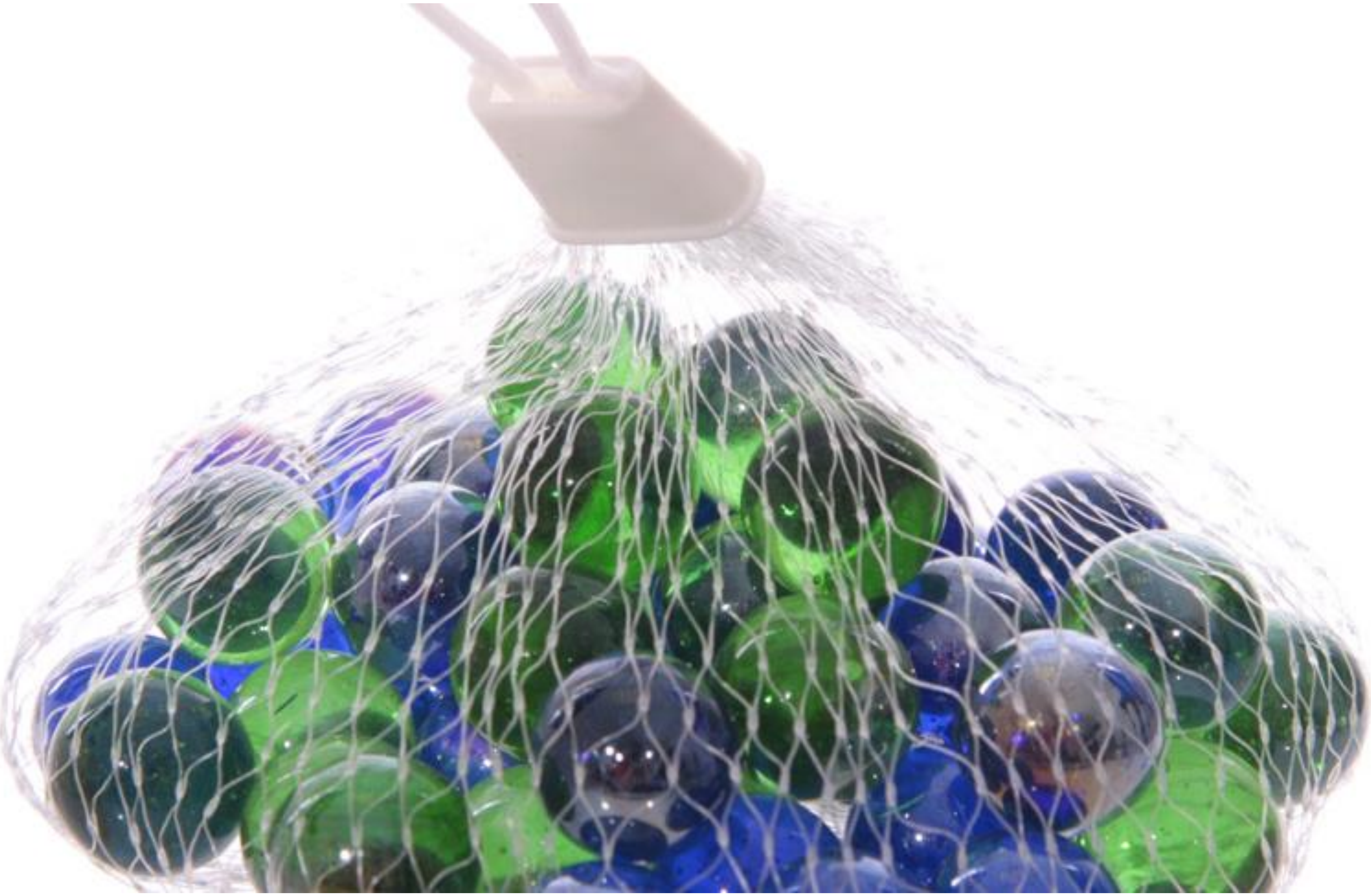
Image classification



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

16-385 Computer Vision
Spring 2018, Lecture 18

Course announcements

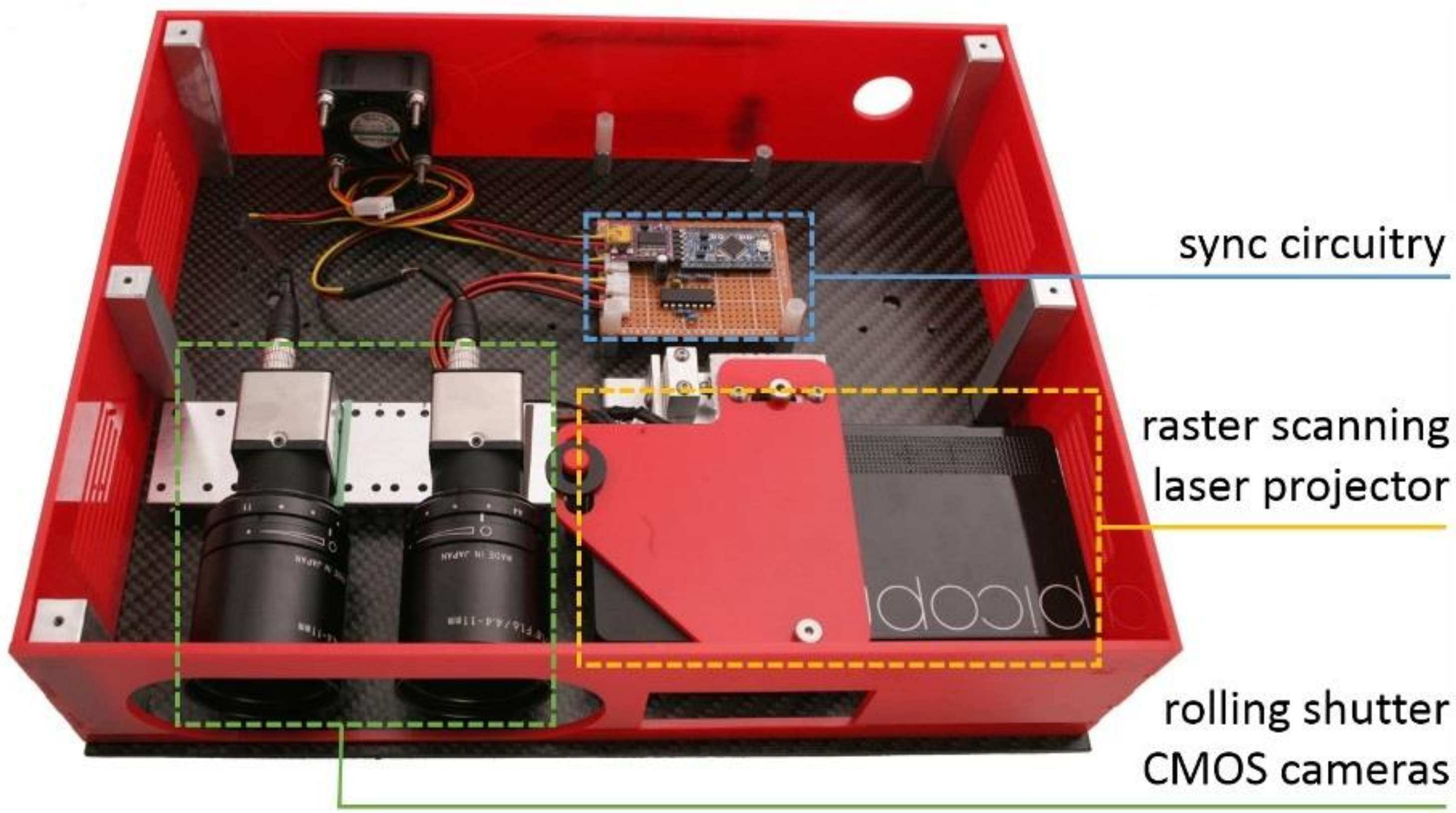
- Homework 5 has been posted and is due on April 6th.
  - Dropbox link because course website is out of space…
  - Any questions about the homework?
  - How many of you have looked at/started/finished homework 5?

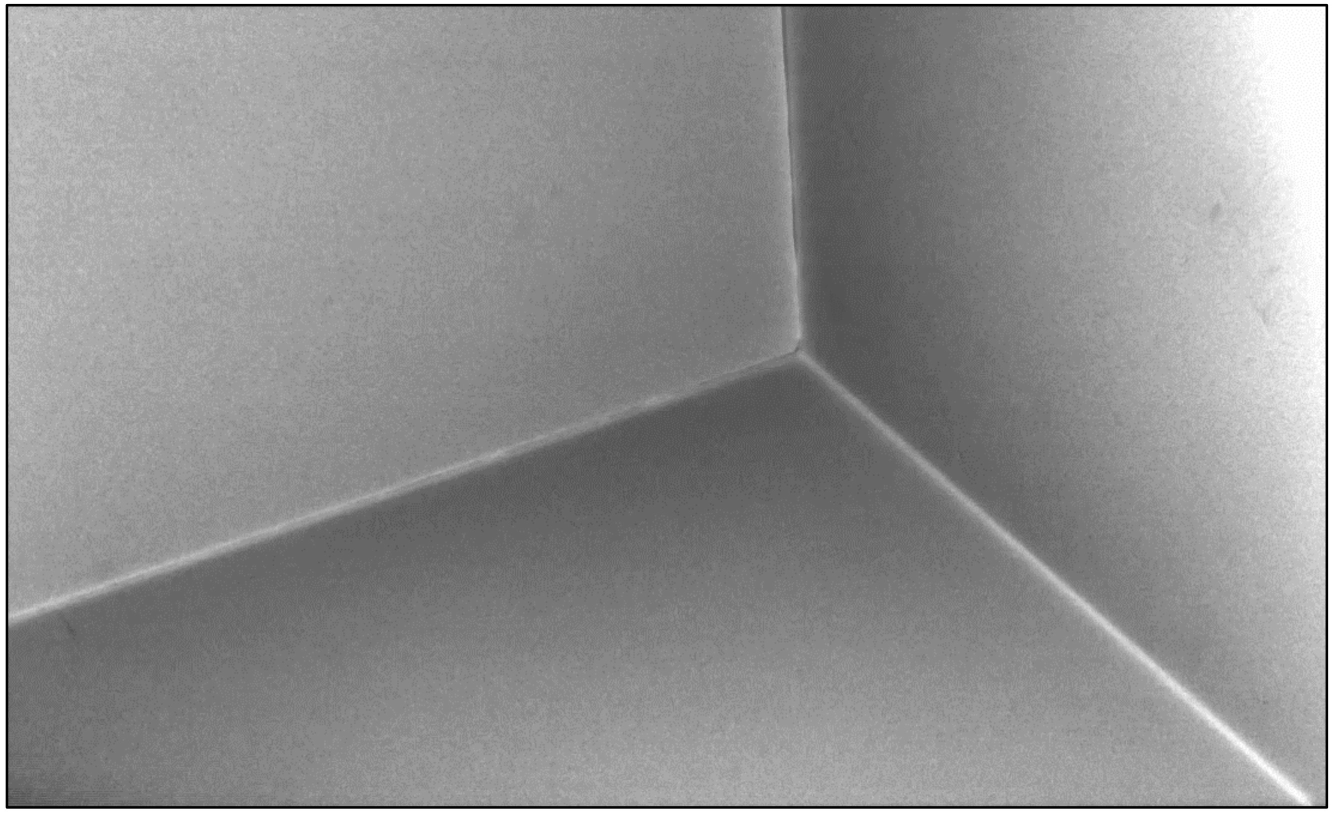
- Yannis' office hours this Friday will be 3 – 4:30pm instead of 3 – 5pm.
  - I'll make it up by scheduling additional office hours next week on Wednesday.

- How many of you went to Saurabh Gupta's talk yesterday?

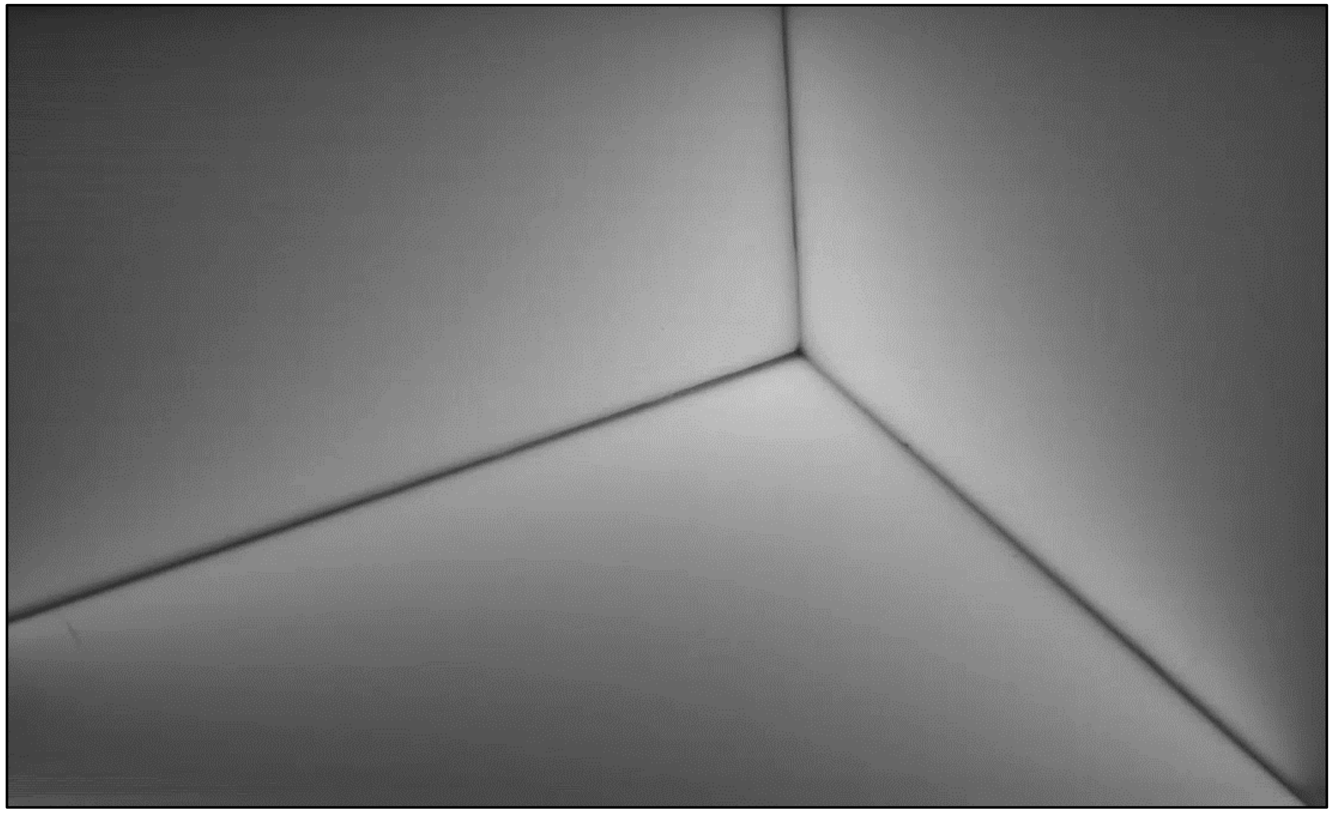
- Talk: Matthew O'Toole, "Probing light transport for 3D shape," Thursday, 1:00 PM, GHC 6115.

Epipolar imaging camera





Video stream of direct reflections



Video stream of indirect reflections

top–left: conventional
top–right: indirect–only
bottom–right: epipolar–only

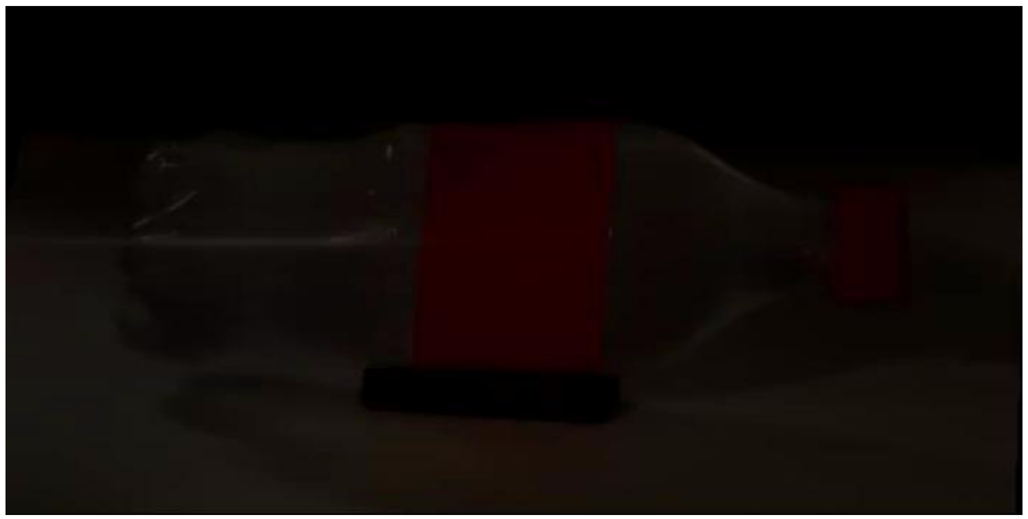
top-left: conventional
top-right: indirect-only
bottom-right: epipolar-only

top–left: conventional
top–right: indirect–only
bottom–right: epipolar–only

top–left: conventional
top–right: indirect–only
bottom–right: epipolar–only

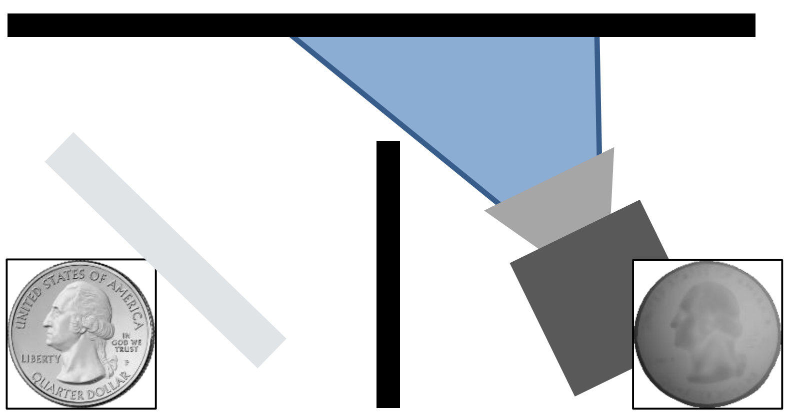
15-463/15-663/15-862 Computational Photography

Learn about this and other unconventional cameras – and build some on your own!



cameras that take video at the speed of light

cameras that measure depth in real time



cameras that see around corners

cameras that capture entire focal stacks

http://graphics.cs.cmu.edu/courses/15-463/

Matt O'Toole will talk about all of these tomorrow

Overview of today's lecture

- Bag-of-words.
- K-means clustering.
- Classification.
- K nearest neighbors.
- Naïve Bayes.
- Support vector machine.

Slide credits

Most of these slides were adapted from:

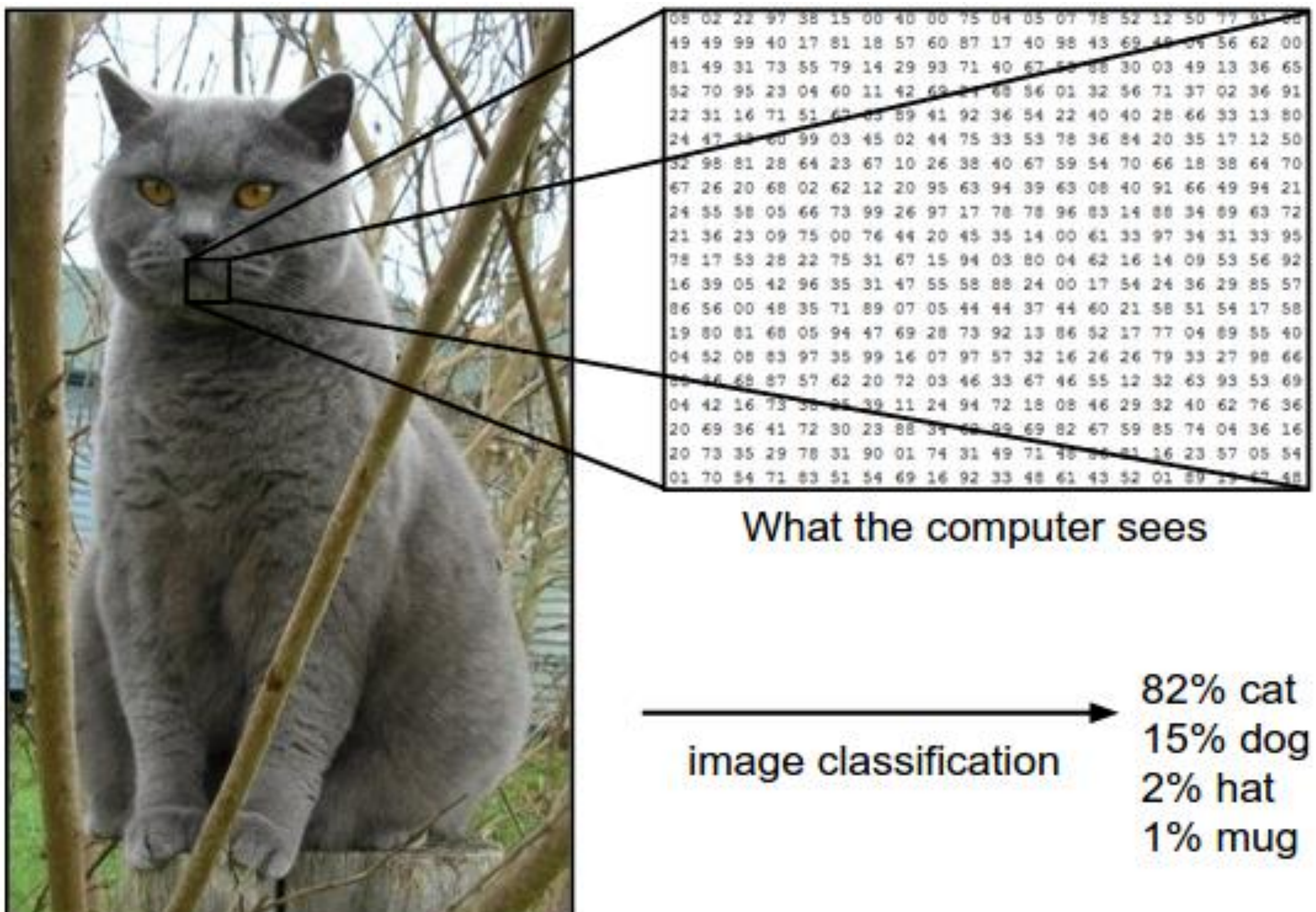
- Kris Kitani (16-385, Spring 2017).
- Noah Snavely (Cornell University).
- Fei-Fei Li (Stanford University).

Image Classification

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

→ cat

Image Classification: Problem

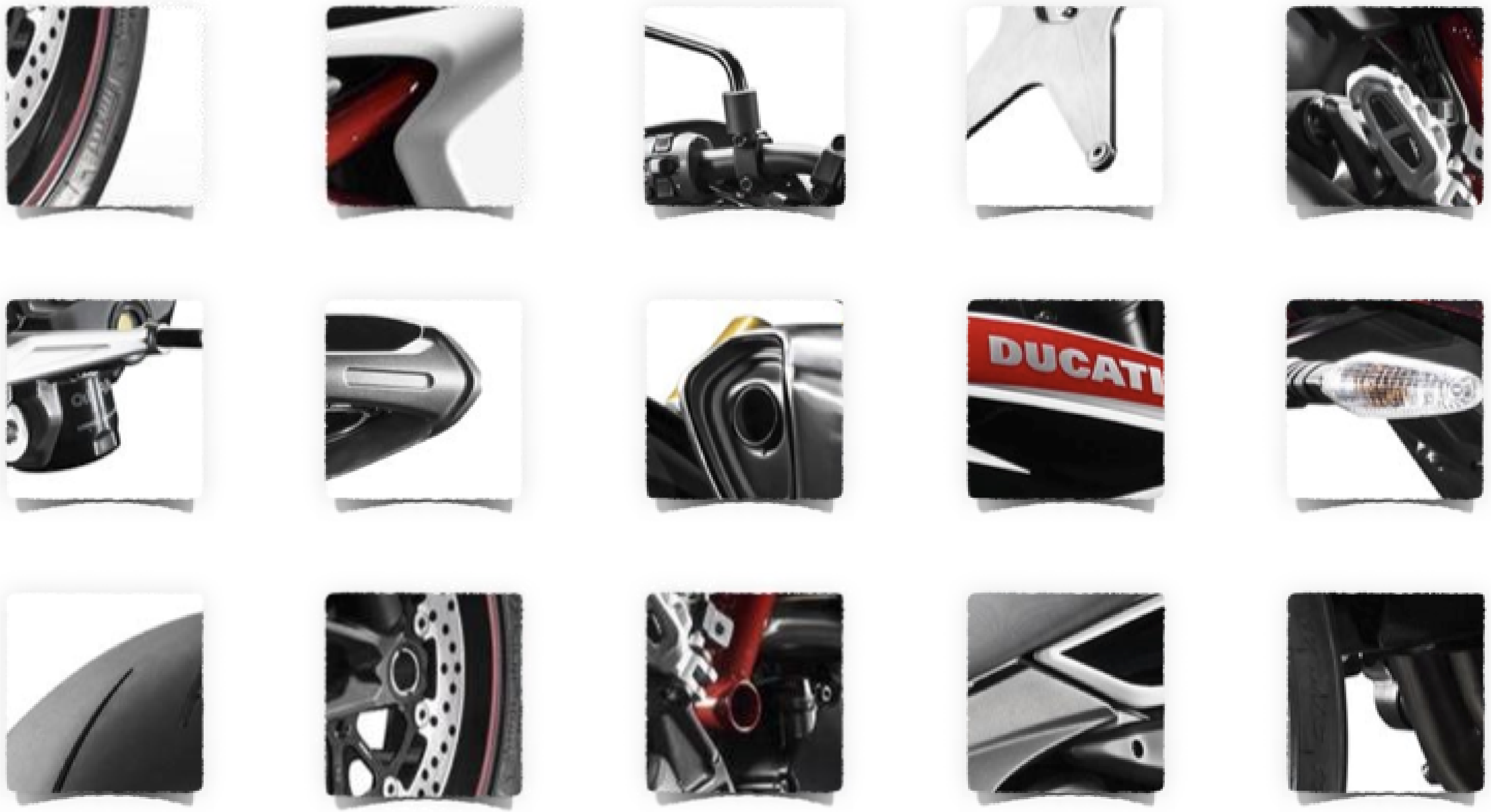


Data-driven approach

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

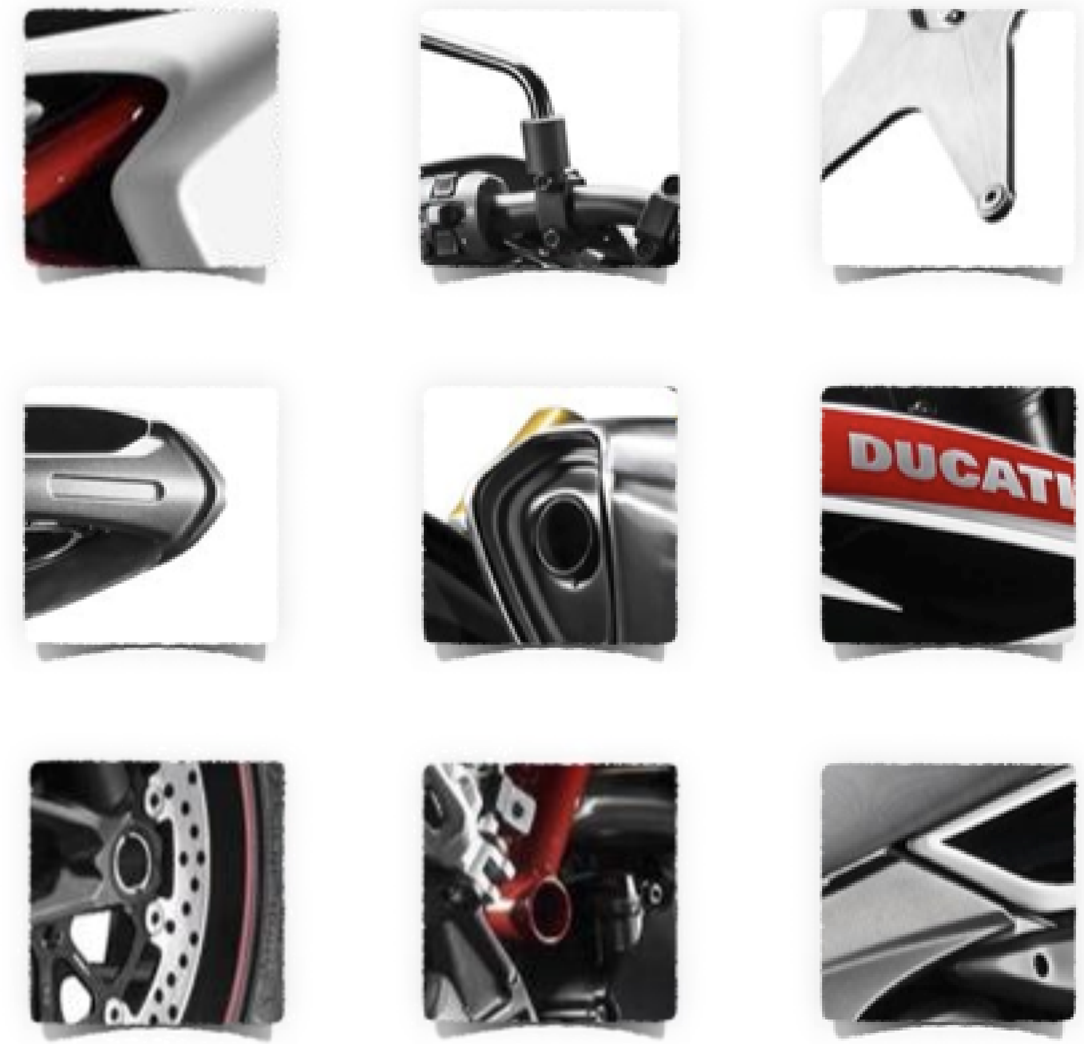
Bag of words

What object do these parts belong to?



Some local feature are very informative

An object as

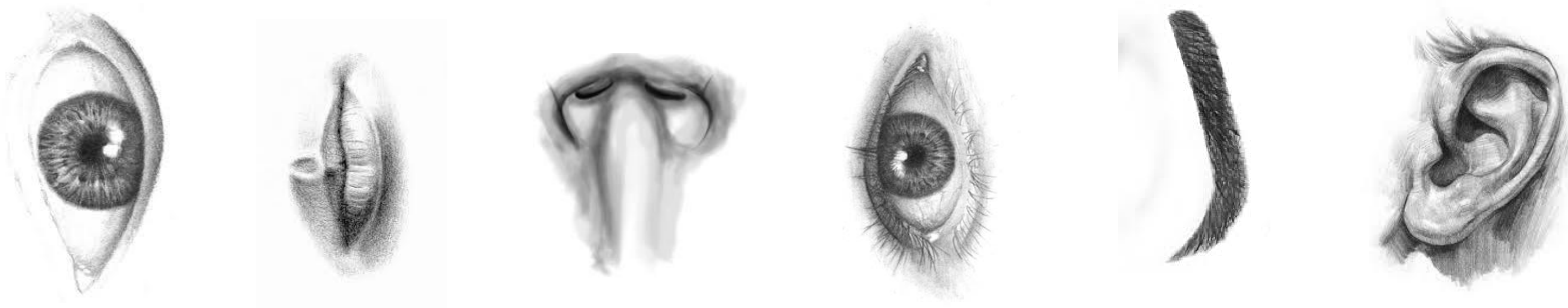


a collection of local features

(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	bag of features	Parts-and-shape model
	Zhang et al. (2005)	Willamowski et al. (2004)	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)

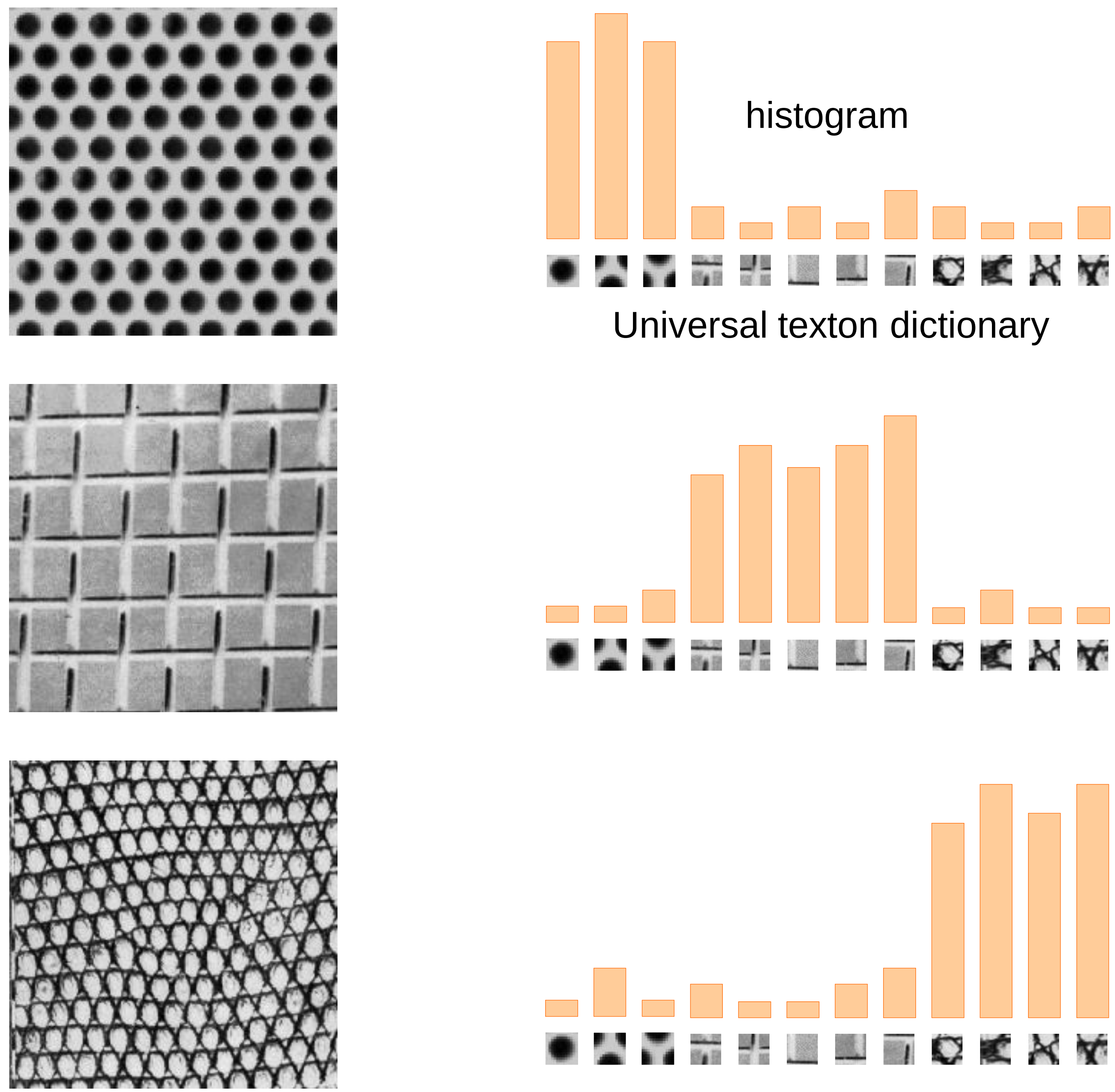
Bag-of-features

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

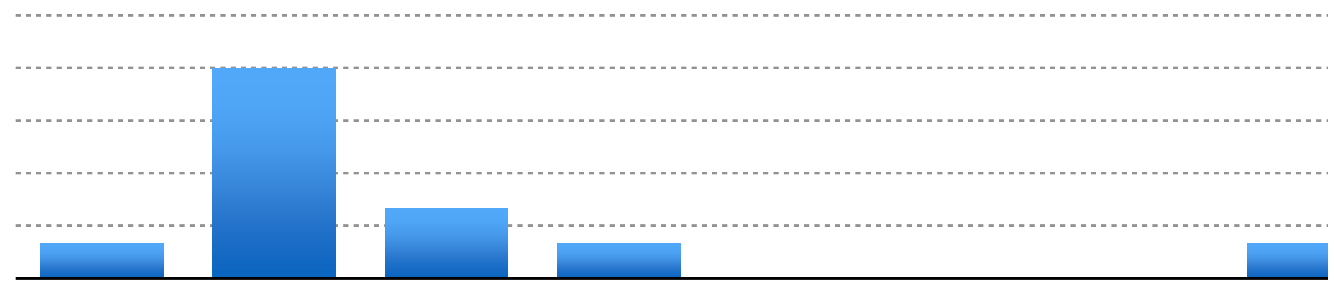
G. Salton. 'Mathematics and Information Retrieval' Journal of Documentation,1979


The Newsp

Sunday, December 22, 2013

DARPA Selects Carnegie Me

The Tartan Rescue Team from Carnegie Mellon University's Robotics National 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.



