

AIN311

Fundamentals of Machine Learning

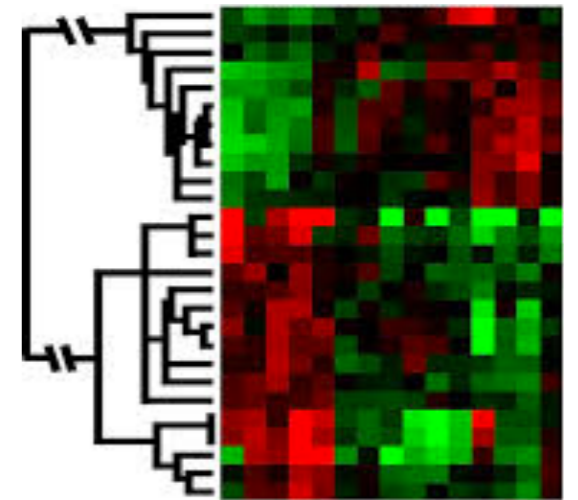
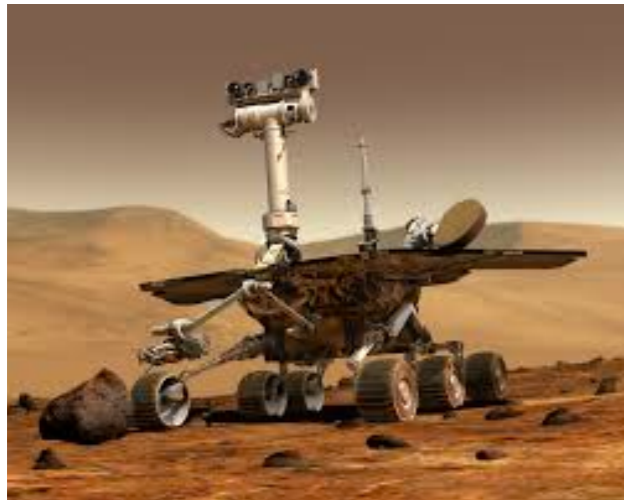
Lecture 2: Machine Learning by Examples, Nearest Neighbor Classifier



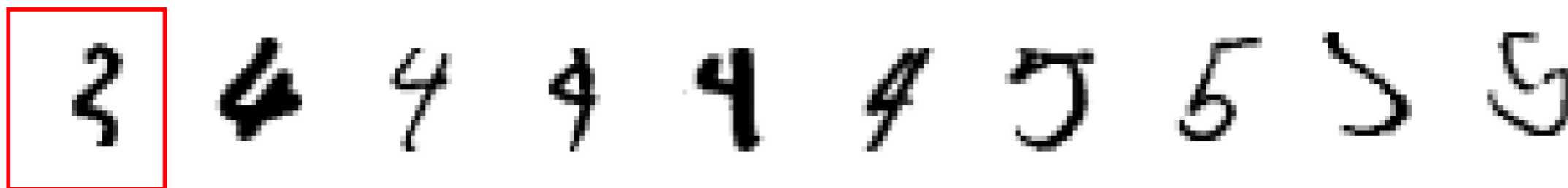
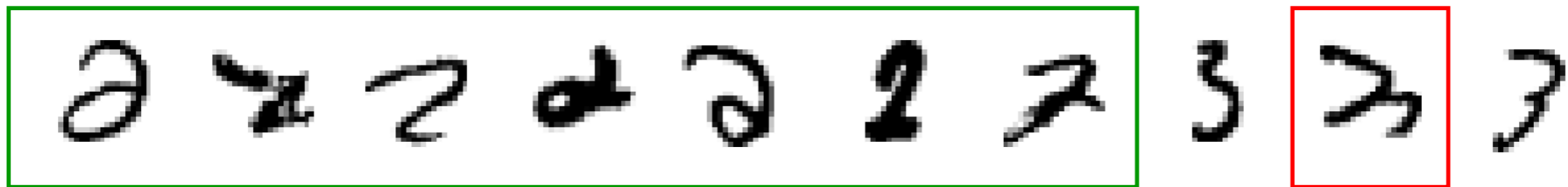
When Do We Use Machine Learning?

ML is used when:

- Human expertise does not exist (navigating on Mars)
- Humans can't explain their expertise (speech recognition)
- Models must be customized (personalized medicine)
- Models are based on huge amounts of data (genomics)



A classic example of a task that requires machine learning: It is very hard to say what makes a 2

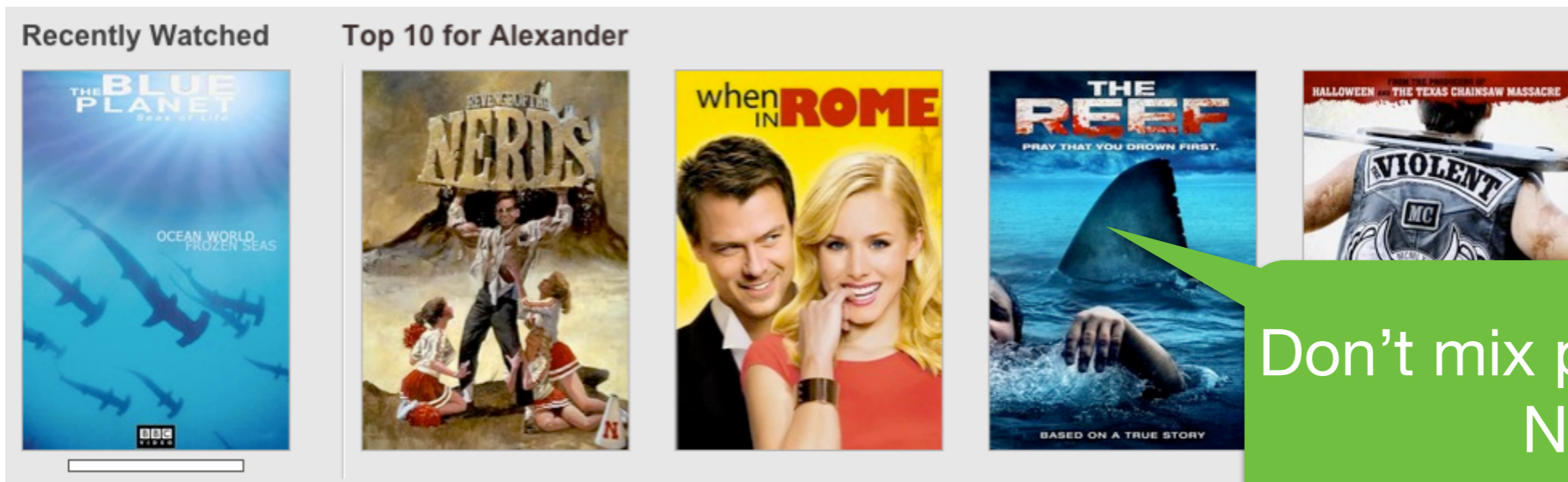


Machine Learning (by examples)

