p(X|z) = \prod_v p(x_v|z)^{c(w_v)}$$

$$= (0.09)^1 (0.55)^6 \cdots (0.09)^1$$

Numbers get really small so use log probabilities

$$\log p(X|z = \text{‘grandchallenge’}) = -2.42 - 3.68 - 3.43 - 2.42 - 0.07 - 0.07 - 0.07 - 2.42 = -14.58$$

$$\log p(X|z = \text{‘softrobot’}) = -7.63 - 9.37 - 15.18 - 2.97 - 0.02 - 0.01 - 0.02 - 2.27 = -37.48$$

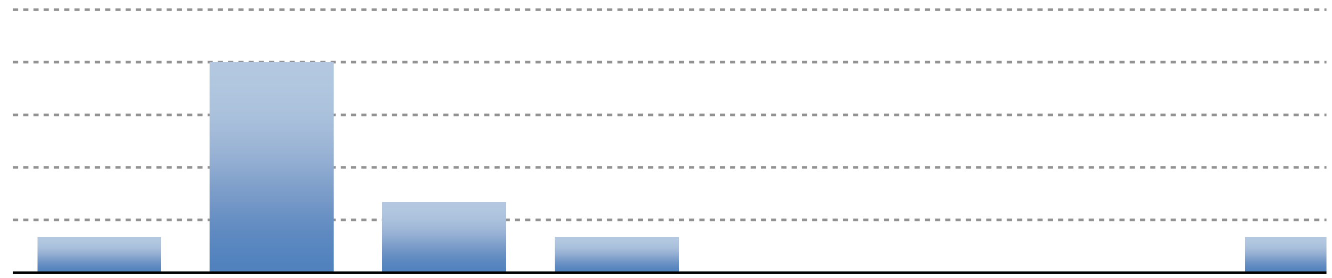
* typically add pseudo-counts (0.001)
** this is an example for computing the likelihood, need to multiply times **prior** to get posterior

The Newspa

Sunday, December 22, 2013

DARPA Selects Carnegie Me

The Tartan Rescue Team from Carnegie Mellon University's National Robotics Engineering Center ranked third among teams competing in the Defense Advanced Research Projects Agency (DARPA) Robotics Challenge Trials this weekend in Homestead, Fla., and was selected by the agency as one of eight teams eligible for DARPA funding to prepare for next December's finals. The team's four-limbed CMU Highly Intelligent Mobile Platform, or CHIMP, robot scored 18 out of a possible 32 points during the two-day trials. It demonstrated its ability to perform such tasks as removing debris, cutting a hole through a wall and closing a series of valves.



count	1	6	2	1	0	0	0	1
word	Tartan	robot	CHIMP	CMU	bio	soft	ankle	sensor
p(x|z)	0.09	0.55	0.18	0.09	0.0	0.0	0.0	0.09

log p(X|z=grand challenge) = **- 14.58**

log p(X|z=bio inspired) = - 37.48

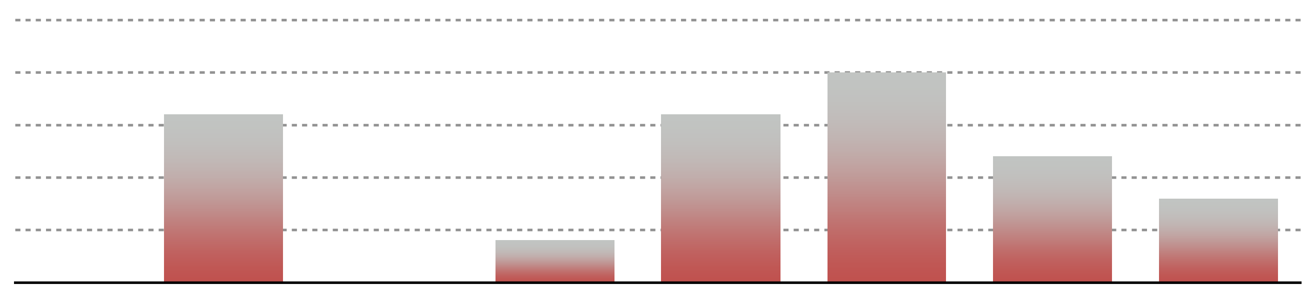
Tartan Tim

Monday, January 20, 2014

Bio-Inspired Robotic Device

PITTSBURGH—A soft, wearable device that mimics the muscles, tendons and ligaments of the lower leg could aid in the rehabilitation of patients with ankle-foot disorders such as drop foot, said Yong-Lae Park, an assistant professor of robotics at Carnegie Mellon University. Park, working with collaborators at Harvard University, the University of Southern California, MIT and BioSensics, developed an active orthotic device using soft plastics and composite materials, instead of a rigid exoskeleton. The soft materials, combined with pneumatic artificial muscles (PAMs), lightweight sensors and advanced control software, made it possible for the robotic device to achieve natural motions in the ankle.

http://www.fodey.com/generators/newspaper/snippet.asp



count	0	4	0	1	4	5	3	2
word	Tartan	robot	CHIMP	CMU	bio	soft	ankle	sensor
p(x|z)	0.0	0.21	0.0	0.05	0.21	0.26	0.16	0.11

log p(X|z=grand challenge) = - 94.06

log p(X|z=bio inspired) = **- 32.41**

* typically add pseudo-counts (0.001)

** this is an example for computing the likelihood, need to multiply times prior to get posterior

Support Vector Machine

Image Classification

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

→ cat

Score function

class scores

Linear Classifier

define a **score function**

data (histogram)

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

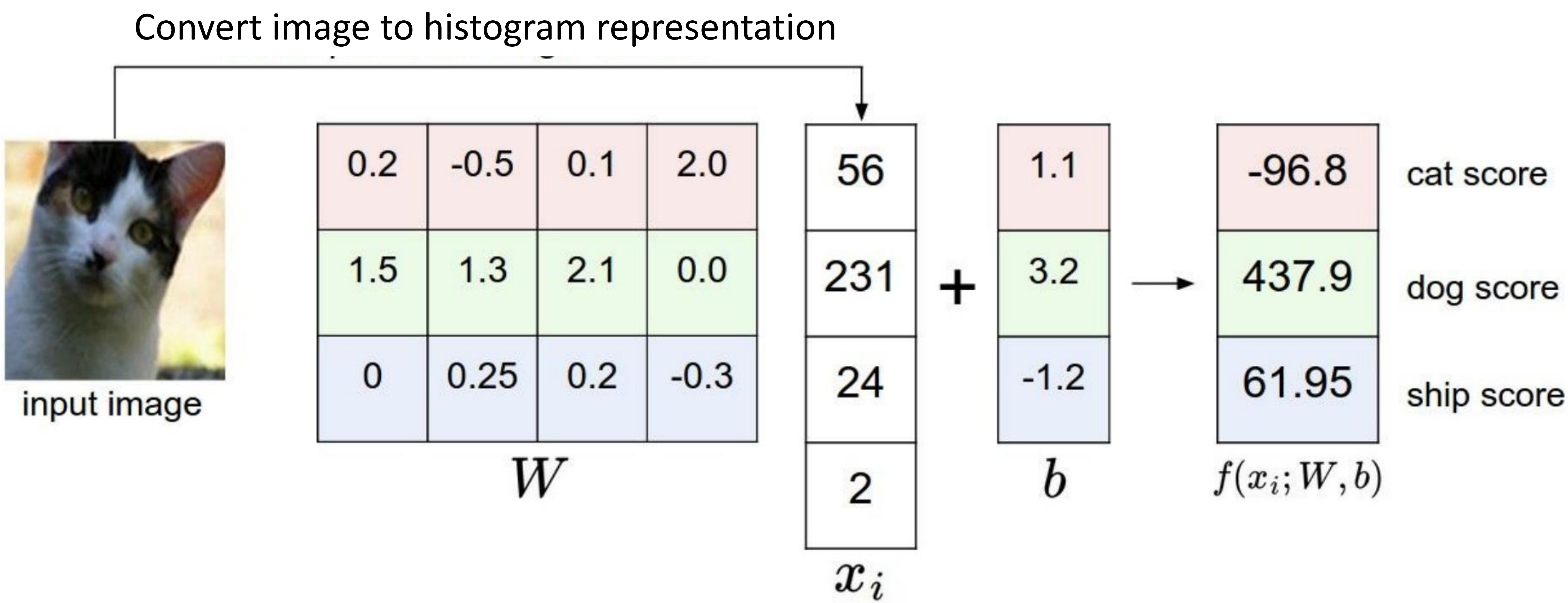
class scores

“weights”

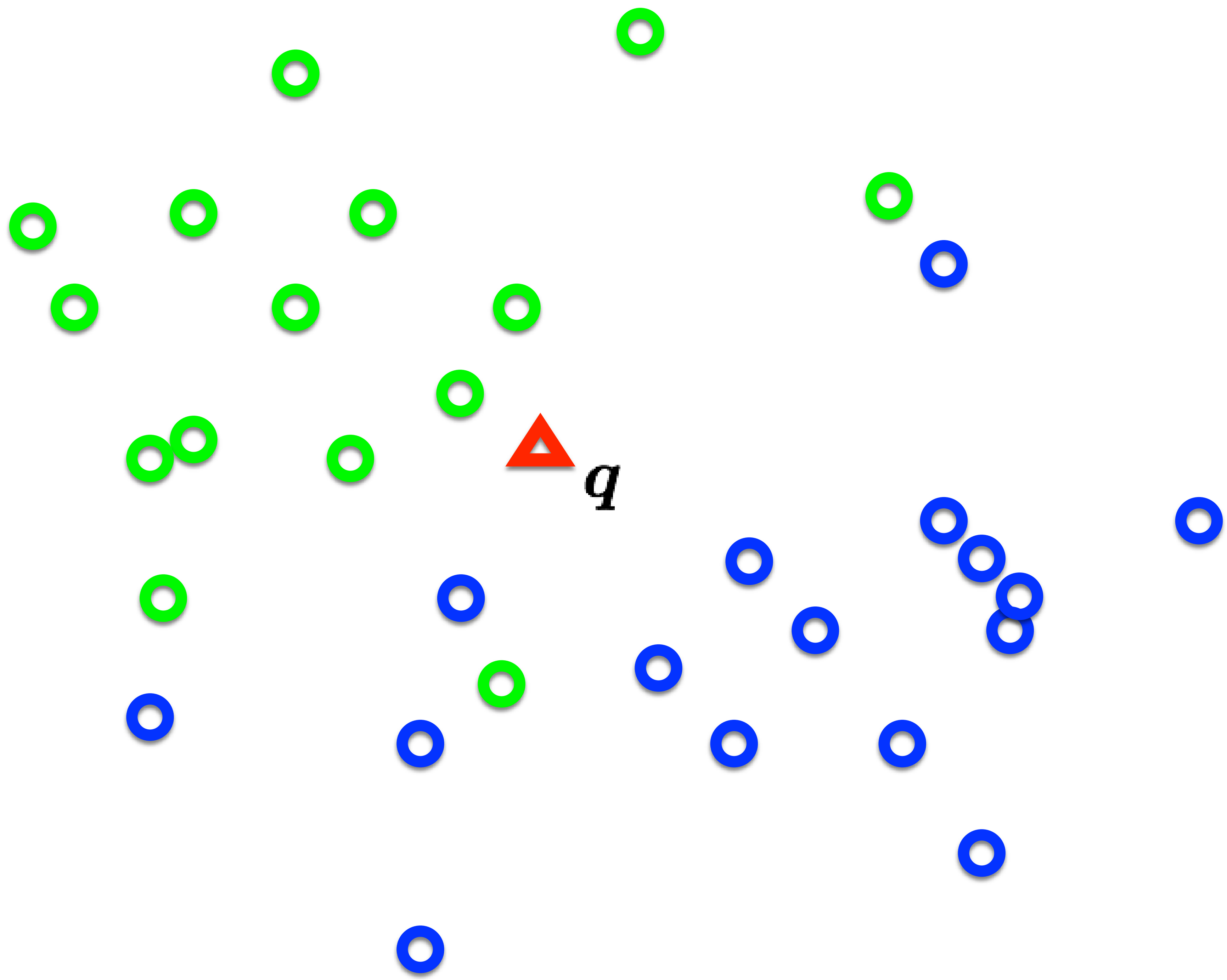
“bias vector”

“parameters”

Example with an image with 4 pixels, and 3 classes (cat/dog/ship)

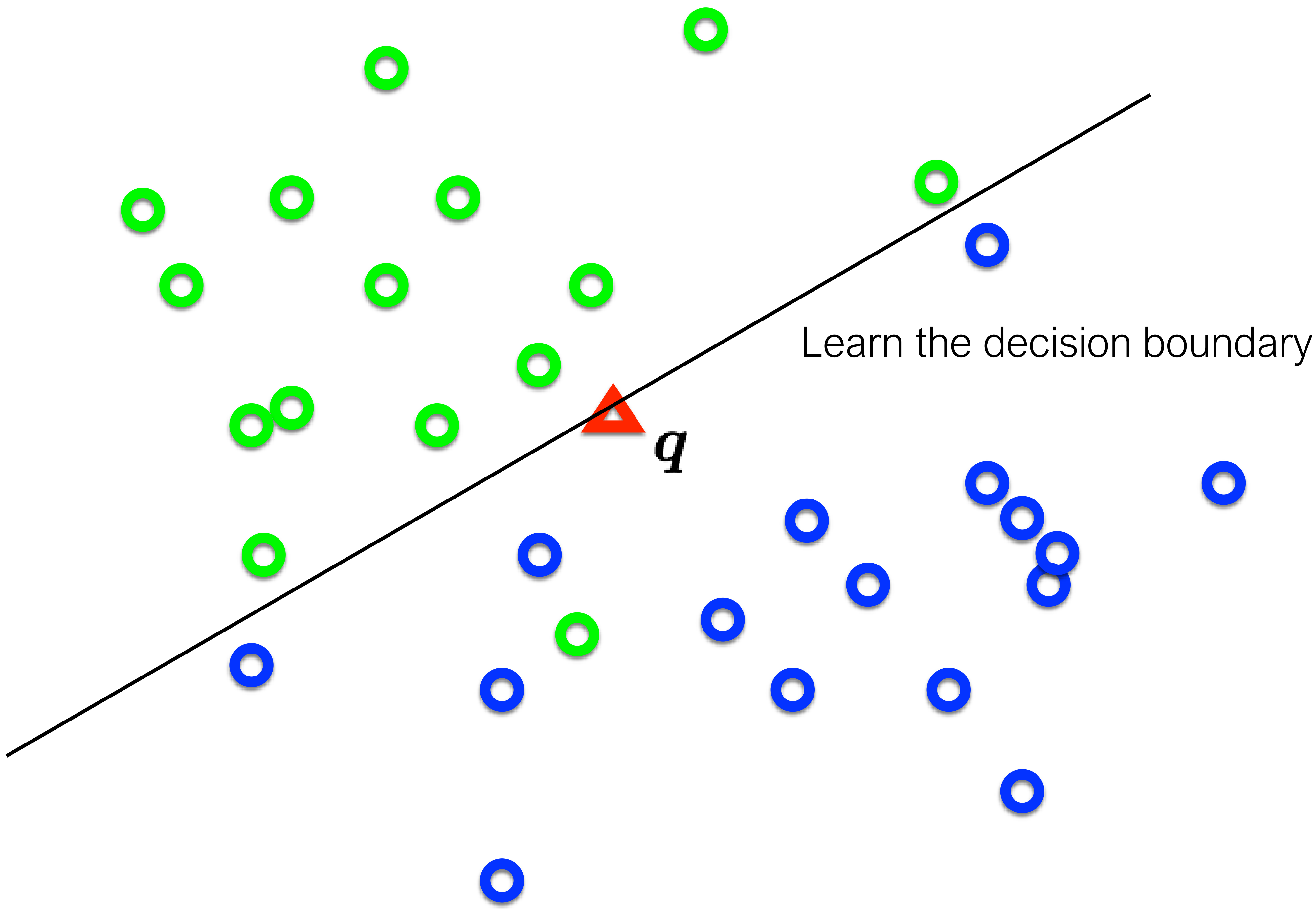


Distribution of data from two classes



Which class does q belong too?