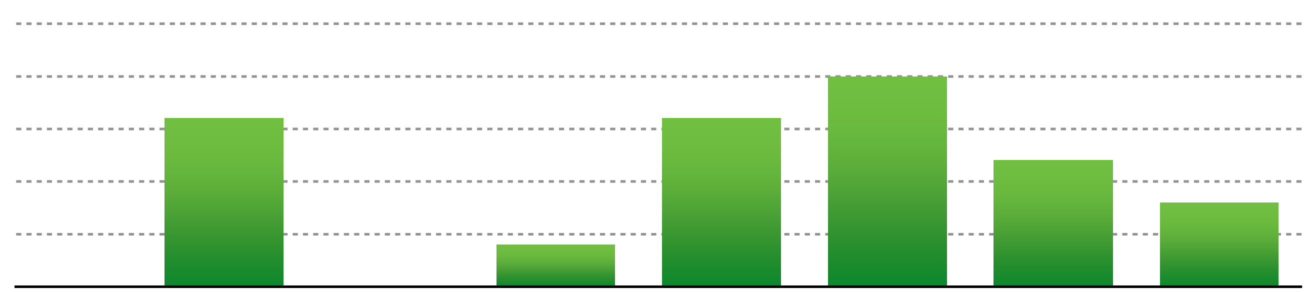
1	6	2	1	0	0	0	1
Tartan	robot	CHIMP	CMU	bio	soft	ankle	sensor


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

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

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?

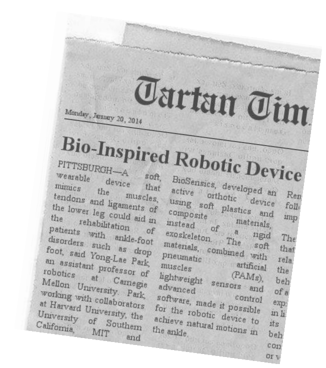
A document (datapoint) is a vector of counts over each word (feature)

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

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

just a histogram over words

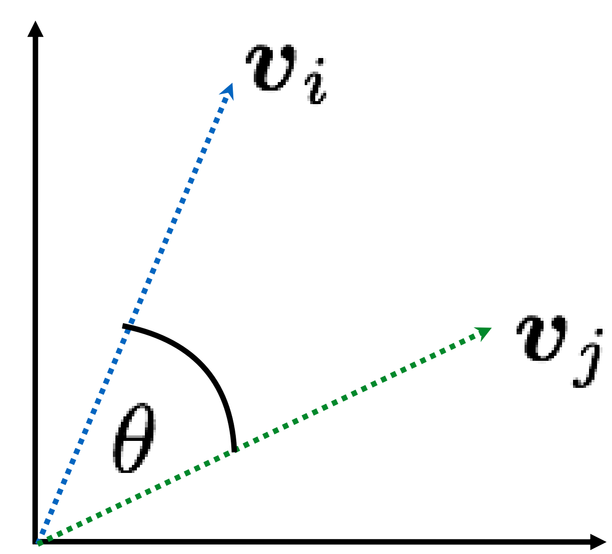
What is the similarity between two documents?



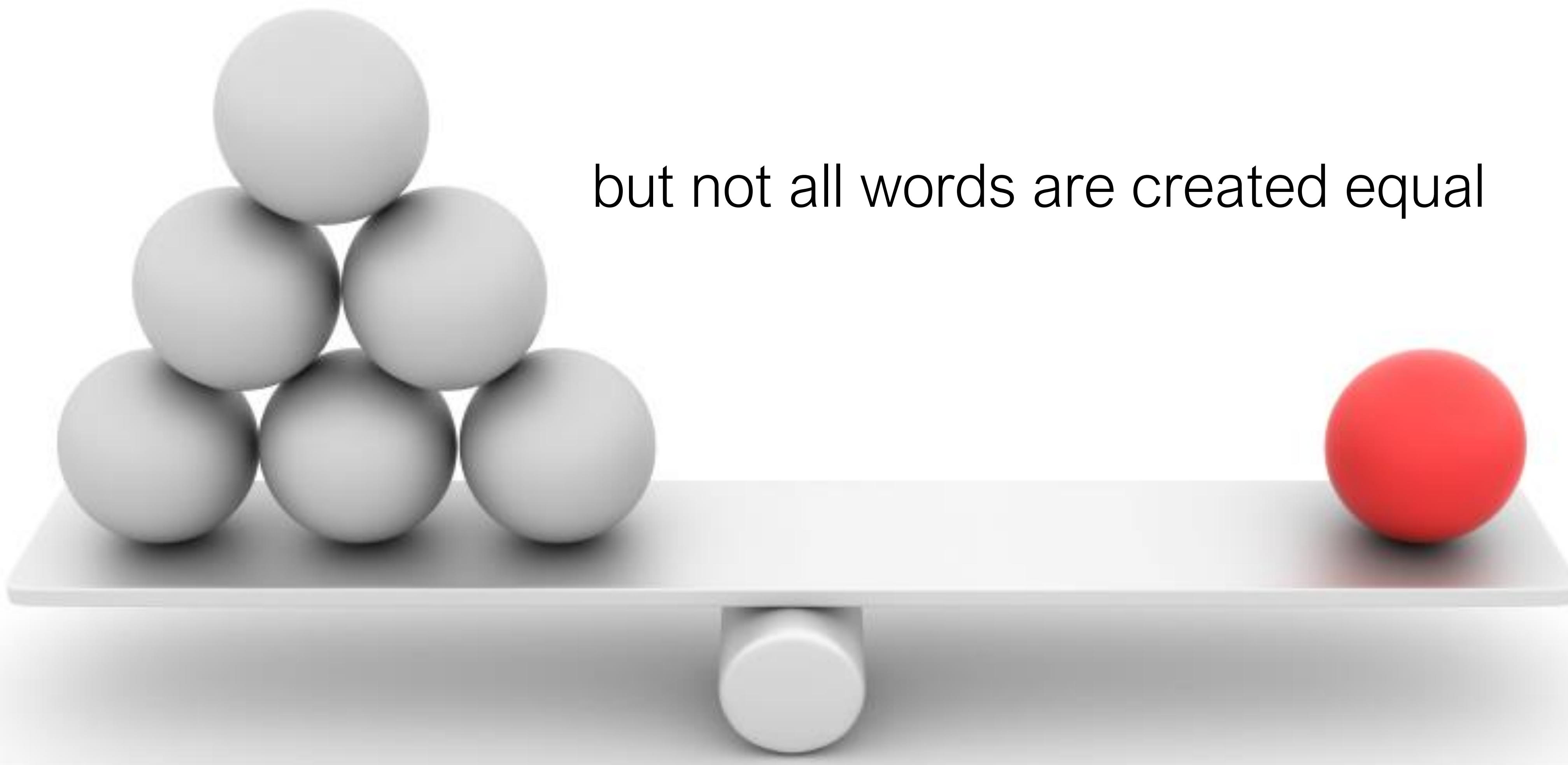
Use any distance you want but the cosine distance is fast.

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

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



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

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

term frequency

inverse document frequency

$$n(w_{i,d})\alpha_i = n(w_{i,d}) \log \left\{ \frac{D}{\sum_{d'} \mathbf{1}[w_i \in d']} \right\}$$

(down-weights **common** terms)

Standard BOW pipeline

(for image classification)

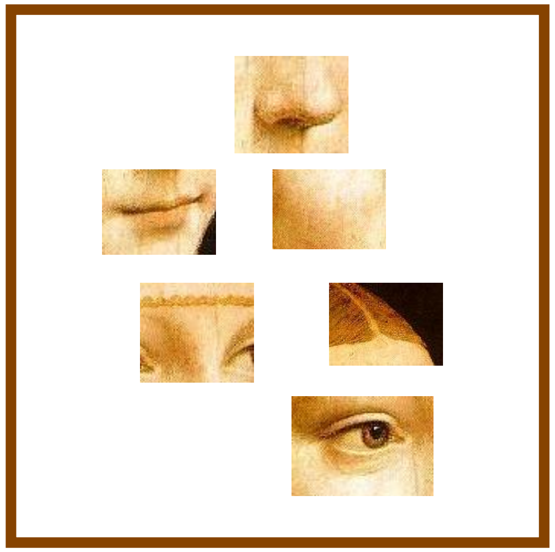
Dictionary Learning:
Learn Visual Words using clustering

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



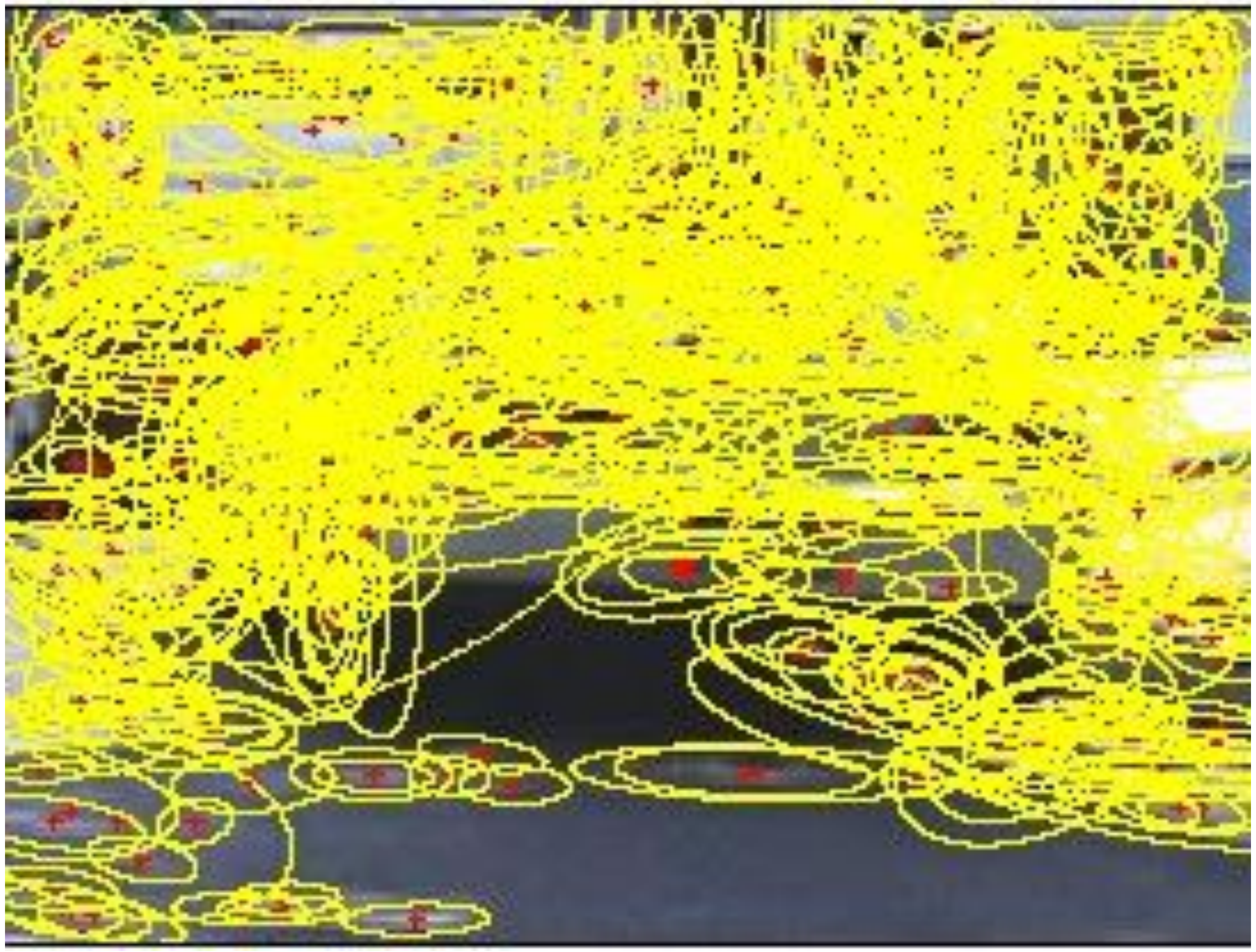
Dictionary Learning:

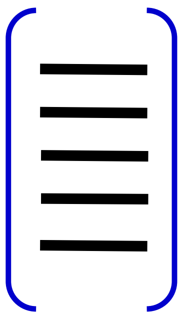
Learn Visual Words using clustering

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

What kinds of features can we extract?

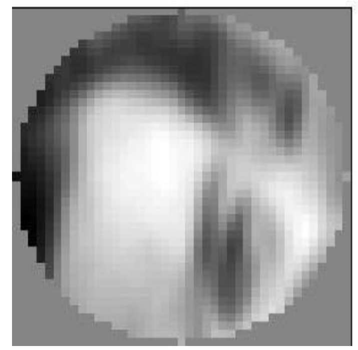
- Regular grid
  - Vogel & Schiele, 2003
  - Fei-Fei & Perona, 2005
- Interest point detector
  - Csurka et al. 2004
  - Fei-Fei & Perona, 2005
  - Sivic et al. 2005
- Other methods
  - Random sampling (Vidal-Naquet & Ullman, 2002)
  - Segmentation-based patches (Barnard et al. 2003)



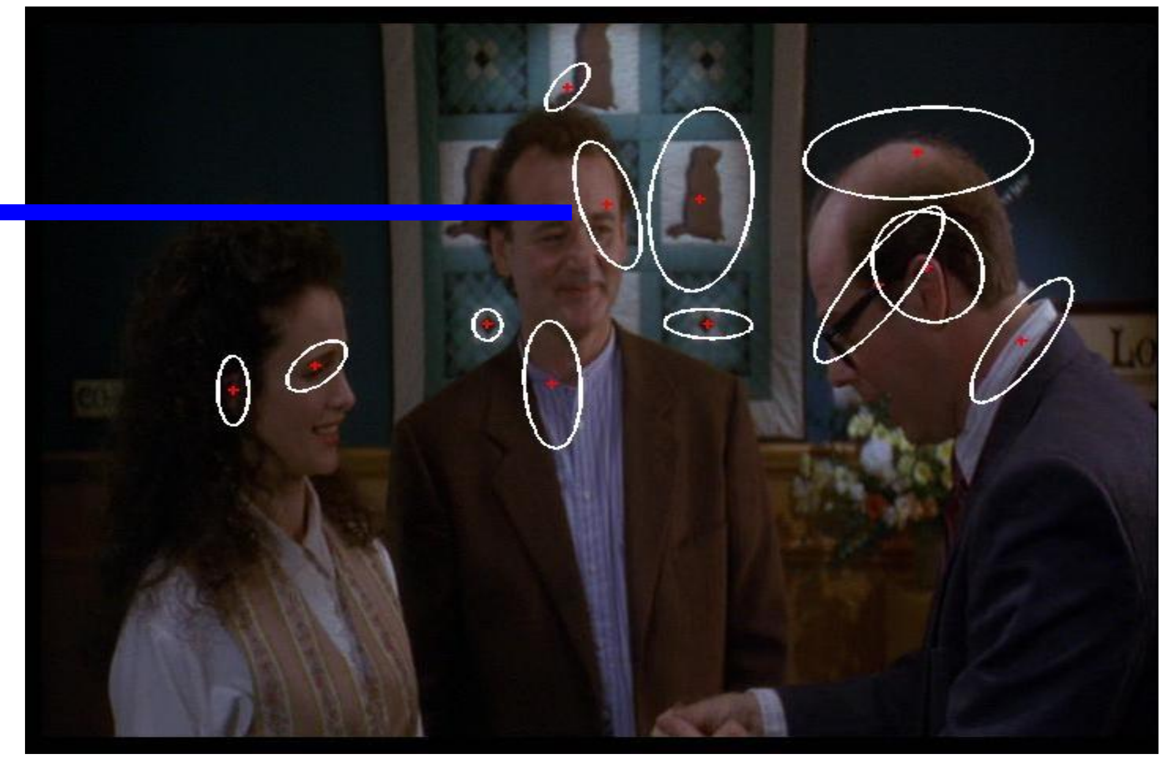


Compute SIFT descriptor

[Lowe'99]



Normalize patch

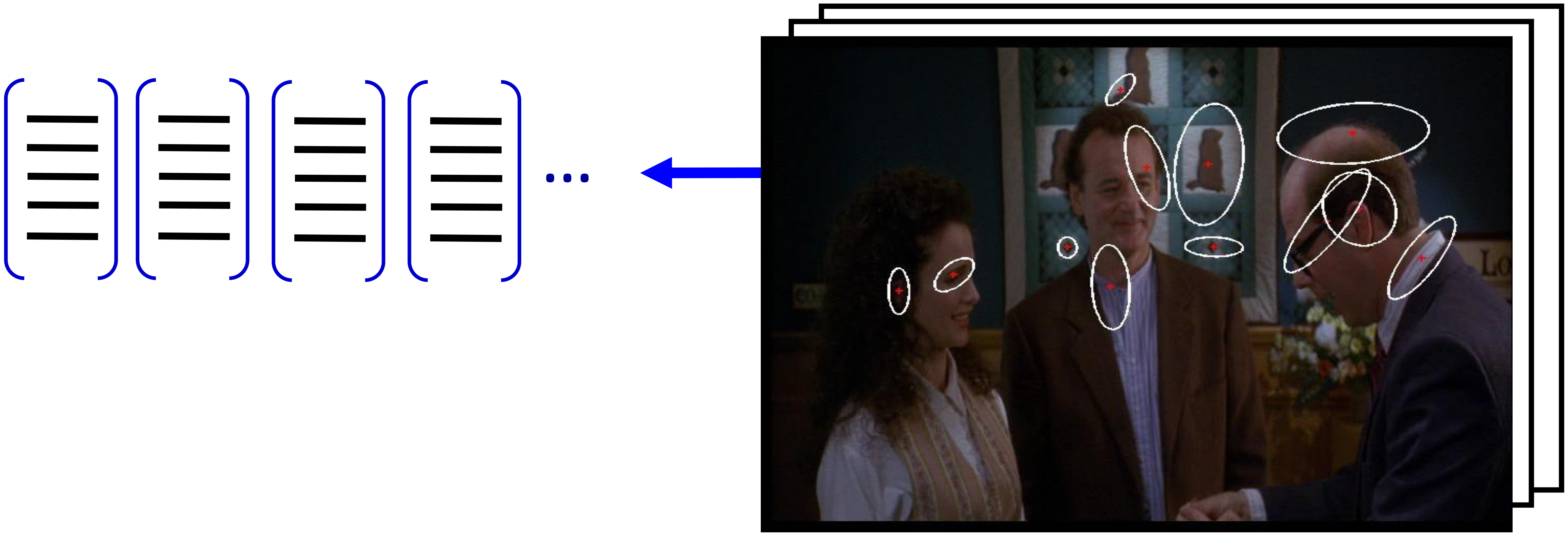


Detect patches

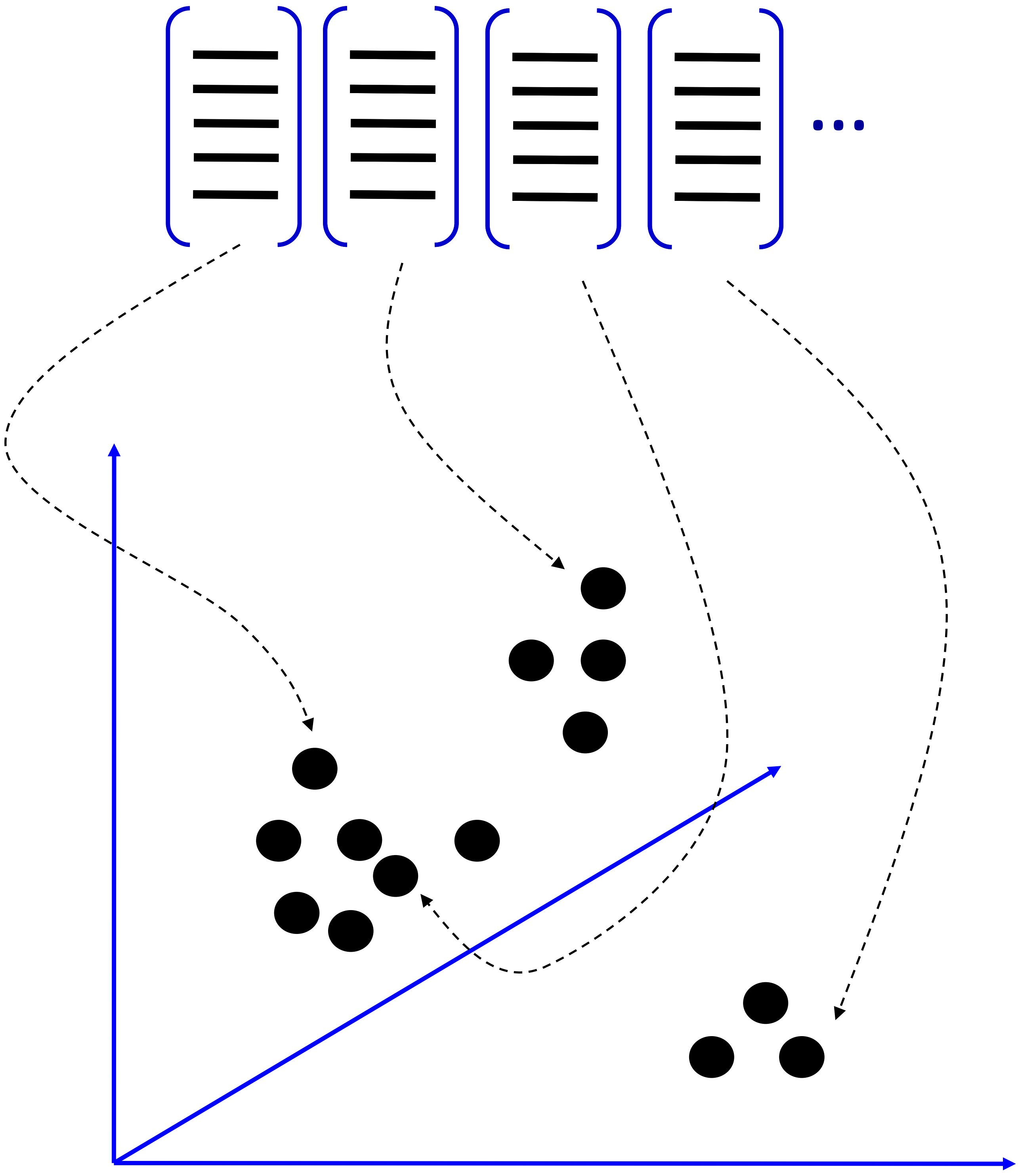
[Mikojaczyk and Schmid '02]

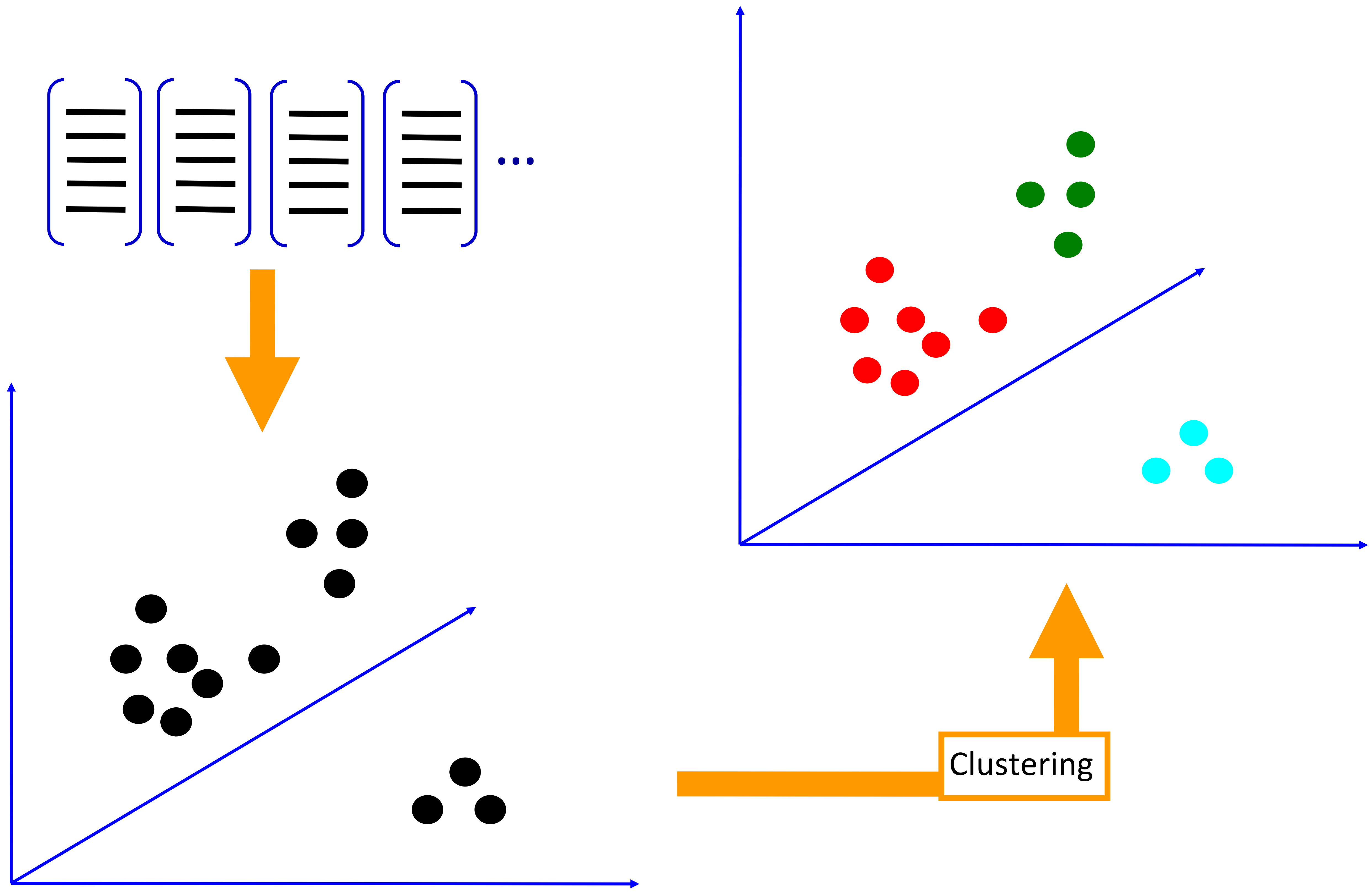
[Mata, Chum, Urban & Pajdla, '02]

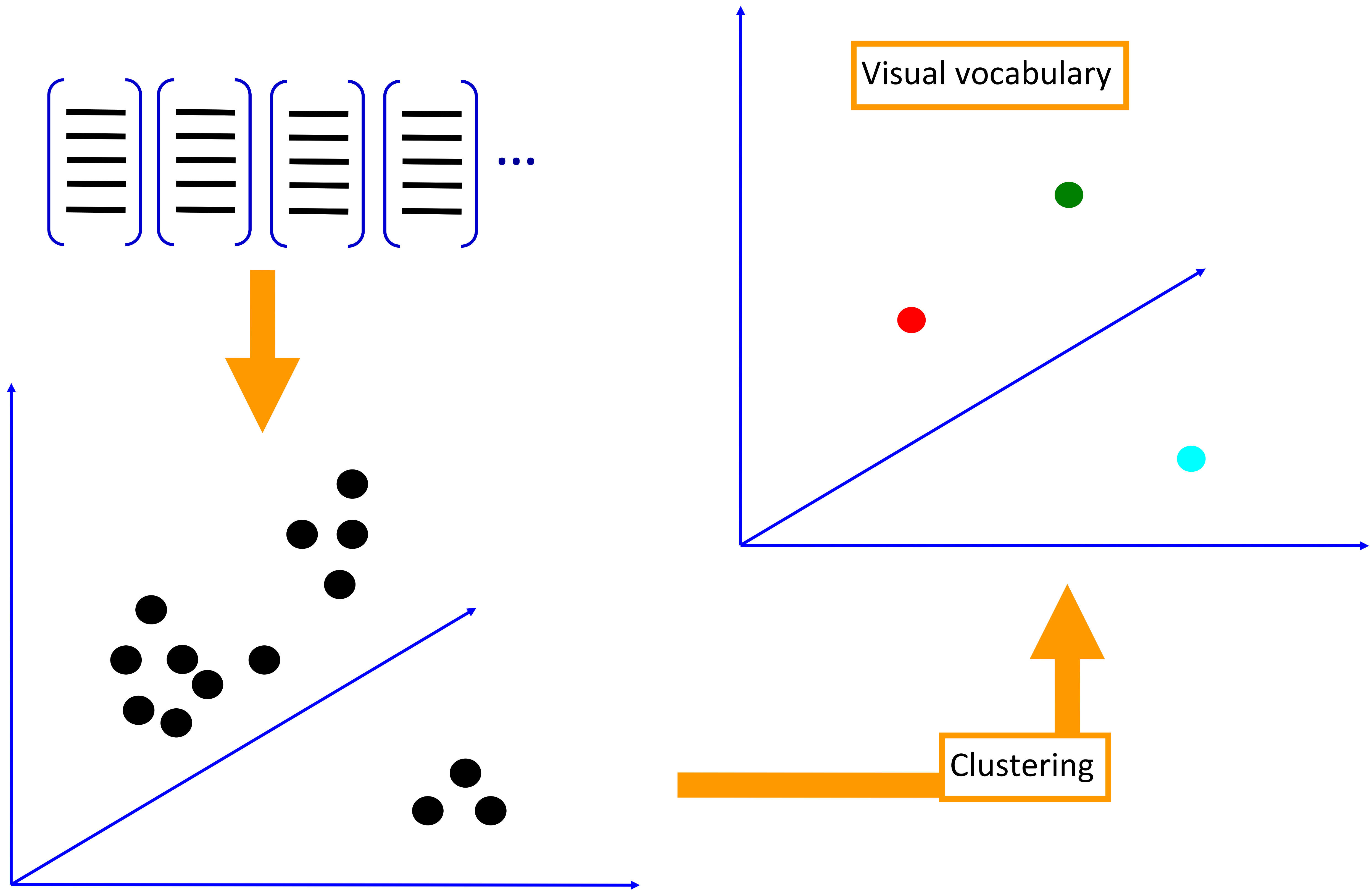
[Sivic & Zisserman, '03]



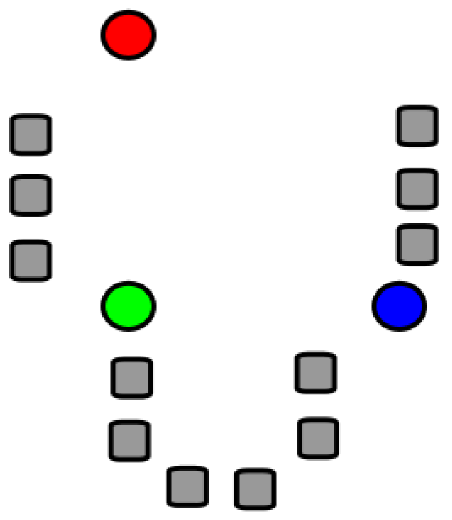
How do we learn the dictionary?