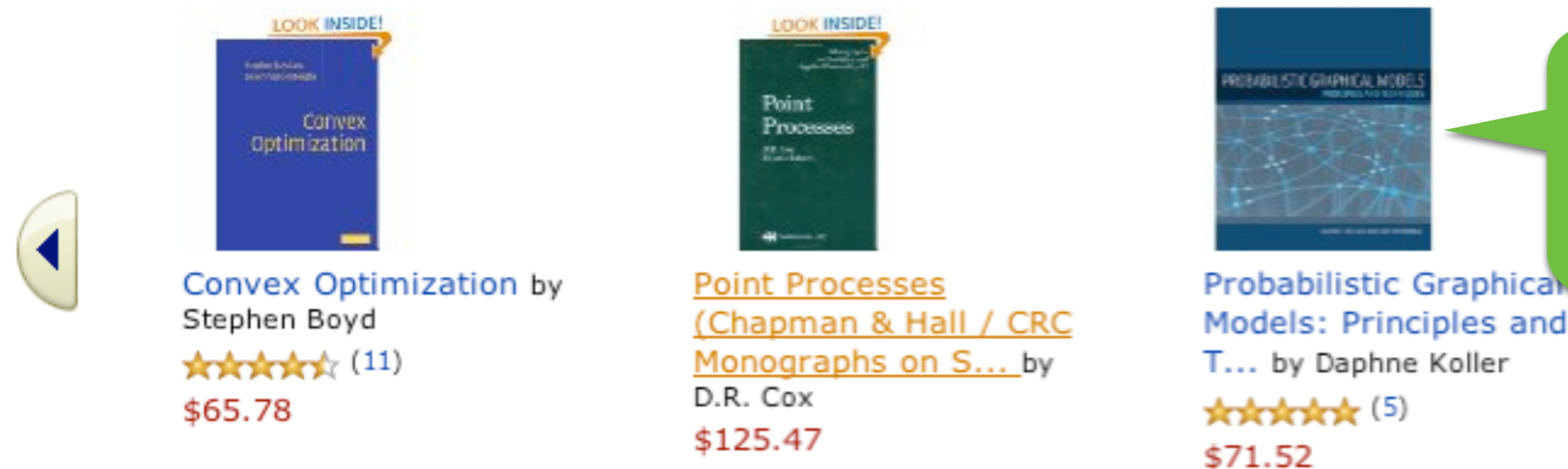
Pose Estimation



Collaborative Filtering



Customers Who Bought This Item Also Bought



Collaborative Filtering



BUSINESS INSIDER

RETAIL

Amazon is being forced to review its website after it reportedly recommended shoppers buy items that can create explosives

Kate Taylor

Sep. 20, 2017, 11:51 AM 6,591

FACEBOOK

LINKEDIN

TWITTER

EMAIL

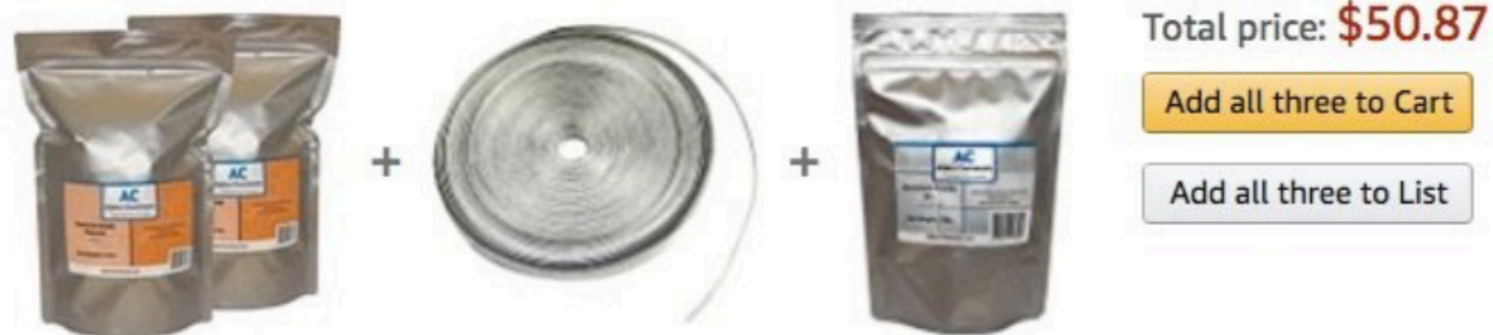
PRINT

Should be careful

Amazon is doing some self-examination after its website suggested customers purchase potentially dangerous groupings of products.

On Wednesday, Amazon told Reuters it was "reviewing its website" after the UK's Channel 4 News reported that the e-commerce giant's algorithm suggests that shoppers pair certain items with products that can be used to create homemade explosives.

Frequently bought together



- ✓ [blurred] \$25.99
- ✓ [blurred] \$3.89
- ✓ [blurred] \$20.99

This chemical compound's "frequently bought together" suggestions are the necessary ingredients to create a dangerous reaction. Amazon.com