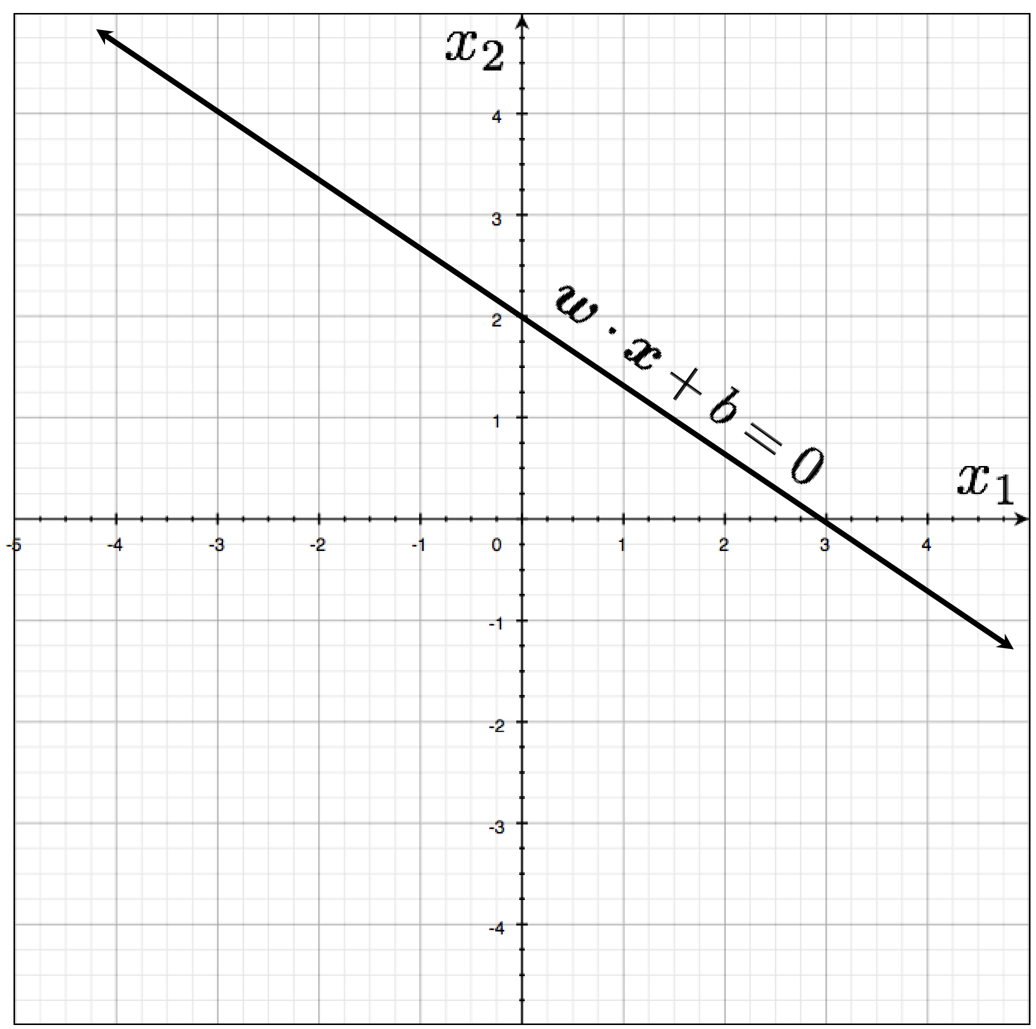
Distribution of data from two classes



First we need to understand hyperplanes…

Hyperplanes (lines) in 2D

$$w_1 x_1 + w_2 x_2 + b = 0$$



a line can be written as
dot product plus a bias

$$\boldsymbol{w} \cdot \boldsymbol{x} + b = 0$$

$$\boldsymbol{w} \in \mathcal{R}^2$$

another version, add a weight 1 and
push the bias inside

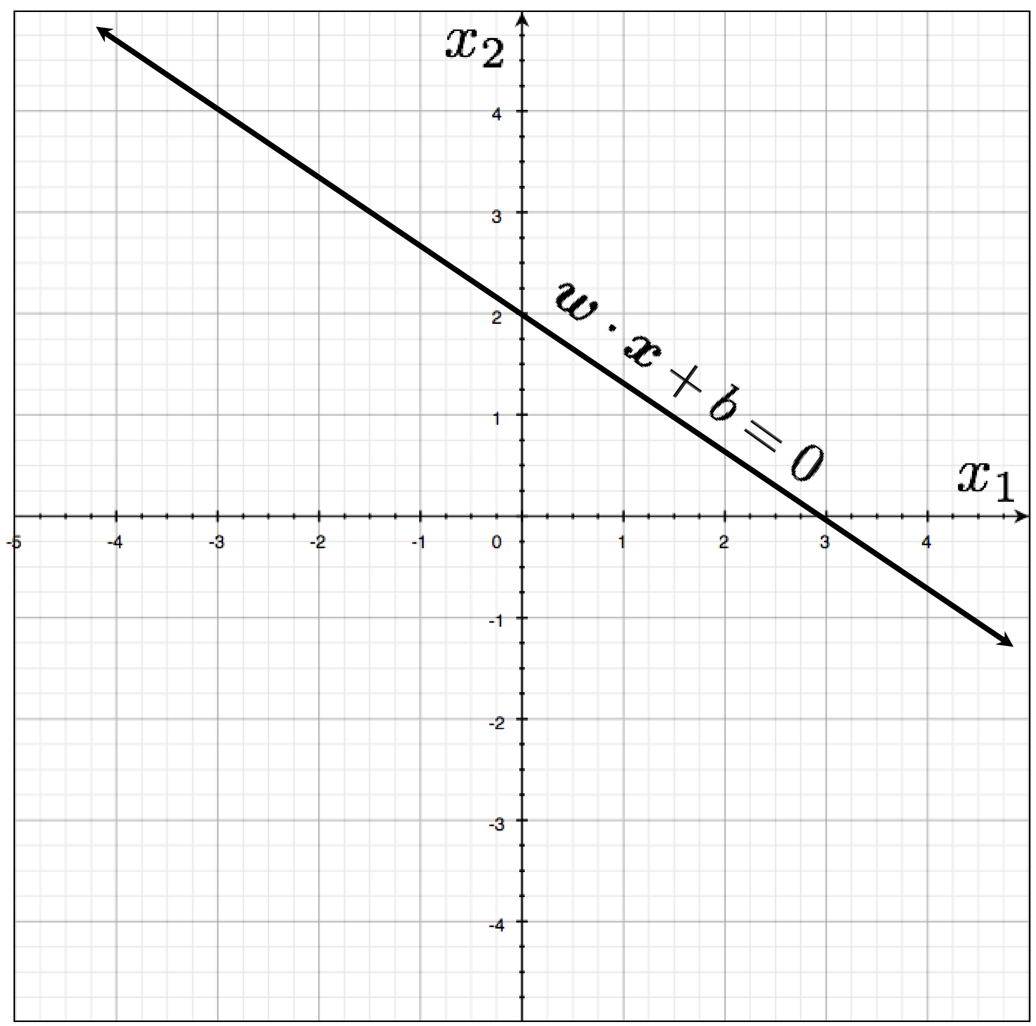
$$\boldsymbol{w} \cdot \boldsymbol{x} = 0$$

$$\boldsymbol{w} \in \mathcal{R}^3$$

Hyperplanes (lines) in 2D

$\boldsymbol{w} \cdot \boldsymbol{x} + b = 0$ (offset/bias outside) $\boldsymbol{w} \cdot \boldsymbol{x} = 0$ (offset/bias inside)

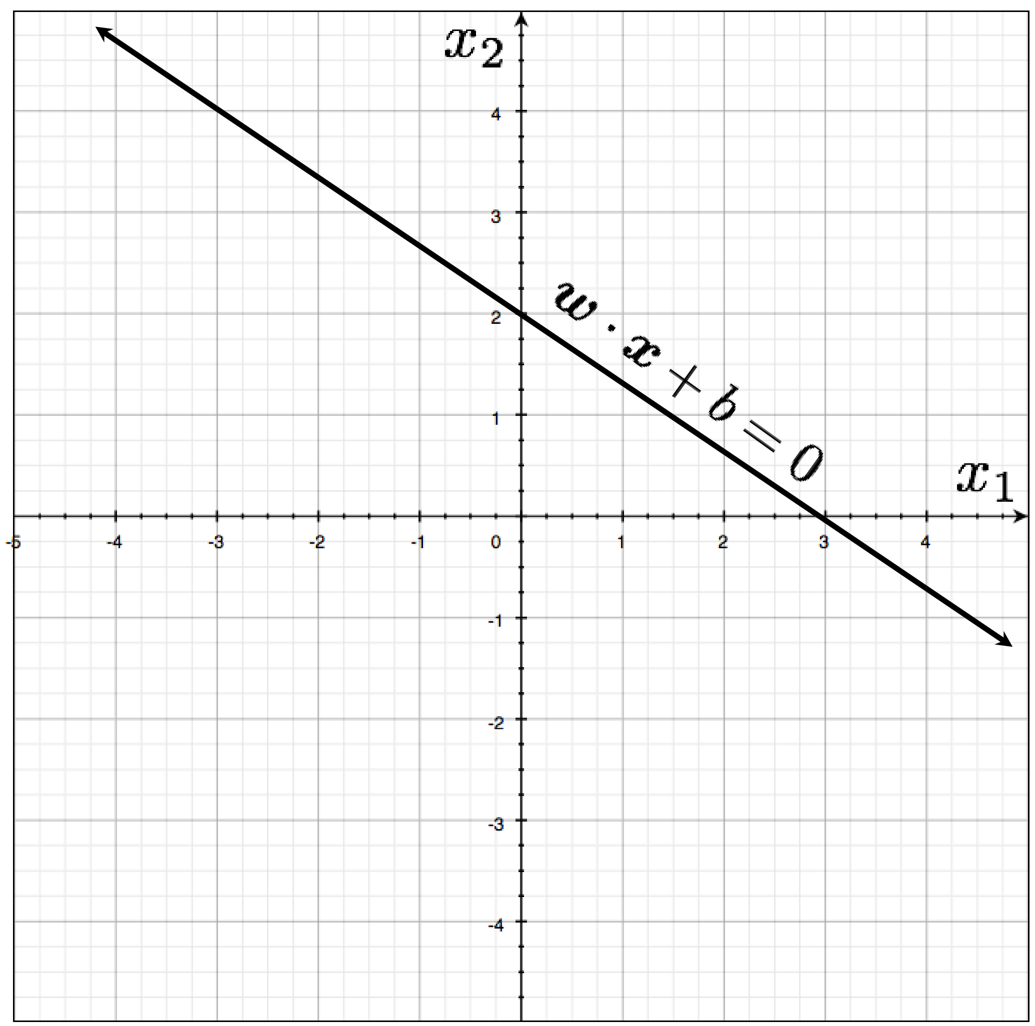
$$w_1 x_1 + w_2 x_2 + b = 0$$



Hyperplanes (lines) in 2D

$$\boldsymbol{w} \cdot \boldsymbol{x} + b = 0$$ (offset/bias outside) $$\boldsymbol{w} \cdot \boldsymbol{x} = 0$$ (offset/bias inside)

$$w_1 x_1 + w_2 x_2 + b = 0$$



Important property:

Free to choose any normalization of w

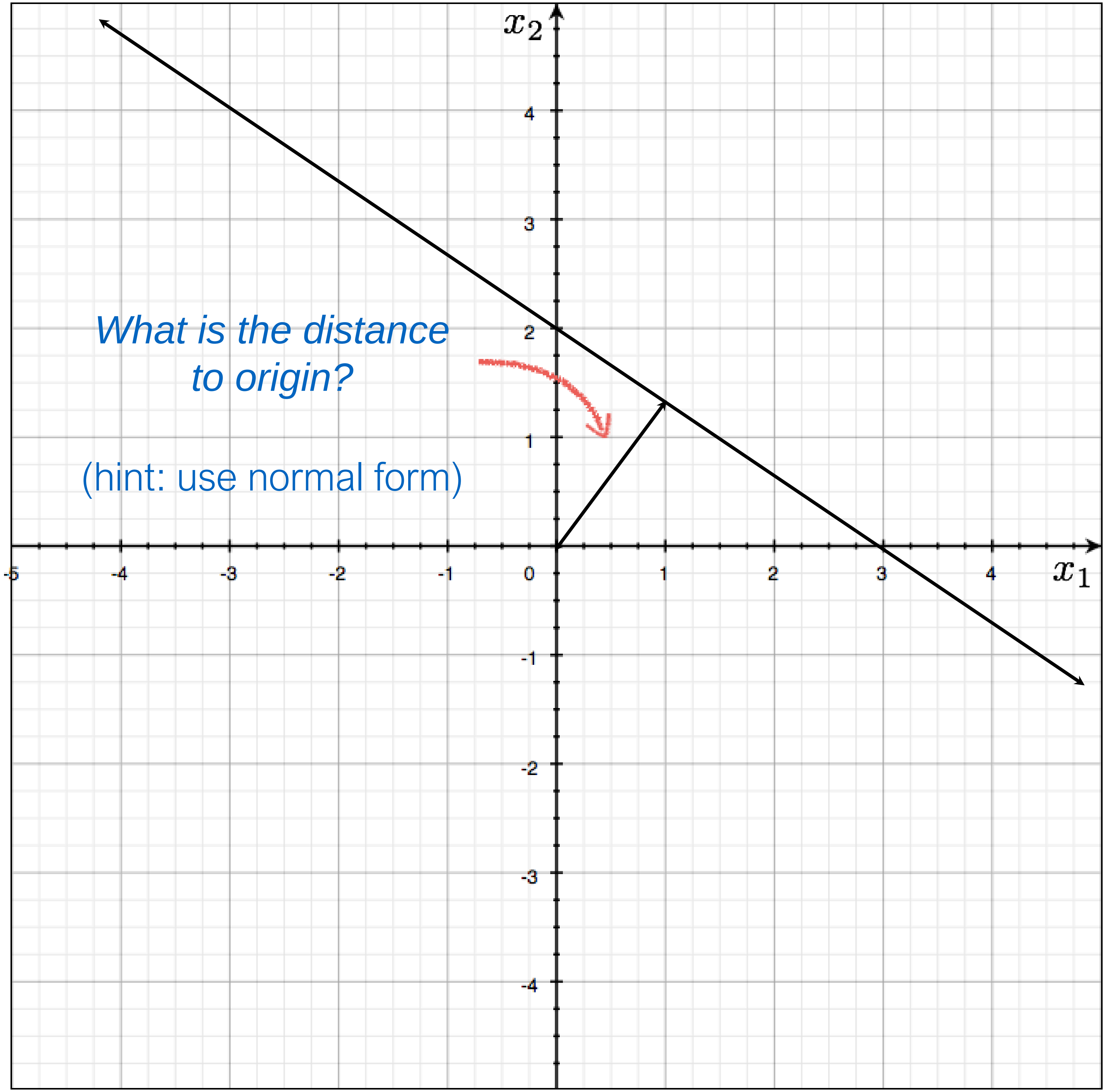
The line

$$w_1 x_1 + w_2 x_2 + b = 0$$

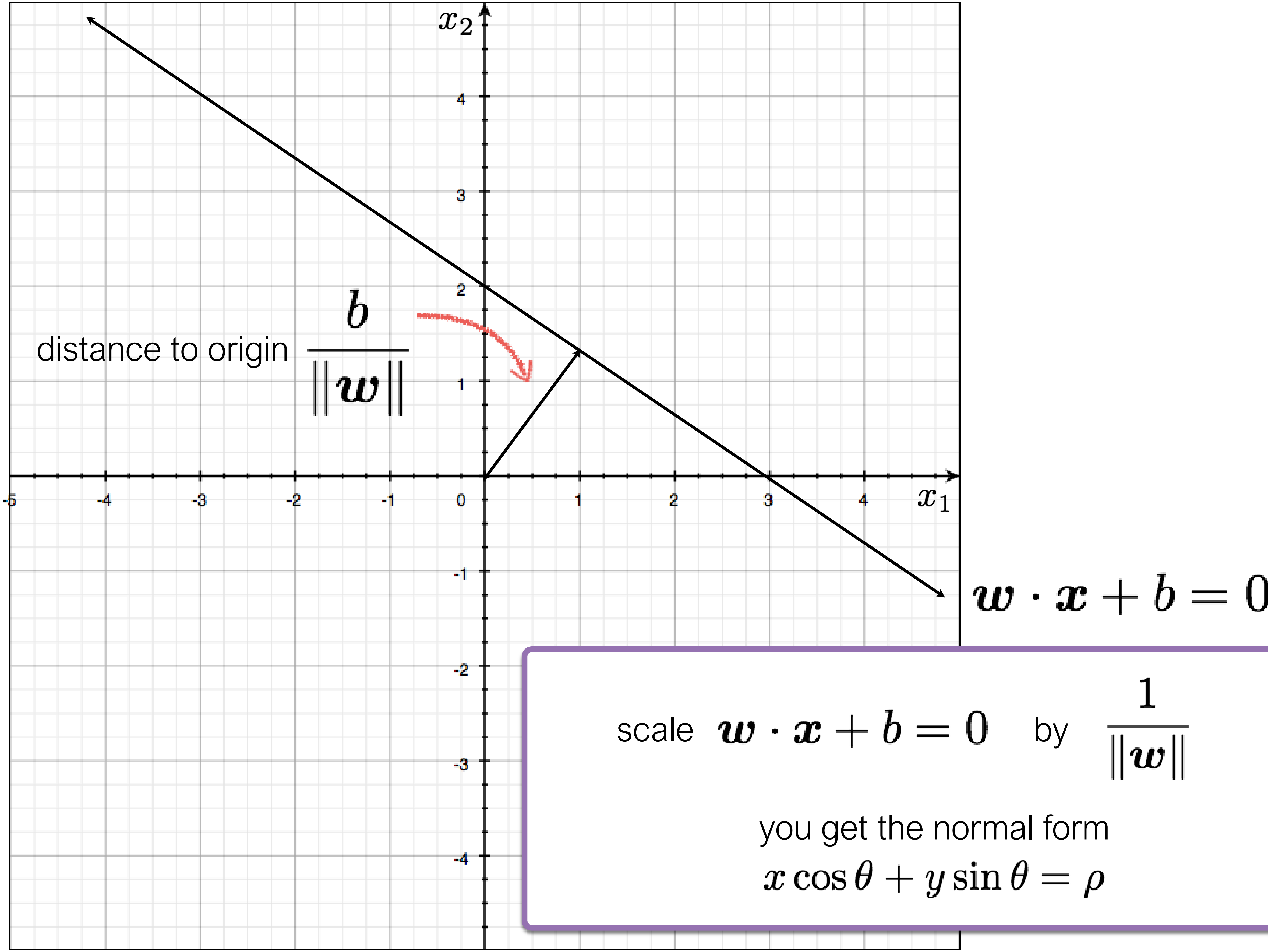
and the line

$$\lambda(w_1 x_1 + w_2 x_2 + b) = 0$$

define the same line



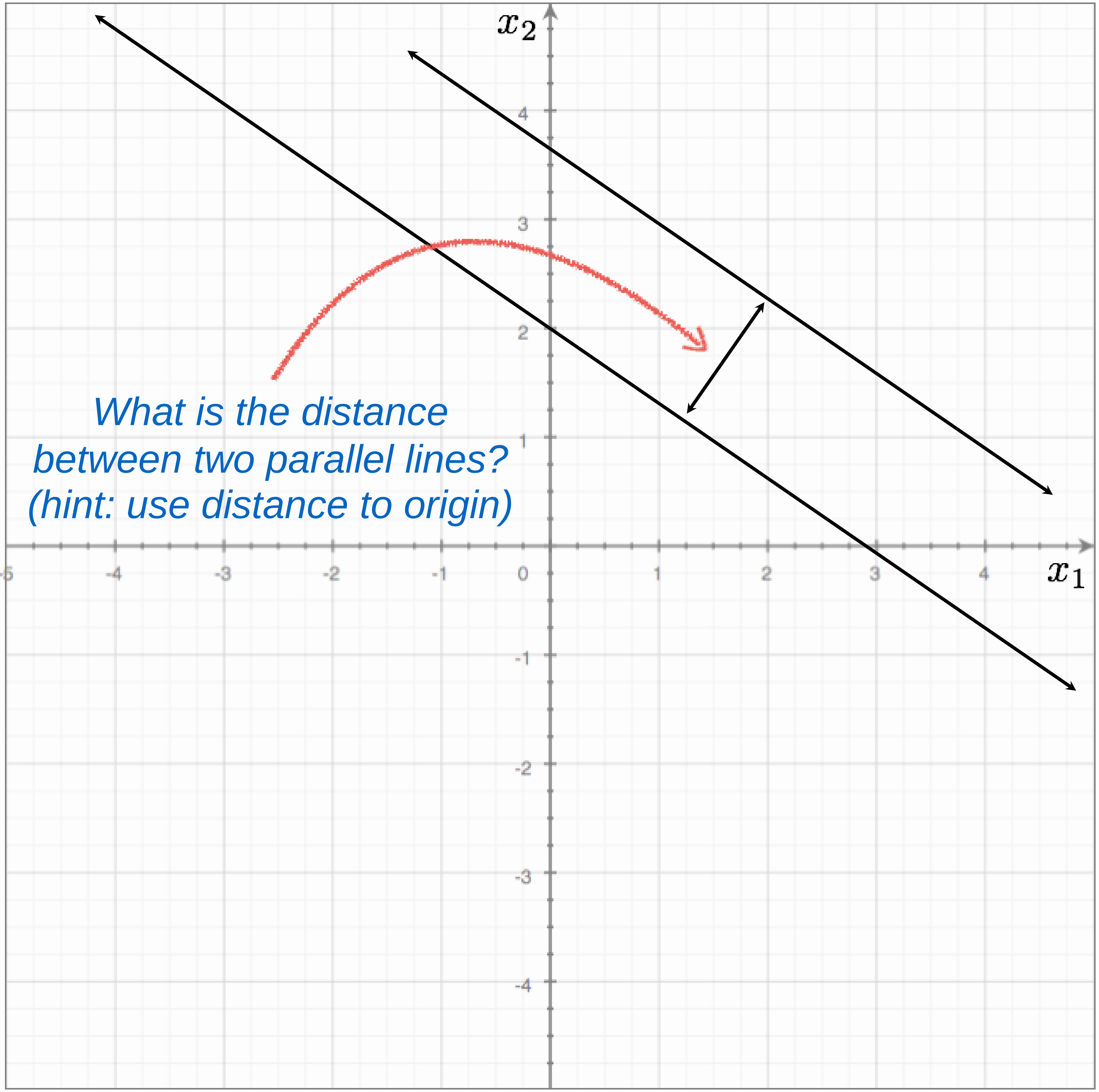
$$\boldsymbol{w} \cdot \boldsymbol{x} + b = 0$$



scale $\boldsymbol{w} \cdot \boldsymbol{x} + b = 0$ by $\dfrac{1}{\|\boldsymbol{w}\|}$

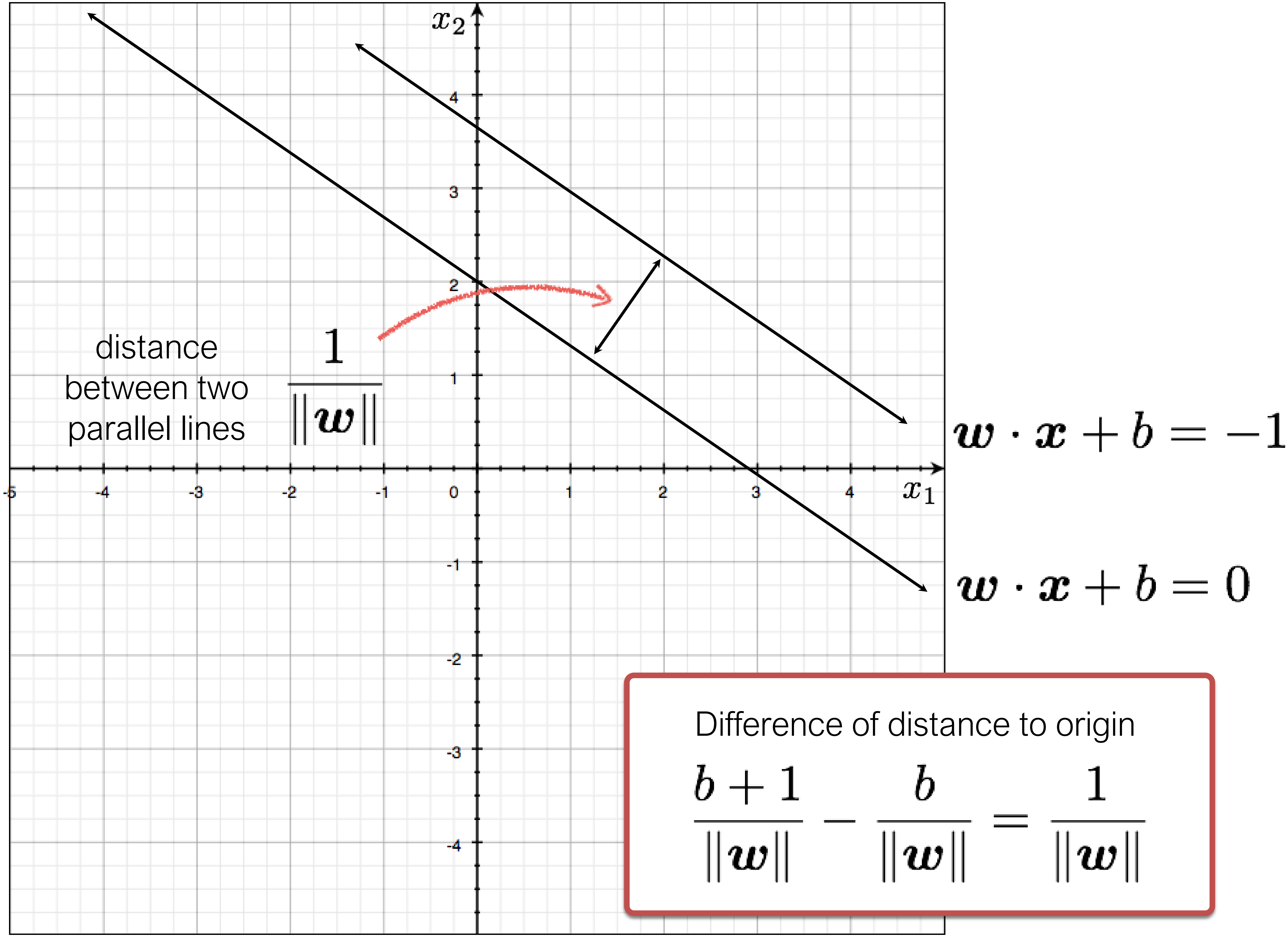
you get the normal form

$$x \cos \theta + y \sin \theta = \rho$$



$$\boldsymbol{w} \cdot \boldsymbol{x} + b = -1$$

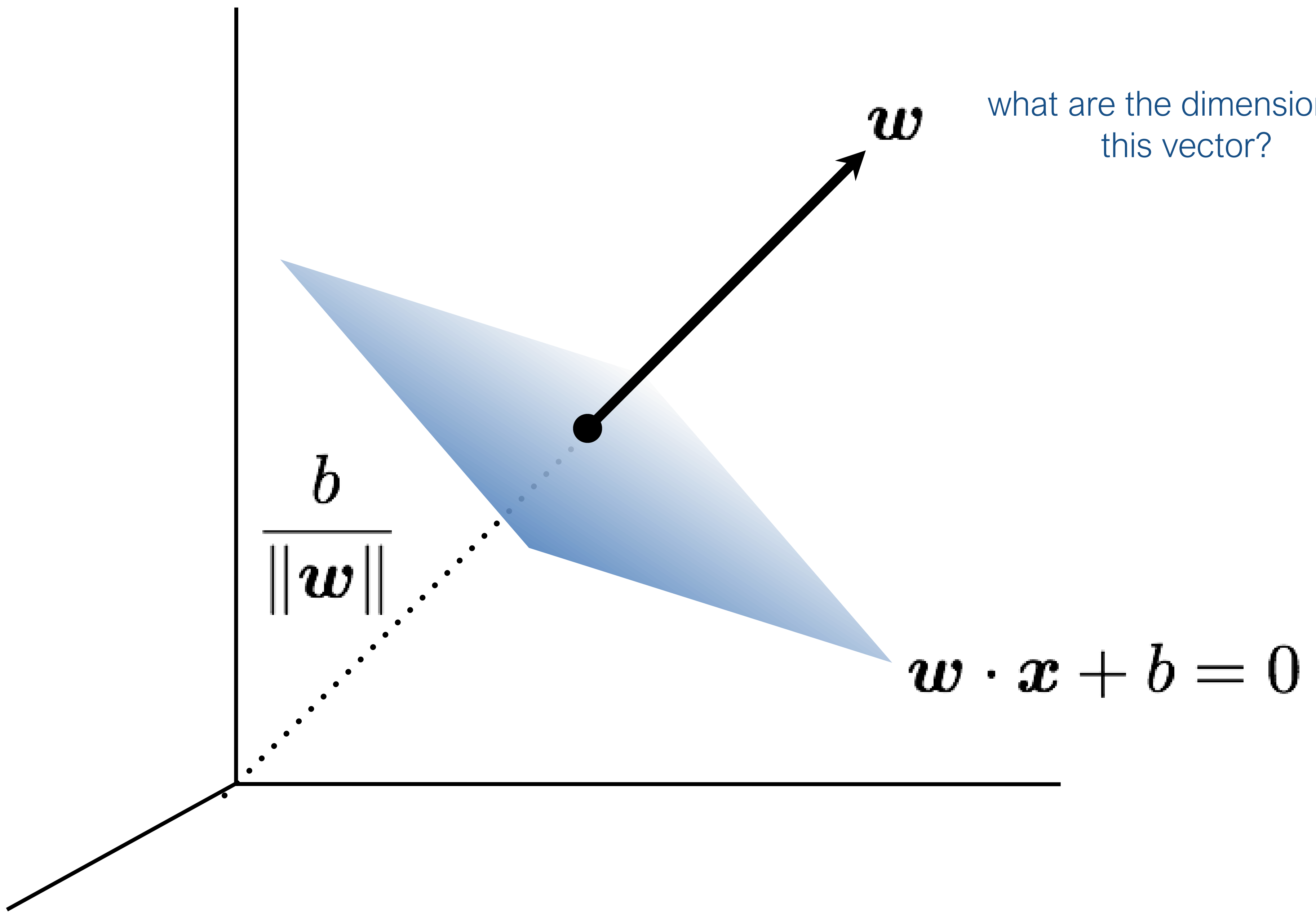
$$\boldsymbol{w} \cdot \boldsymbol{x} + b = 0$$



x_2
x_1
4
3
2
1
0
-1
-2
-3
-4
-5
distance between two parallel lines
$\frac{1}{\|\boldsymbol{w}\|}$
$\boldsymbol{w} \cdot \boldsymbol{x} + b = -1$
$\boldsymbol{w} \cdot \boldsymbol{x} + b = 0$
Difference of distance to origin
$\frac{b+1}{\|\boldsymbol{w}\|} - \frac{b}{\|\boldsymbol{w}\|} = \frac{1}{\|\boldsymbol{w}\|}$