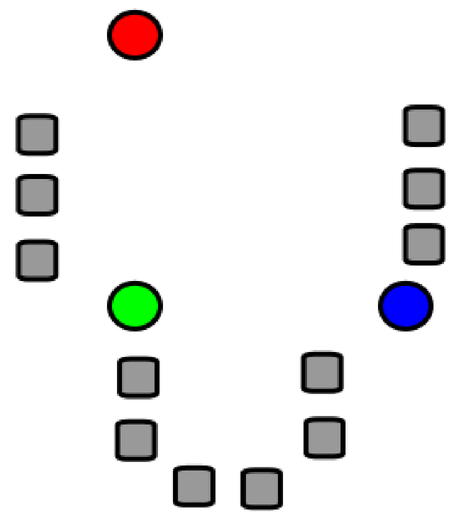

Clustering


Visual vocabulary
Clustering

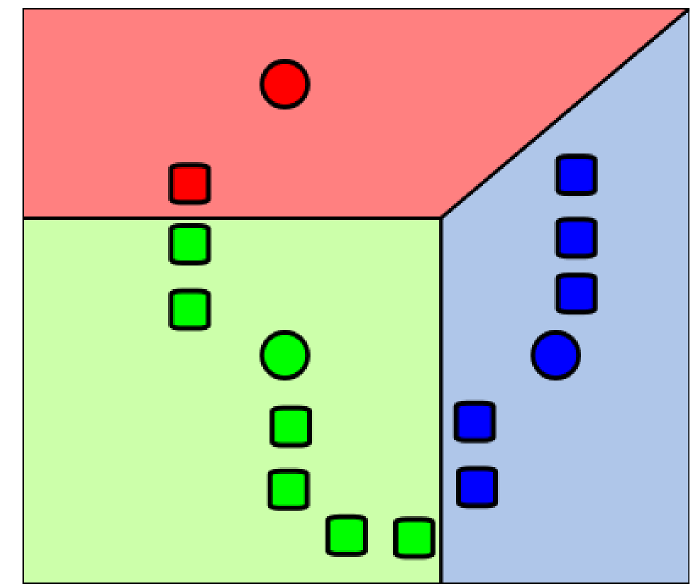
K-means clustering



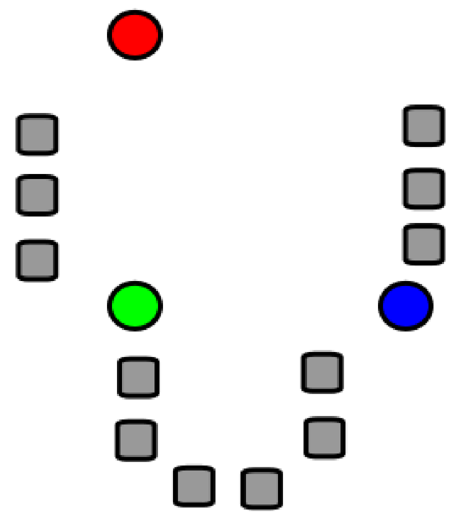
1. Select initial
centroids at random



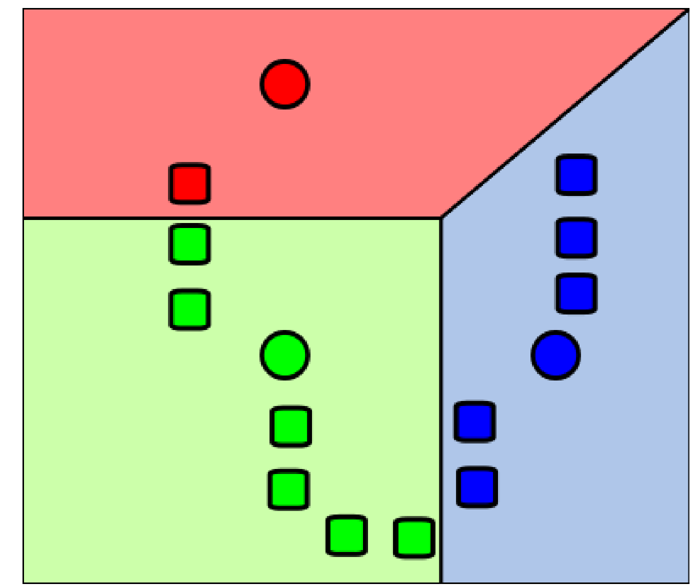
1. Select initial centroids at random



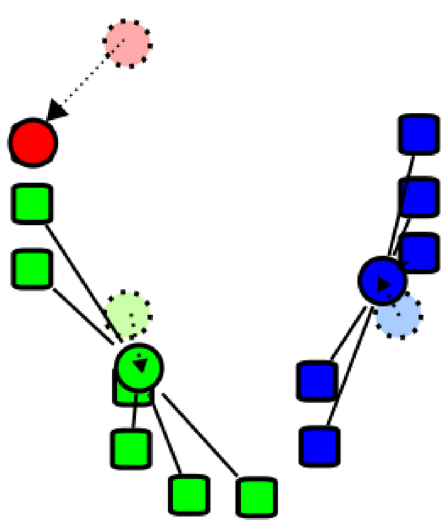
2. Assign each object to the cluster with the nearest centroid.



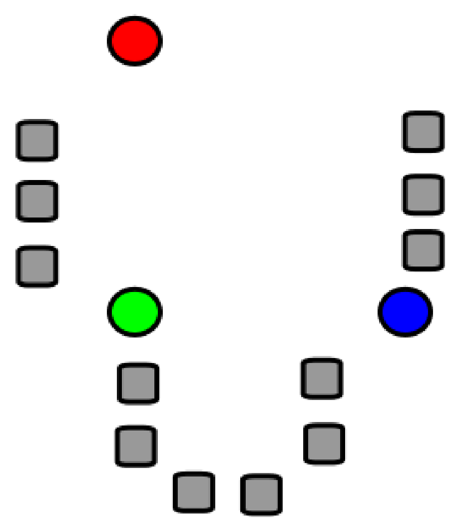
1. Select initial centroids at random



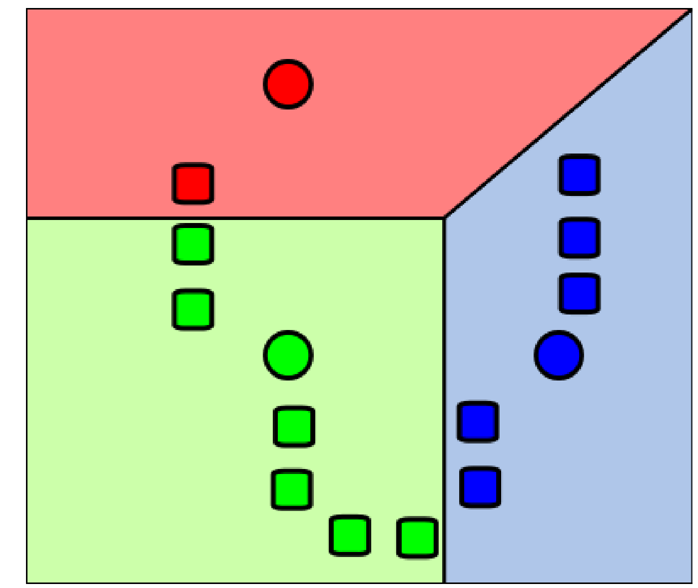
2. Assign each object to the cluster with the nearest centroid.



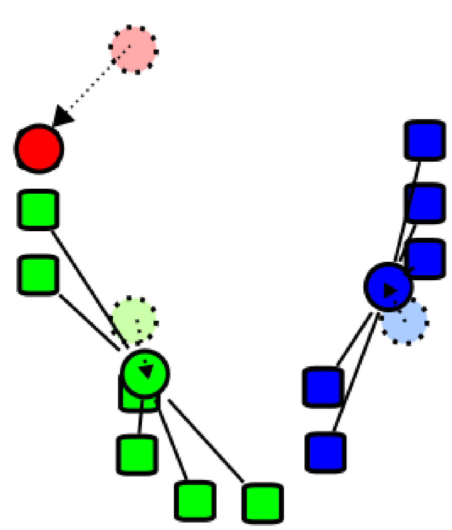
3. Compute each centroid as the mean of the objects assigned to it (go to 2)



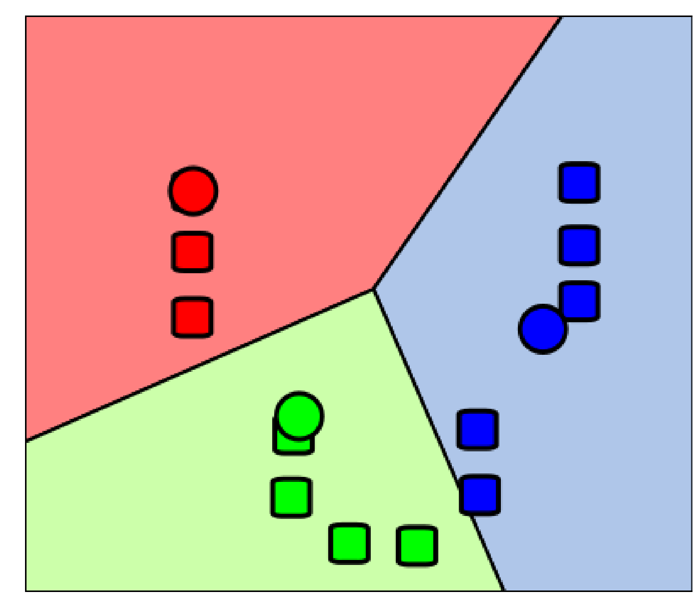
1. Select initial centroids at random



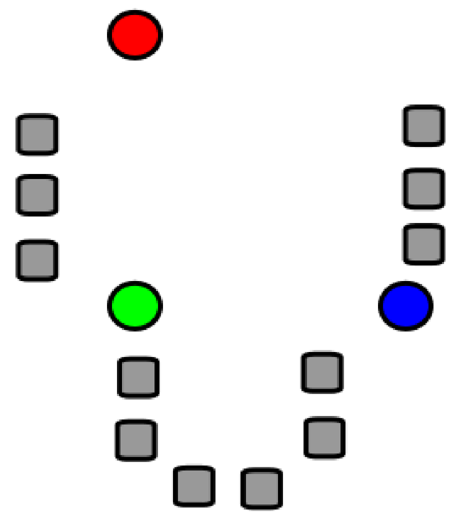
2. Assign each object to the cluster with the nearest centroid.



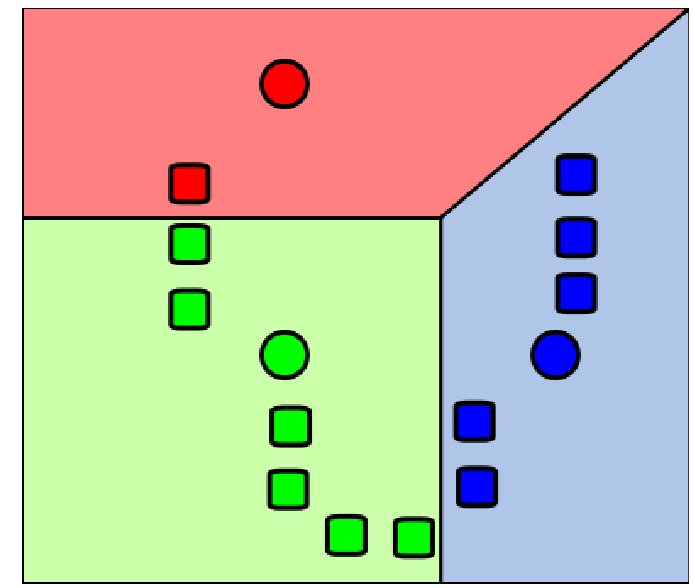
3. Compute each centroid as the mean of the objects assigned to it (go to 2)



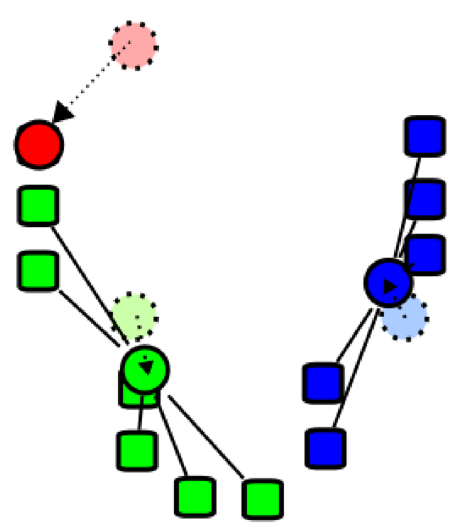
2. Assign each object to the cluster with the nearest centroid.



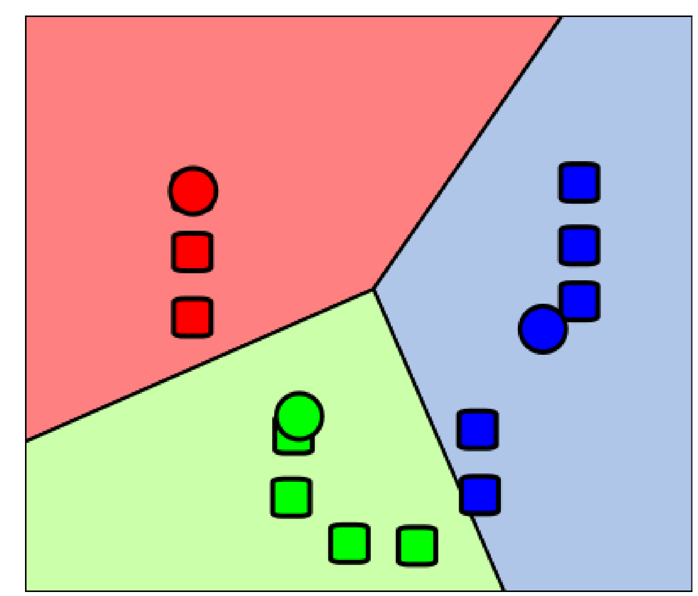
1. Select initial centroids at random



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.

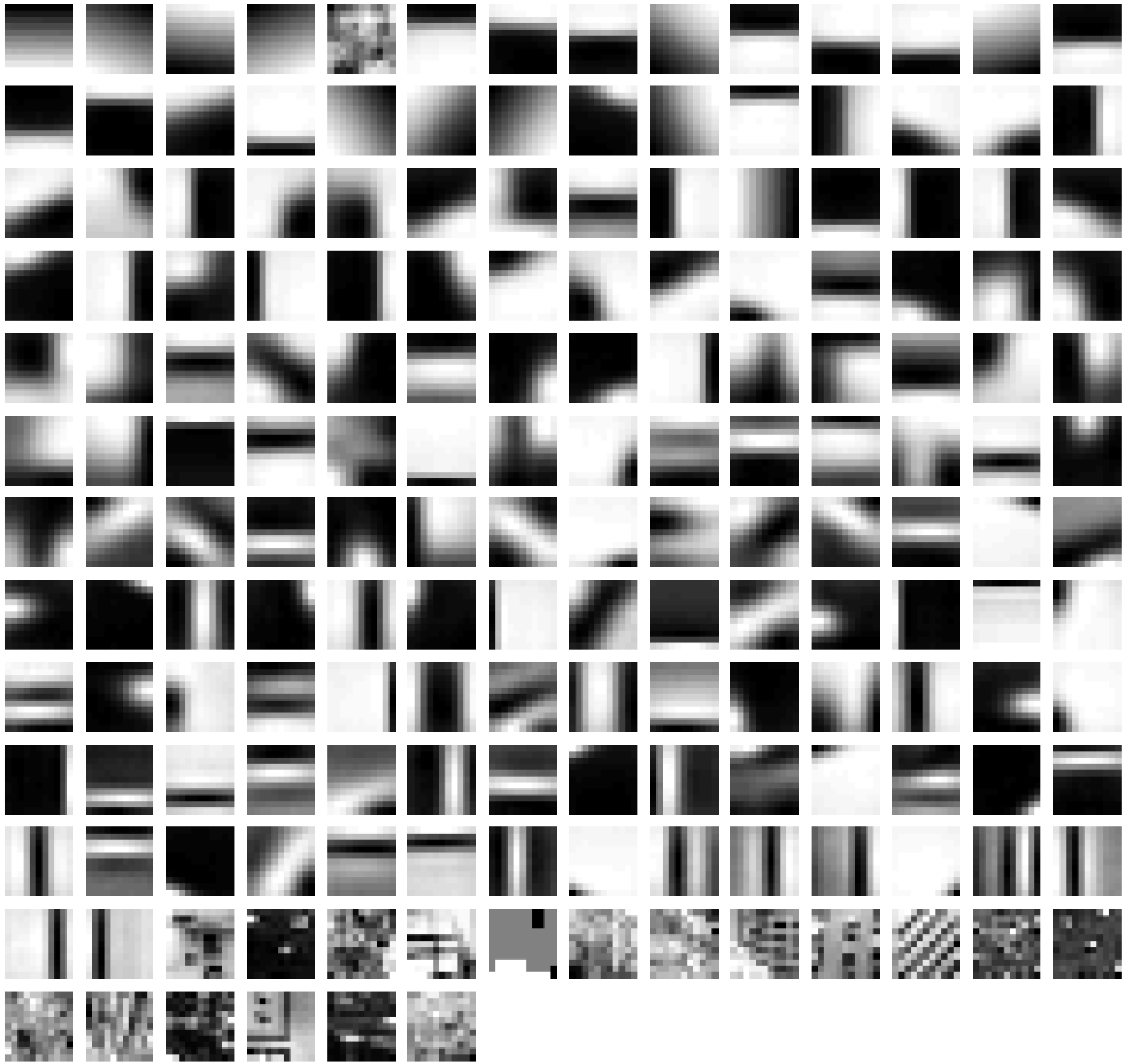
4. Repeat previous 2 steps until no change.

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

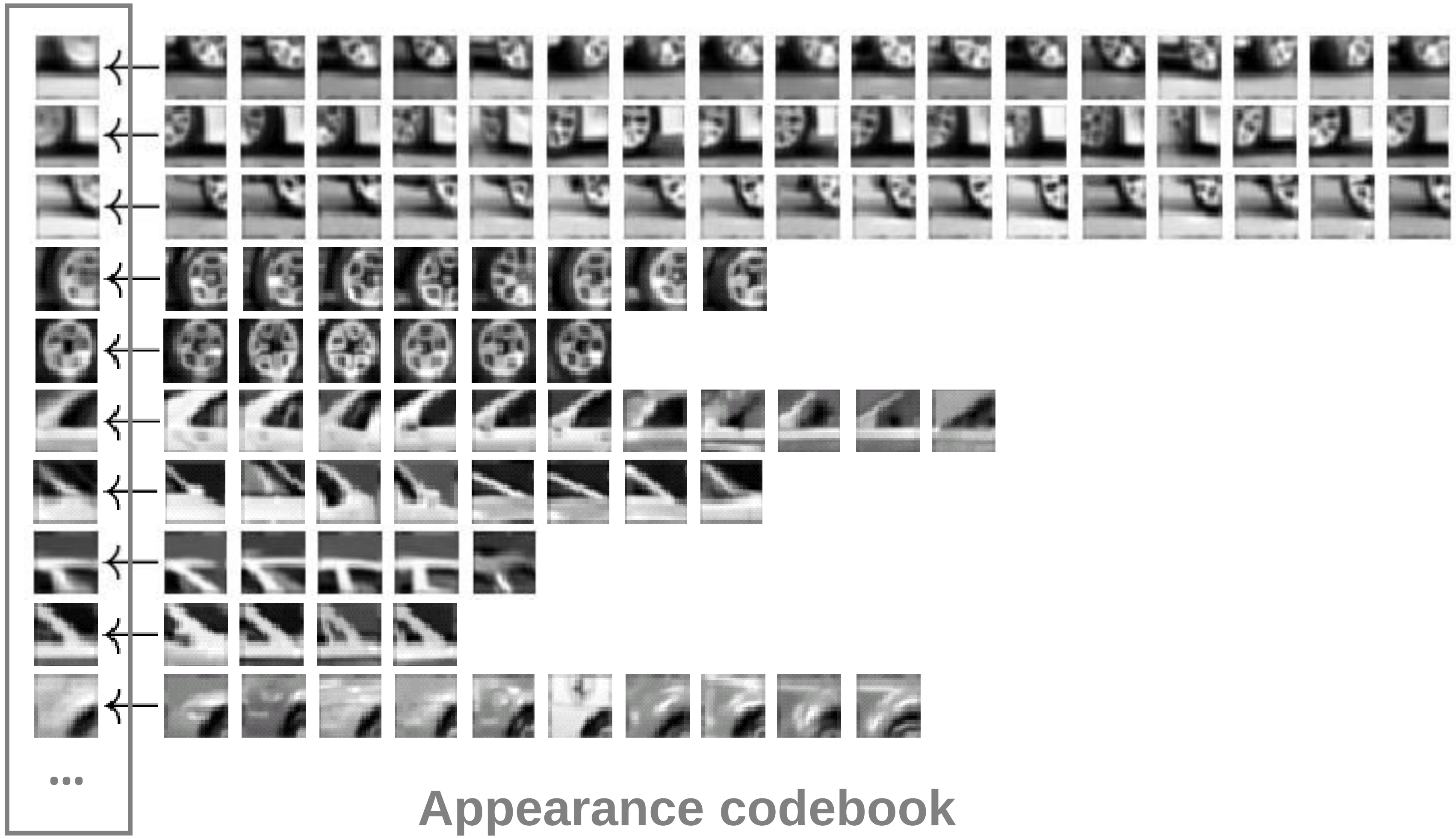
*From what **data** should I learn the dictionary?*

- 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

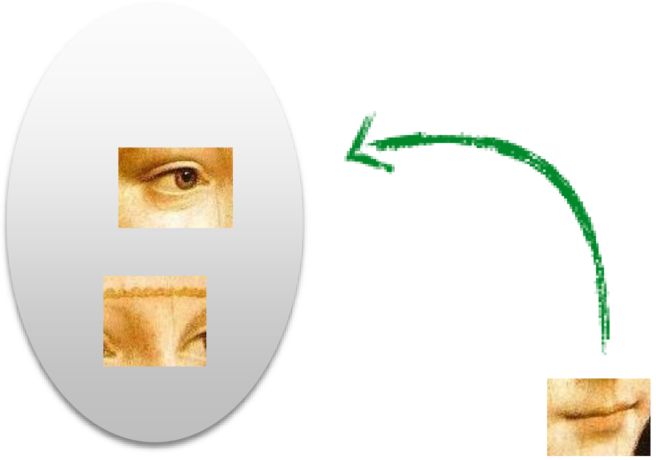
Appearance codebook

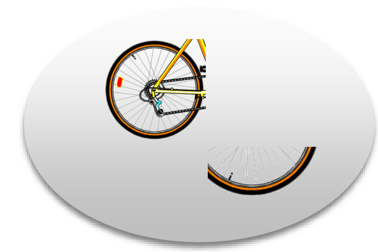
Source: B. Leibe

Dictionary Learning:
Learn Visual Words using clustering

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

Classify:
Train and test data using BOWs

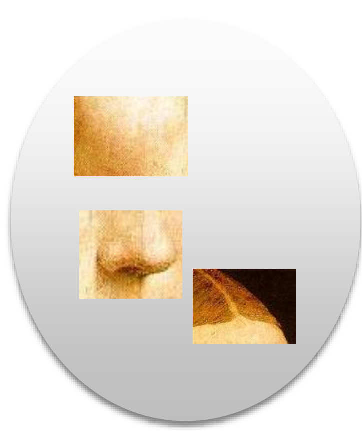




1. Quantization: image features gets associated to a visual word (nearest cluster center)

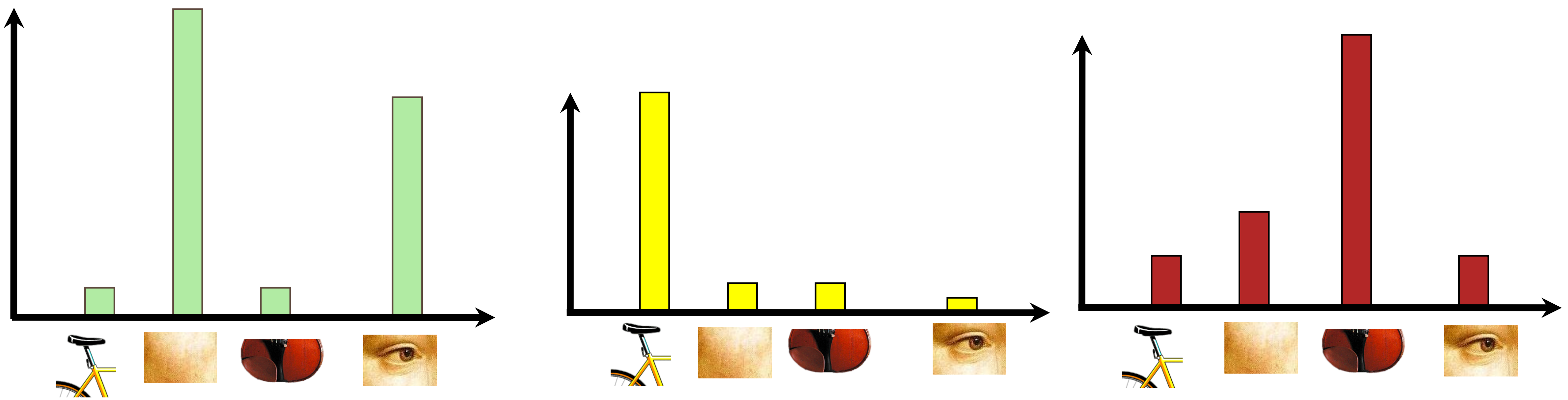
Encode:

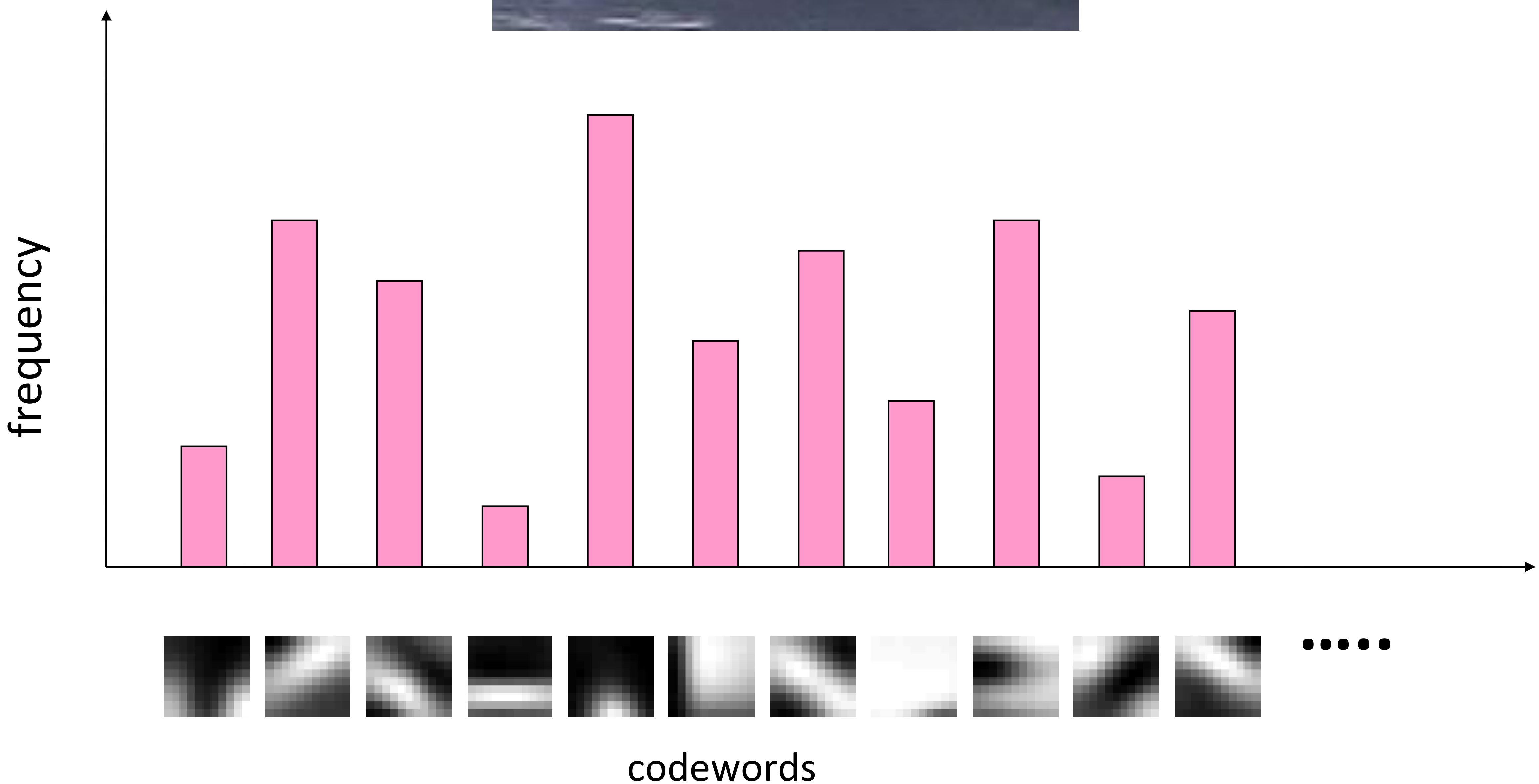
build Bags-of-Words (BOW) vectors for each image



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

2. Histogram: count the number of visual word occurrences

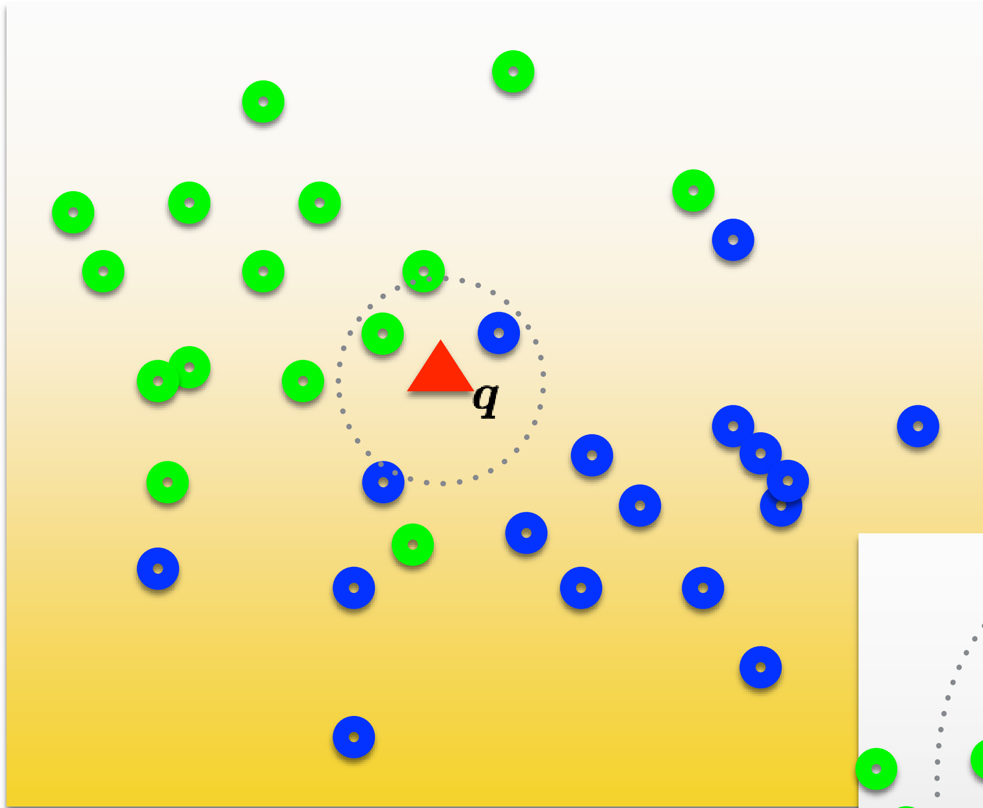



frequency
codewords

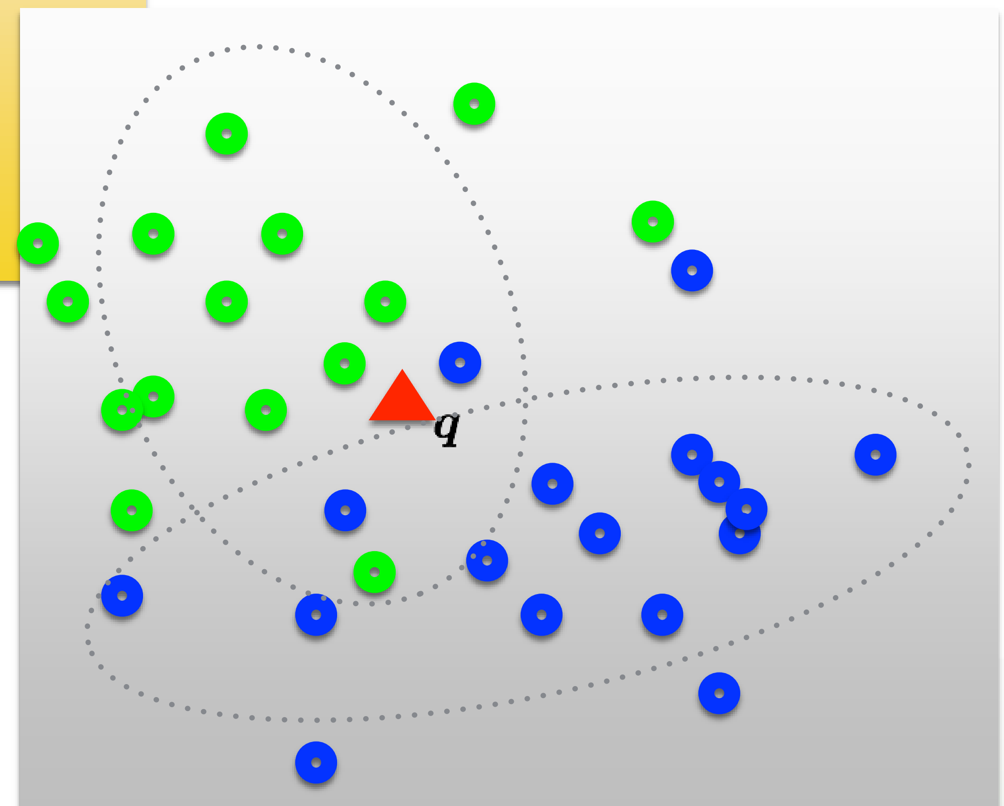
Dictionary Learning:
Learn Visual Words using clustering