Imitation Learning in Games

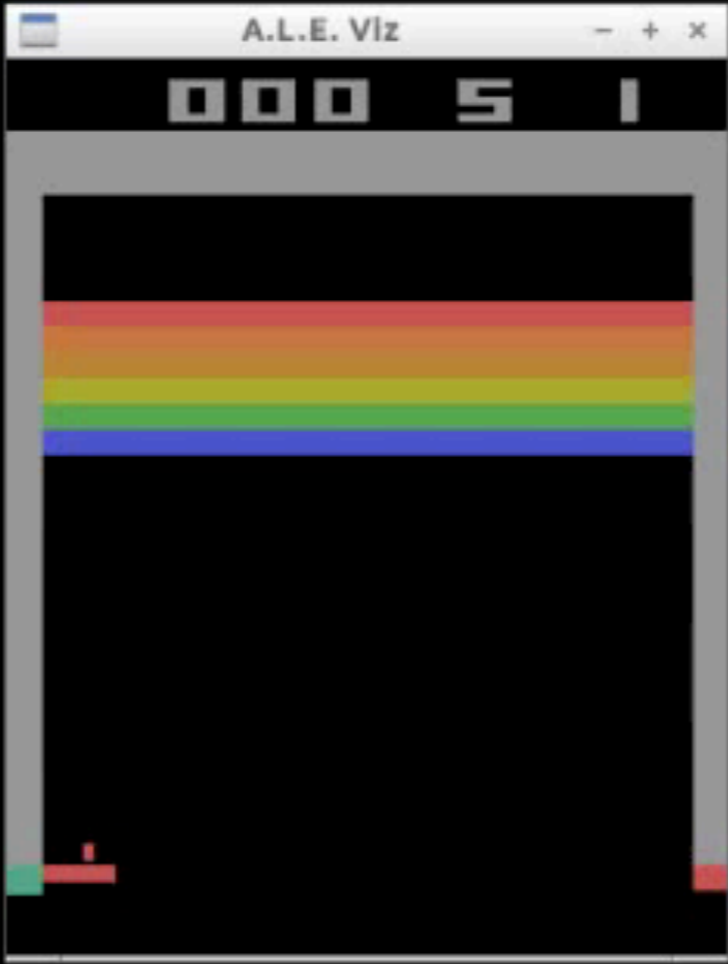


Avatar learns from your behavior

Black & White
Lionsgate Studios

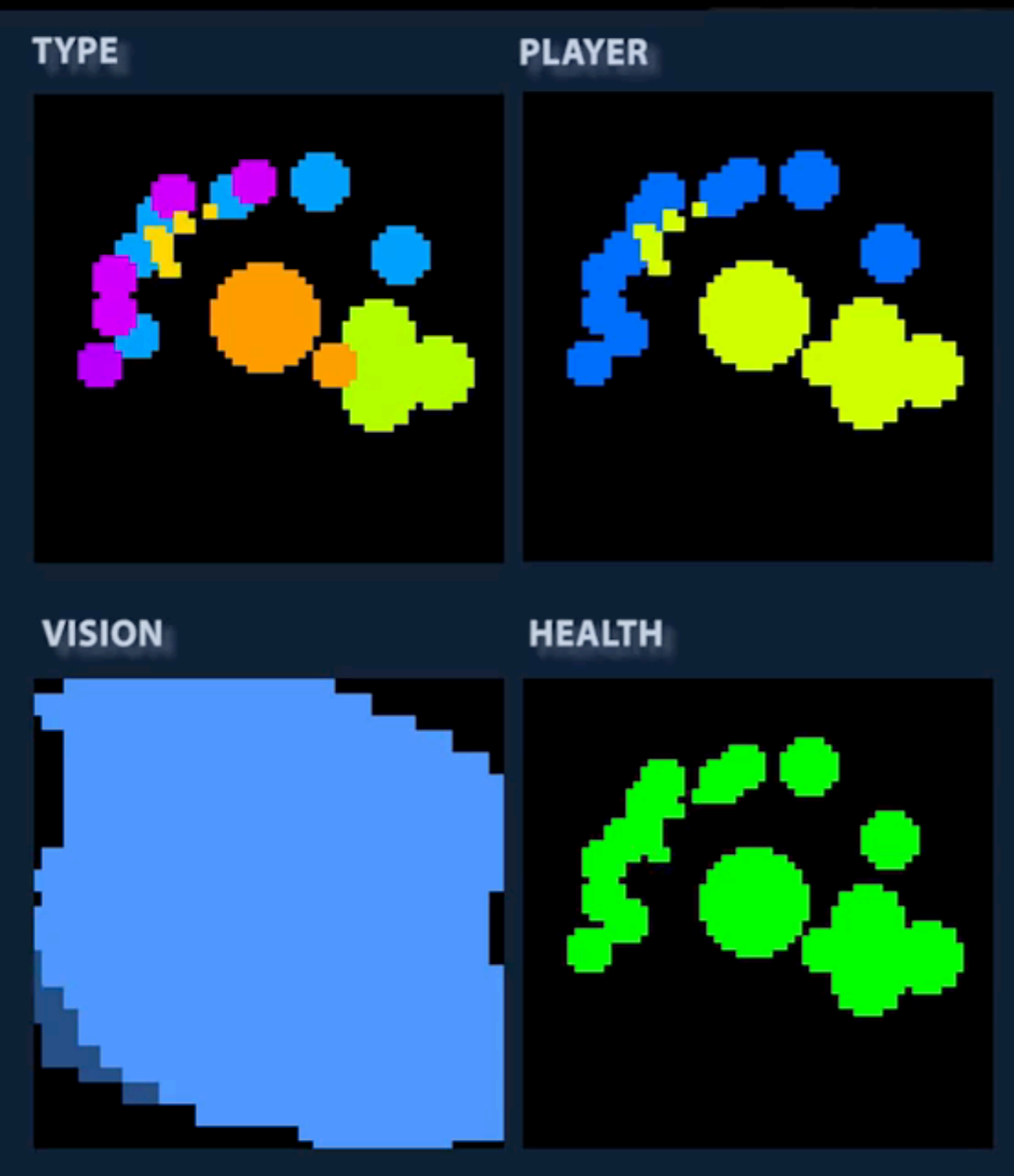
Reinforcement Learning

```
Game will be controlled through named FIFO pipes.  
Size 160-210  
OK  
<type 'str'> 67200  
<type 'numpy.ndarray'> 84  
S: 1 A: 0 R: 0 D: 0  
Start  
  
action: 1  
S: 2 A: 1 R: 1 D: 0  
Reward 0  
  
action: 1  
S: 3 A: 2 R: 2 D: 0  
Reward 0  
  
action: 1  
S: 4 A: 3 R: 3 D: 0  
Reward 0  
  
action NEURALNET: 3  
S: 5 A: 4 R: 4 D: 1  
Reward 0  
  
action NEURALNET: 3  
S: 6 A: 5 R: 5 D: 2  
Reward 0  
  
action NEURALNET: 0  
S: 7 A: 6 R: 6 D: 3  
Reward 0  
  
action NEURALNET: 3  
S: 8 A: 7 R: 7 D: 4  
Reward 0  
  
action NEURALNET: 0  
S: 9 A: 8 R: 8 D: 5  
Reward 0  
  
action NEURALNET: 3
```



<https://www.youtube.com/watch?v=IleRKHsJBJ0>

Reinforcement Learning



Spam Filtering

+Alex Search Images Maps Play YouTube News Gmail Drive Calendar More

Google

Alex Smola 0 + Share

Gmail More

ham

1-50 of 15,803

COMPOSE

Inbox (7,180)
Important
Sent Mail
Drafts (61)

<input type="checkbox"/>	<input type="star"/>	<input type="arrow"/>	Southwest Airlines	Your trip is around the corner! - You're all set for your San Jose trip! My Account View My Itinerary Online	2:12 pm
<input type="checkbox"/>	<input type="star"/>	<input type="arrow"/>	DiscountMags.com	\$3.99 Business & Finance Sale.. starts now! - Trouble Seeing This Email? View as Webpage STOP these e-r	12:03 pm
<input type="checkbox"/>	<input type="star"/>	<input checked="" type="arrow"/>	support, Alex (3)	Your order has shipped... - please send to the address below for an exchange remoteresremotes.com(exchange)	7:22 am
<input type="checkbox"/>	<input type="star"/>	<input type="arrow"/>	American Airlines AAdvan.	AAdvantage eSummary - January 2013 - VIEW IN WEB BROWSER >> http://americanairlines.ed10.net/r/JC	1:17 am
<input type="checkbox"/>	<input type="star"/>	<input checked="" type="arrow"/>	Taesup, Alex, Taesup (3)	Happy new year! - Hi Alex, Thanks for your condolence. I will arrive at Berkeley on 16th (wed) night. So, I car	Jan 11

+Alex Search Images Maps Play YouTube News Gmail Drive Calendar More

Google

Alex Smola 0 + Share

Gmail More

spam

1-50 of 244

COMPOSE

Inbox (7,180)
Important
Sent Mail
Drafts (61)
All Mail

▶ Circles

▼ [Gmail]
Done (1,006)
[Imap]/Drafts
[Imap]/Sent
alex.smola@yah...

Search people...
Barak Pearlmut...