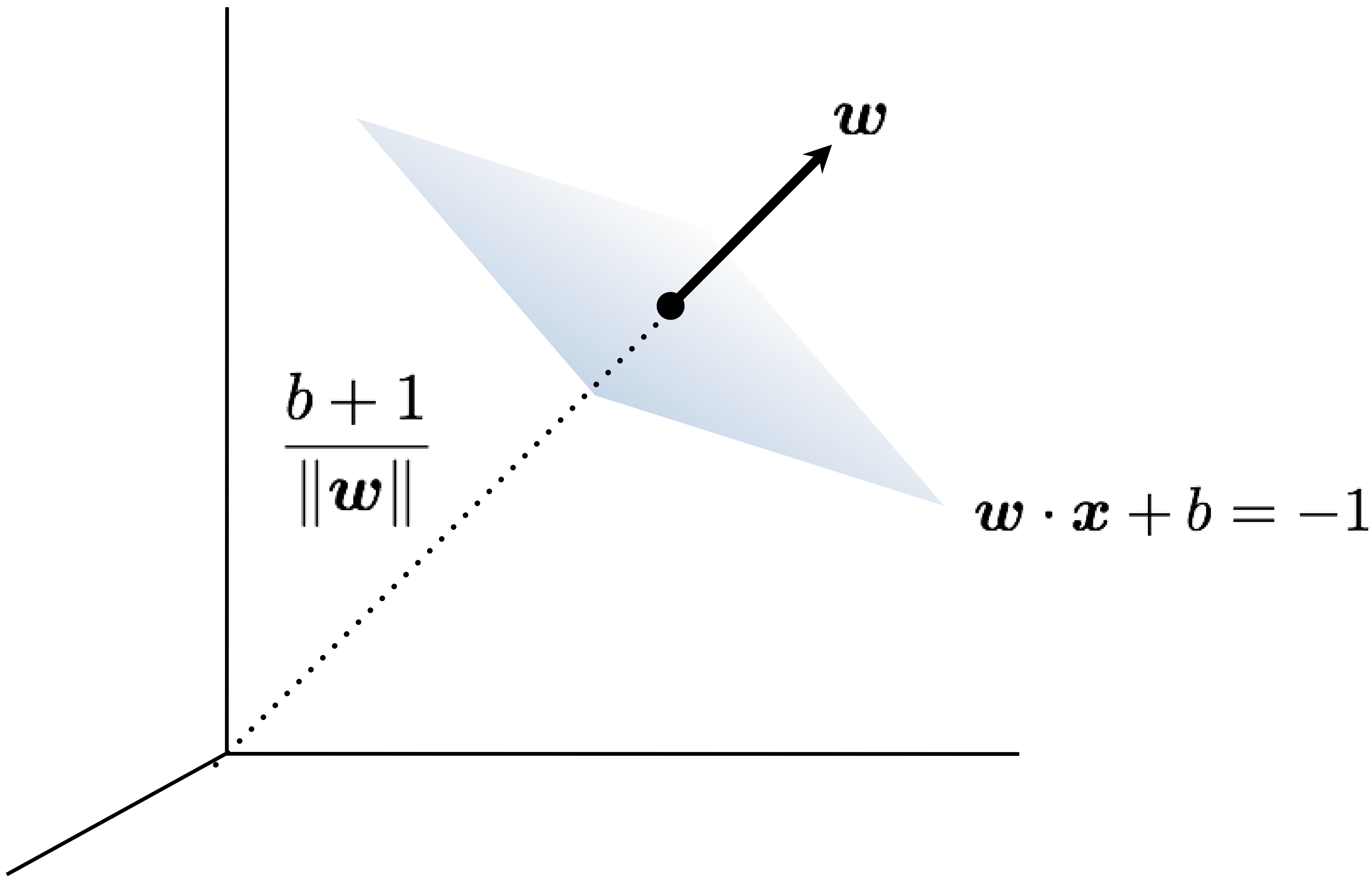
Now we can go to 3D …

Hyperplanes (planes) in 3D



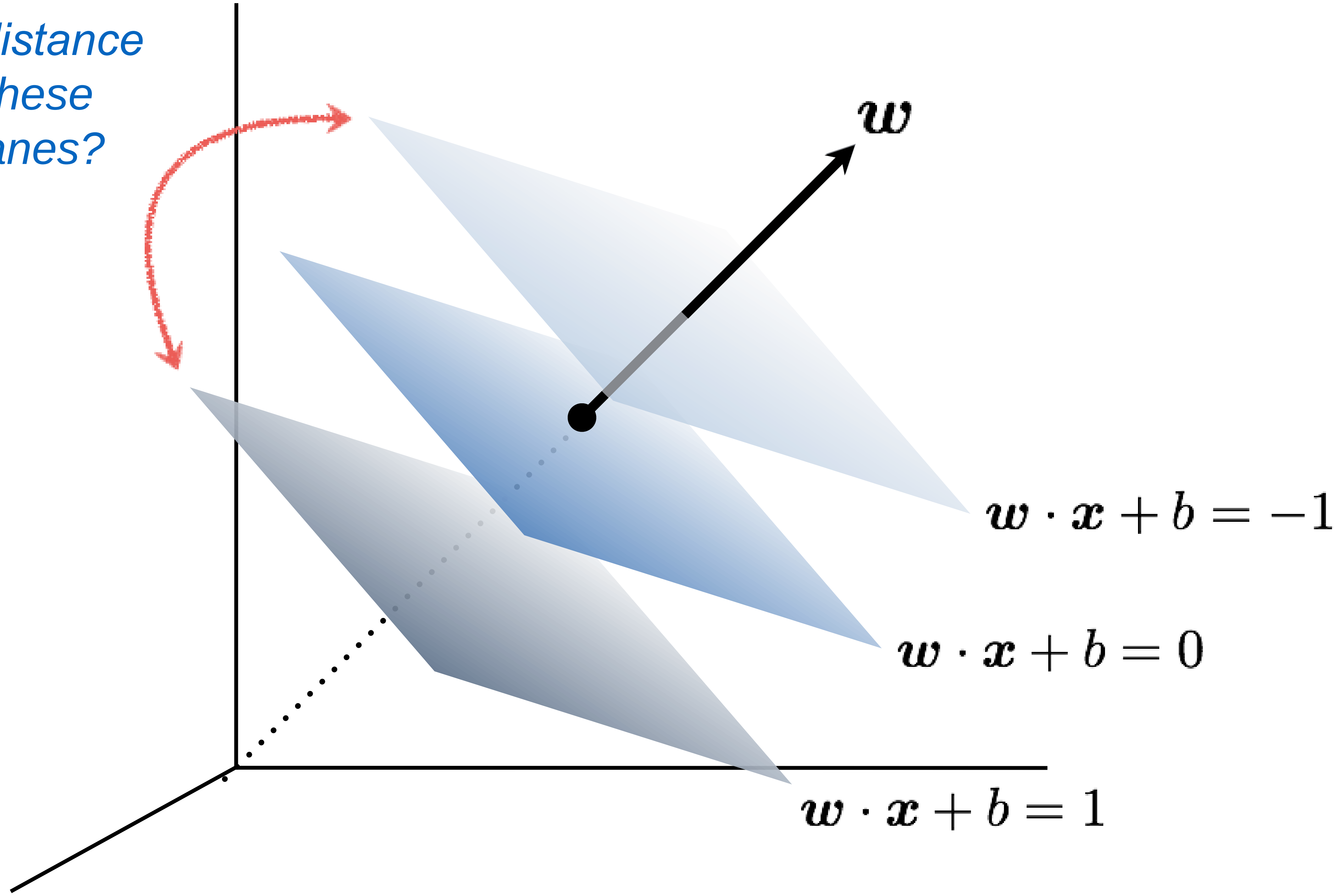
*What happens if you change **b**?*

Hyperplanes (planes) in 3D

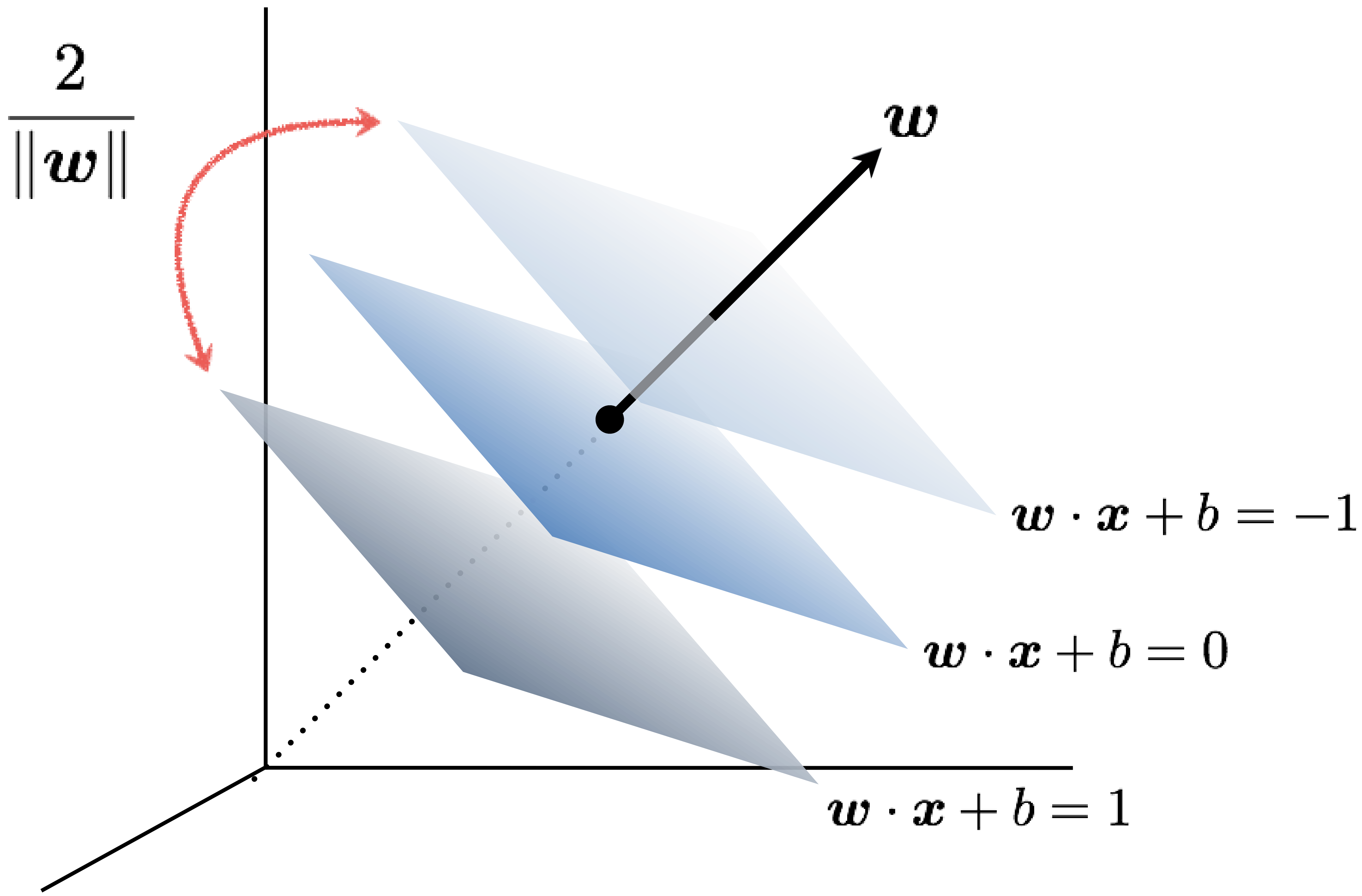


Hyperplanes (planes) in 3D

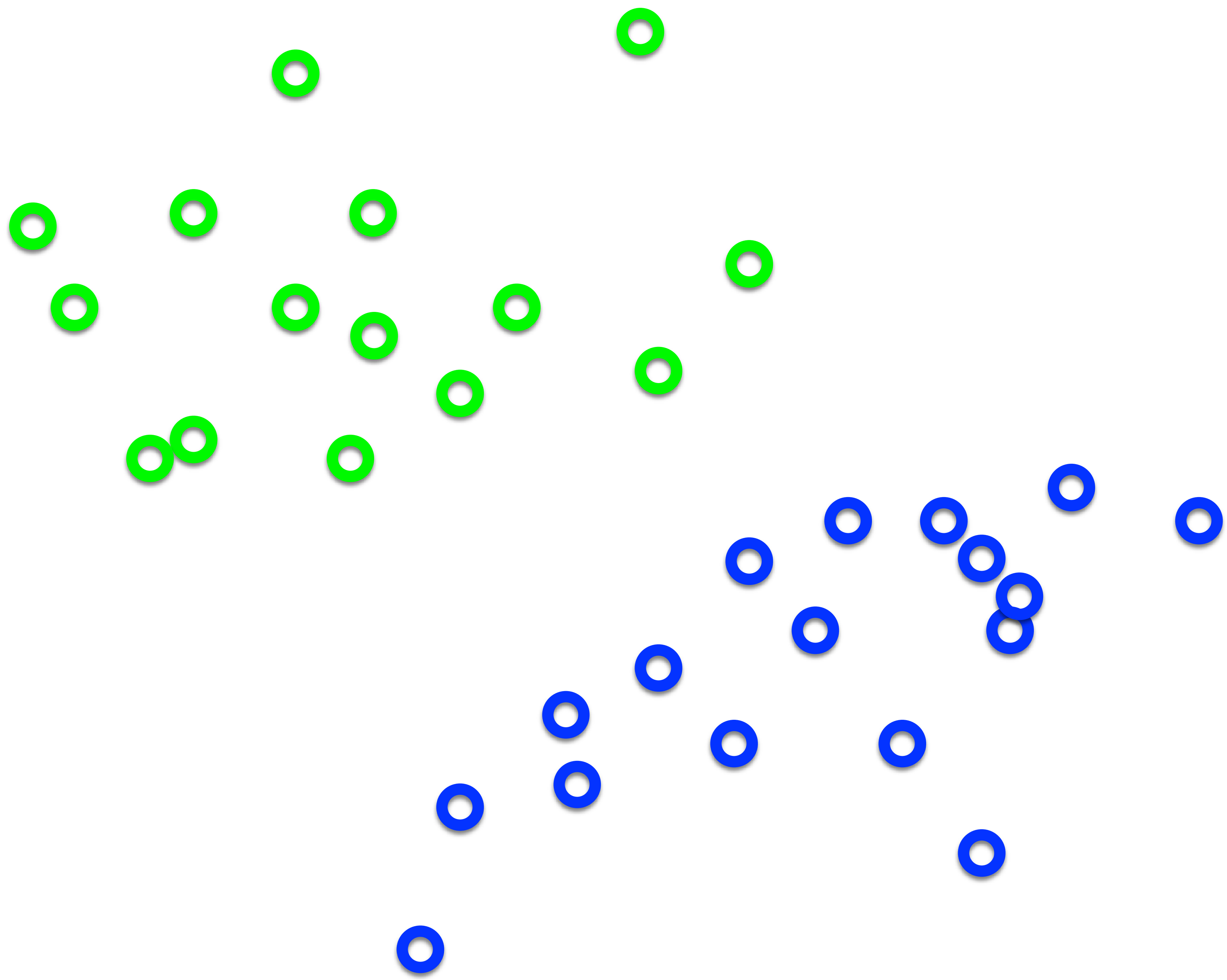
What's the distance between these parallel planes?