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

Classify:
Train and test data using BOWs

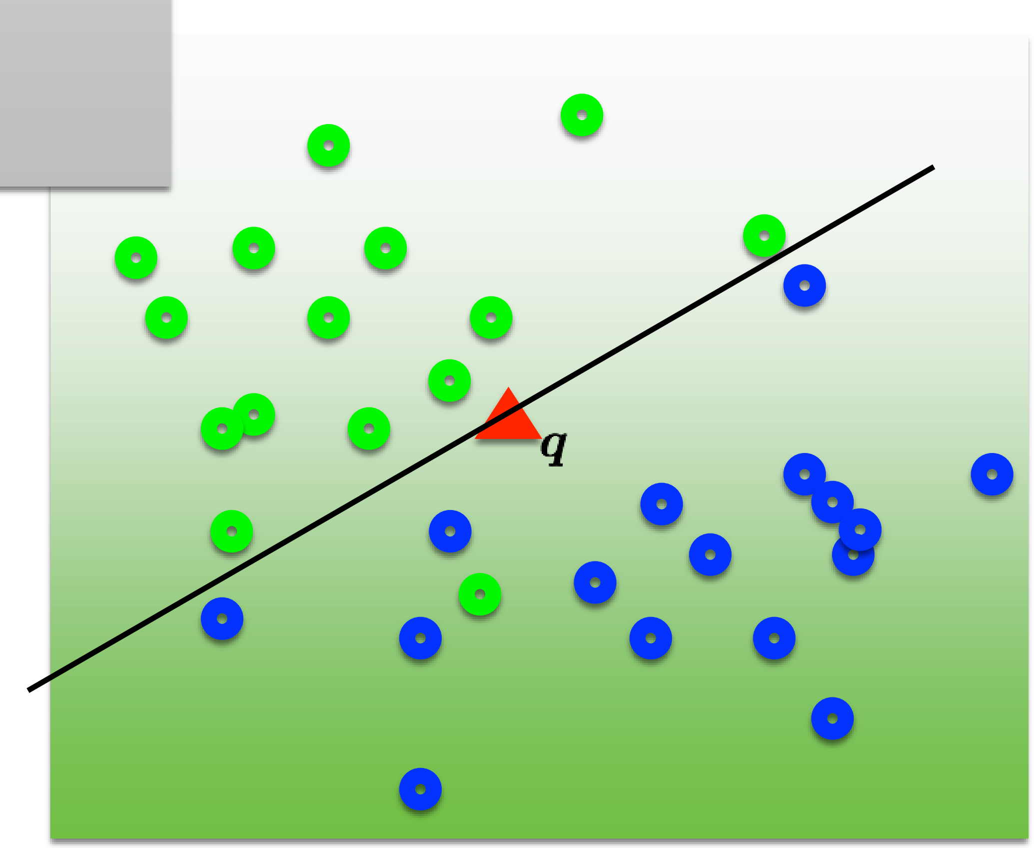


K nearest neighbors



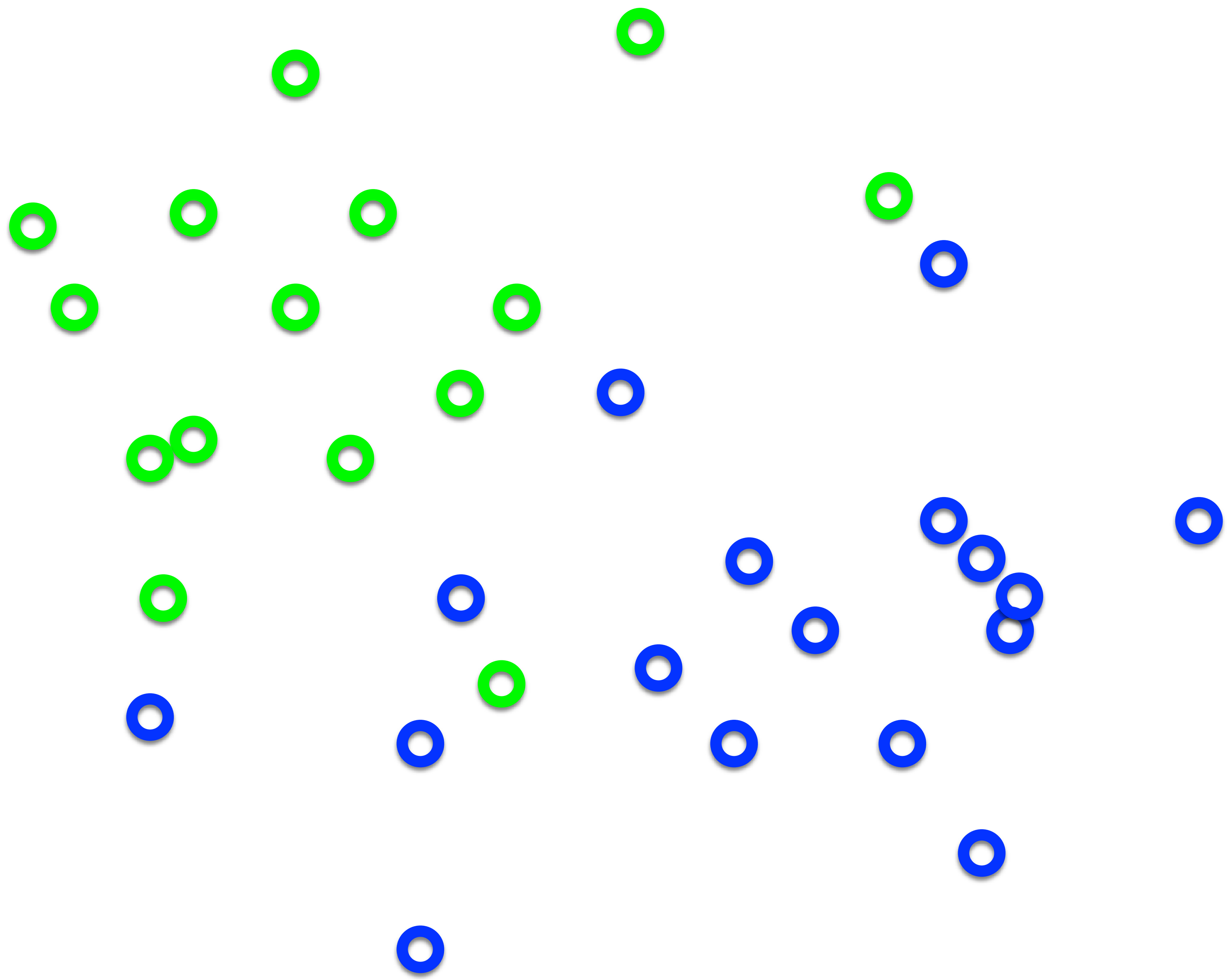
Naïve Bayes

Support Vector Machine

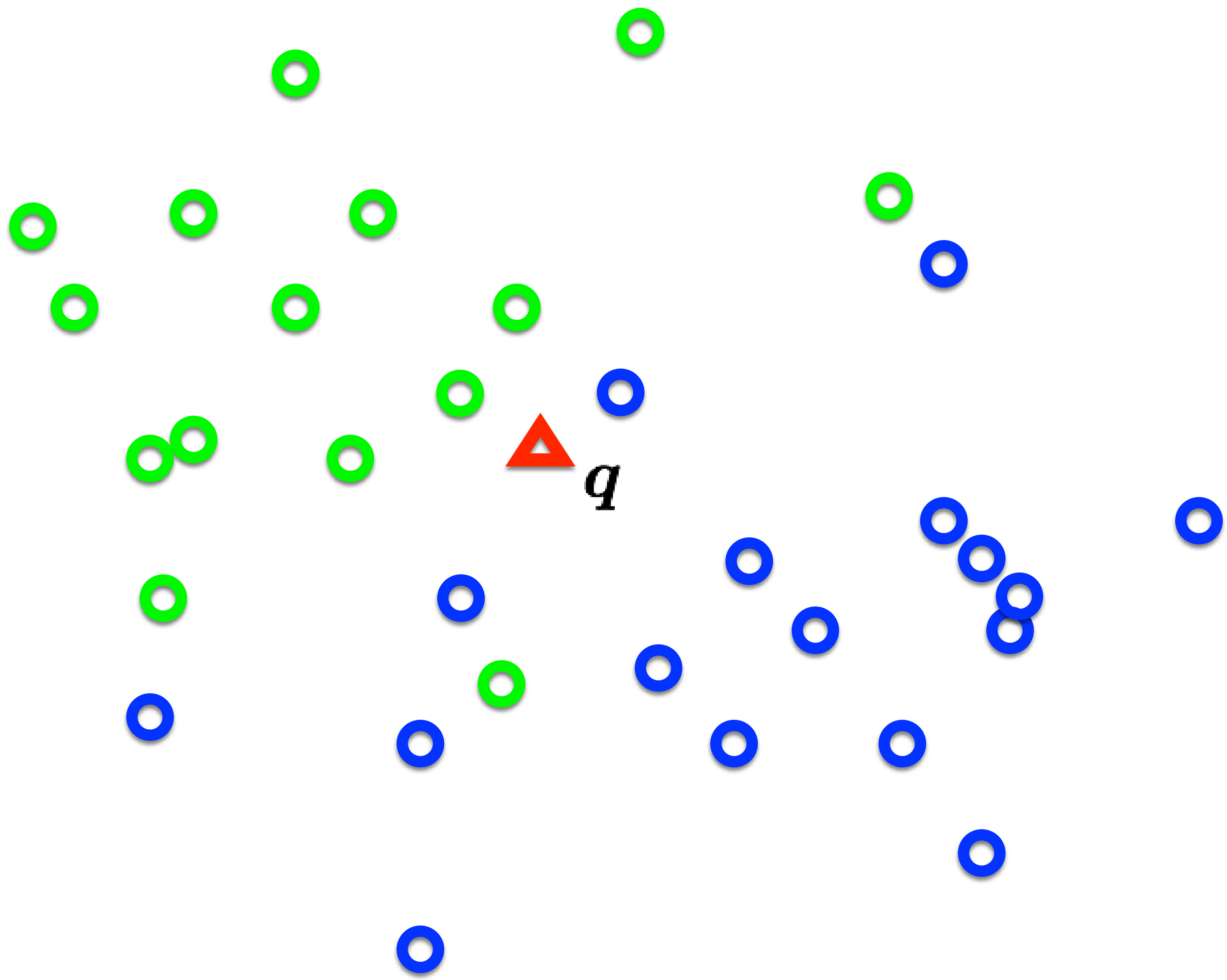


K nearest neighbors

Distribution of data from two classes

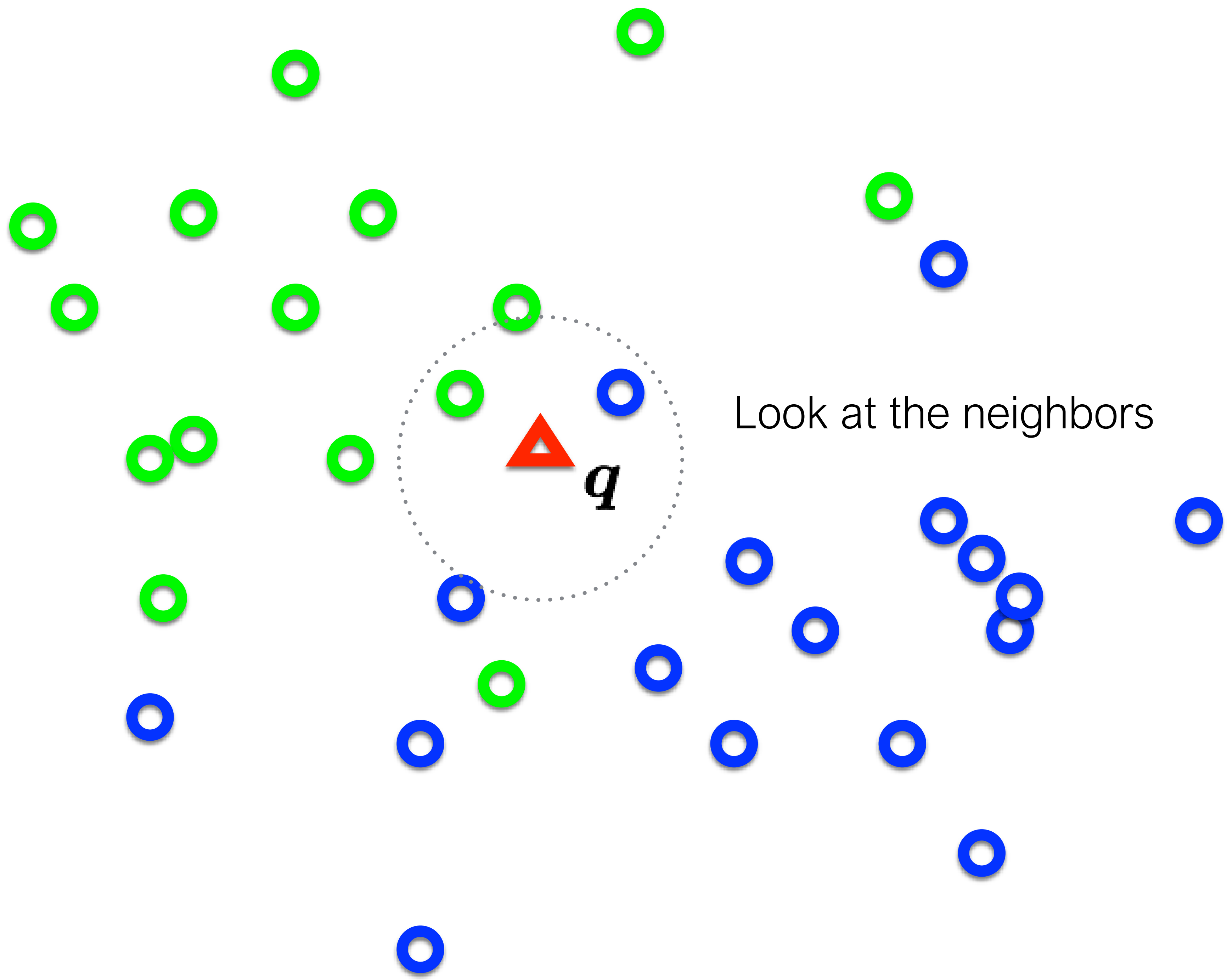


Distribution of data from two classes

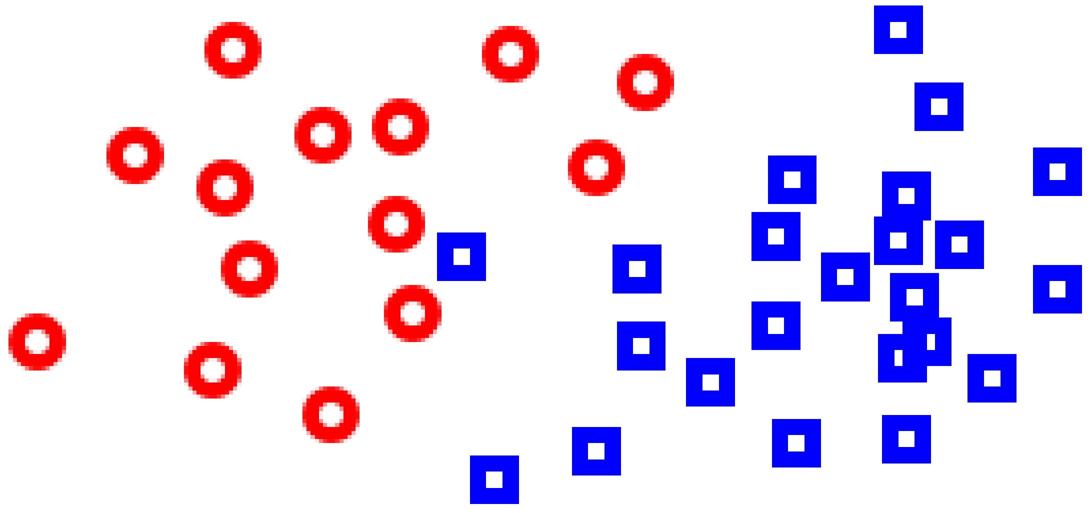


Which class does q belong too?

Distribution of data from two classes



K-Nearest Neighbor (KNN) Classifier

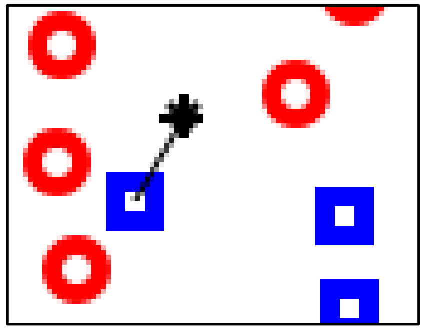


Non-parametric pattern classification approach

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

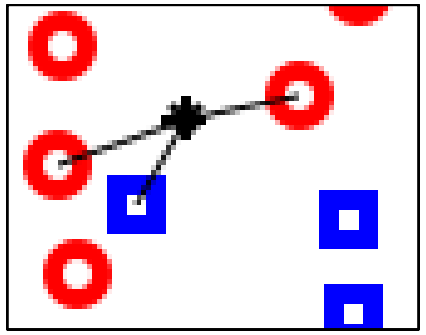
For a given query point q, assign the class of the nearest neighbor

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
SVM deg 4 polynomial	1.1
2-layer NN, 300 hidden units	4.7
2-layer NN, 300 HU, [deskewing]	1.6
LeNet-5, [distortions]	0.8
Boosted LeNet-4, [distortions]	0.7

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

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

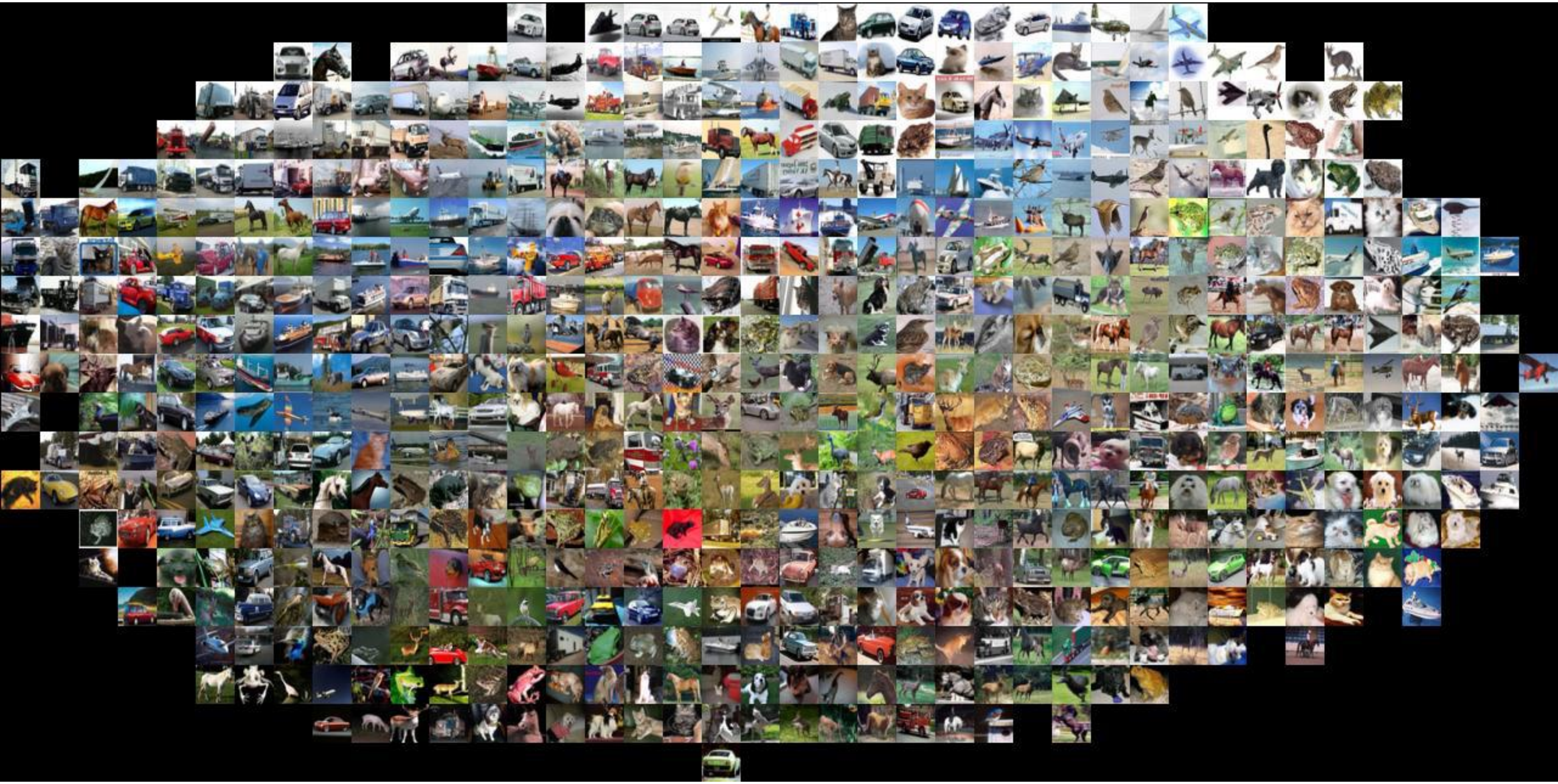
L2 (Euclidean) distance

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

- 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

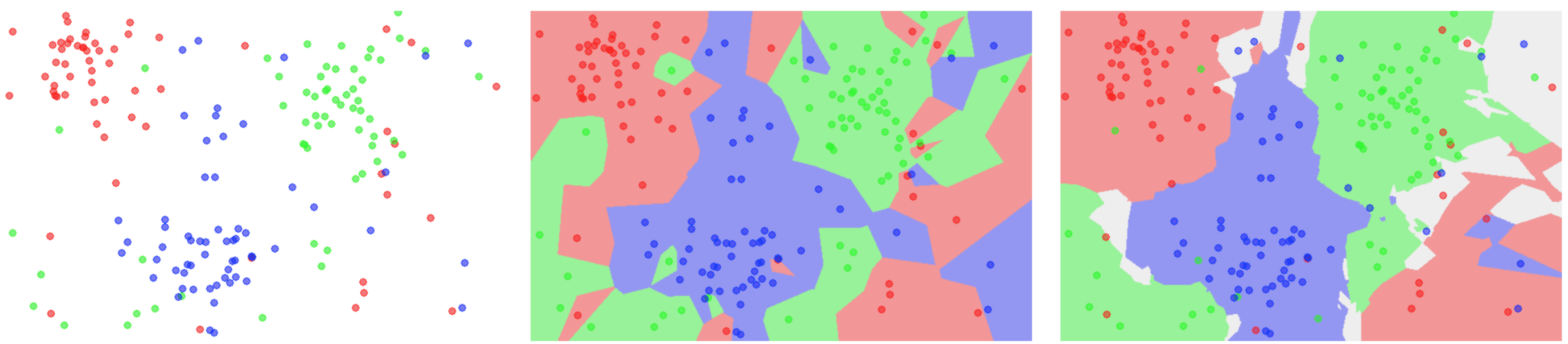
Example dataset: **CIFAR-10**
10 labels
50,000 training images
10,000 test images.

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

the data NN classifier 5-NN classifier



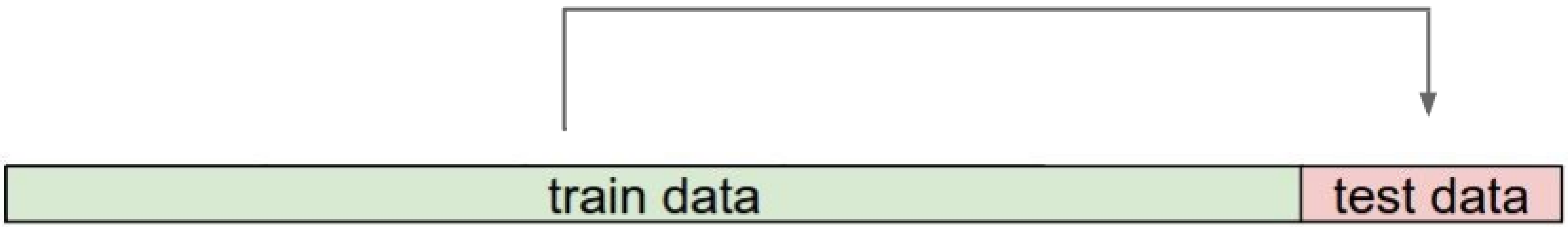
Hyperparameters

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

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

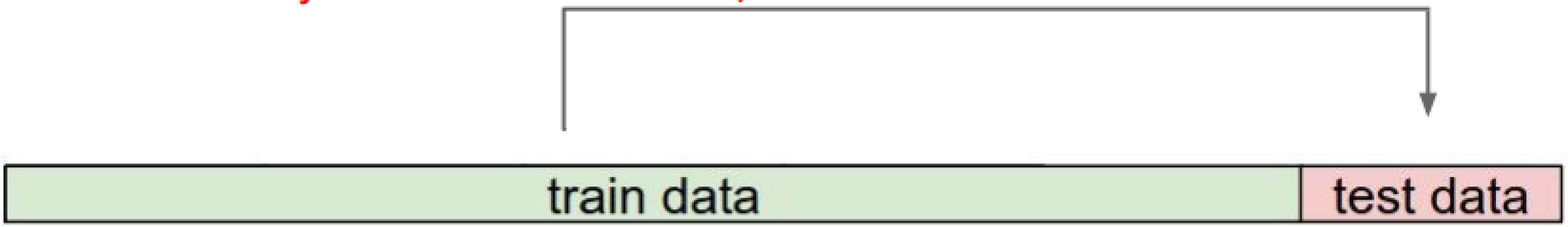
- Very problem-dependent
- Must try them all and see what works best

Try out what hyperparameters work best on test set.

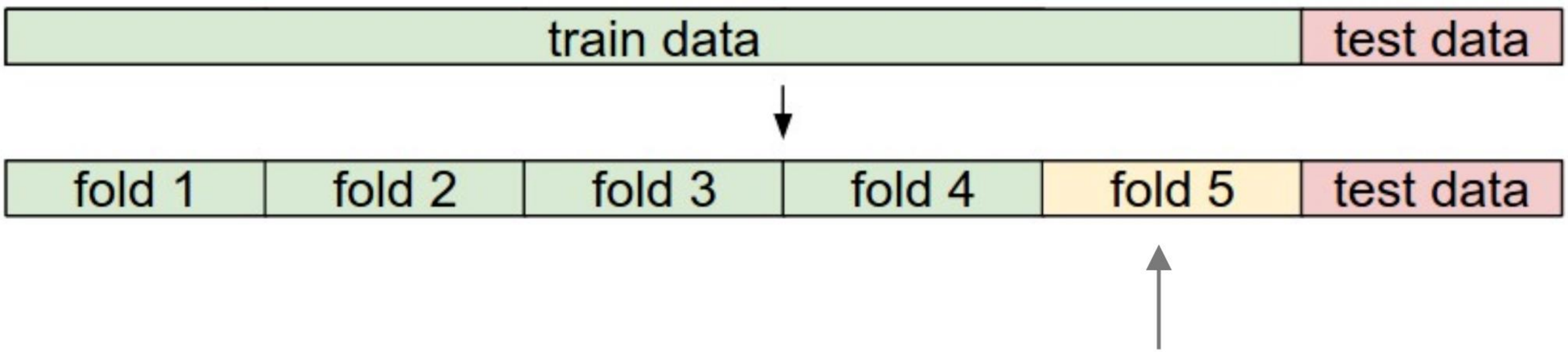


Trying out what hyperparameters work best on test set:

Very bad idea. The test set is a proxy for the generalization performance!
Use only **VERY SPARINGLY,** at the end.