Delete all spam messages now (messages that have been in Spam more than 30 days will be automatically deleted)

<input type="checkbox"/>	<input type="star"/>	<input checked="" type="arrow"/>	maee	(Ei&ISTP Index)2013机械与自动化工程国际会议征文: [alex.smola@gmail.com] - 尊敬的老师, 您好: 机械与	Jan 11
<input type="checkbox"/>	<input type="star"/>	<input type="arrow"/>	Dear Valued Customers,	Low Interest Rate Loan - Dear Valued Customers, Do you need a loan or funding for any of the following reas	Jan 11
<input type="checkbox"/>	<input type="star"/>	<input type="arrow"/>	garjeti	Call for Research Papers - GLOBAL ADVANCED RESEARCH JOURNAL OF ENGINEERING, TECHNOLOG	Jan 11
<input type="checkbox"/>	<input type="star"/>	<input checked="" type="arrow"/>	Steven Cooke	Congratulations Alex, \$150 awaits you - Alex: IMPORTANT - NOTICE OF WINNINGS Please make sure yo	Jan 11
<input type="checkbox"/>	<input type="star"/>	<input checked="" type="arrow"/>	paper18	【2013-1-15截稿】 【2013年机电与控制工程亚太地区学术研讨会APCMCE 2013】 【EI】 【香港】 【不参-不要.	Jan 10
<input type="checkbox"/>	<input type="star"/>	<input checked="" type="arrow"/>	First-Class Mail Service	Tracking ID (G)BGD35 849 603 4893 4550 - Fed Ex Order: JN-3339-28981768 Order Date: Thursday, 3 Janua	Jan 10
<input type="checkbox"/>	<input type="star"/>	<input type="arrow"/>	garjeti	Call for Research Papers - GLOBAL ADVANCED RESEARCH JOURNAL OF ENGINEERING, TECHNOLOG	Jan 10
<input type="checkbox"/>	<input type="star"/>	<input checked="" type="arrow"/>	Candy.Li	中层,不只当老板的代言人	Jan 9
<input type="checkbox"/>	<input type="star"/>	<input checked="" type="arrow"/>	Ronan Morgan	Ronan Morgan just sent you a personal message. - LinkedIn Ronan Morgan just sent you a private messag	Jan 9
<input type="checkbox"/>	<input type="star"/>	<input type="arrow"/>	RE/MAX®	2013 Valueable Offer! - Hello Friend, RE/MAX® has issued 2013 valuable property offer in your resident from	Jan 9
<input type="checkbox"/>	<input type="star"/>	<input type="arrow"/>	newsletter	newsletter WWW2013 - Newsletter 6 - See the Portuguese and Spanish version right after the English versior	Jan 9
<input type="checkbox"/>	<input type="star"/>	<input checked="" type="arrow"/>	CJCR editor	Chinese Journal of Cancer Research (CJCR) has been indexed by Pubmed and PMC - Click here if this e-mail	Jan 9
<input type="checkbox"/>	<input type="star"/>	<input type="arrow"/>	garjeti (2)	Call for Research Papers - GLOBAL ADVANCED RESEARCH JOURNAL OF ENGINEERING, TECHNOLOG	Jan 9
<input type="checkbox"/>	<input type="star"/>	<input checked="" type="arrow"/>	Wayne Smith	Wayne Smith has sent you a message - Linked In Wayne Smith just sent you a message Date: 1/09/2013 hi	Jan 9

Cheque Reading

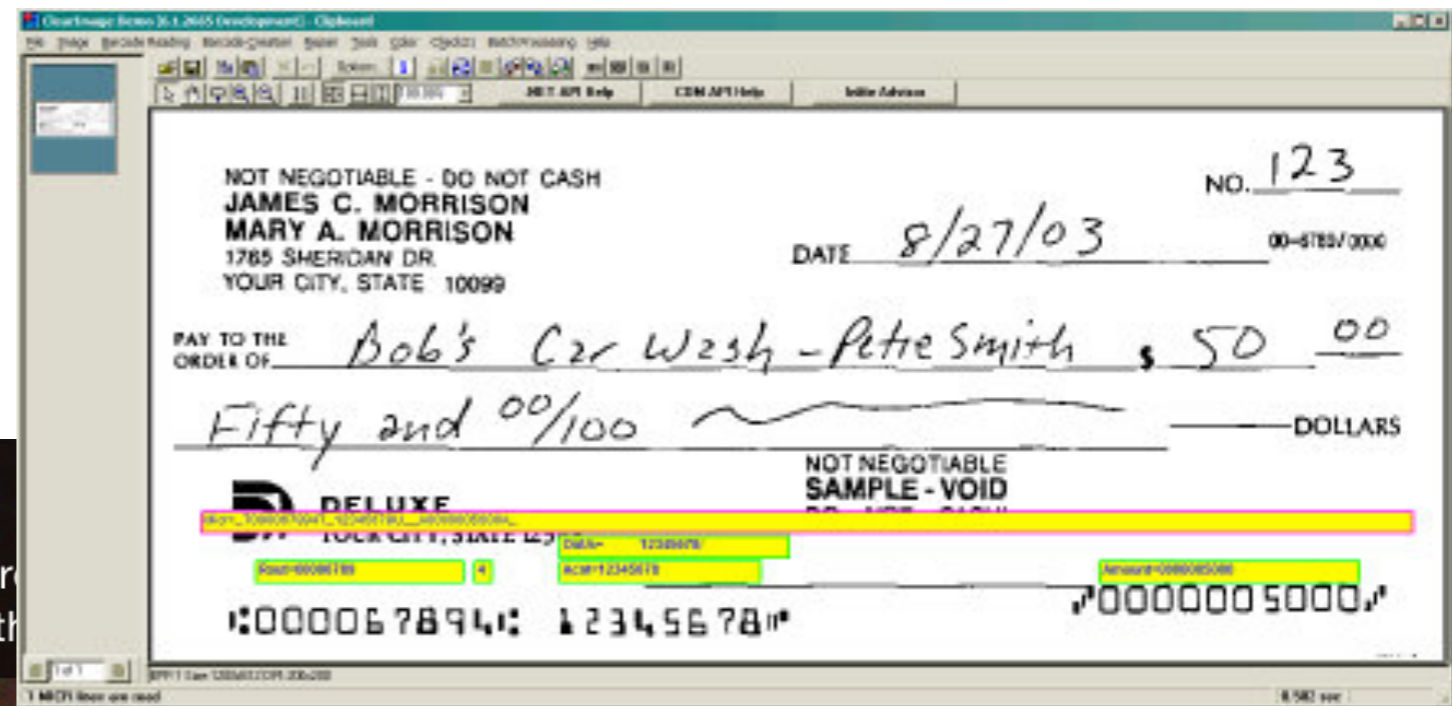
segment image

Photograph Front of Check

Place the check on a dark background in a well-lit area, hold the camera steady and align the check's edges with the frame.



Note: Fidelity cannot act on any written instructions



recognize
handwriting

Image Layout



- Raw set of images from several cameras
- Joint layout based on image similarity

Search Ads

The image shows a Google search interface for the term "mesothelioma". The search bar contains the text "mesothelioma" and the name "Alex" is visible in the top right corner. Below the search bar, navigation tabs for "Web", "Images", "Maps", "Shopping", "News", "More", and "Search tools" are present. The search results are displayed below, starting with "About 10,600,000 results (0.25 seconds)".

Ads related to mesothelioma

- Mesothelioma Symptoms - Lung cancer from Asbestos.**
www.mesothelioma-lung-cancer.org/
It can take 20-30 years to develop
What Is It? Symptoms
Portal Entrance Treatments
- Mesothelioma Symptoms - 101 Facts about Mesothelioma.**
www.mesothelioma-answer.org/
By Anna Kaplan, M.D.
Free Mesothelioma Book - Nutrition Book - Free Mesothelioma DVDs - Asbestos
- Mesothelioma Diagnosis? - Get the money you deserve fast**
www.mesotheliomaclaimscenter.info/
File with **Mesothelioma** Claim Center
Mesothelioma Compensation Amounts - File a Mesothelioma Claim
- Mesothelioma - Wikipedia, the free encyclopedia**
en.wikipedia.org/wiki/Mesothelioma
Mesothelioma (or, more precisely, malignant **mesothelioma**) is a rare form of cancer that develops from transformed cells originating in the mesothelium, the ...
Signs and symptoms - Cause - Diagnosis - Screening
- Mesothelioma Cancer Alliance | The Authority on Asbestos Cancer**
www.mesothelioma.com/
Mesothelioma treatment, diagnosis and related information for patients and families. Legal options for those diagnosed with malignant **mesothelioma**.