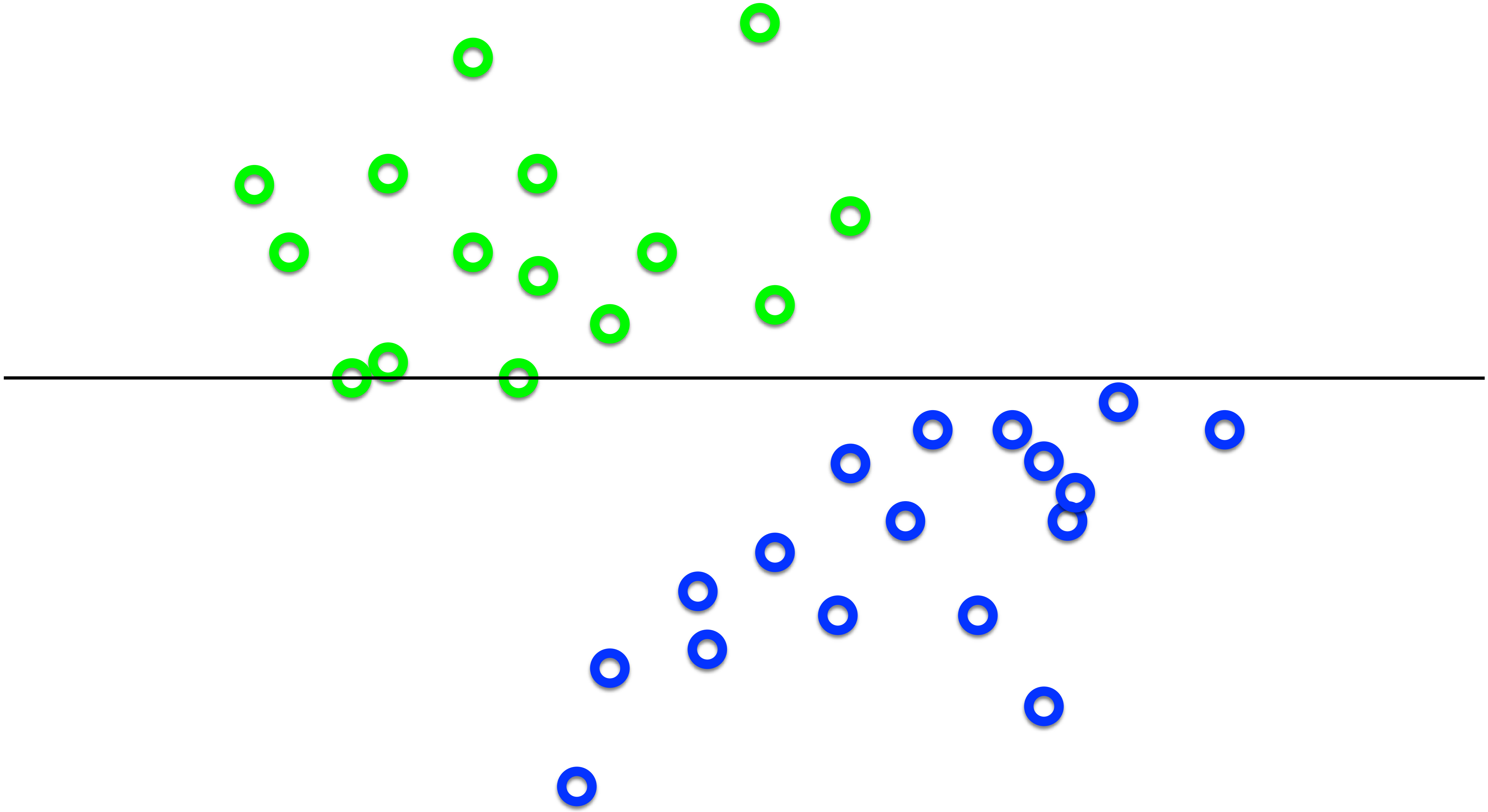
Hyperplanes (planes) in 3D



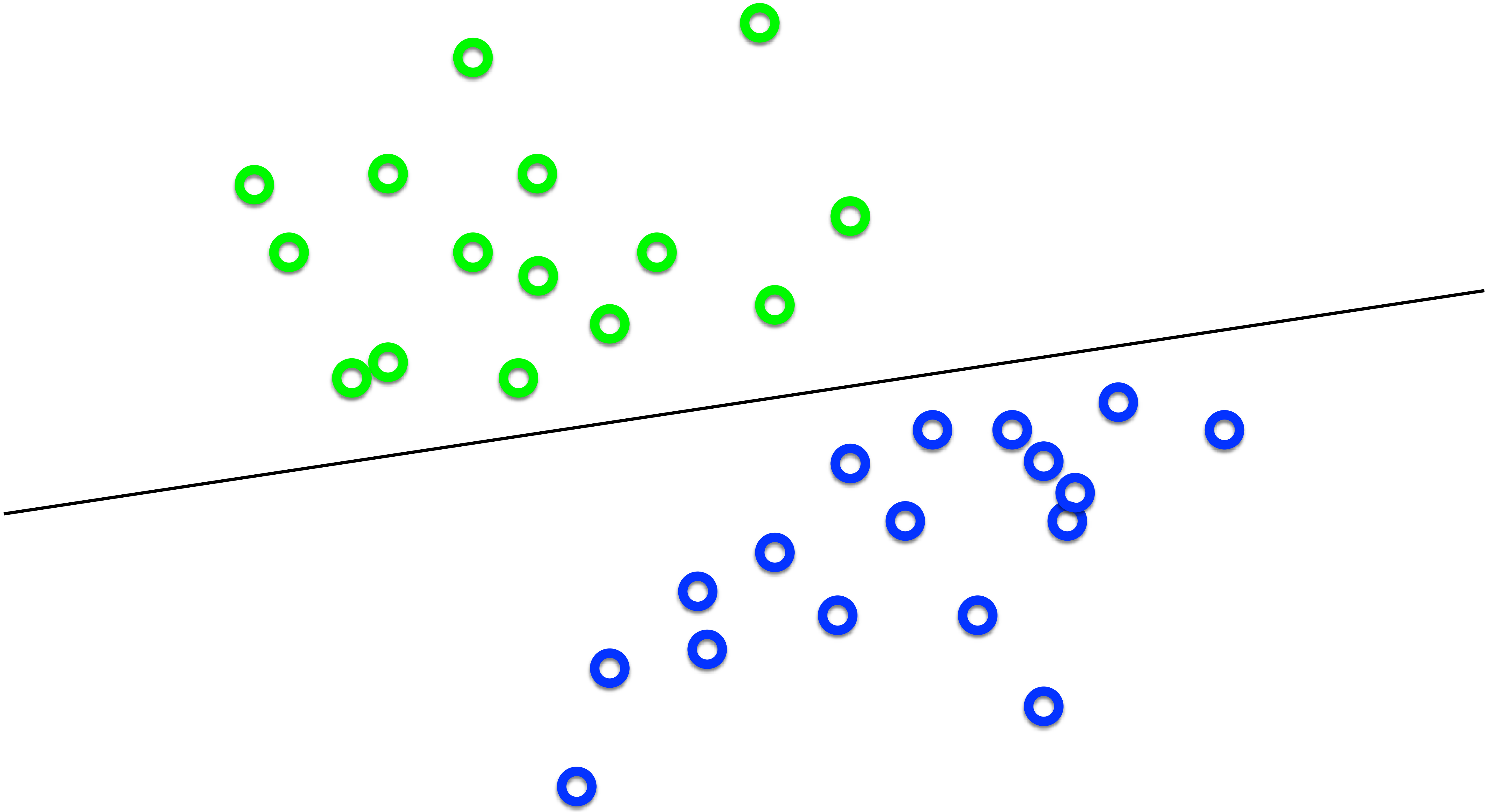
What's the best **w**?



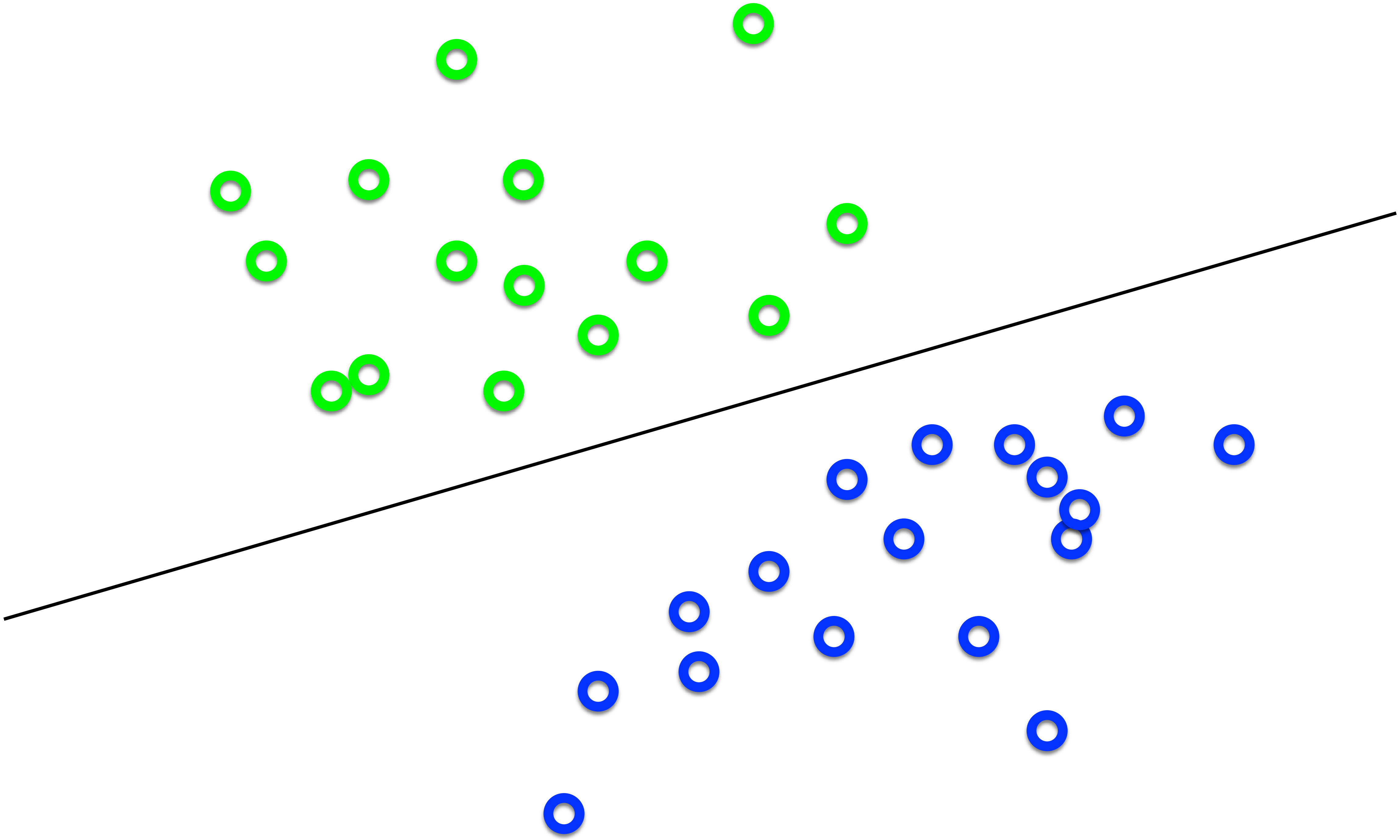
What’s the best **w**?



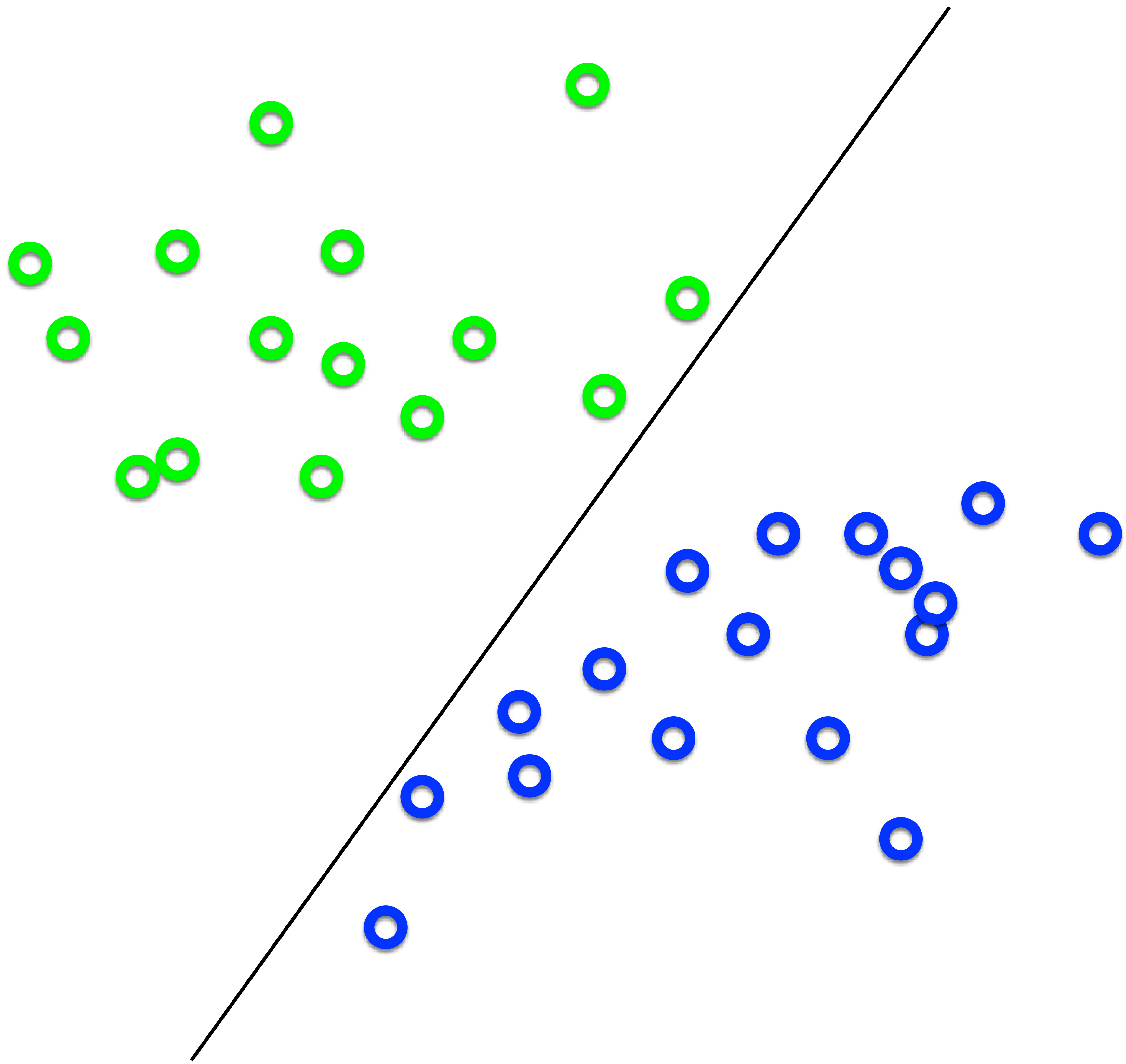
What's the best **w**?