Validation

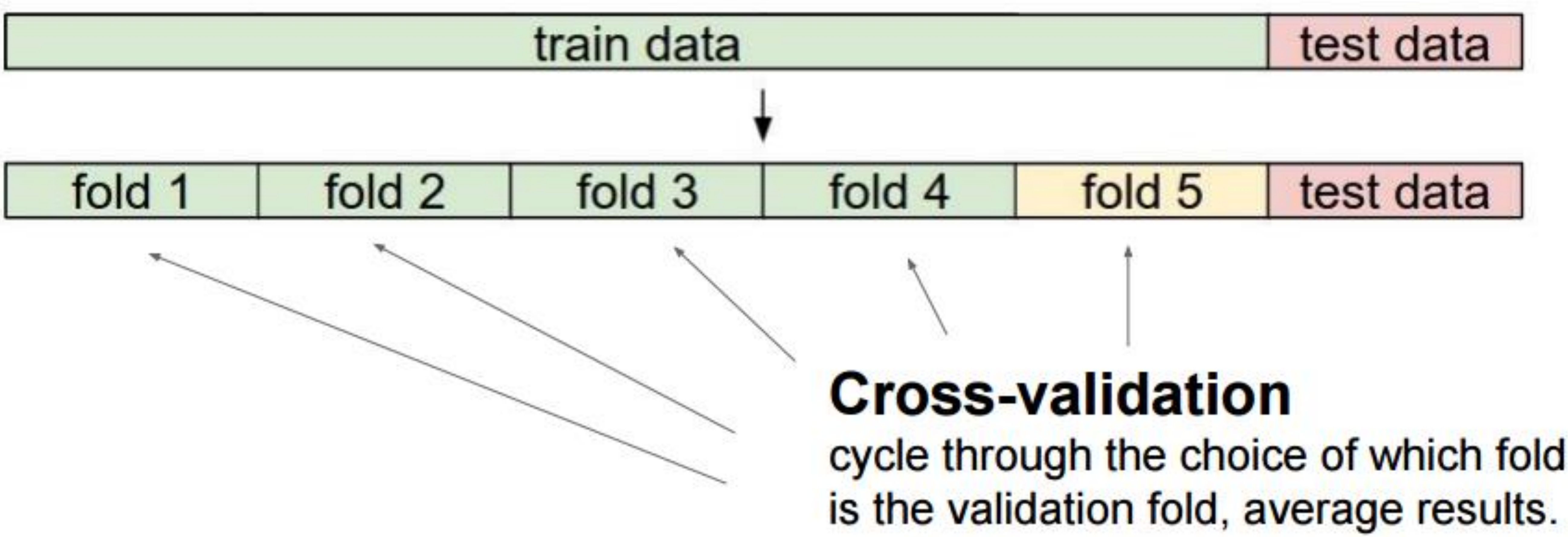


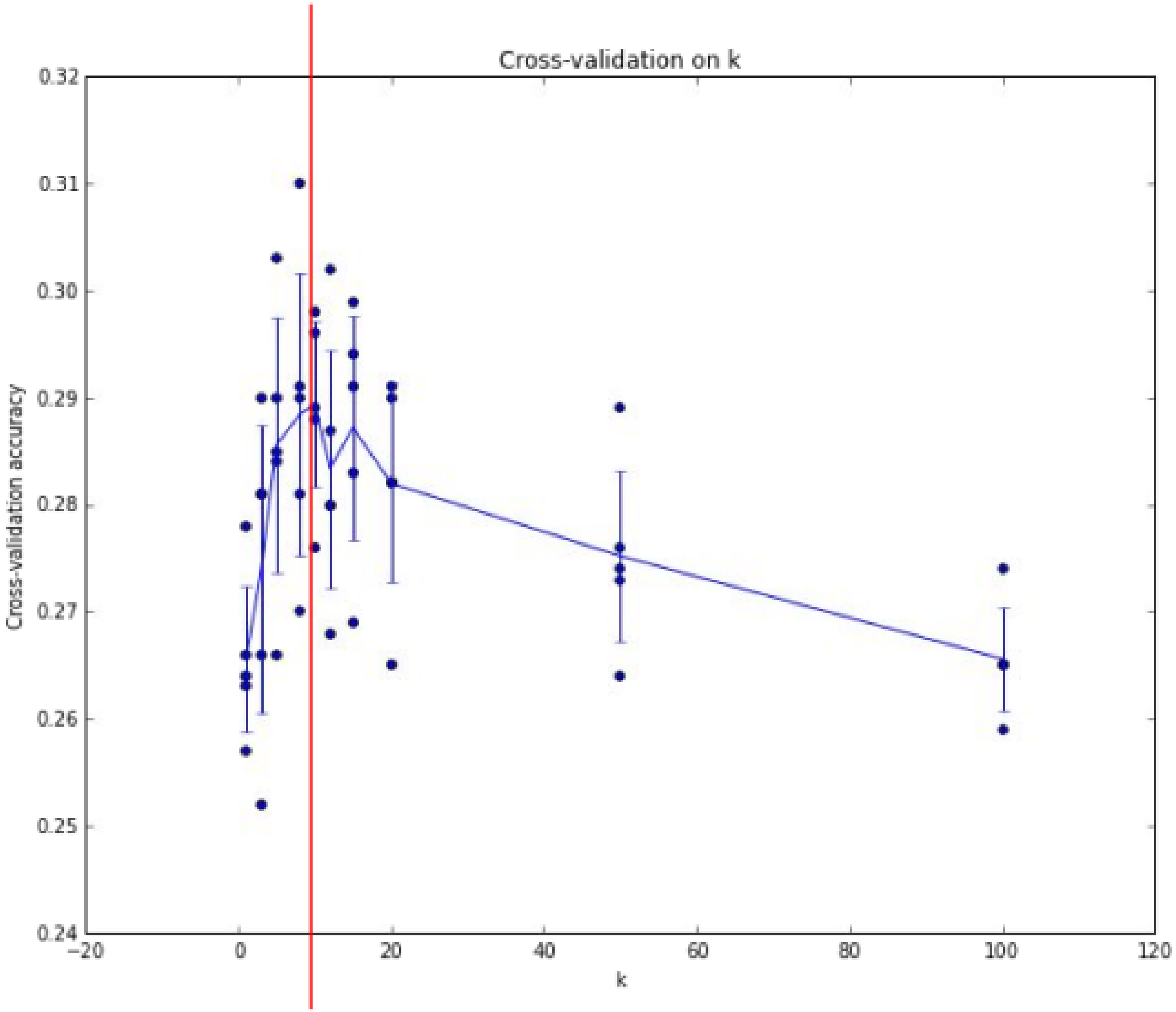
Validation data

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

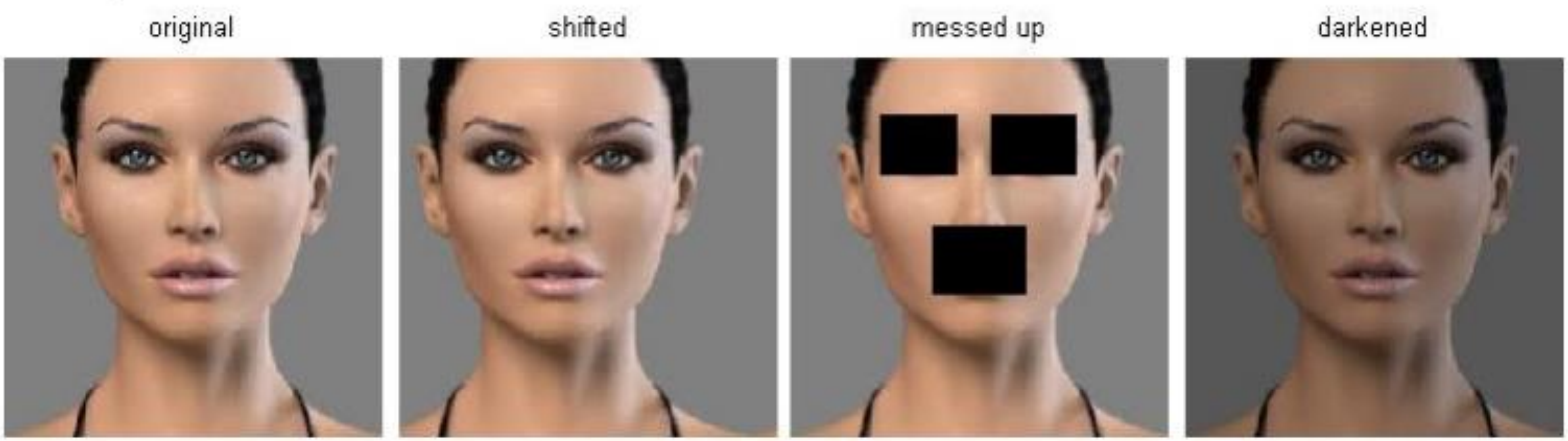
- N training images, M test images

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

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

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

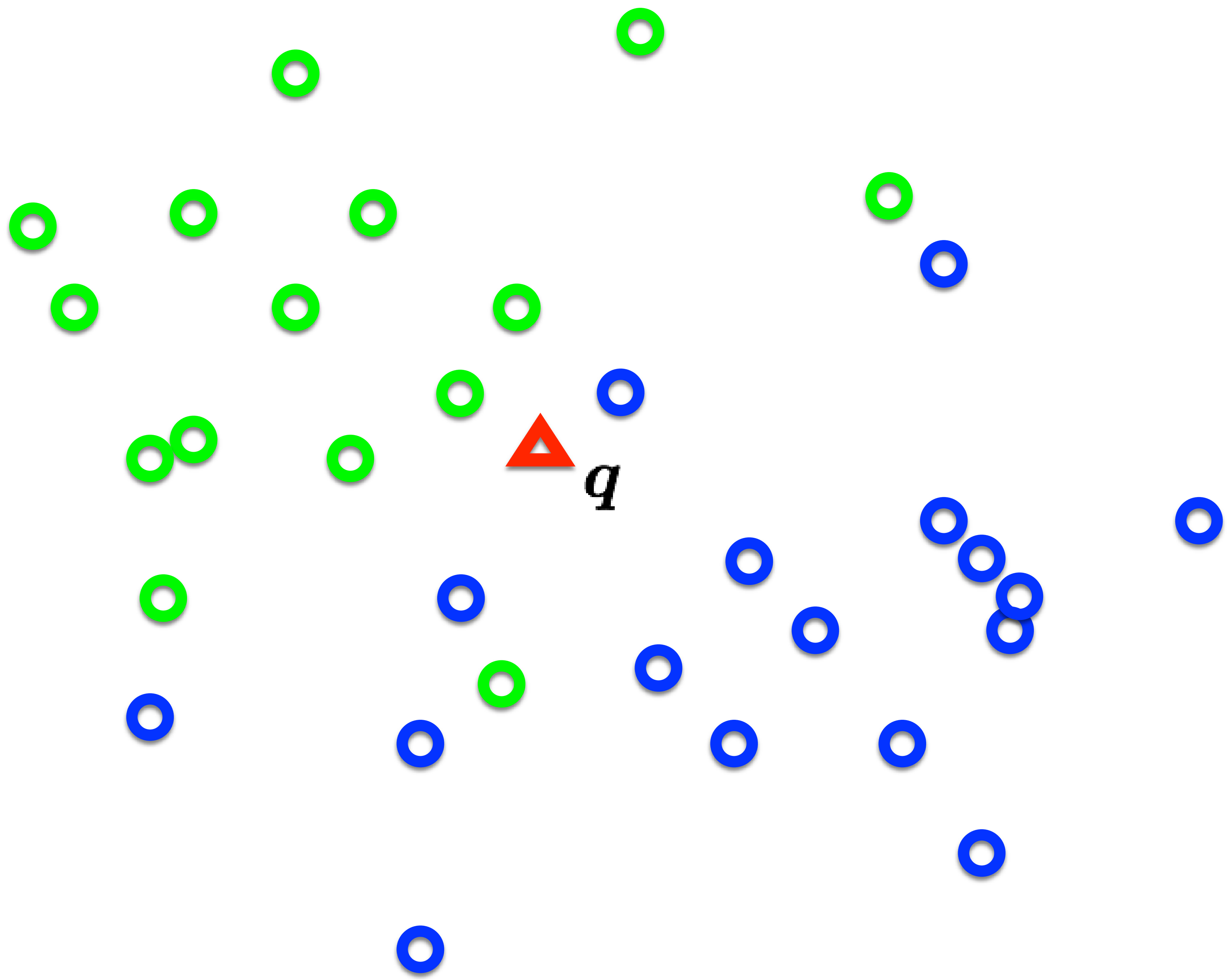
- terrible performance at test time
- distance metrics on level of whole images can be very unintuitive



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

Naïve Bayes

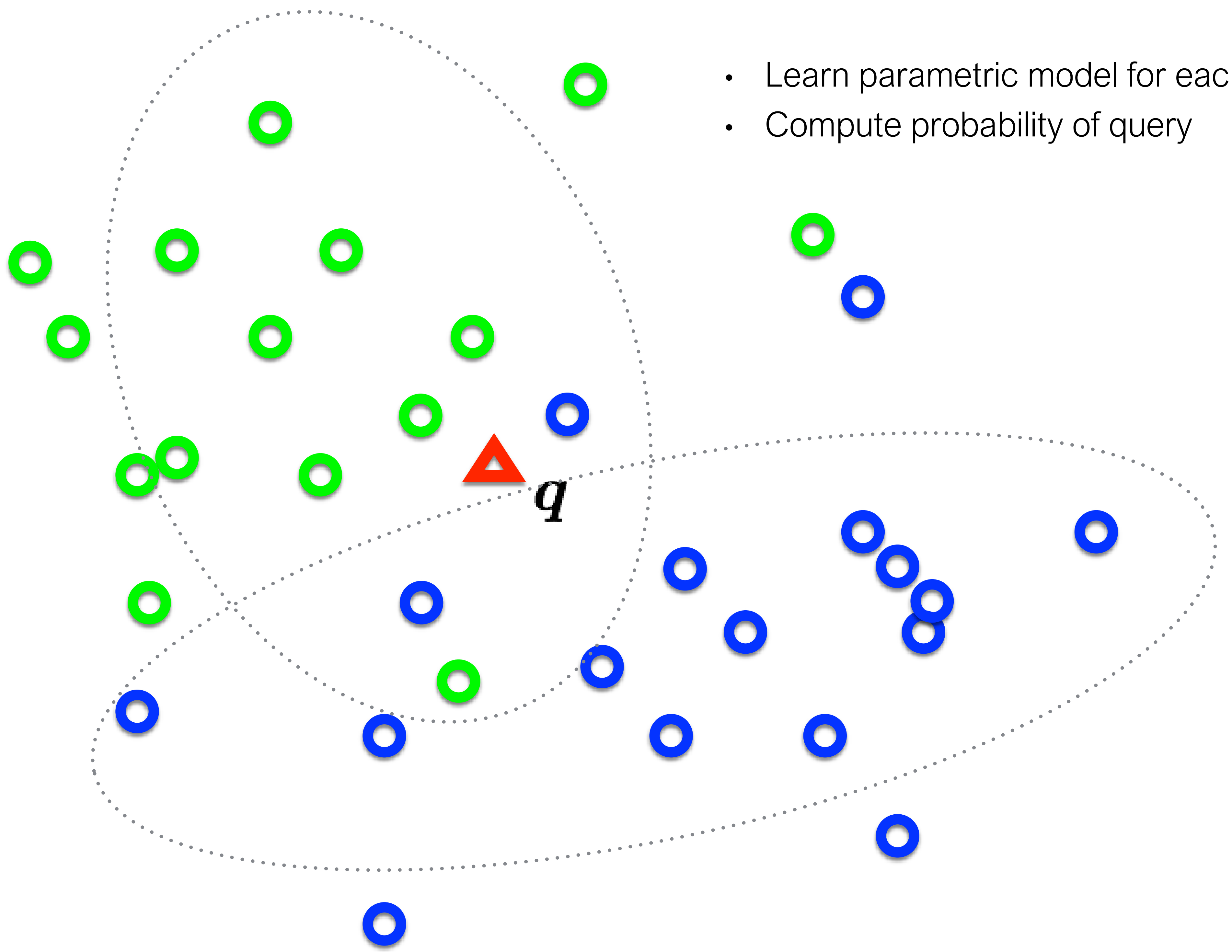
Distribution of data from two classes



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**

$$\underbrace{p(A|B)}_{\text{posterior}} = \frac{\overbrace{p(B|A)}^{\text{likelihood}}\overbrace{p(A)}^{\text{prior}}}{p(B)}$$

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

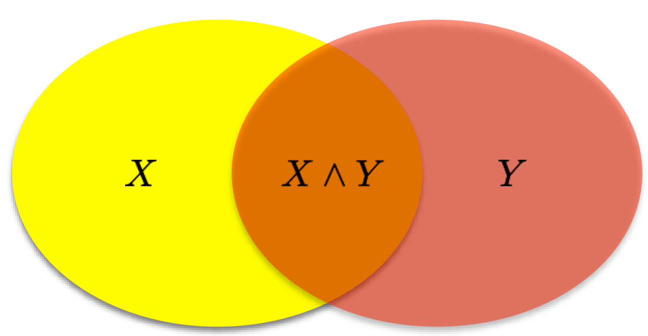
To optimize this…we need to compute this

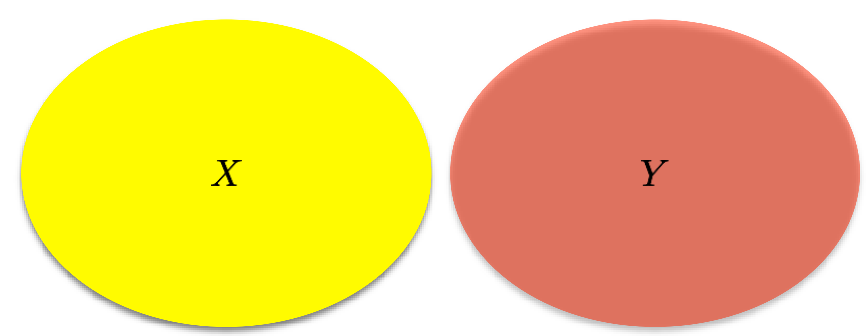
Compute the likelihood…

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:





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

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

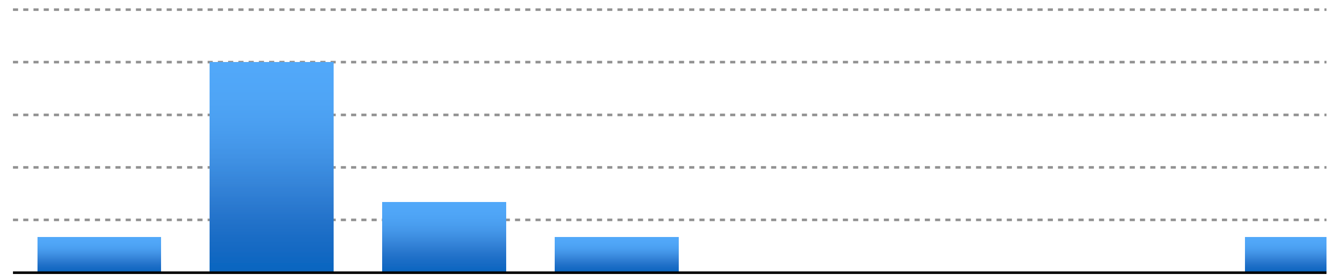
$$\hat{z} = \arg\max_{z \in \boldsymbol{z}} p(z) \prod_n p(x_n|z)$$

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

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

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

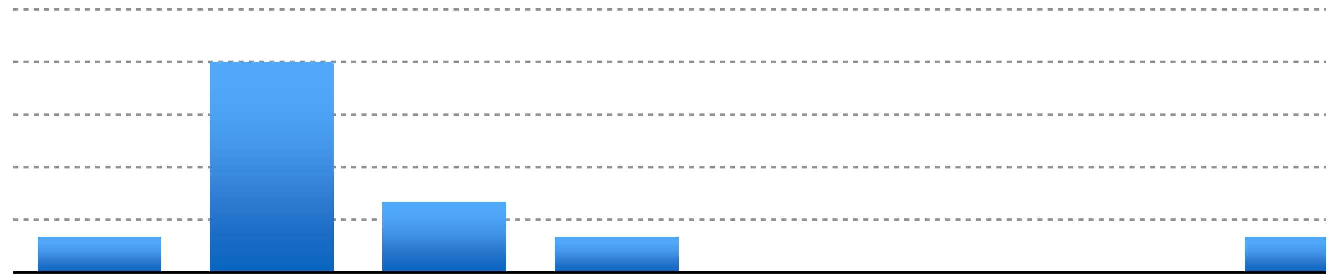
* typically add pseudo-counts (0.001)
** this is an example for computing the likelihood, need to multiply times **prior** to get posterior


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

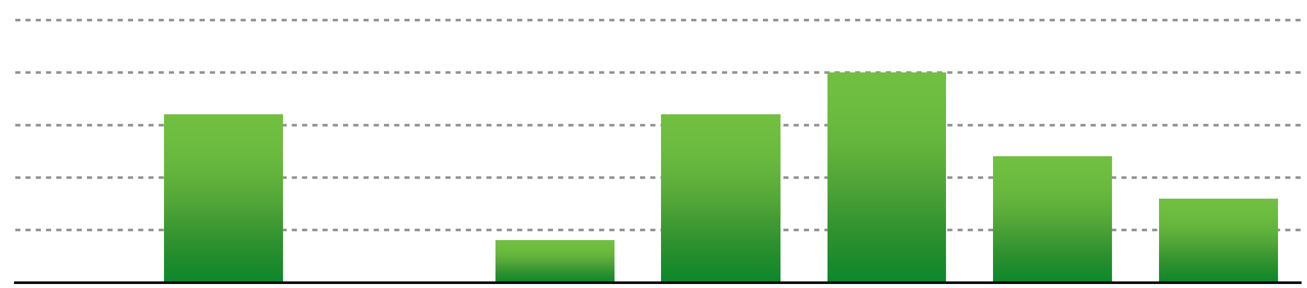

Tartan Tim

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

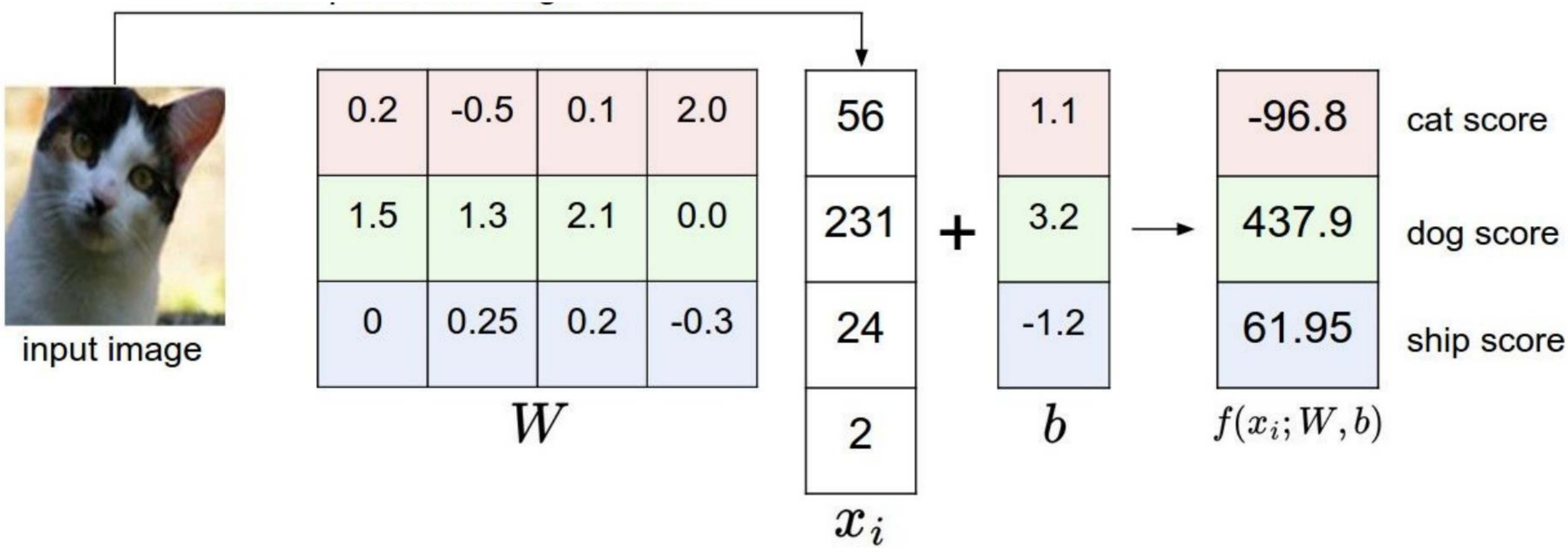
"weights"

"bias vector"

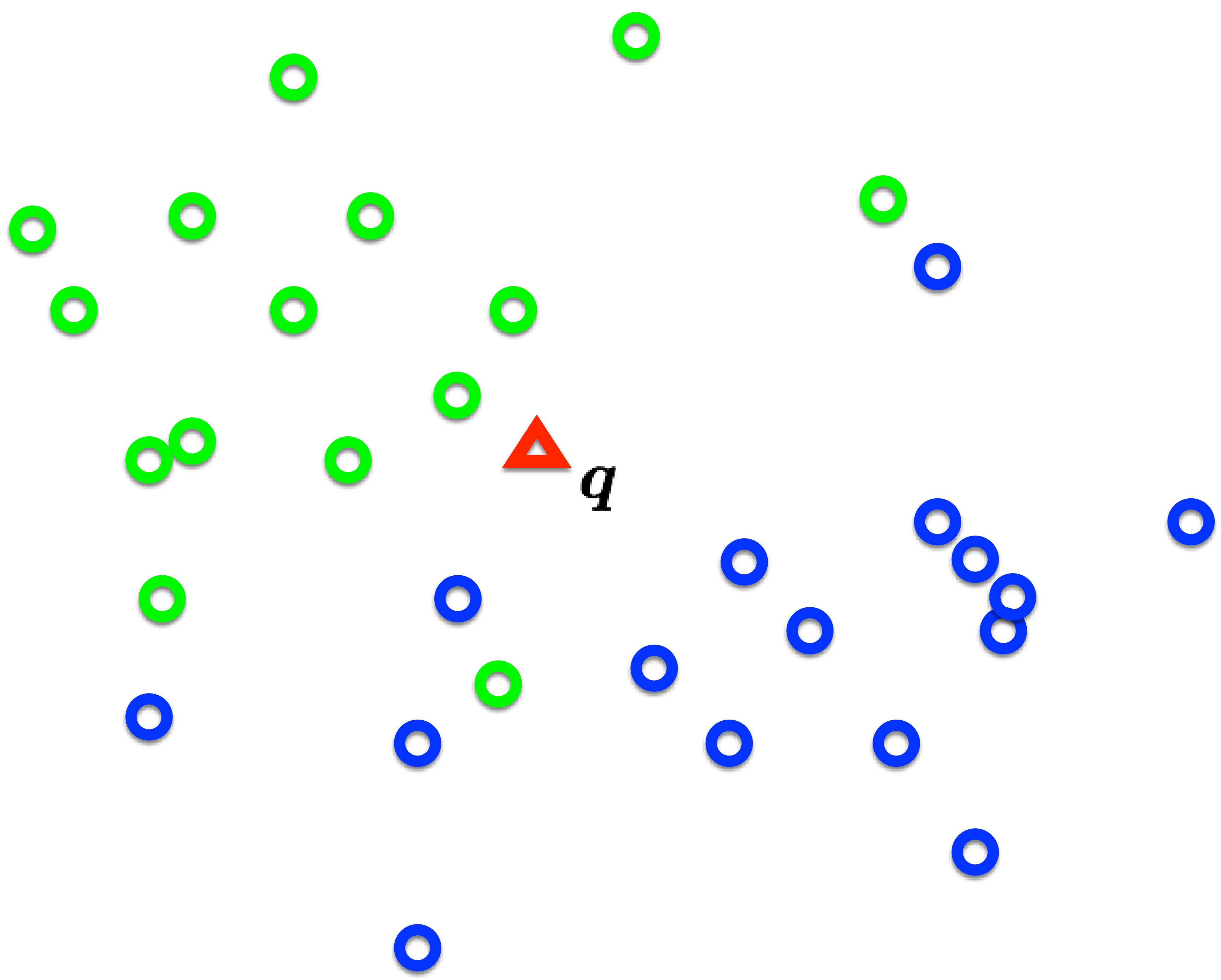
"parameters"

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

Convert image to histogram representation

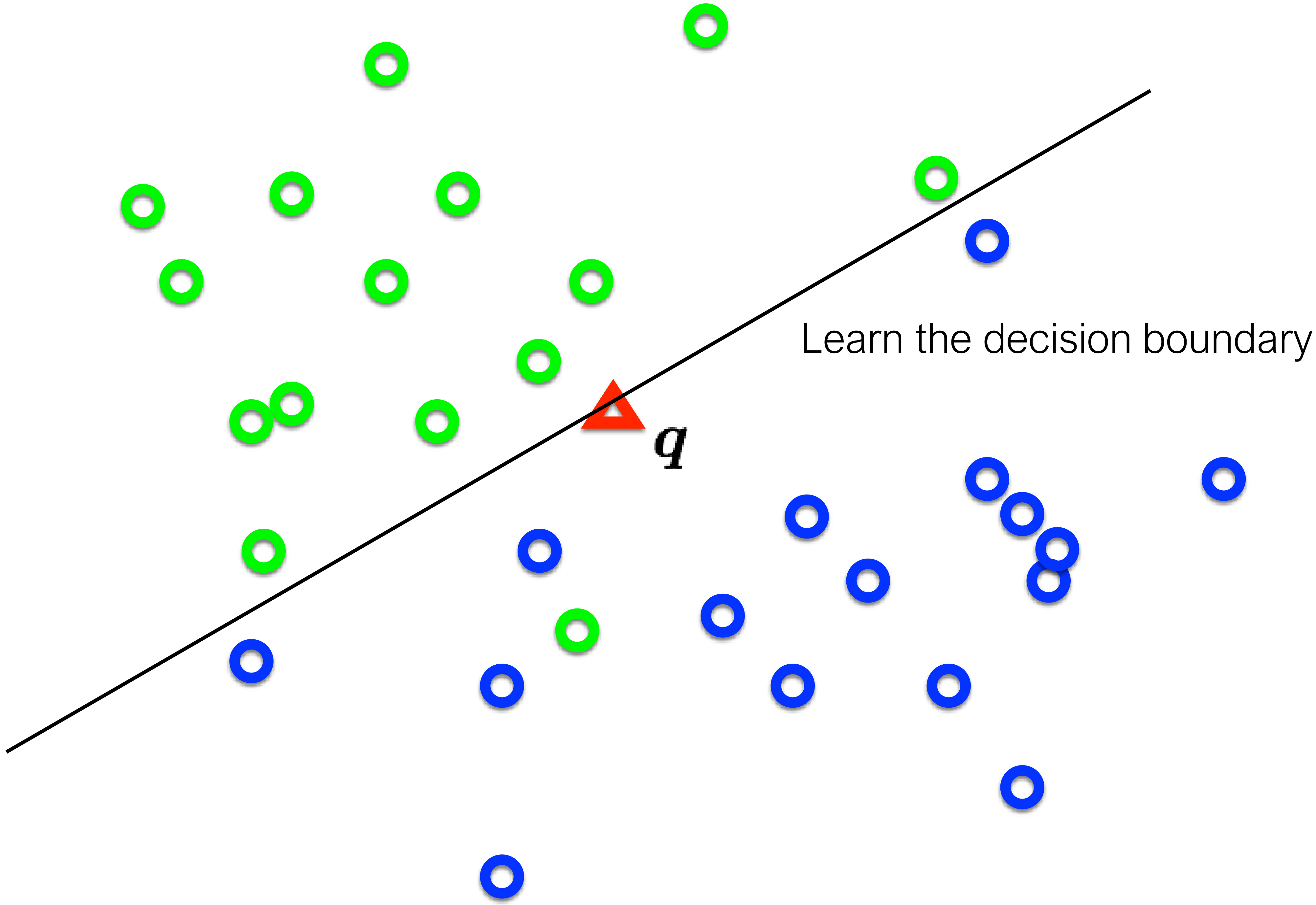


Distribution of data from two classes



Which class does q belong too?

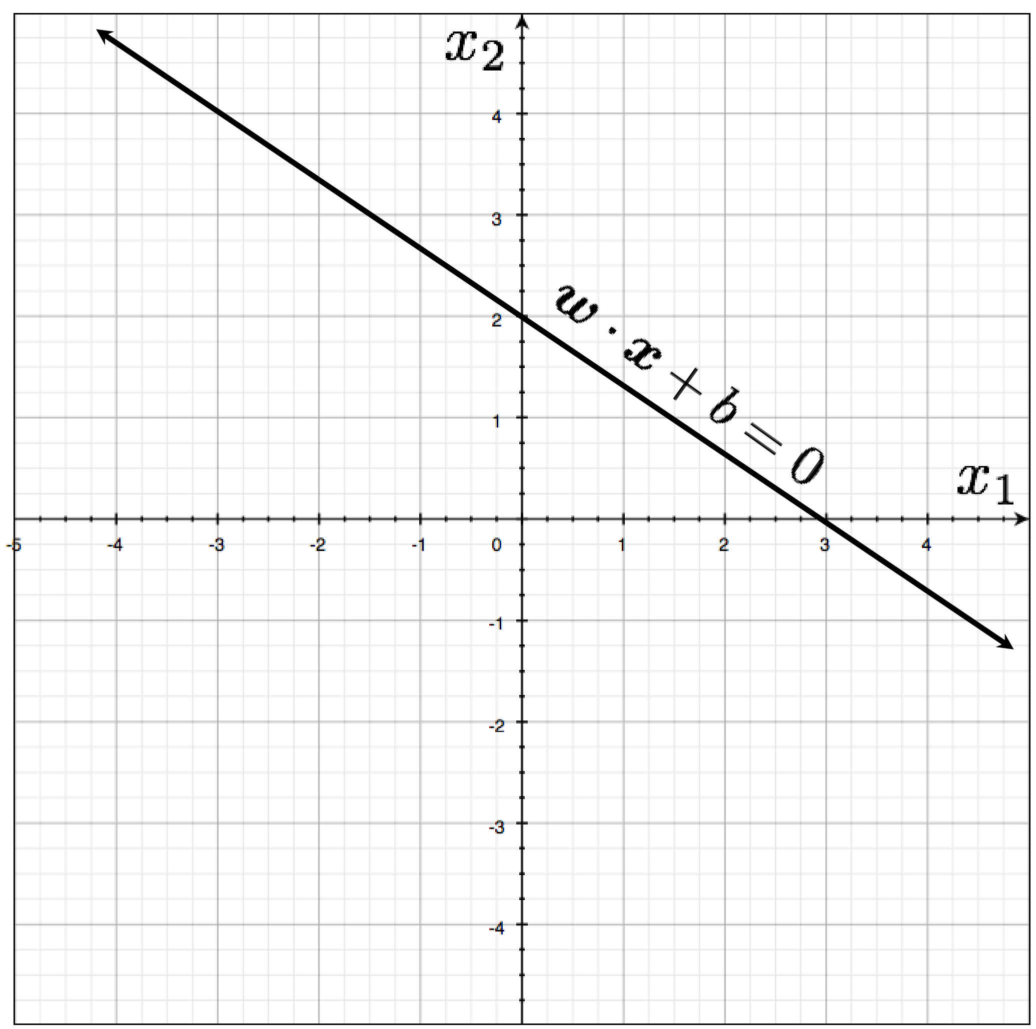
Distribution of data from two classes



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

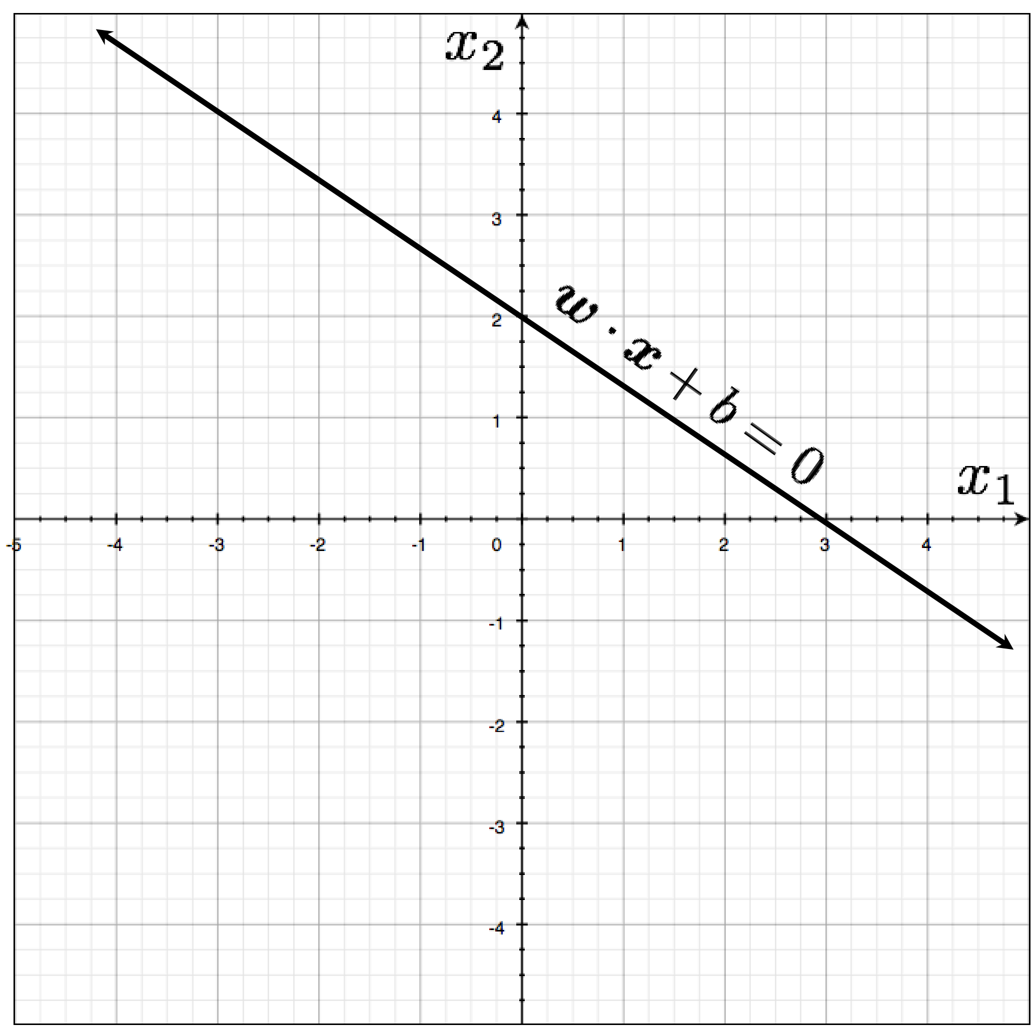
$$\boldsymbol{w} \cdot \boldsymbol{x} = 0$$