Ads

- Mesothelioma compensation**
www.simmonsfirm.com/888-360-4189
Free Consultation with Lawyers that Focus on **Mesothelioma** Cases.
- Mesothelioma Compensation**
www.sokolovelaw.com/Call_Now
Mesothelioma Diagnosis? Get the Money You Deserve! [800-581-8243](tel:800-581-8243)
- Mesothelioma 800-582-0706**
- You Don't Have To Sue Anyone.**
\$30 Billion Asbestos Trust Fund
- Mesothelioma & Asbestos**
www.navy-veterans-mesothelioma.org/
Important info for Navy Vets.
Learn About **Mesothelioma** Claims
- Asbestos Exposure?**
www.mesotheliomalawfirm.com/
Mesothelioma victims are entitled

why these ads?

Self-Driving Cars

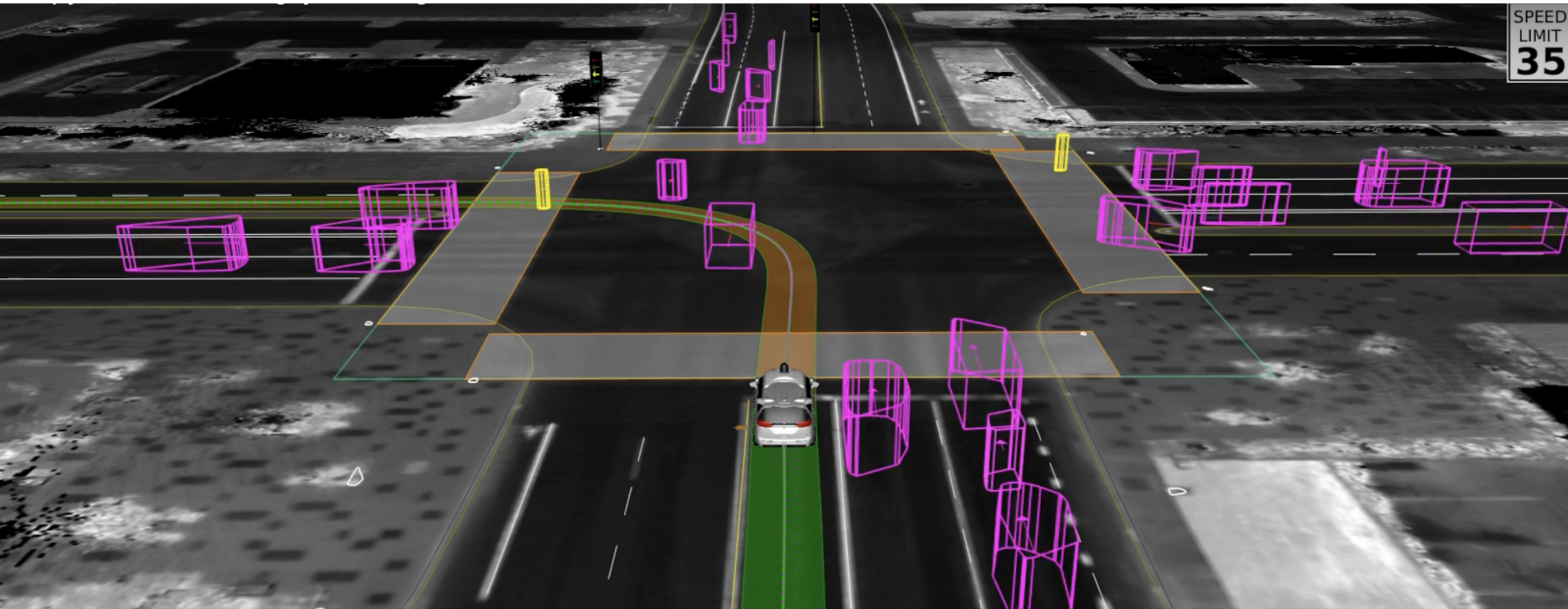
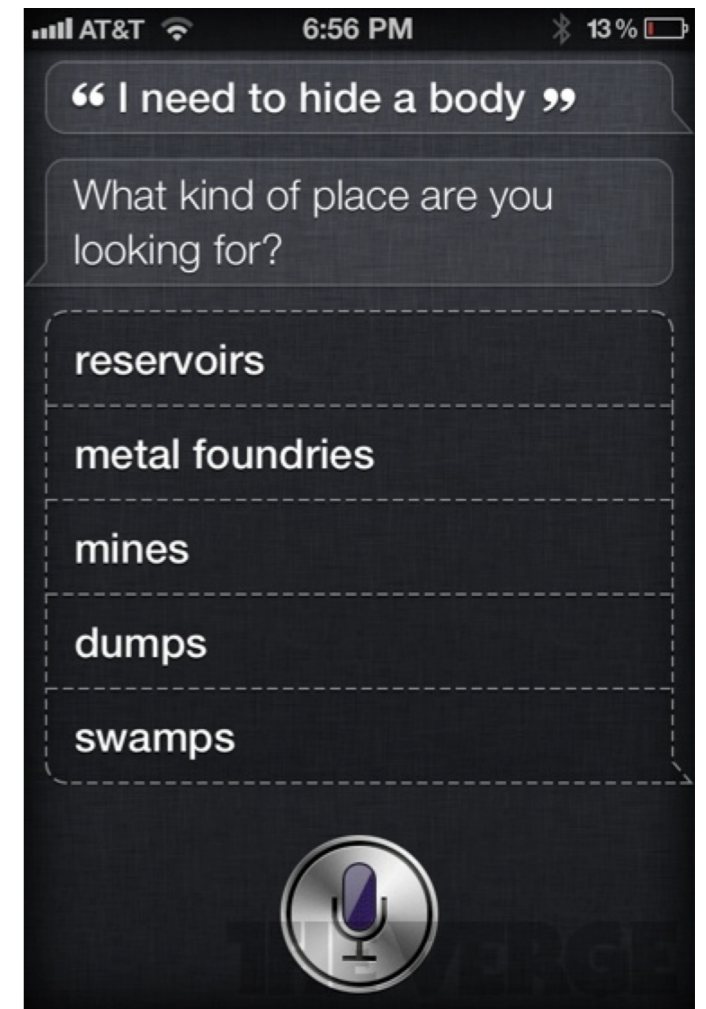
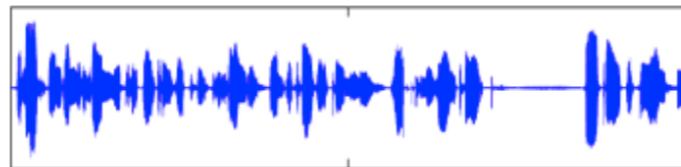


Image: <https://medium.com/waymo/simulation-how-one-flashing-yellow-light-turns-into-thousands-of-hours-of-experience-a7a1cb475565>