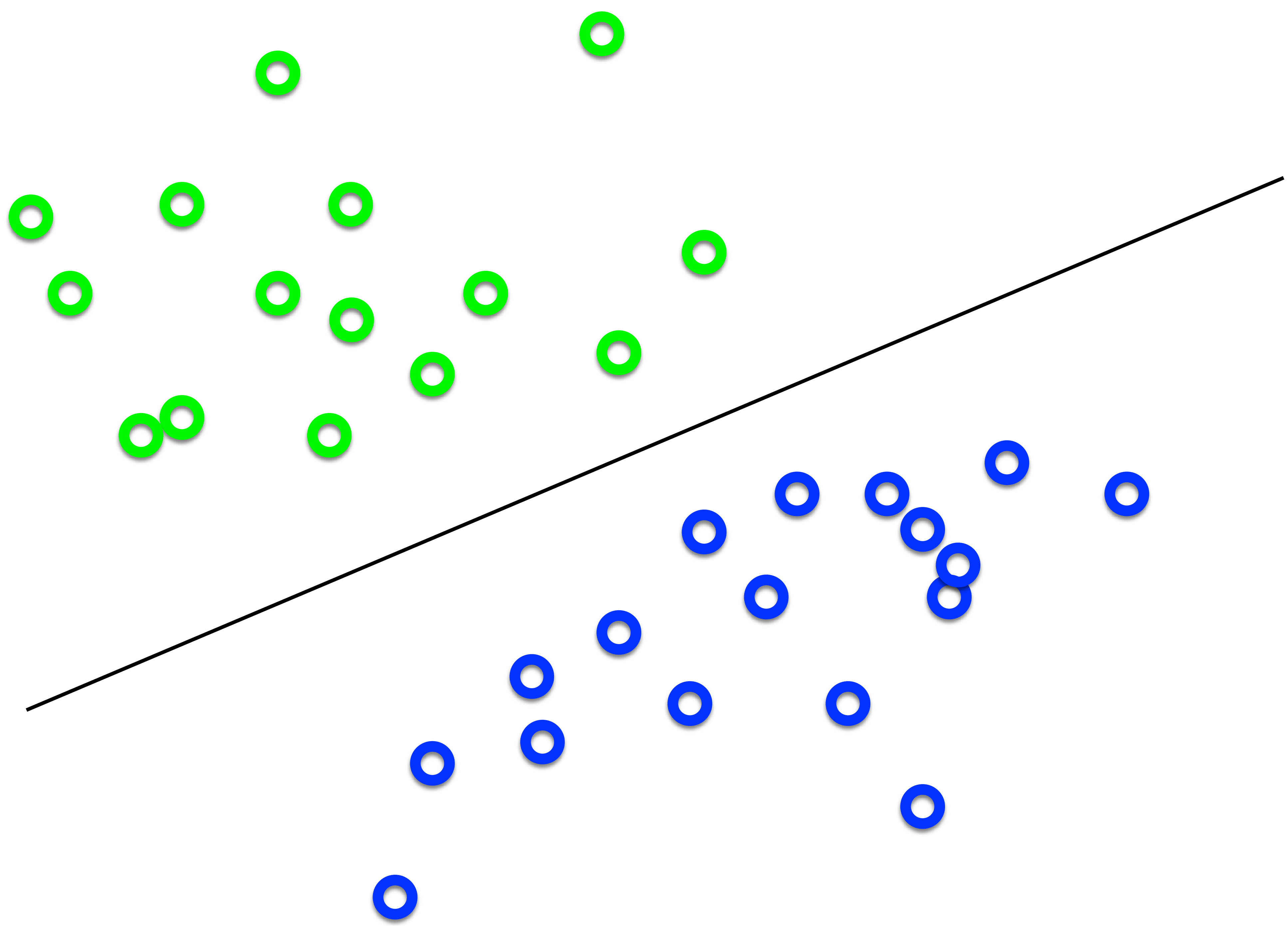
What's the best **w**?



What's the best **w**?

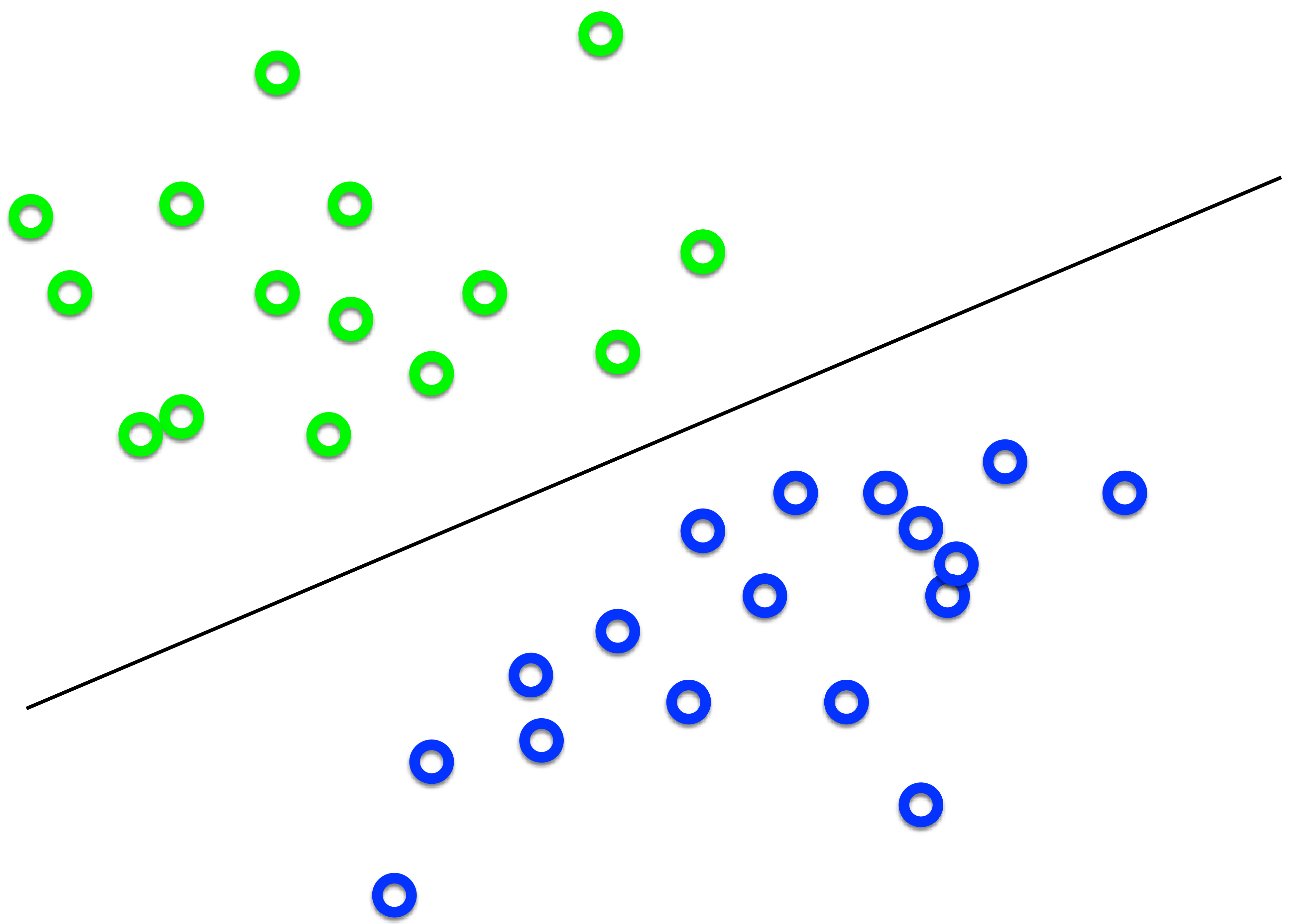


What's the best **w**?



Intuitively, the line that is the farthest from all interior points

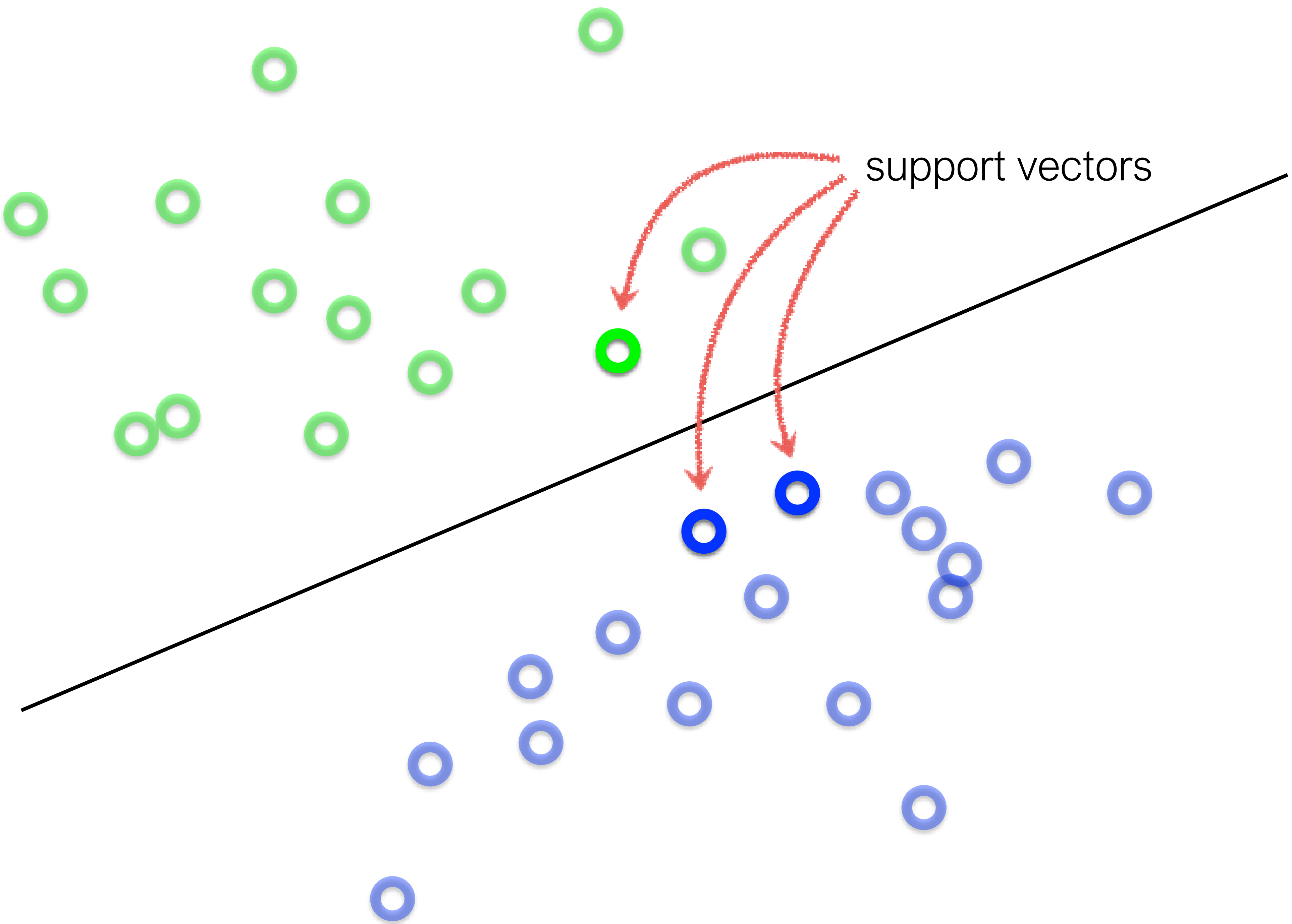
What's the best **w**?



Maximum Margin solution:

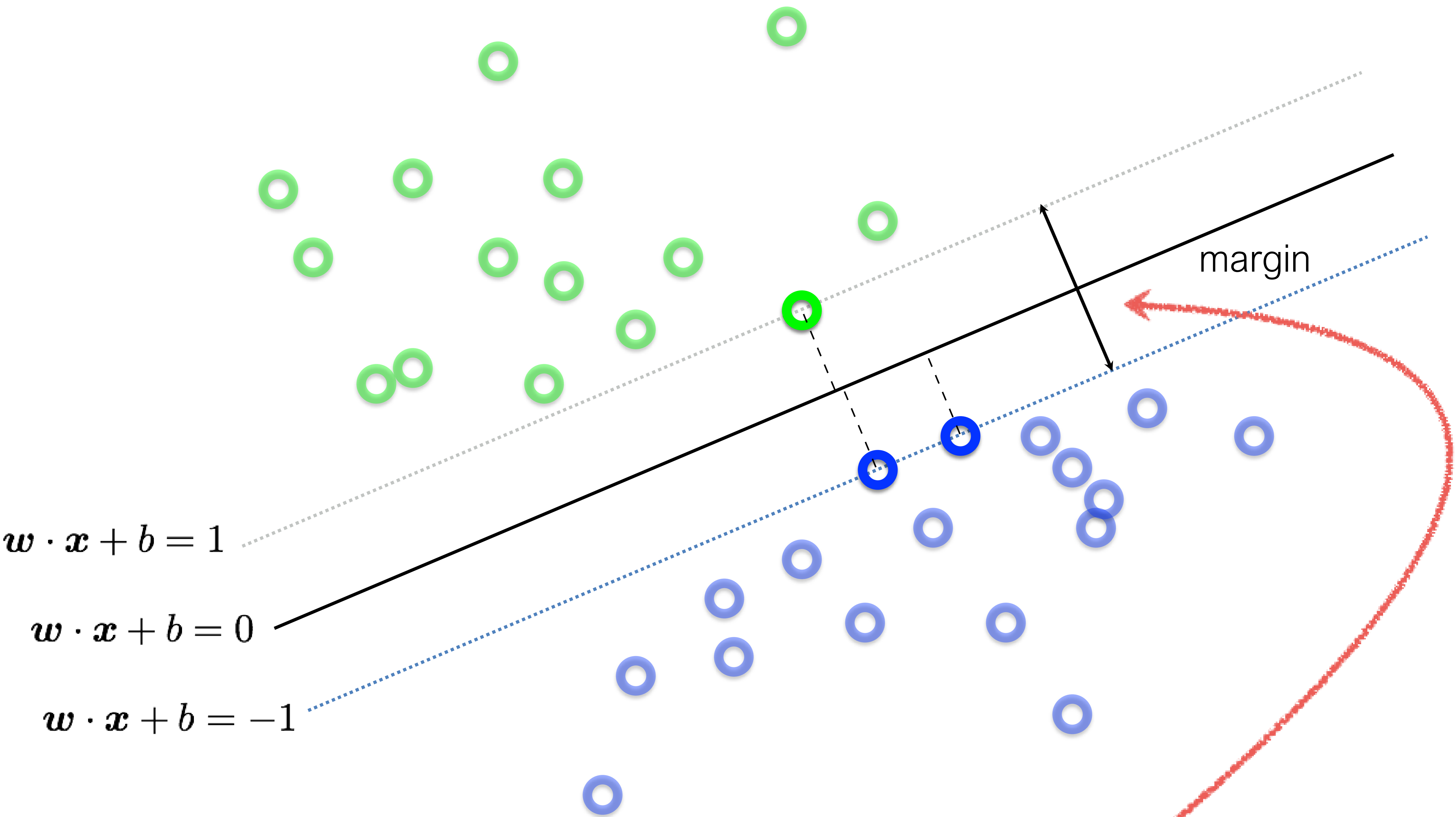
most stable to perturbations of data

What's the best **w**?



Want a hyperplane that is far away from 'inner points'

Find hyperplane **w** such that …



the gap between parallel hyperplanes $\dfrac{2}{\|\boldsymbol{w}\|}$ is maximized

Can be formulated as a maximization problem

$$\max_{\boldsymbol{w}} \frac{2}{||\boldsymbol{w}||}$$

$$\text{subject to } \boldsymbol{w} \cdot \boldsymbol{x}_i + b \begin{array}{l} \geq +1 \quad \text{if} \quad y_i = +1 \\ \leq -1 \quad \text{if} \quad y_i = -1 \end{array} \text{ for } \quad i = 1, \ldots, N$$

What does this constraint mean?

label of the data point

Why is it +1 and -1?

Can be formulated as a maximization problem

$$\max_{\boldsymbol{w}} \frac{2}{\|\boldsymbol{w}\|}$$

$$\text{subject to } \boldsymbol{w} \cdot \boldsymbol{x}_i + b \begin{array}{l} \geq +1 \text{ if } y_i = +1 \\ \leq -1 \text{ if } y_i = -1 \end{array} \text{ for } i = 1, \ldots, N$$

Equivalently,

Where did the 2 go?

$$\min_{\boldsymbol{w}} \|\boldsymbol{w}\|$$

$$\text{subject to } y_i(\boldsymbol{w} \cdot \boldsymbol{x}_i + b) \geq 1 \text{ for } i = 1, \ldots, N$$

What happened to the labels?