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

Hyperplanes (lines) in 2D

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

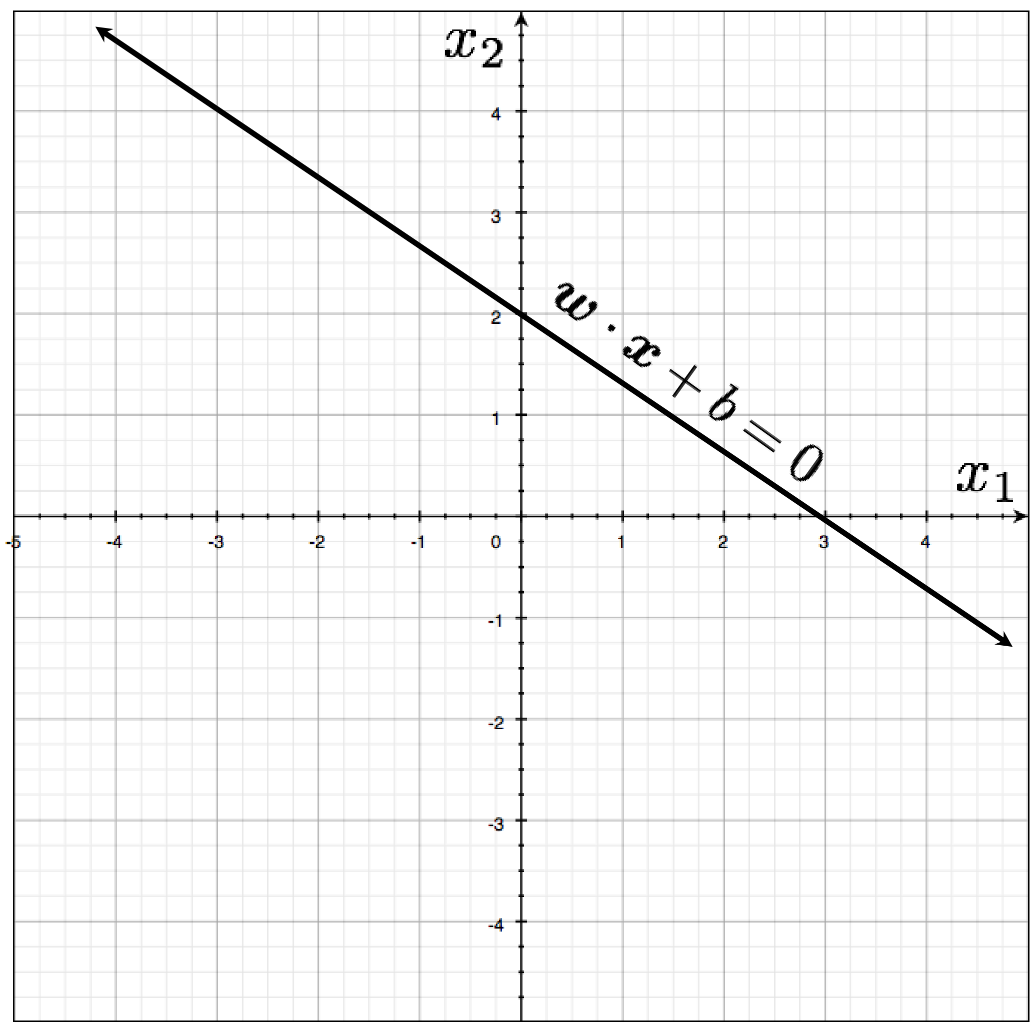
$$w_1 x_1 + w_2 x_2 + b = 0$$



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

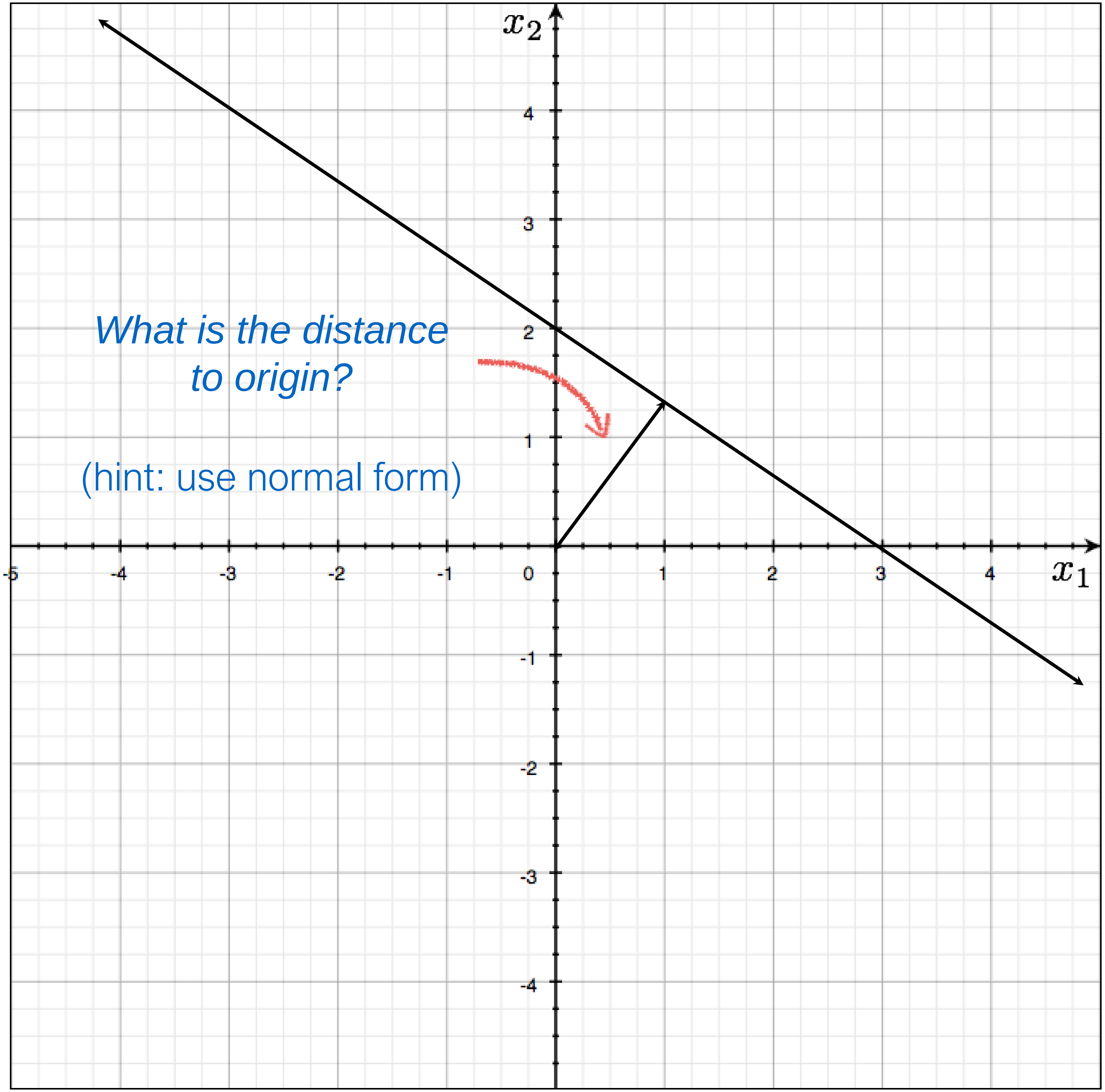
The line

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

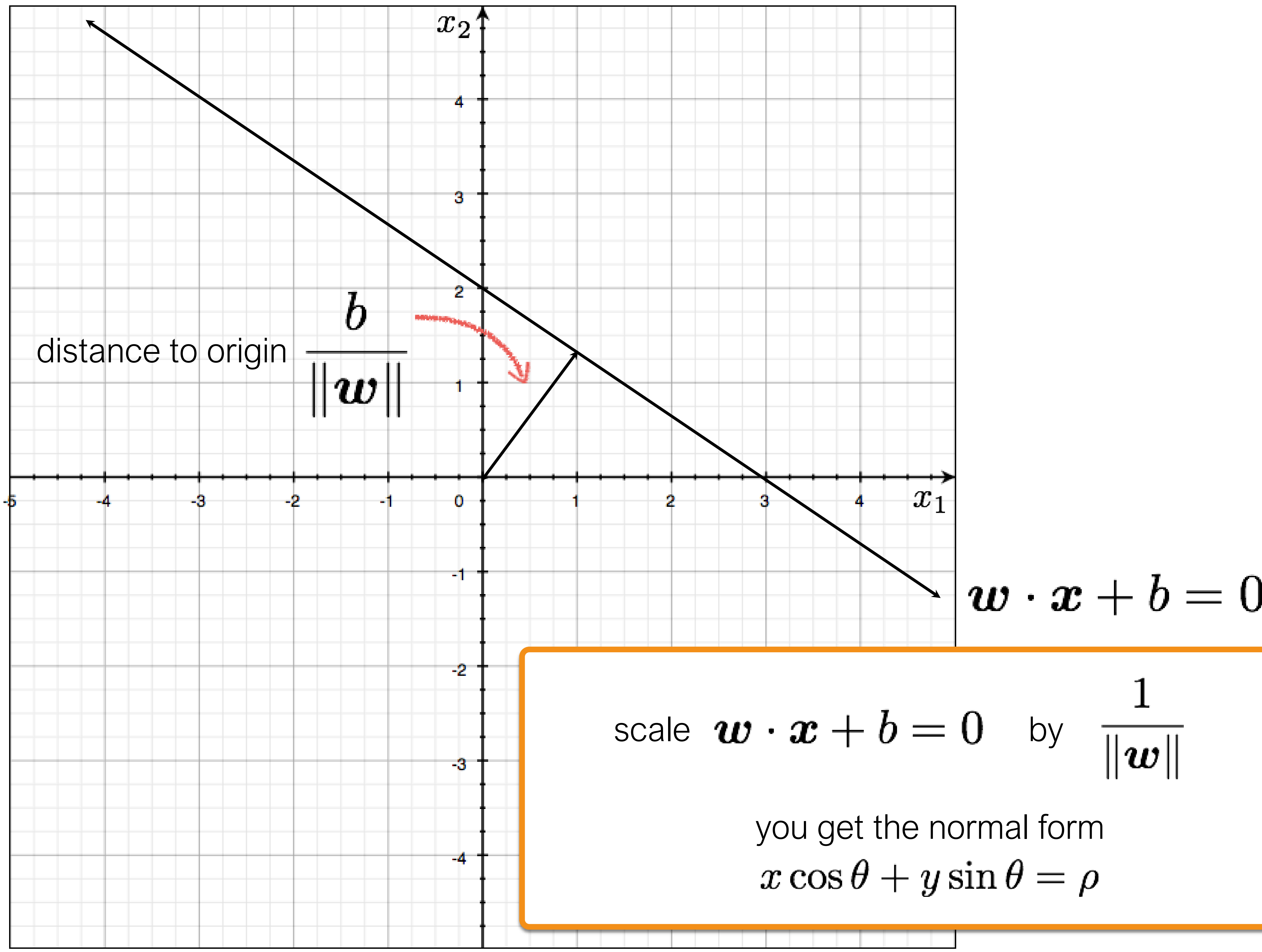
and the line

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

define the same line



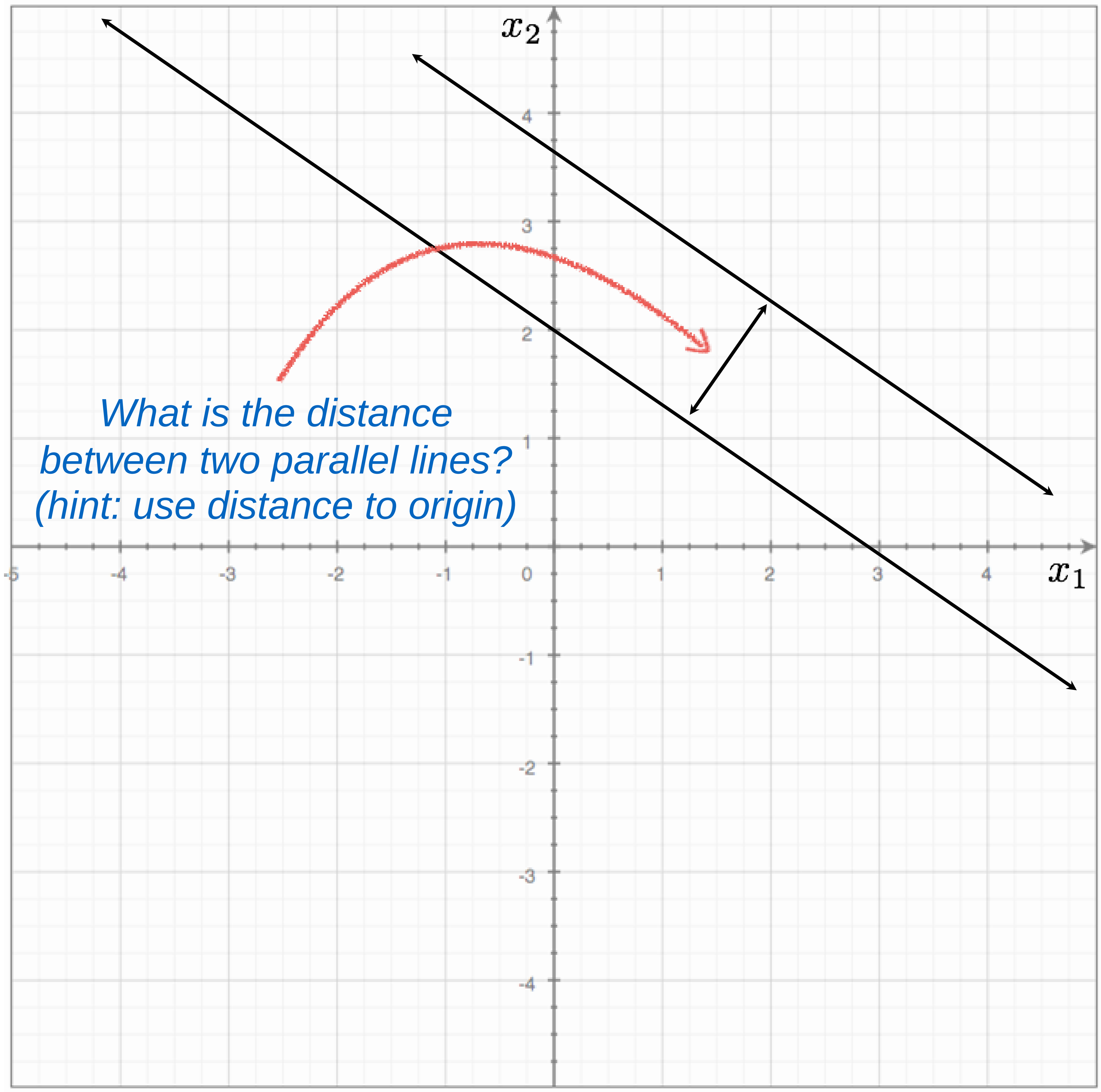
$$\boldsymbol{w} \cdot \boldsymbol{x} + b = 0$$



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

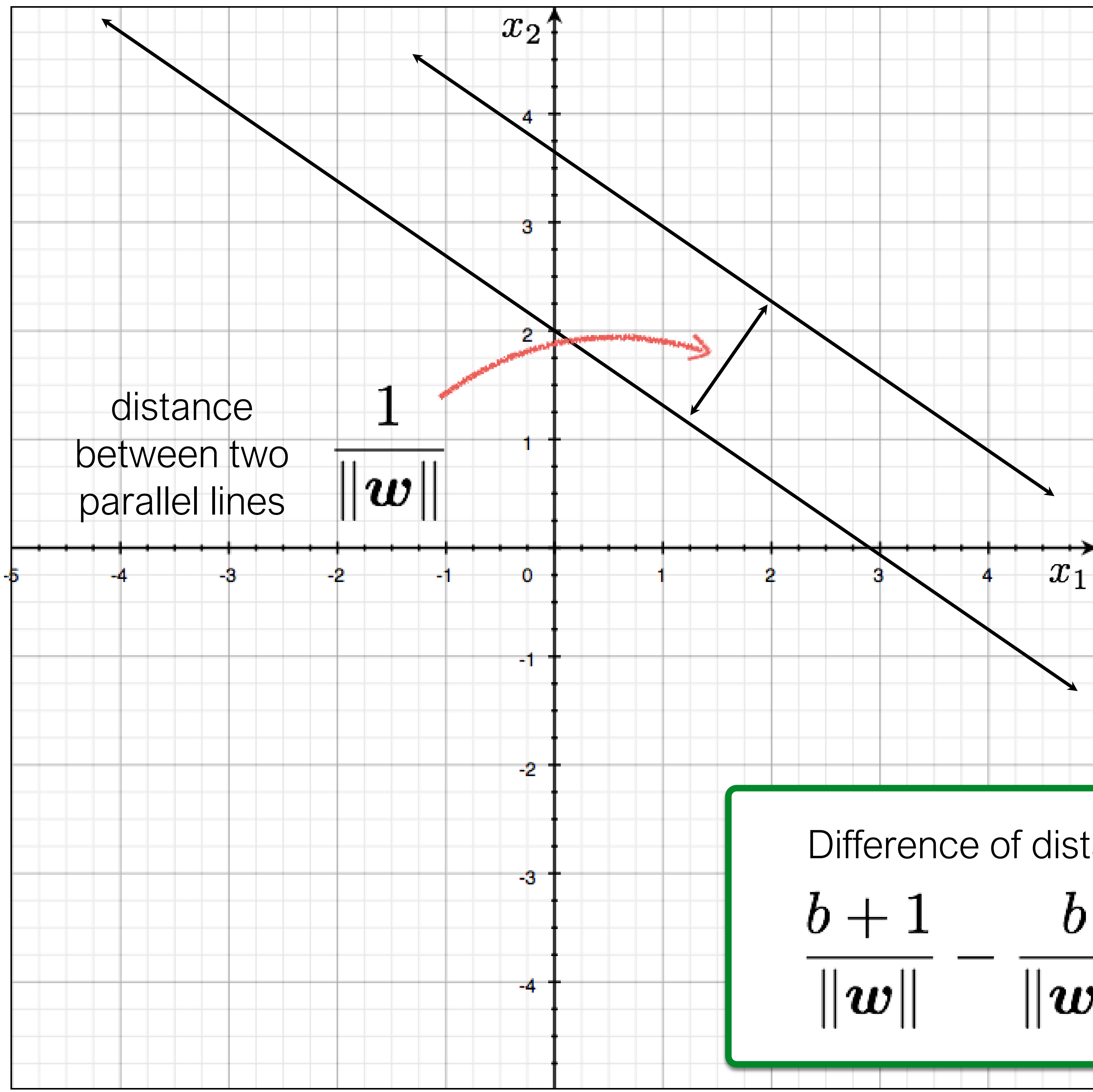
you get the normal form

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



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

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



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

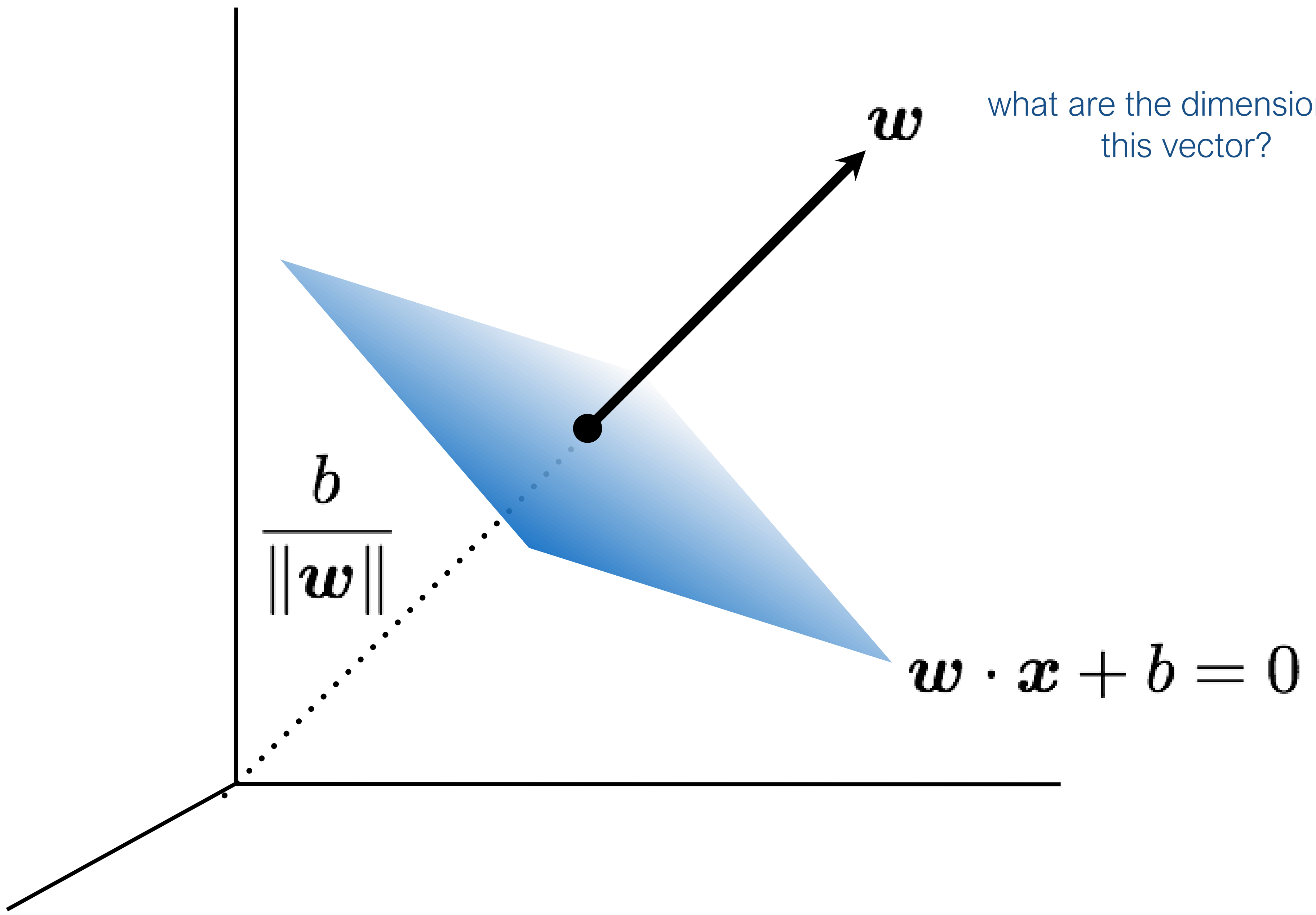
$$\boldsymbol{w} \cdot \boldsymbol{x} + b = 0$$

Difference of distance to origin

$$\frac{b+1}{\|\boldsymbol{w}\|} - \frac{b}{\|\boldsymbol{w}\|} = \frac{1}{\|\boldsymbol{w}\|}$$

Now we can go to 3D …

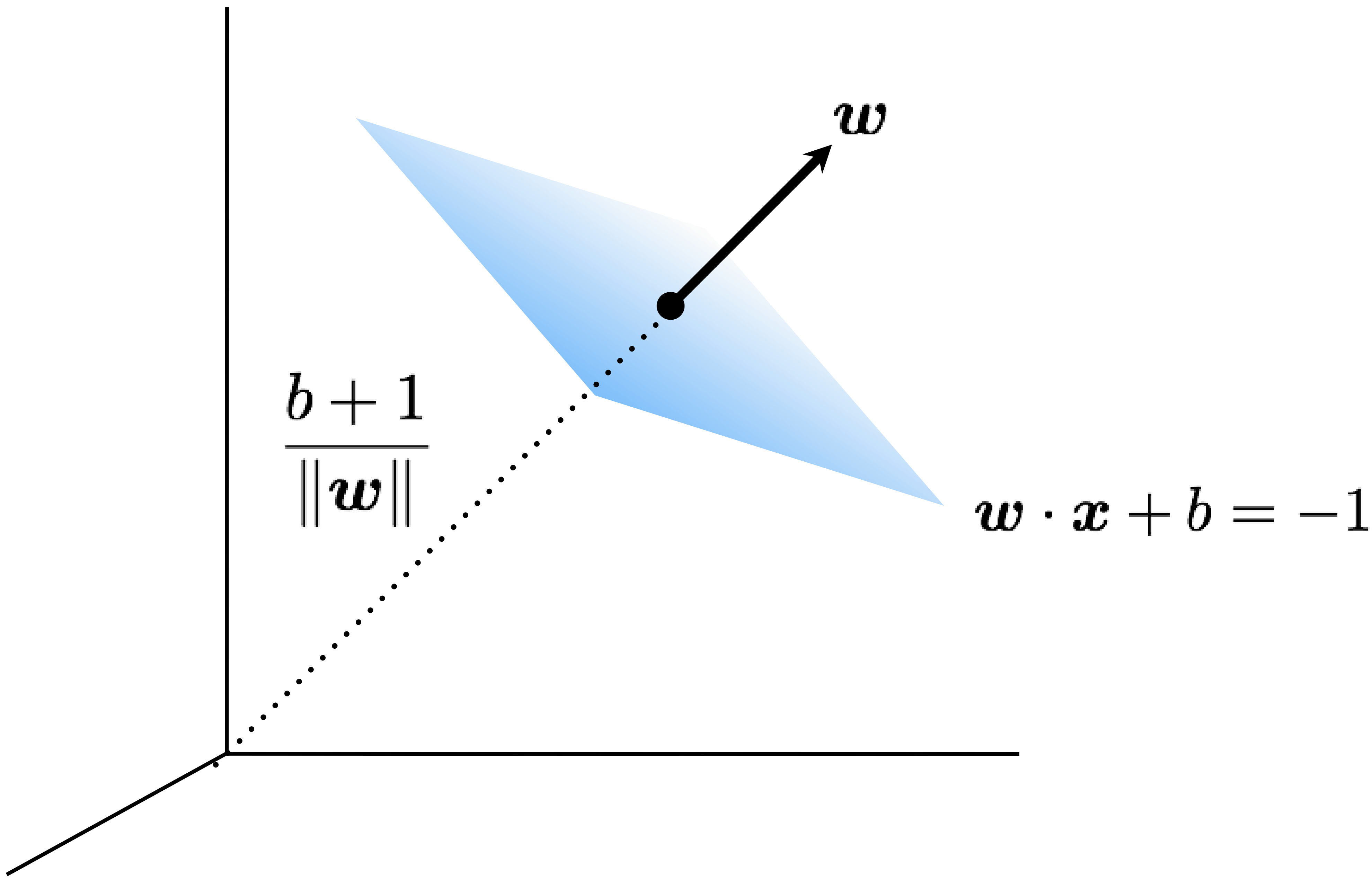
Hyperplanes (planes) in 3D



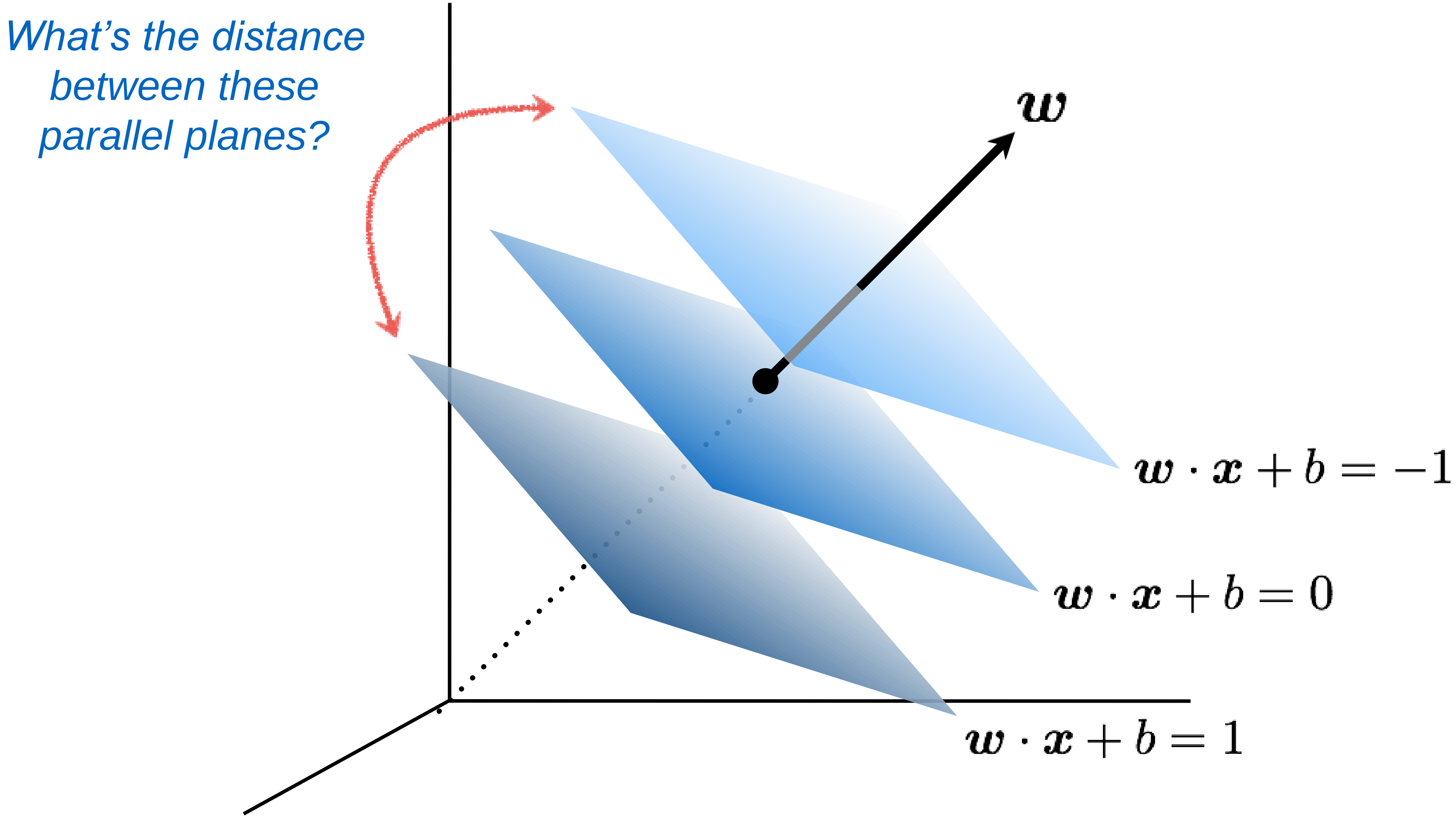
what are the dimensions of this vector?