Speech Recognition

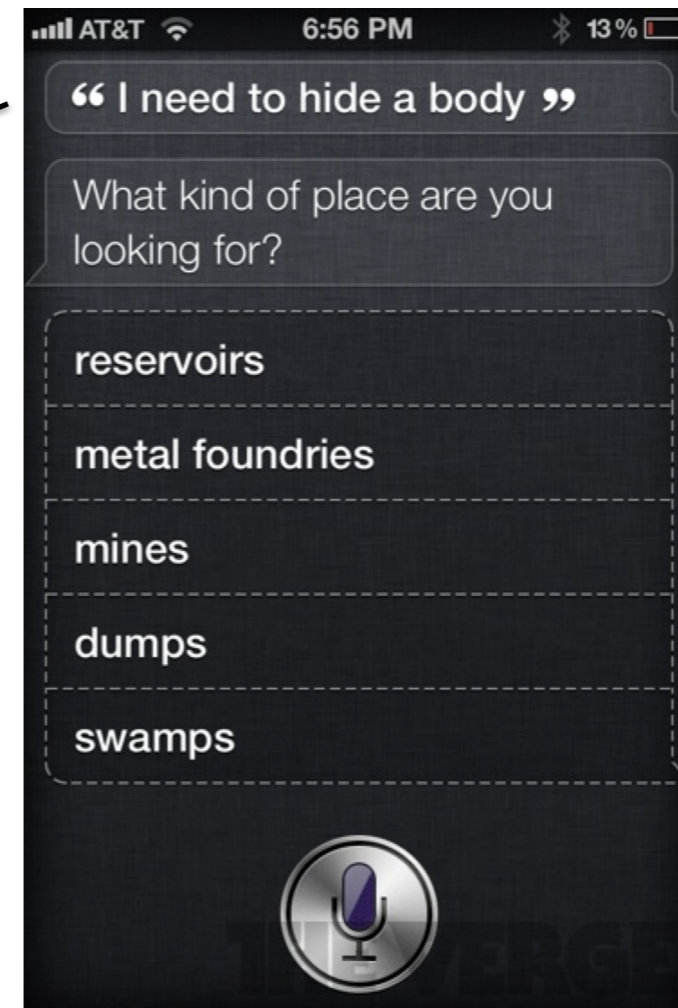
Given an audio waveform, robustly extract & recognize any spoken words



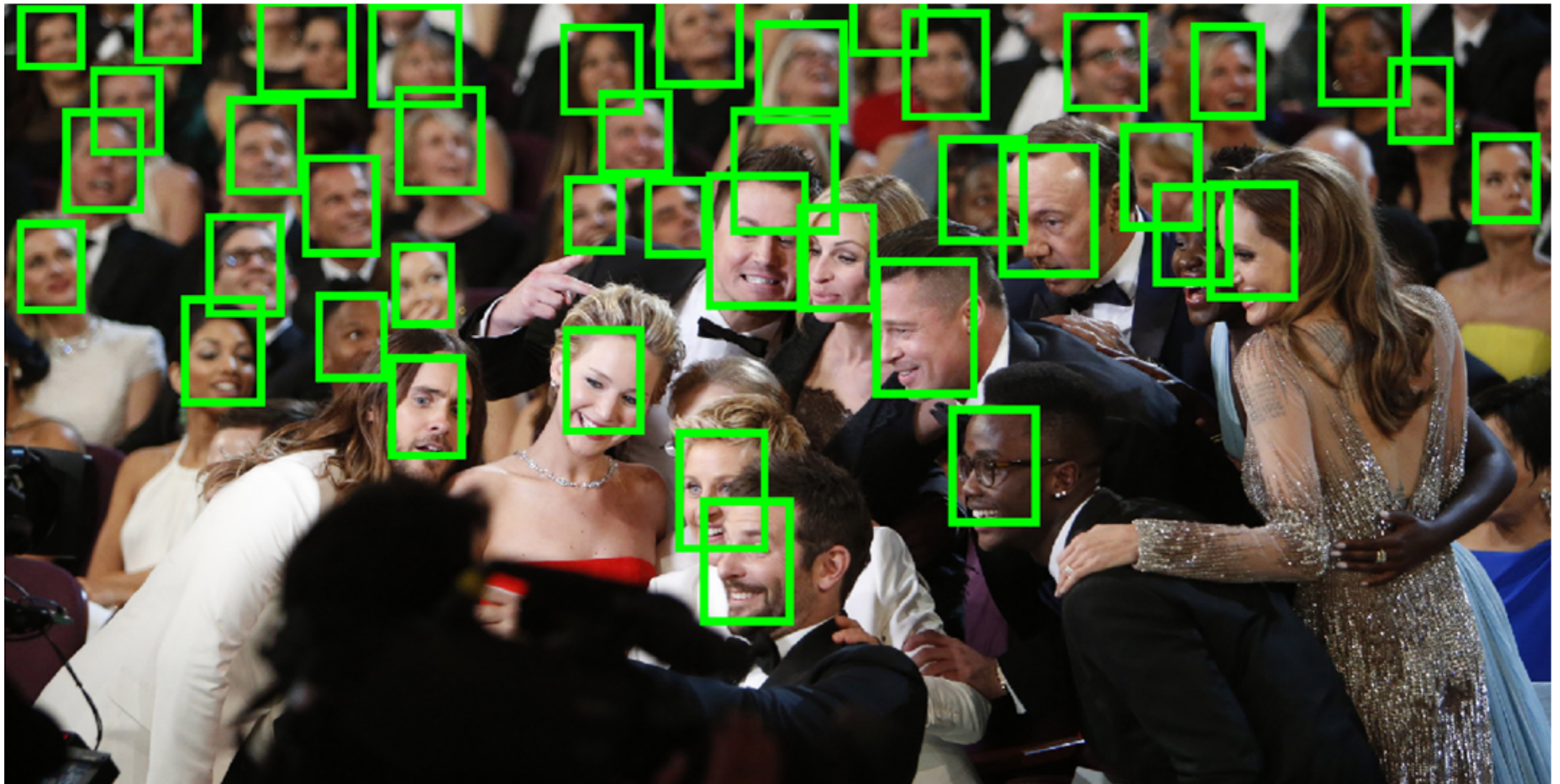
- Statistical models can be used to
 - Provide greater robustness to noise
 - Adapt to accent of different speakers
 - Learn from training

Natural Language Processing

I need to hide a body
noun, verb, preposition, ...