‘Primal formulation’ of a linear SVM

$$\min_{\boldsymbol{w}} ||\boldsymbol{w}||$$

Objective Function

$$\text{subject to } y_i(\boldsymbol{w} \cdot \boldsymbol{x}_i + b) \geq 1 \text{ for } i = 1, \ldots, N$$

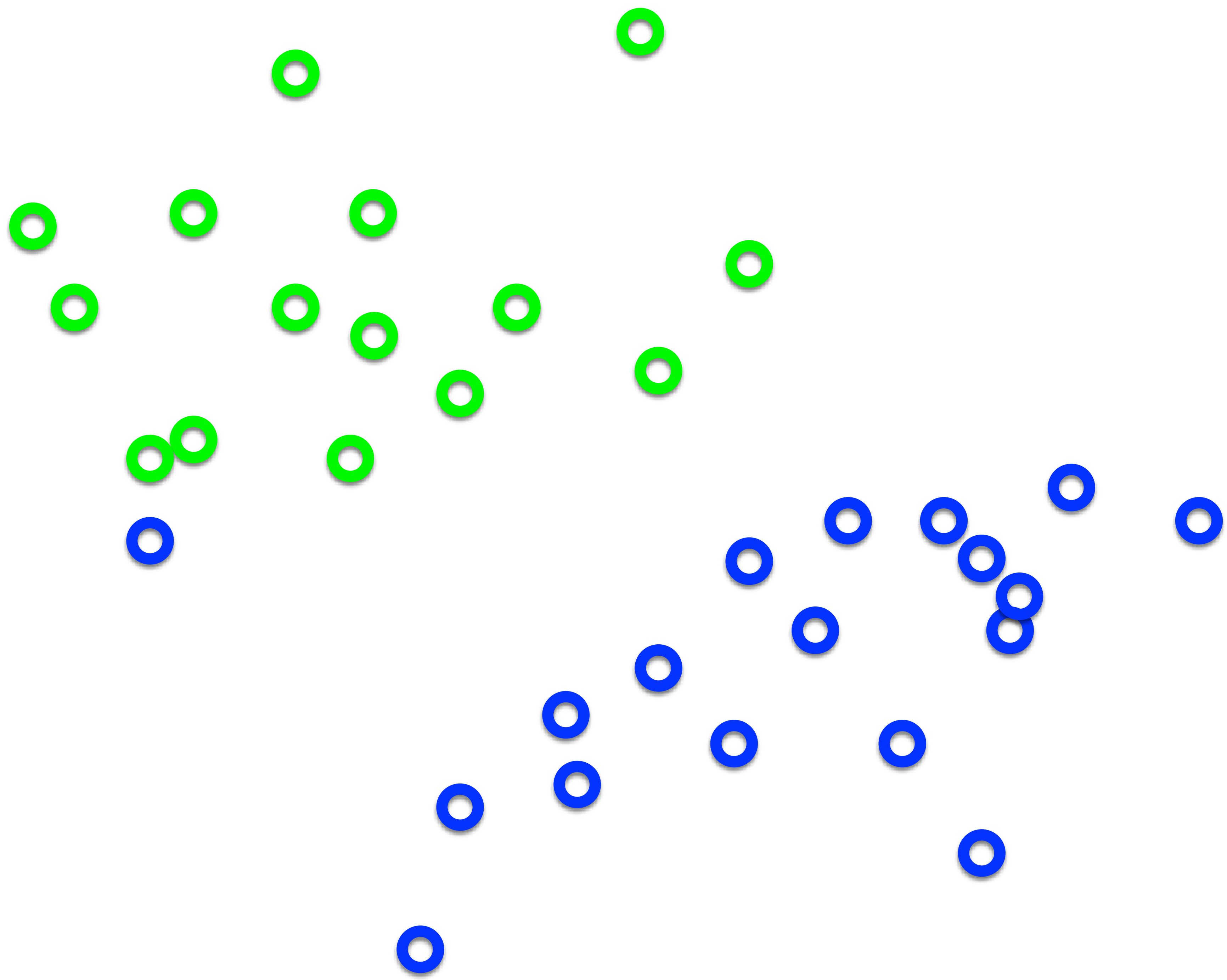
Constraints

This is a convex quadratic programming (QP) problem

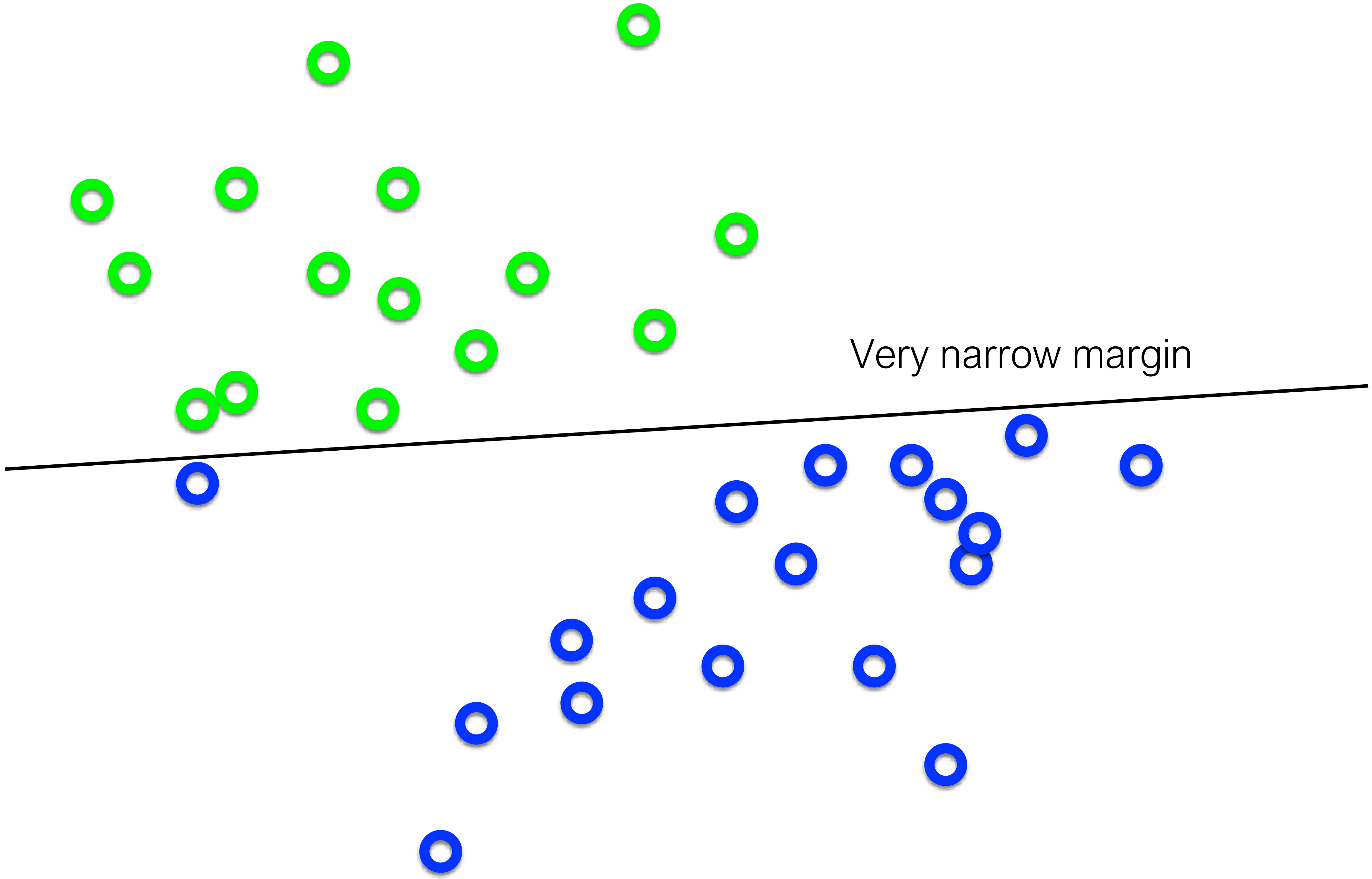
(a unique solution exists)

‘soft’ margin

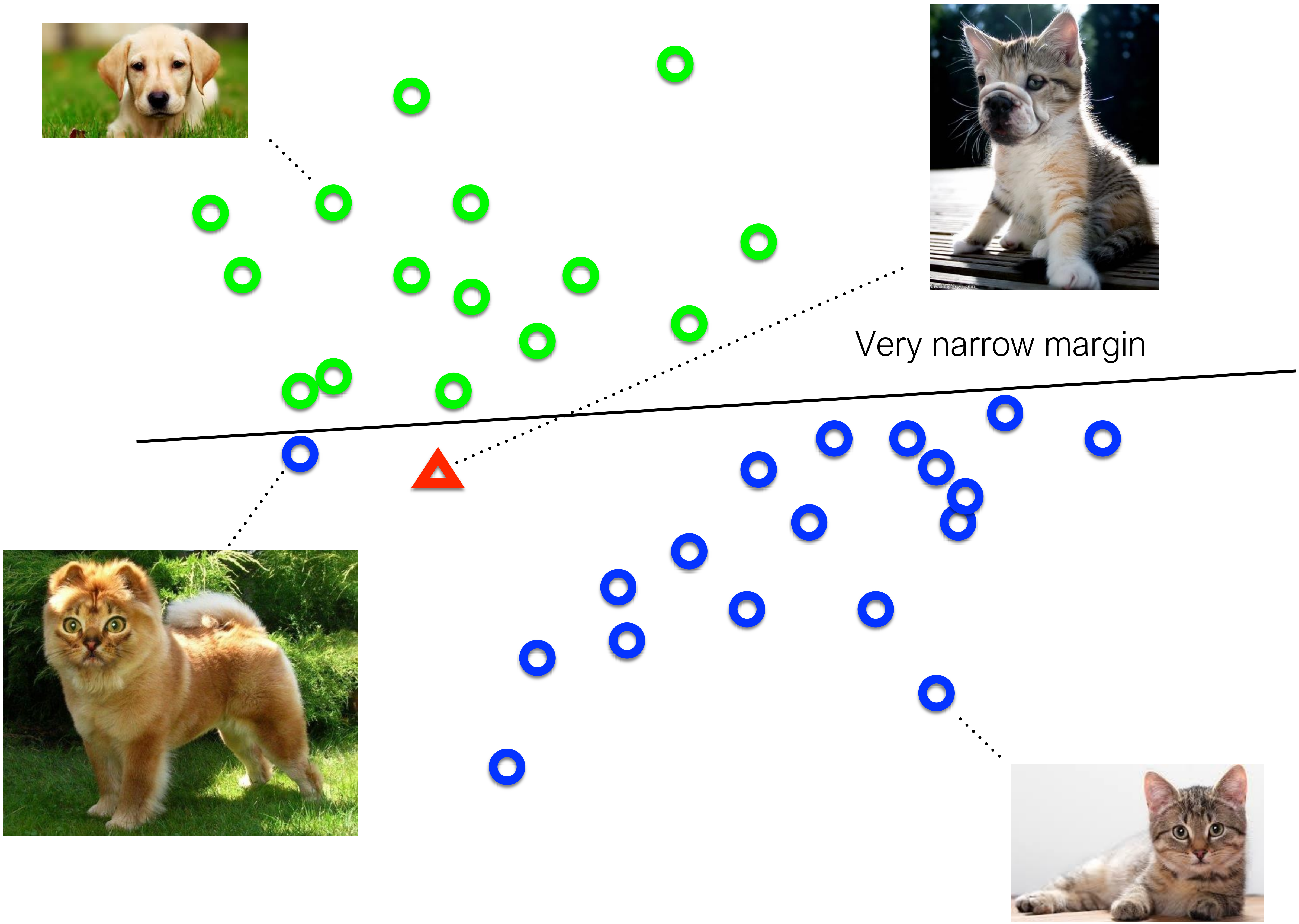
What's the best **w**?



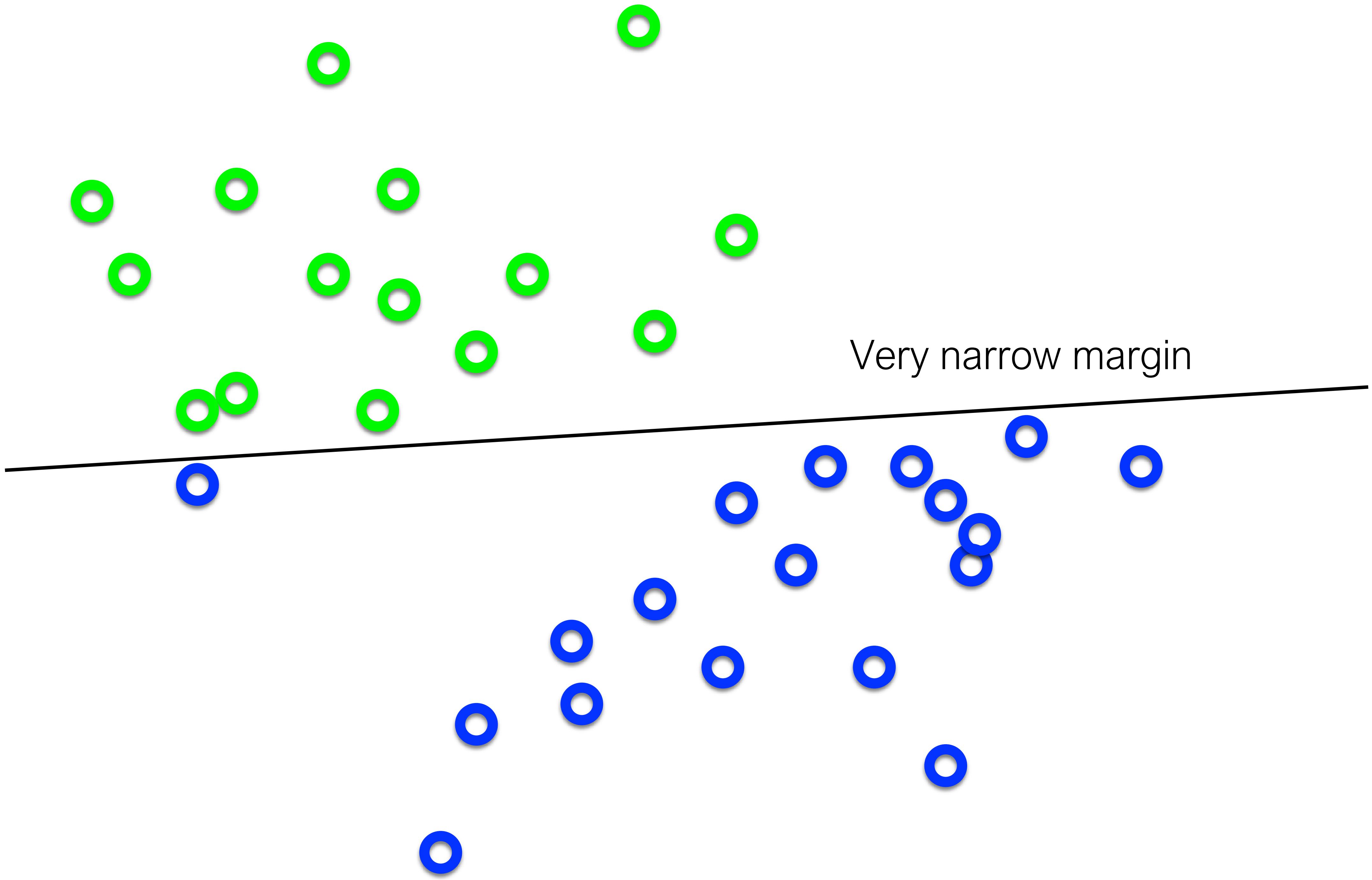
What's the best **w**?