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

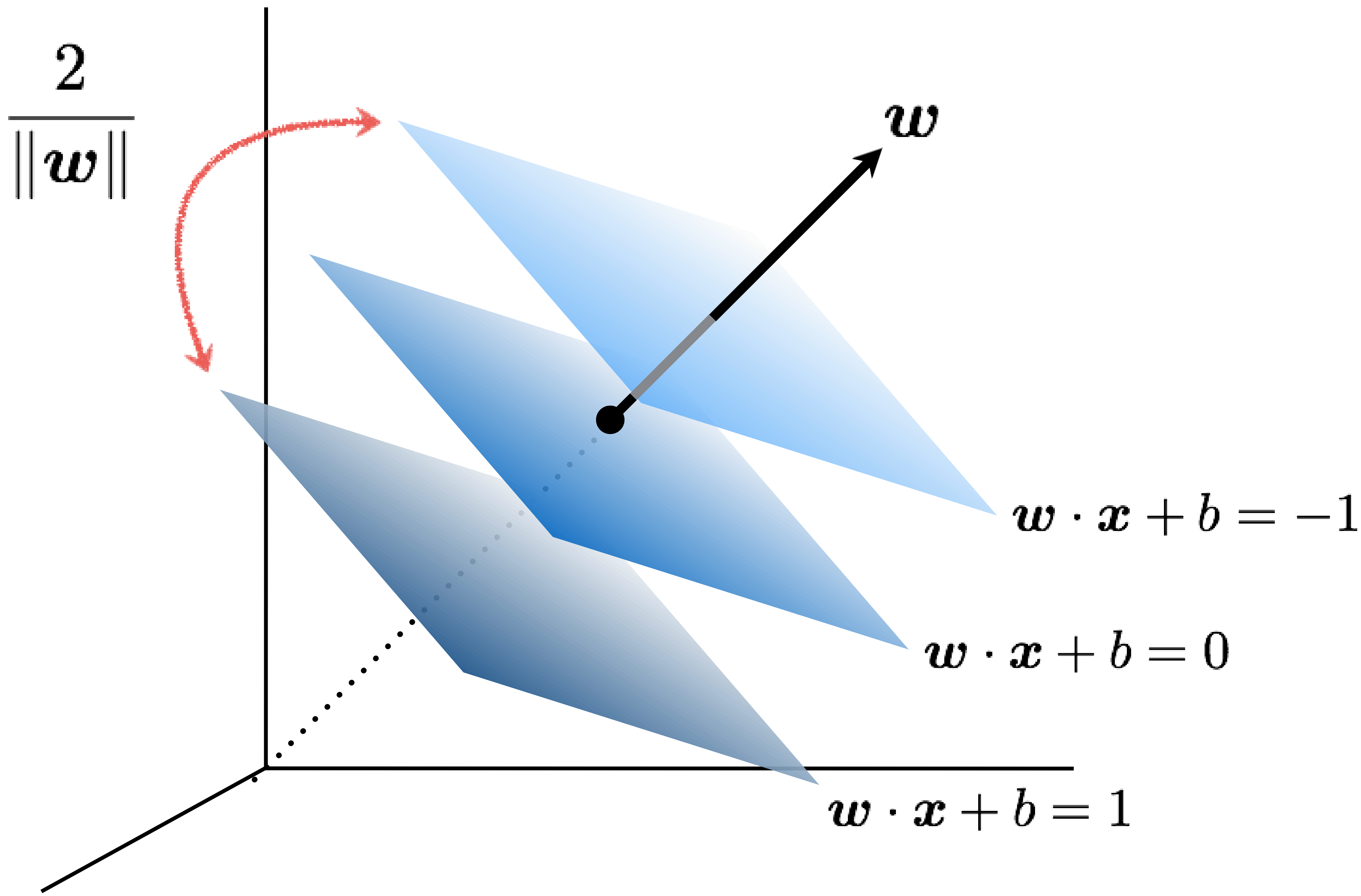
Hyperplanes (planes) in 3D



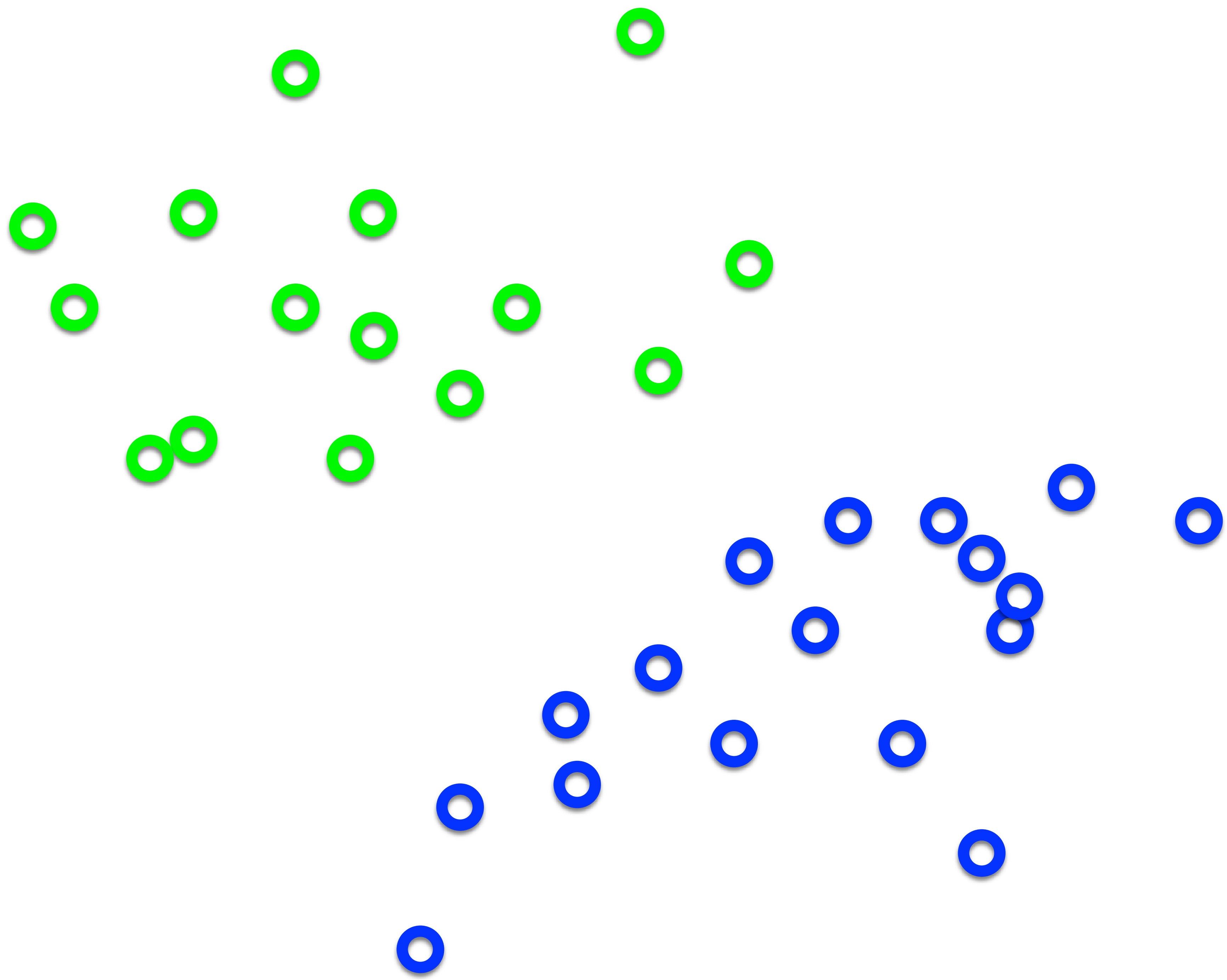
Hyperplanes (planes) in 3D



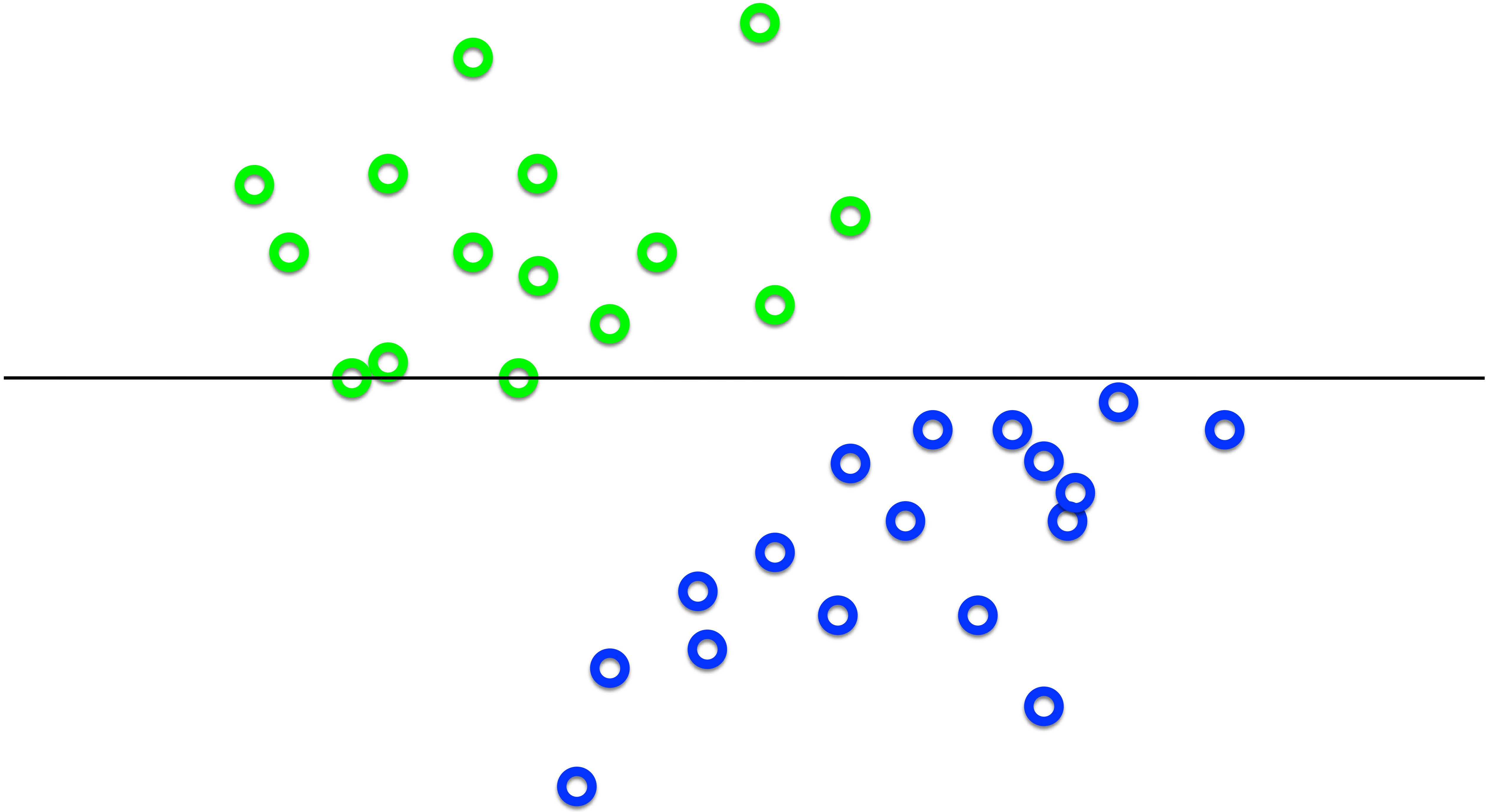
Hyperplanes (planes) in 3D



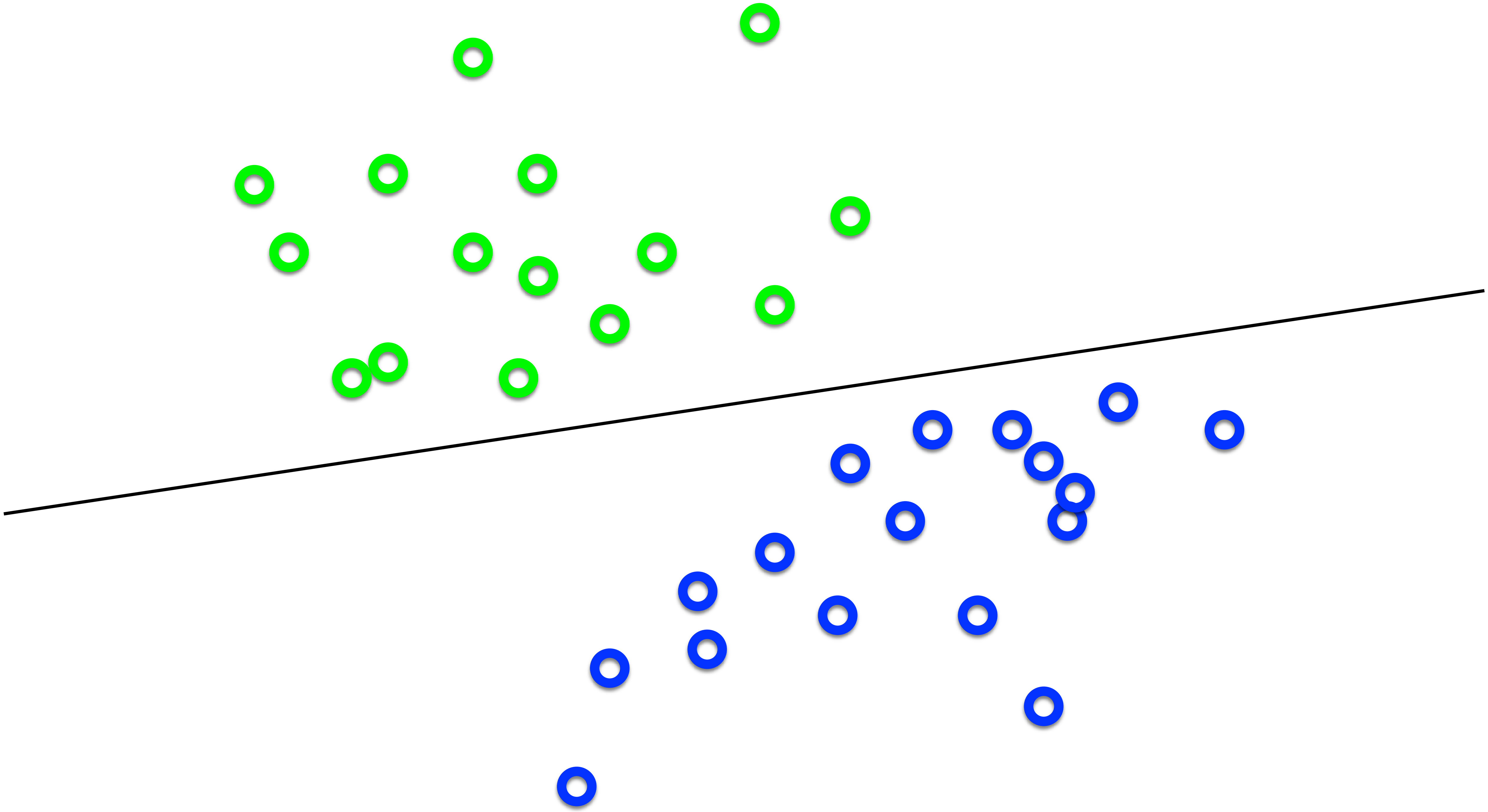
What's the best **w**?



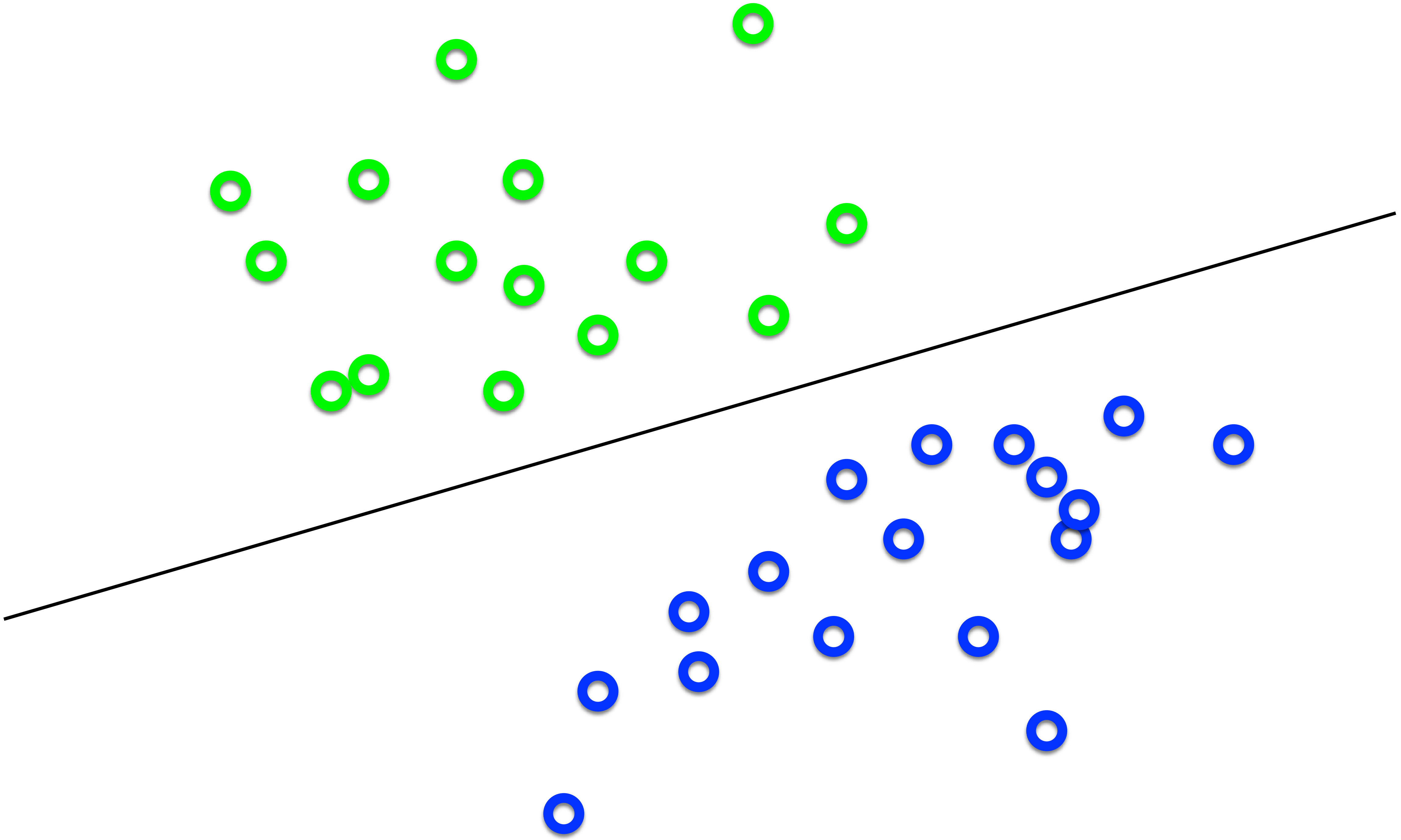
What’s the best **w**?



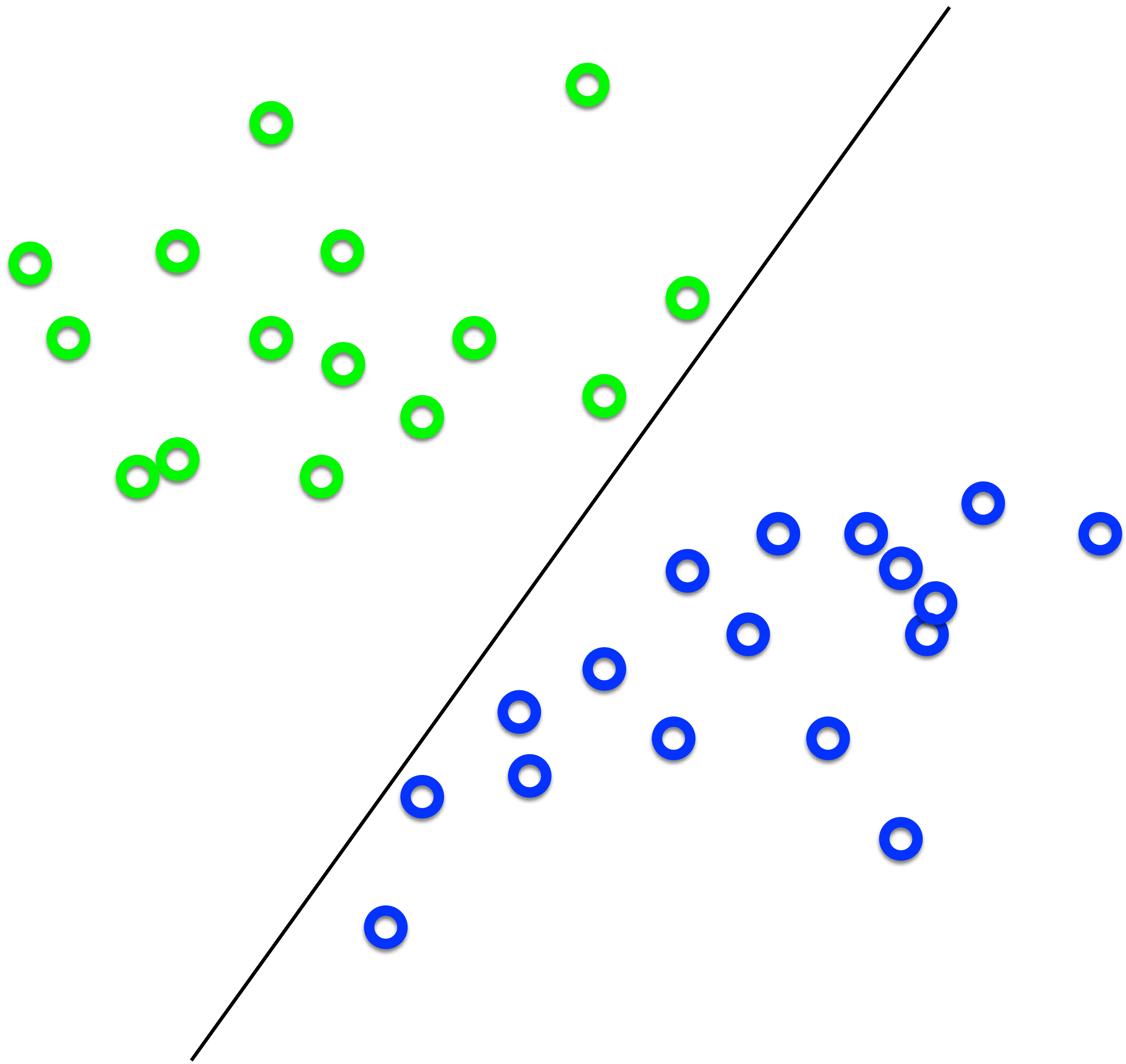
What’s the best **w**?



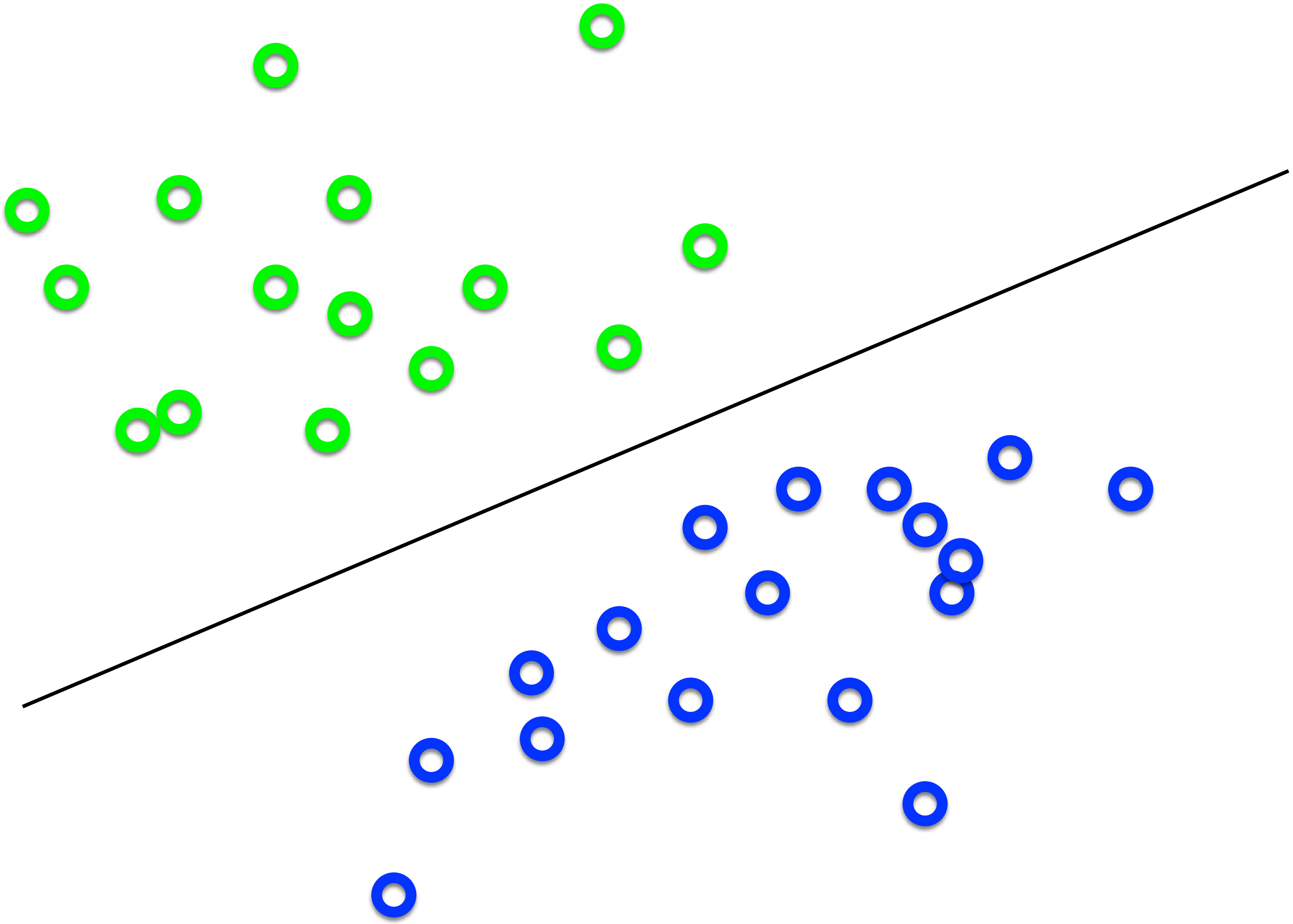
What's the best **w**?



What’s the best **w**?

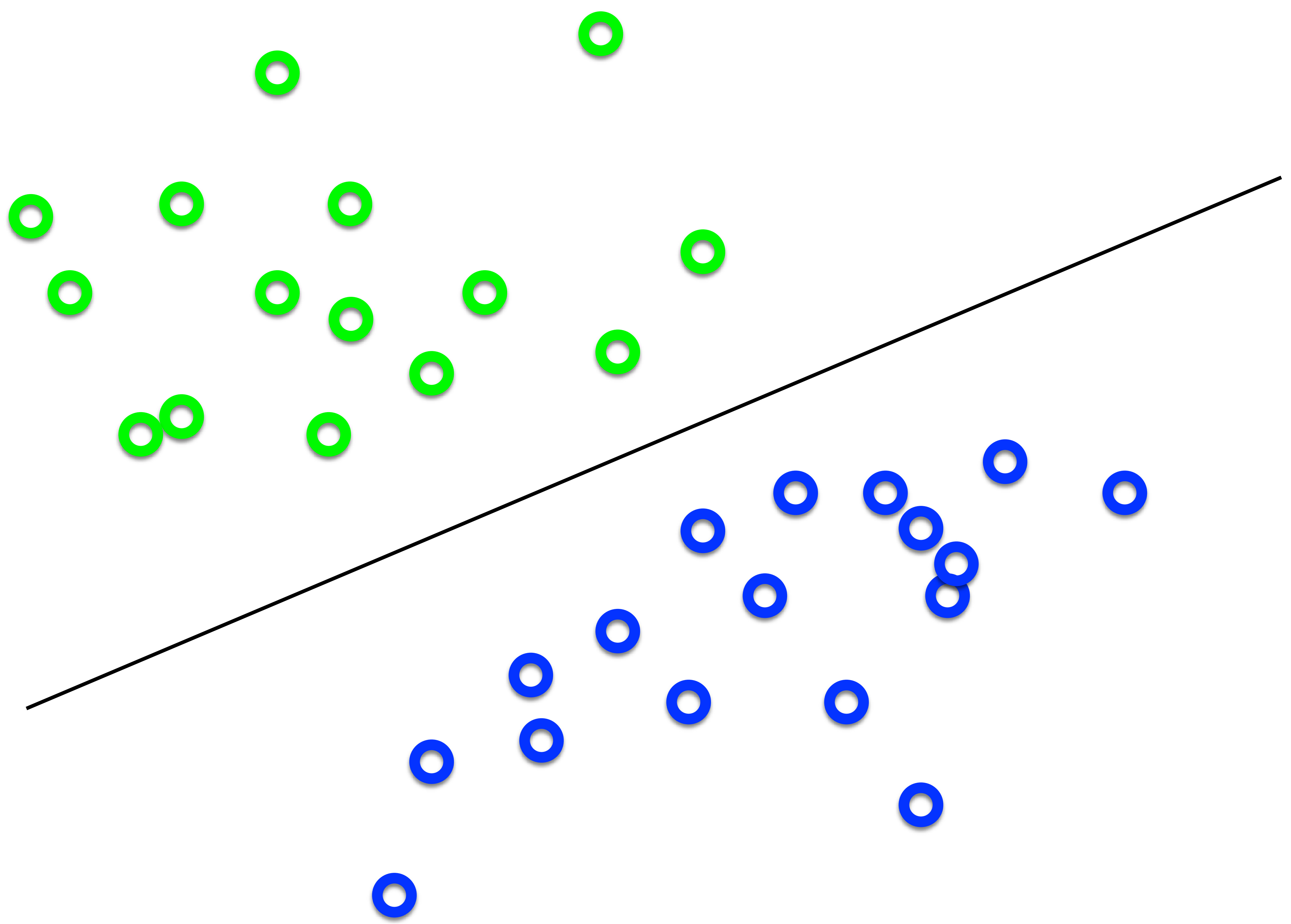


What's the best **w**?



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

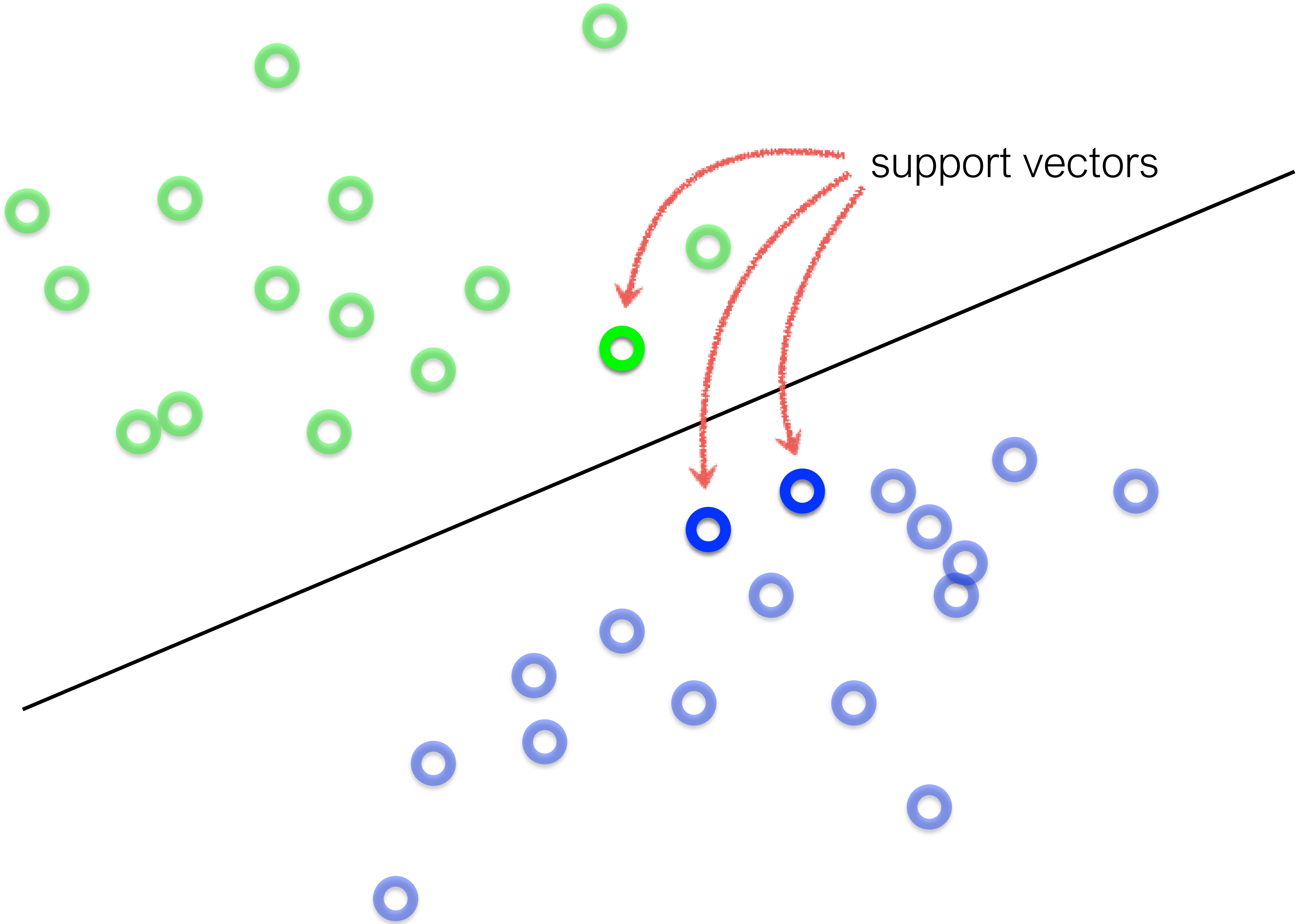
What’s the best **w**?



Maximum Margin solution:

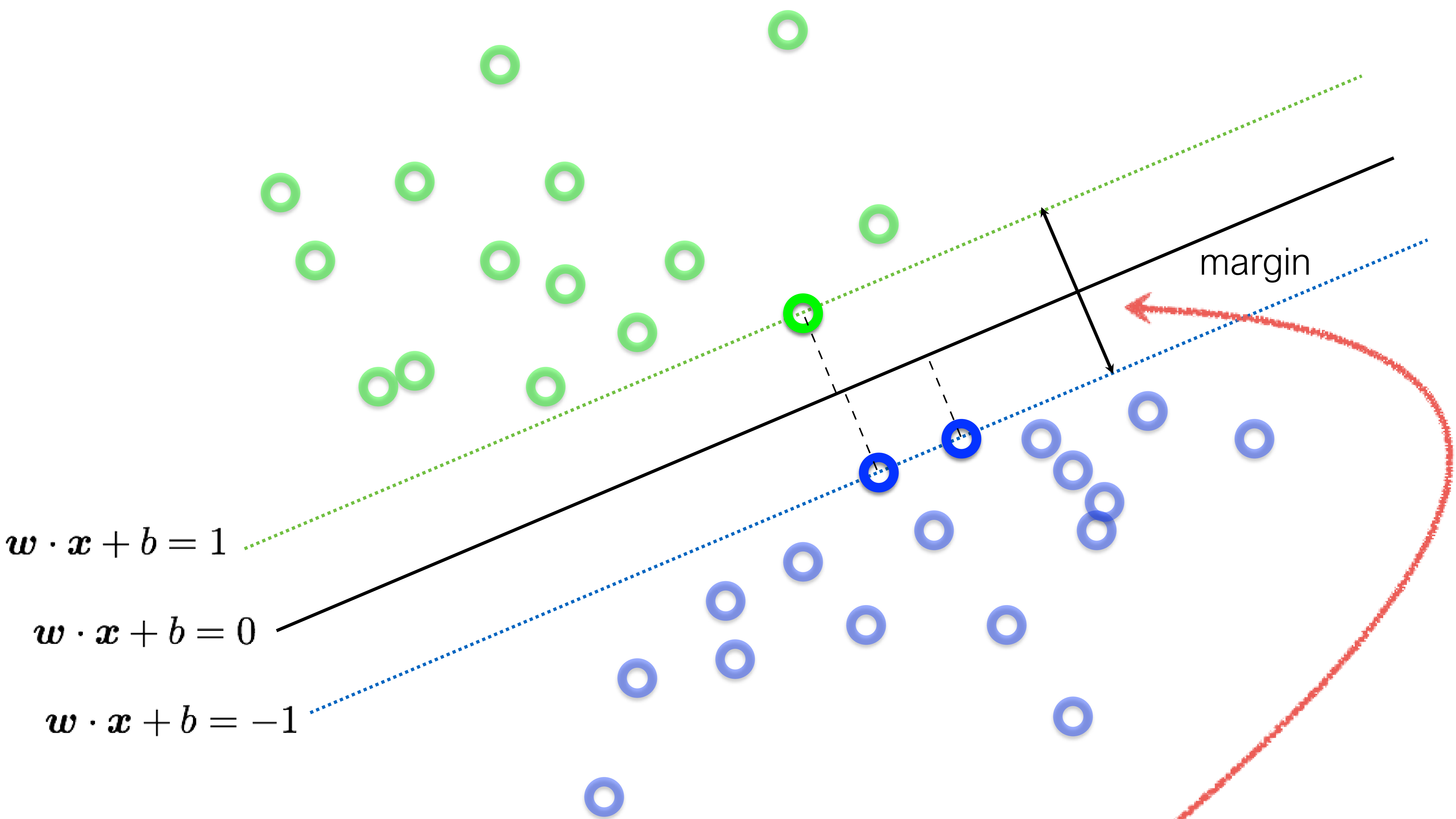
most stable to perturbations of data

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 $\frac{2}{\|\boldsymbol{w}\|}$ is maximized

Can be formulated as a maximization problem

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

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

What does this constraint mean?

label of the data point

Why is it +1 and -1?