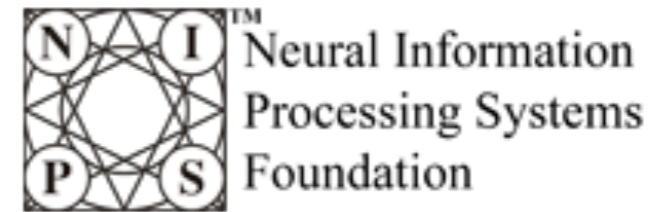
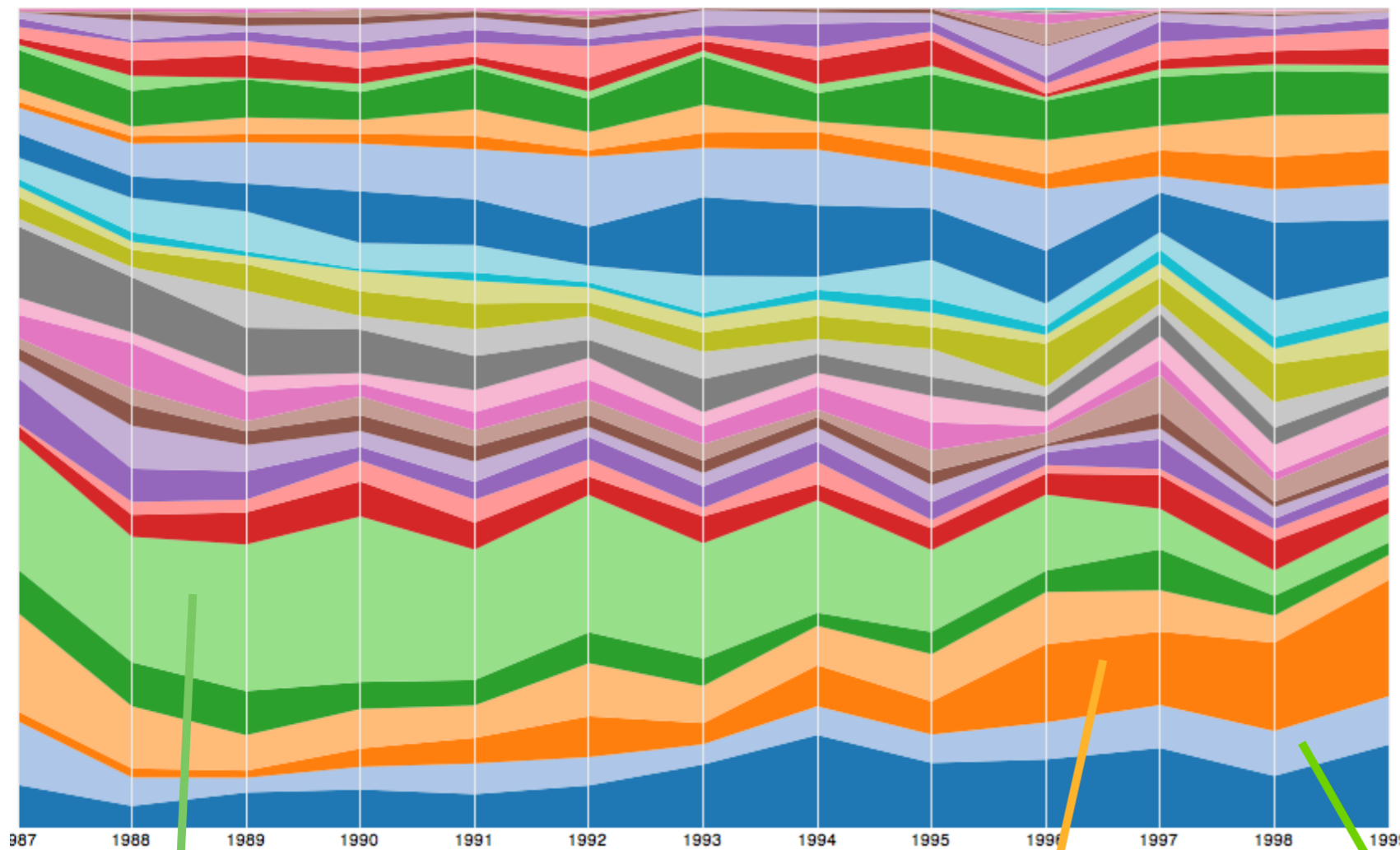


Face Detection

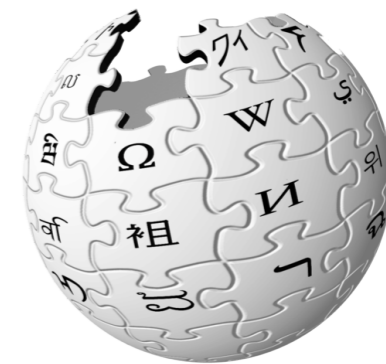


Yang et al., From Facial Parts Responses to Face Detection: A Deep Learning Approach, ICCV 2015

Topic Models of Text Documents



**The
New York
Times**



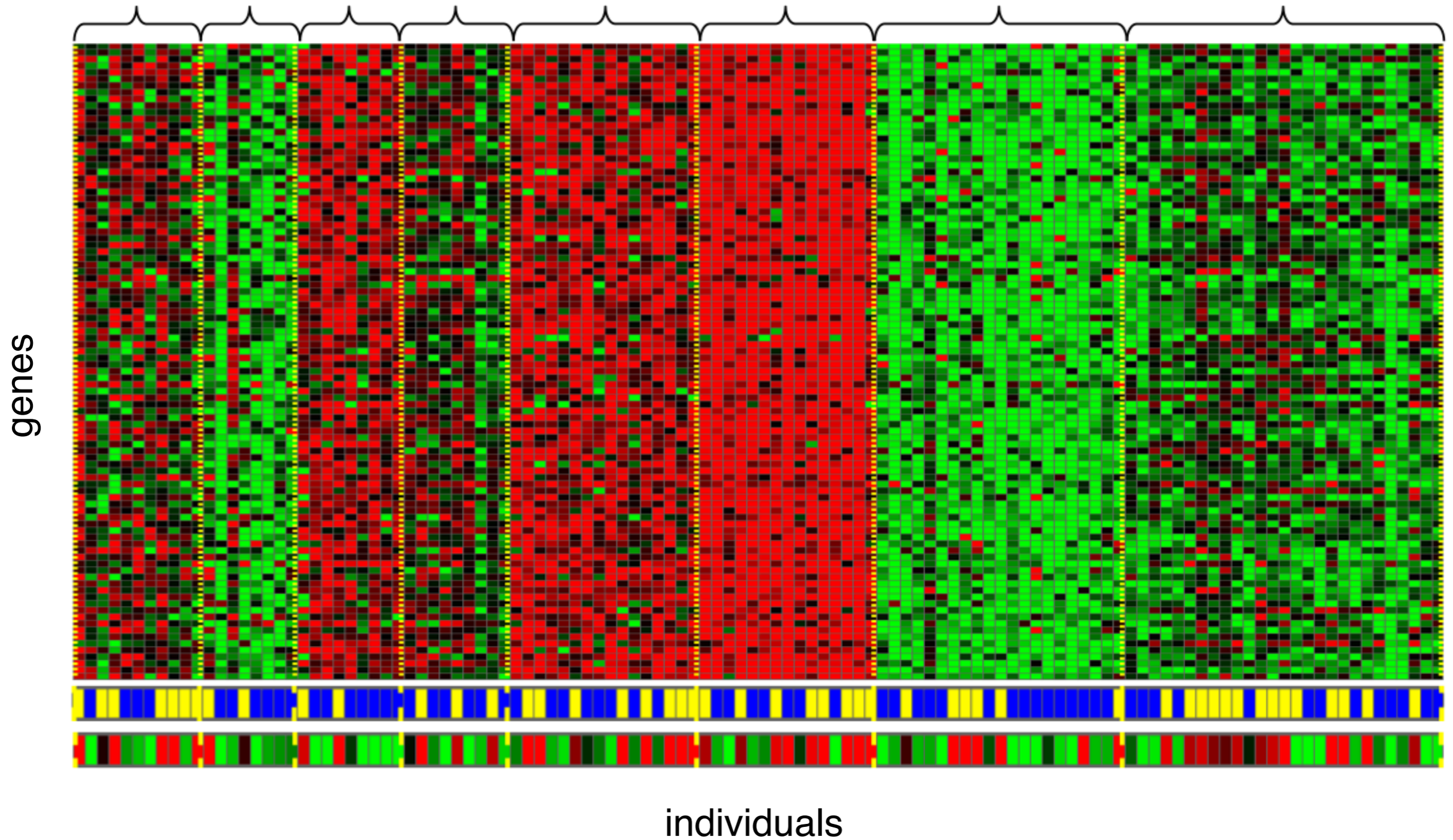
WIKIPEDIA
The Free Encyclopedia

weight neural figure inputs error unit
input layer **network** units
 weights training learning net hidden
 architecture set **networks** output
 number