Separating cats and dogs

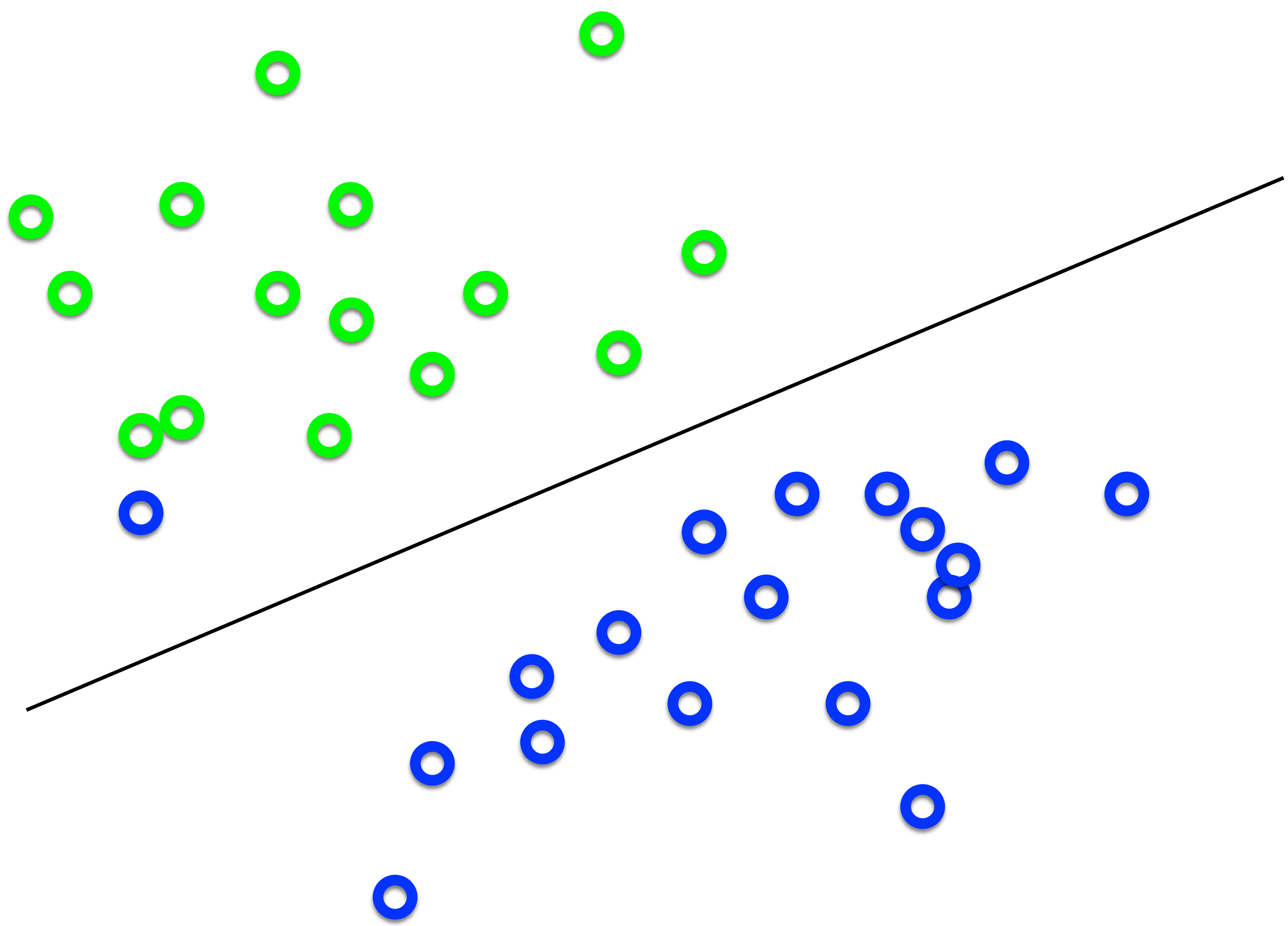


What’s the best **w**?



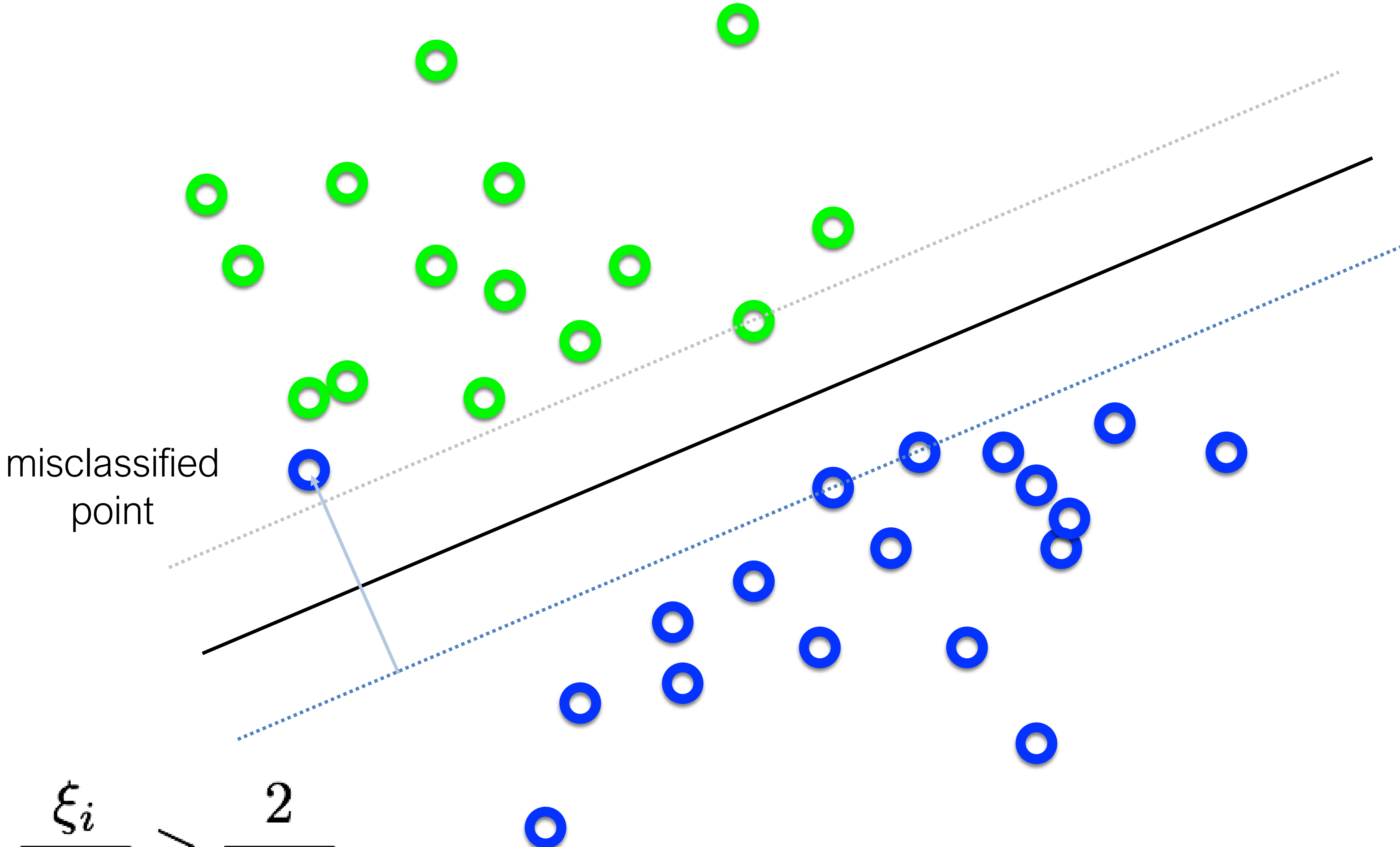
Intuitively, we should allow for some misclassification if we can get more robust classification

What's the best **w**?



Trade-off between the MARGIN and the MISTAKES
(might be a better solution)

Adding slack variables $\xi_i \geq 0$



$$\frac{\xi_i}{\|\boldsymbol{w}\|} > \frac{2}{\|\boldsymbol{w}\|}$$

'soft' margin

objective

$$\min_{\boldsymbol{w},\boldsymbol{\xi}} \|\boldsymbol{w}\|^2 + C\sum_i \xi_i$$

subject to

$$y_i(\boldsymbol{w}^\top \boldsymbol{x}_i + b) \geq 1 - \xi_i$$

for $i = 1, \ldots, N$

'soft' margin

objective

$$\min_{\boldsymbol{w},\boldsymbol{\xi}} \|\boldsymbol{w}\|^2 + C\sum_i \xi_i$$

subject to

$$y_i(\boldsymbol{w}^\top \boldsymbol{x}_i + b) \geq 1 - \xi_i$$

$$\text{for} \quad i = 1, \ldots, N$$

The slack variable allows for mistakes,
as long as the inverse margin is minimized.

‘soft’ margin

objective

$$\min_{\boldsymbol{w},\boldsymbol{\xi}} \|\boldsymbol{w}\|^2 + C \sum_i \xi_i$$

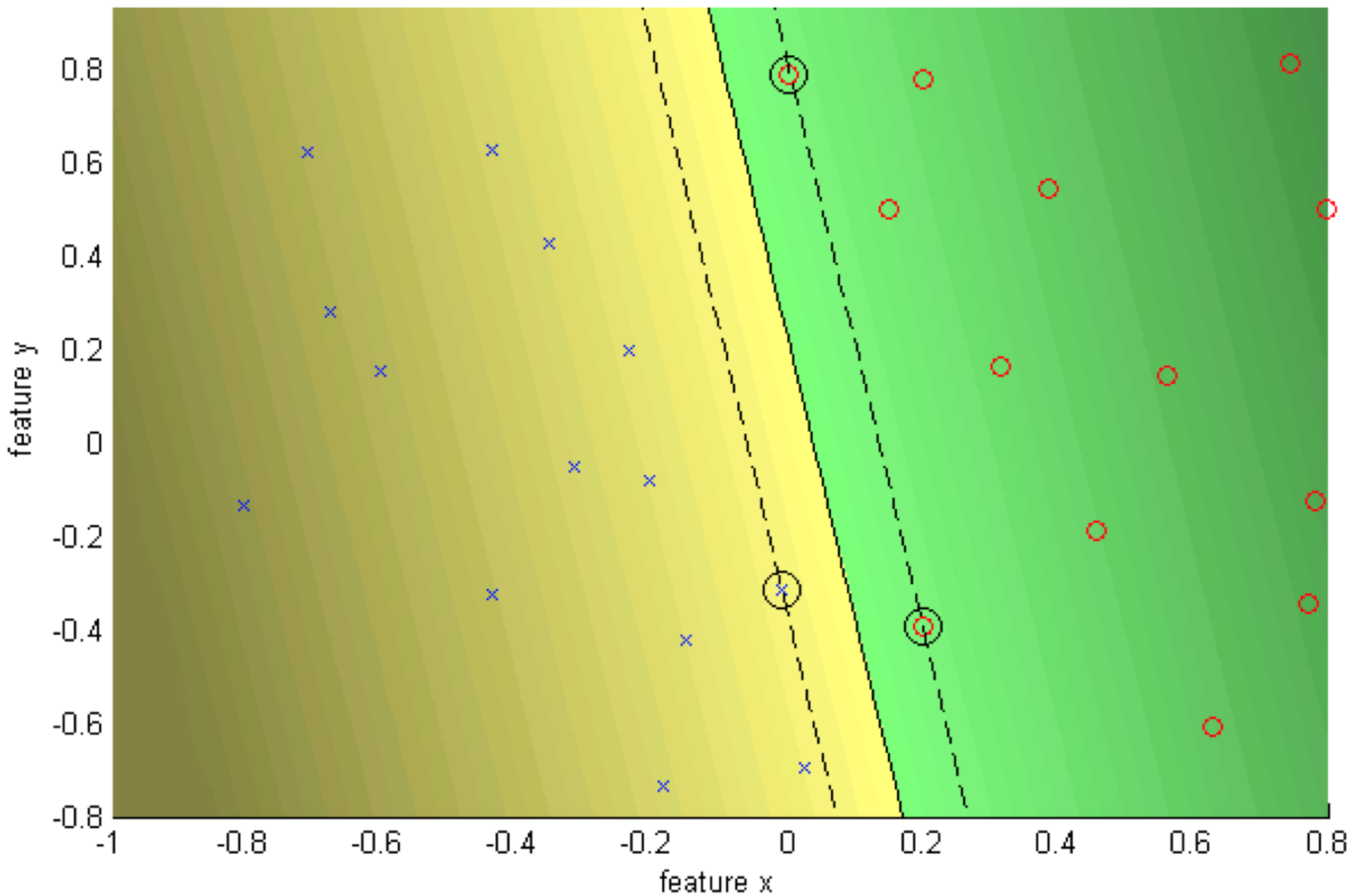
subject to

$$y_i(\boldsymbol{w}^\top \boldsymbol{x}_i + b) \geq 1 - \xi_i$$

$$\text{for} \quad i = 1, \ldots, N$$

- Every constraint can be satisfied if slack is large
- C is a regularization parameter
  - Small C: ignore constraints (larger margin)
  - Big C: constraints (small margin)
- Still QP problem (unique solution)

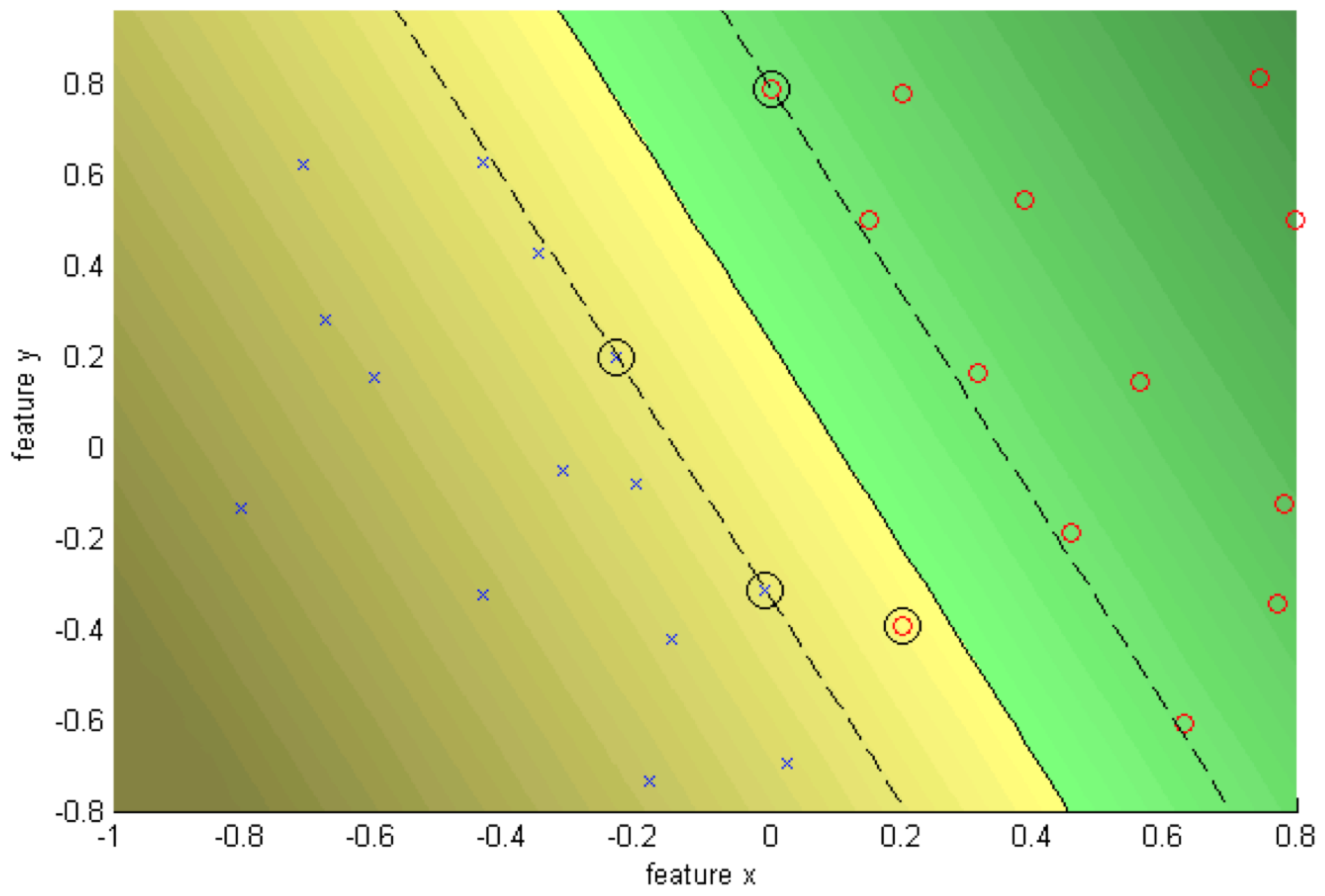
C = Infinity   hard margin



Comment Window

SVM (L1) by Sequential Minimal Optimizer
Kernel: linear (-), C: Inf
Kernel evaluations: 971
Number of Support Vectors: 3
Margin: 0.0966
Training error: 0.00%

C = 10   soft margin



Comment Window

SVM (L1) by Sequential Minimal Optimizer
Kernel: linear (-), C: 10.0000
Kernel evaluations: 2645
Number of Support Vectors: 4
Margin: 0.2265
Training error: 3.70%

References

Basic reading:
- Szeliski, Chapter 14.