Can be formulated as a maximization problem

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

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

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

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

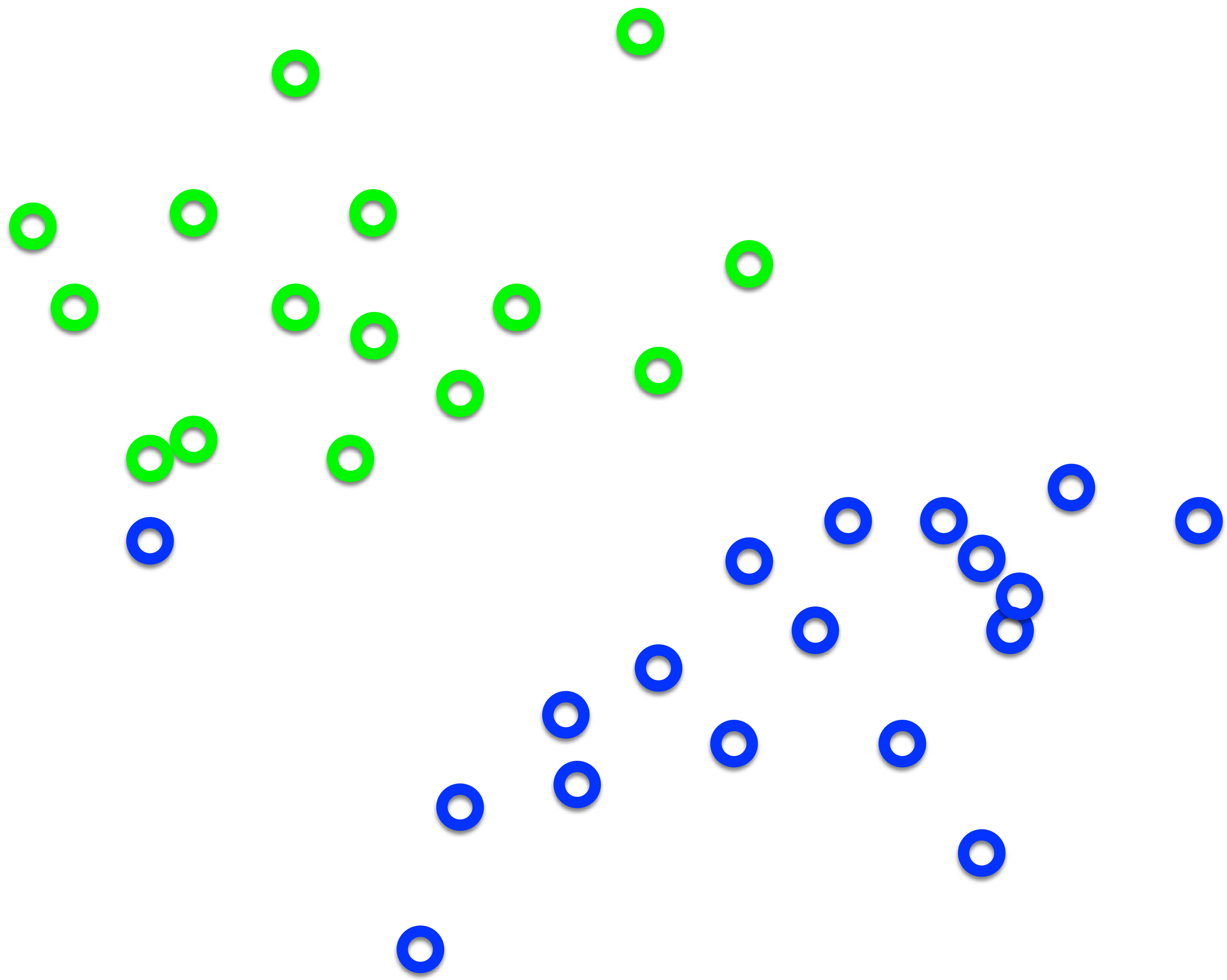
Constraints

This is a convex quadratic programming (QP) problem

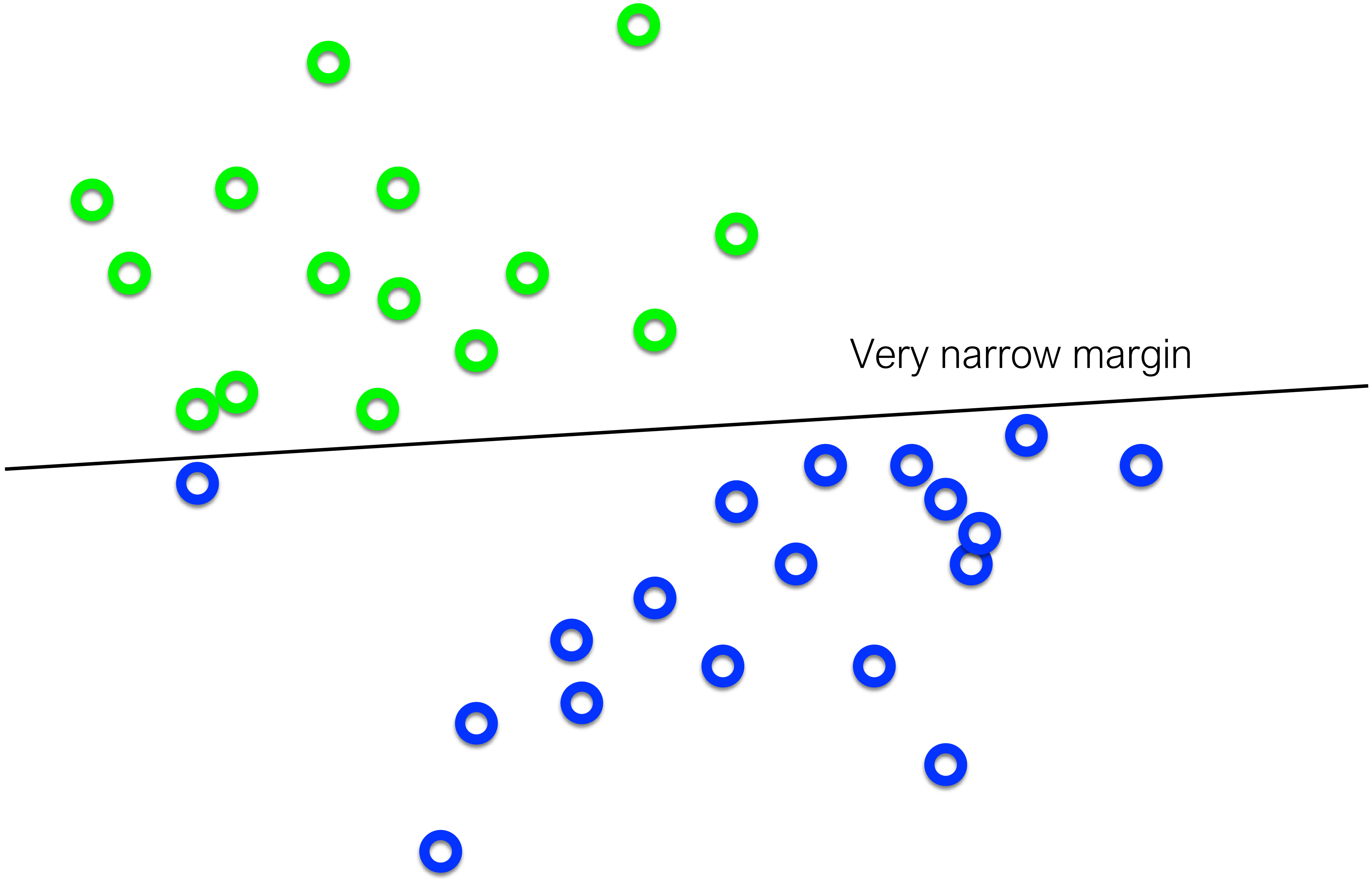
(a unique solution exists)

‘soft’ margin

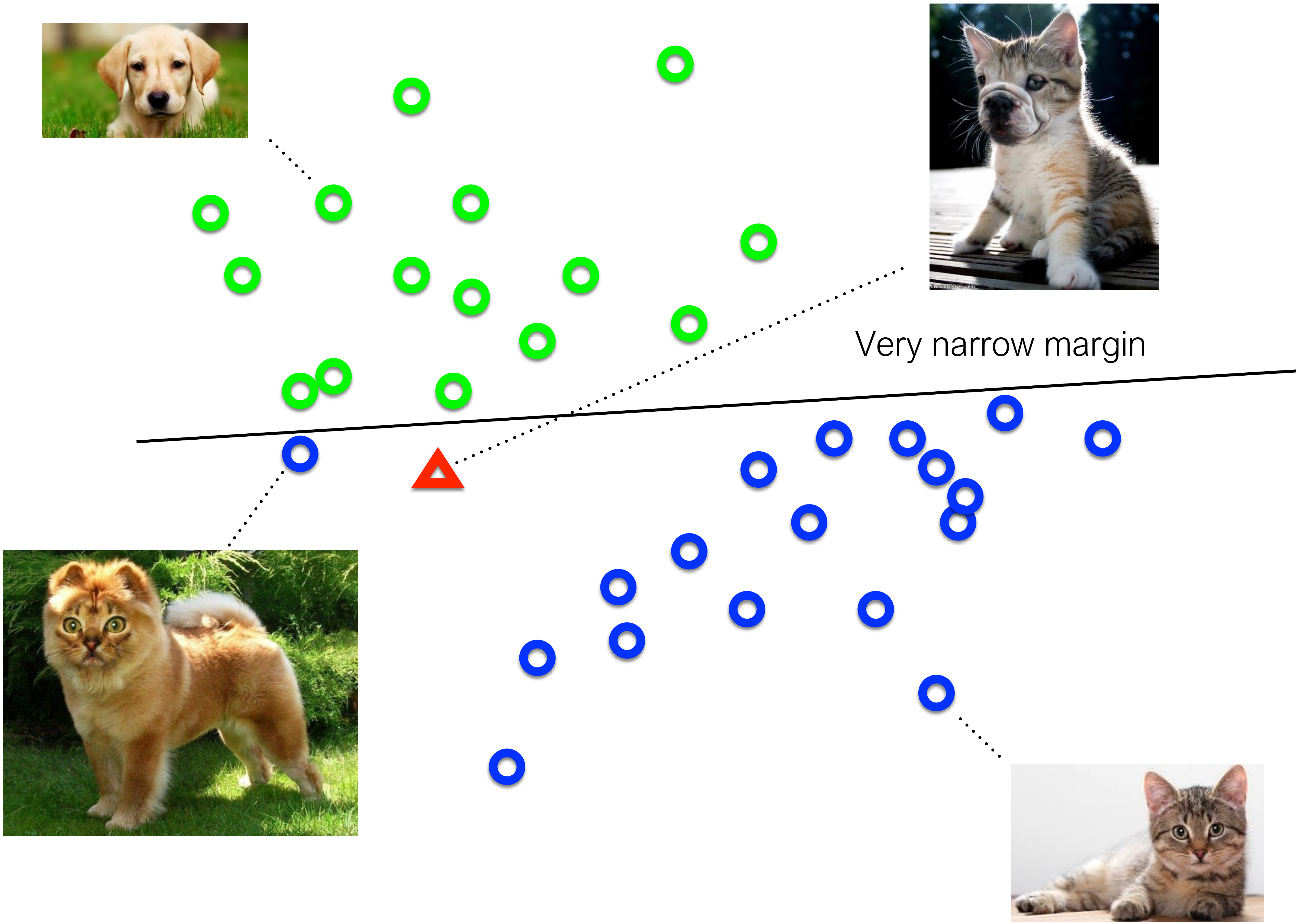
What's the best **w**?



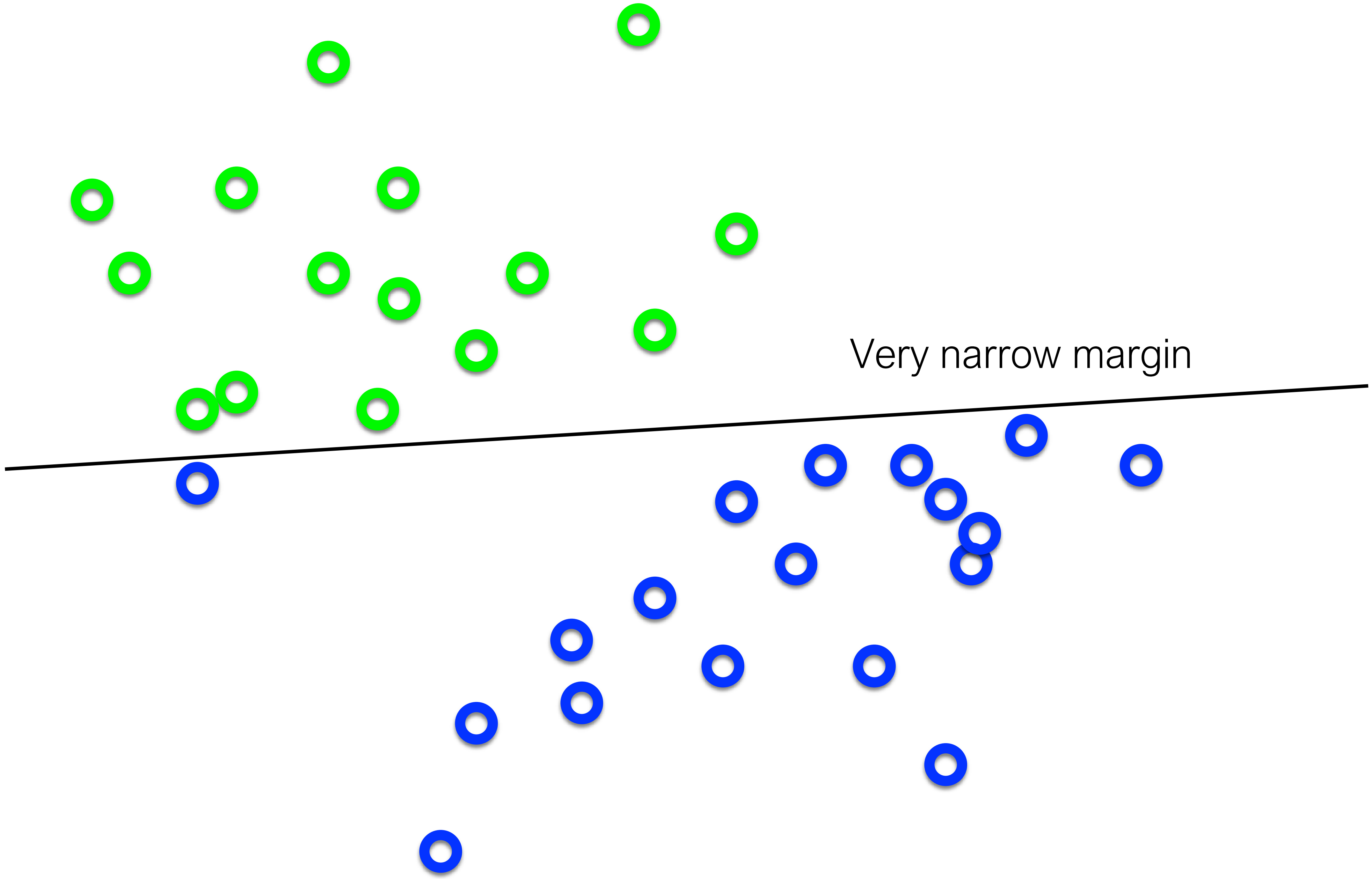
What’s the best **w**?



Separating cats and dogs

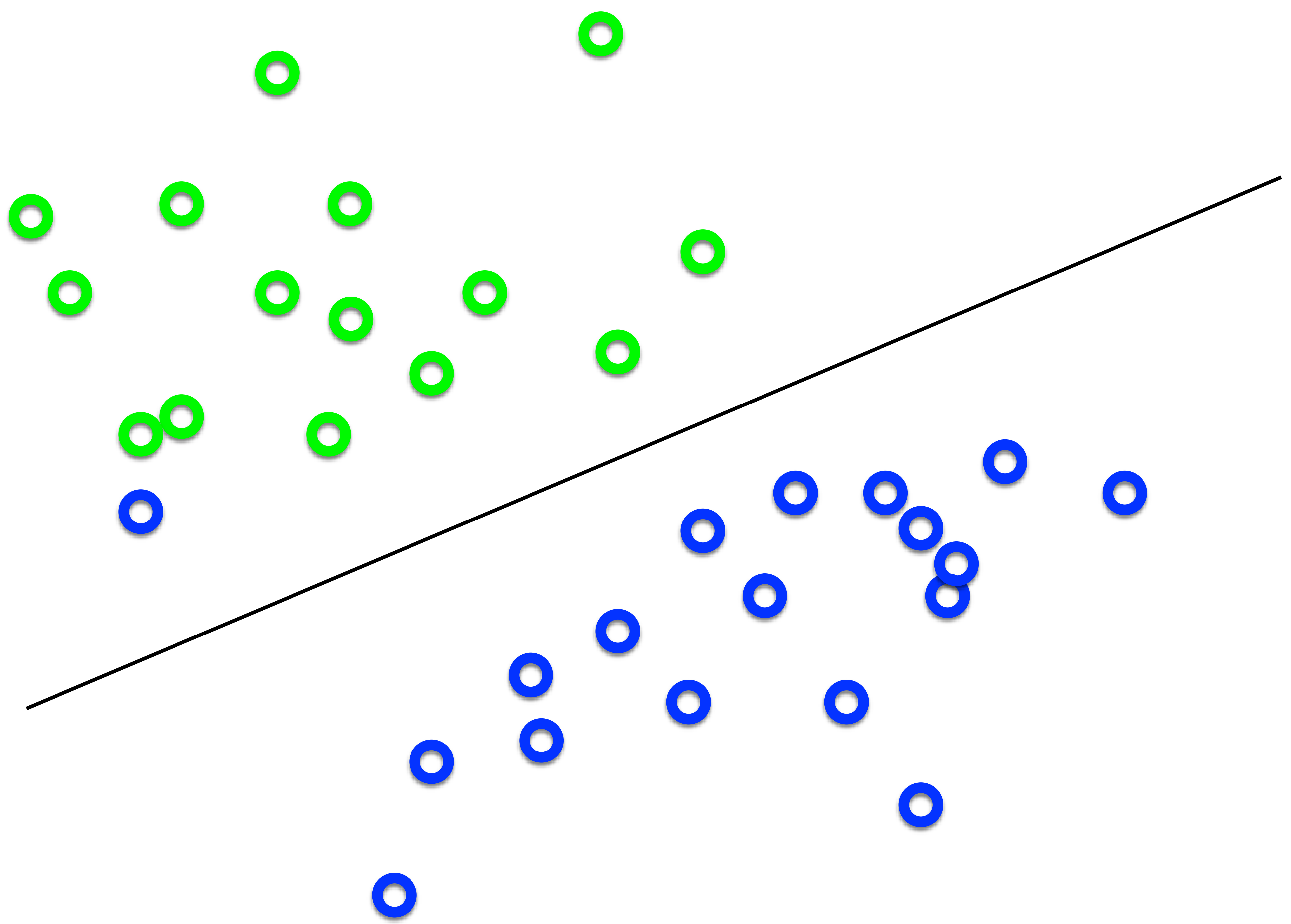


What's the best **w**?



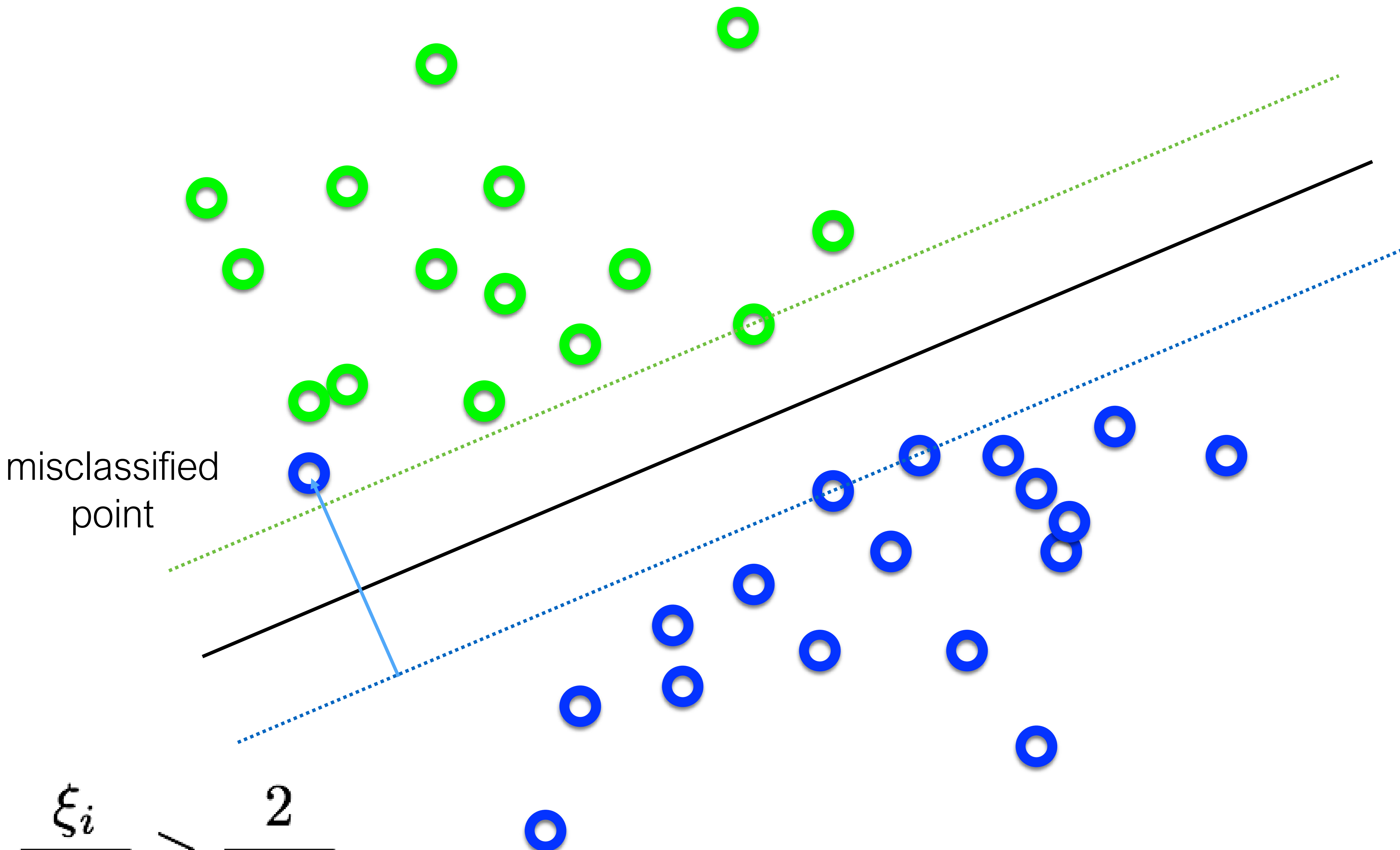
Intuitively, we should allow for some misclassification if we can get more robust classification

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

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

for $i = 1, \ldots, N$

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

‘soft’ margin

objective

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

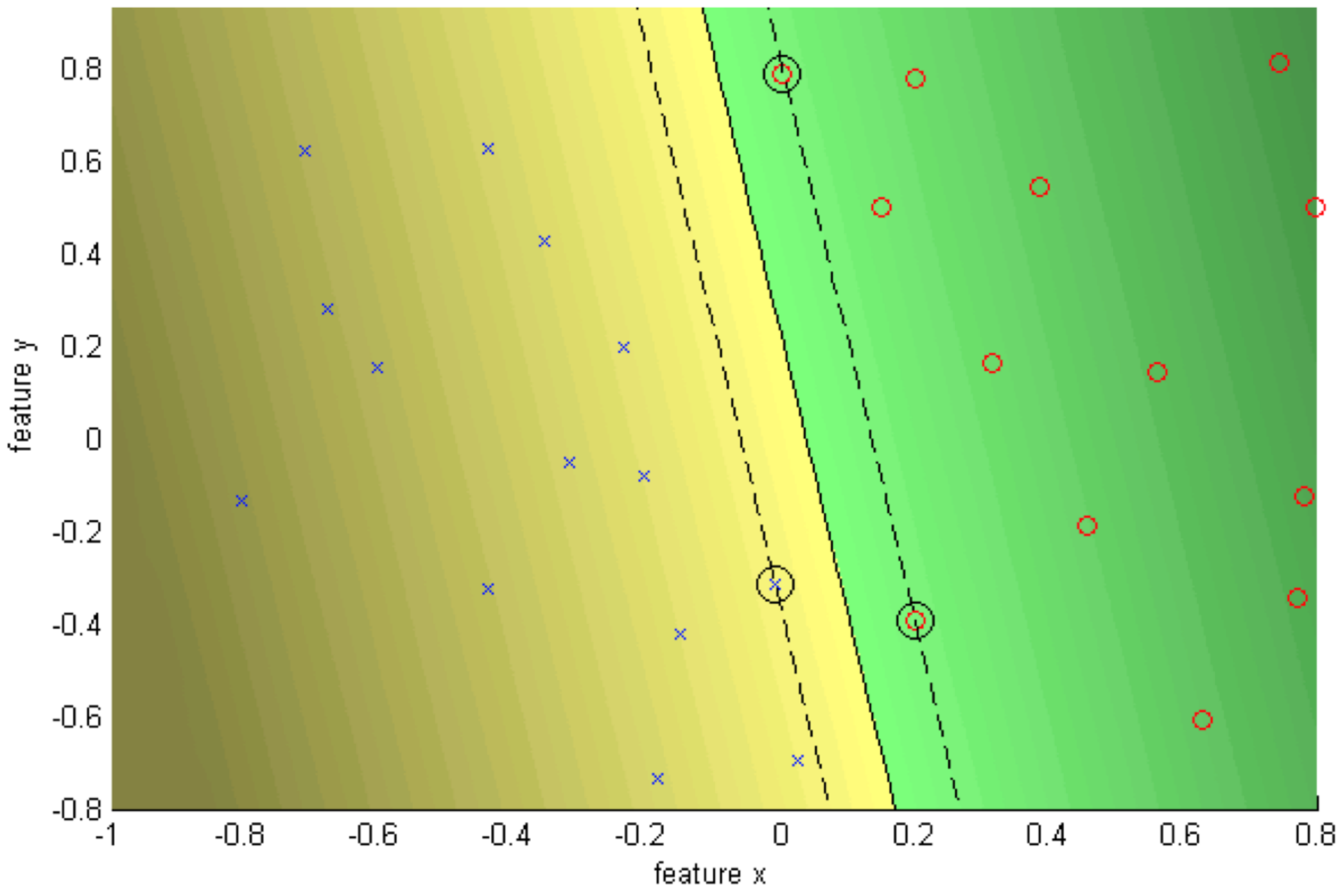
subject to

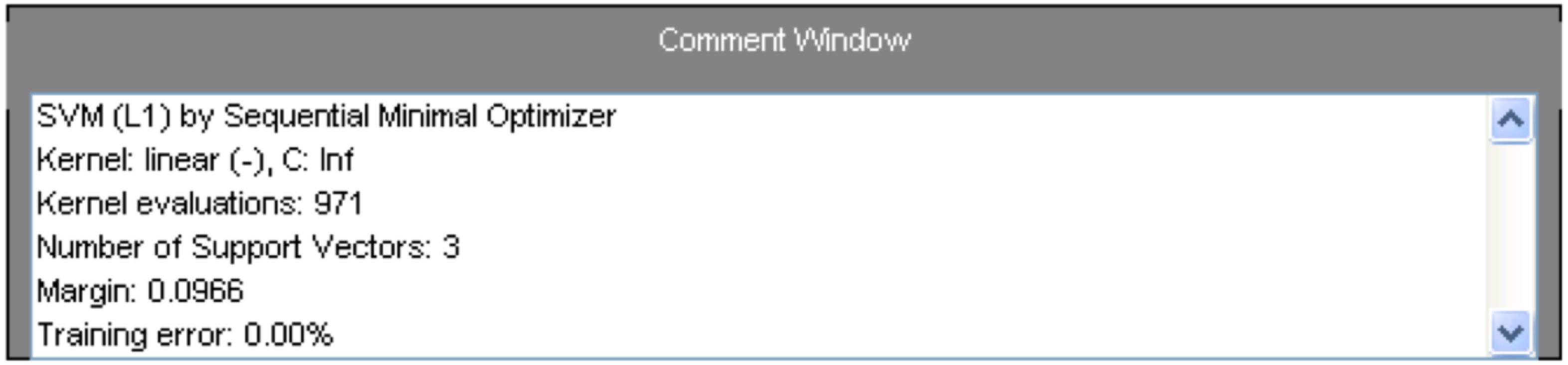
$$y_i(\boldsymbol{w}^\top \boldsymbol{x}_i + b) \geq 1 - \xi_i$$

for $i = 1, \ldots, N$

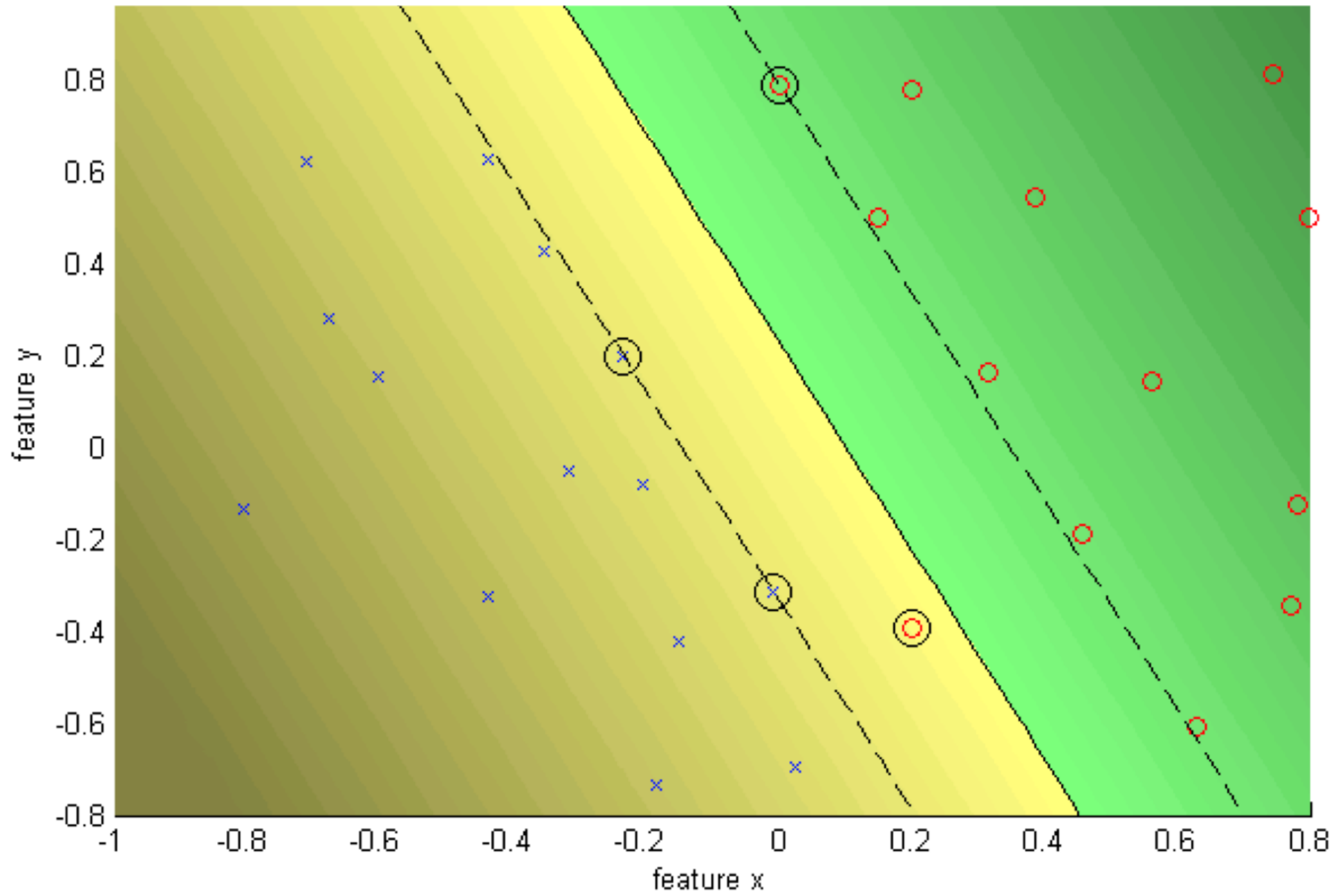
- Every constraint can be satisfied if slack is large
- C is a regularization parameter
    - Small C: ignore constraints (larger margin)
    - Big C: constraints (small margin)
- Still QP problem (unique solution)

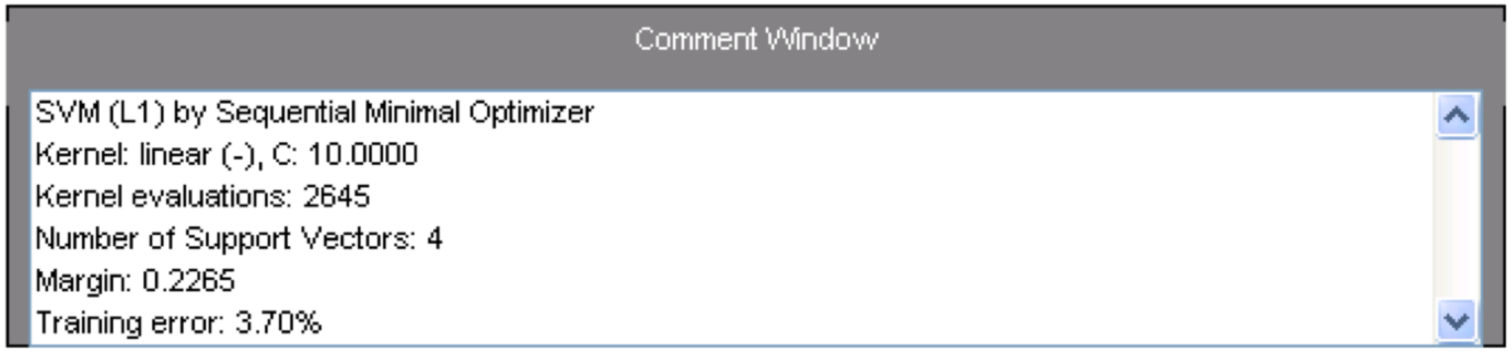
C = Infinity   hard margin





C = 10   soft margin





References

Basic reading:
- Szeliski, Chapter 14.