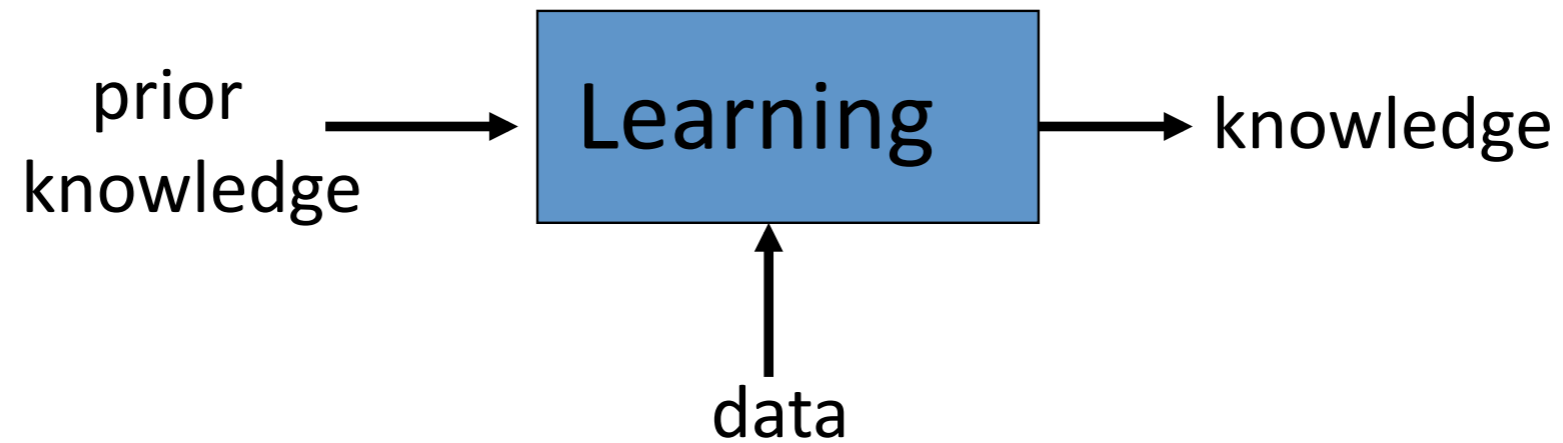
estimation density approach em
data probability **model** number set
 mixture gaussian posterior bayesian distribution
 figure parameters models
 log likelihood prior

cell responses motion
 field **cells**
 receptive **visual** tuning
 input model stimulus response
 neurons direction orientation stimuli
 spatial cortex cortical
 figure

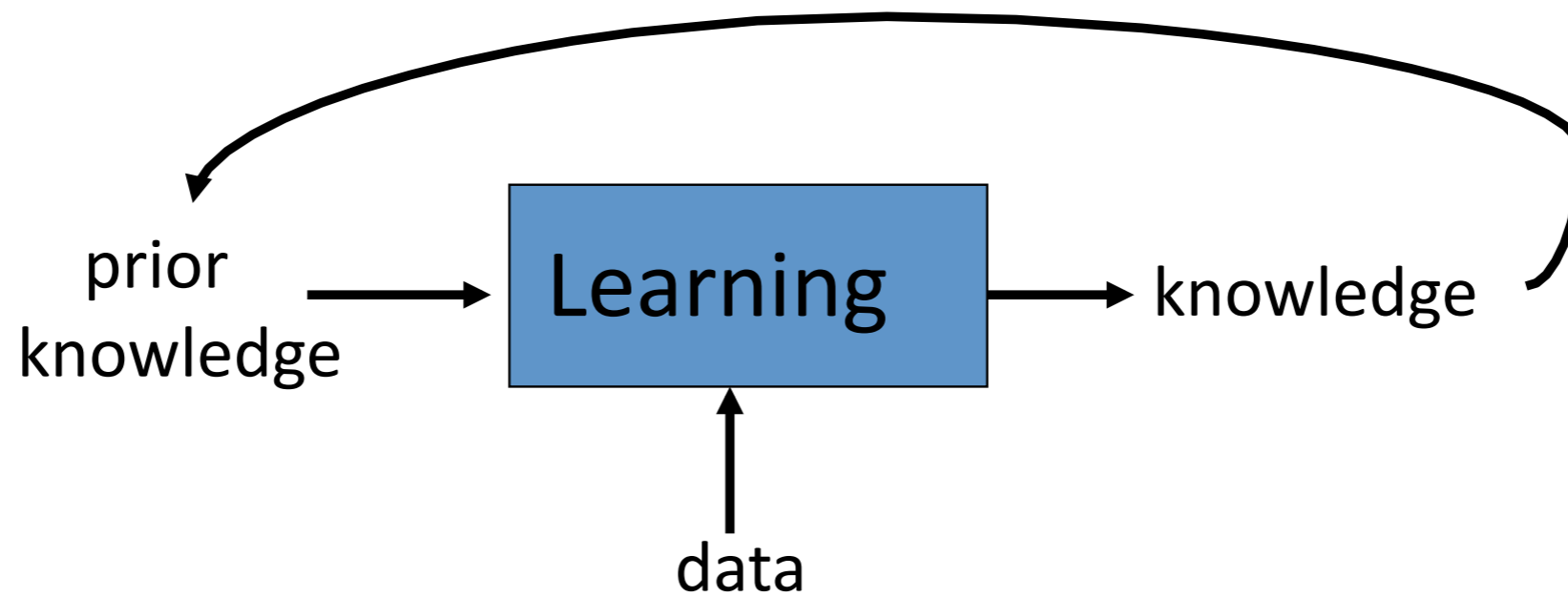
Genomics: group individuals by genetic similarity



Learning - revisited



Learning - revisited



Programming with Data

- Want adaptive robust and fault tolerant systems
- Rule-based implementation is (often)
 - difficult (for the programmer)
 - brittle (can miss many edge-cases)
 - becomes a nightmare to maintain explicitly
 - often doesn't work too well (e.g. OCR)
- Usually easy to obtain examples of what we want
IF x THEN DO y
- Collect many pairs (x_i, y_i)
- Estimate function f such that $f(x_i) = y_i$ (supervised learning)
- Detect patterns in data (unsupervised learning)

Objectives of Machine Learning

- **Algorithms:** design of efficient, accurate, and general learning algorithms to
 - deal with large-scale problems.
 - make accurate predictions (unseen examples).
 - handle a variety of different learning problems.
- **Theoretical questions:**
 - what can be learned? Under what conditions?
 - what learning guarantees can be given?
 - what is the algorithmic complexity?

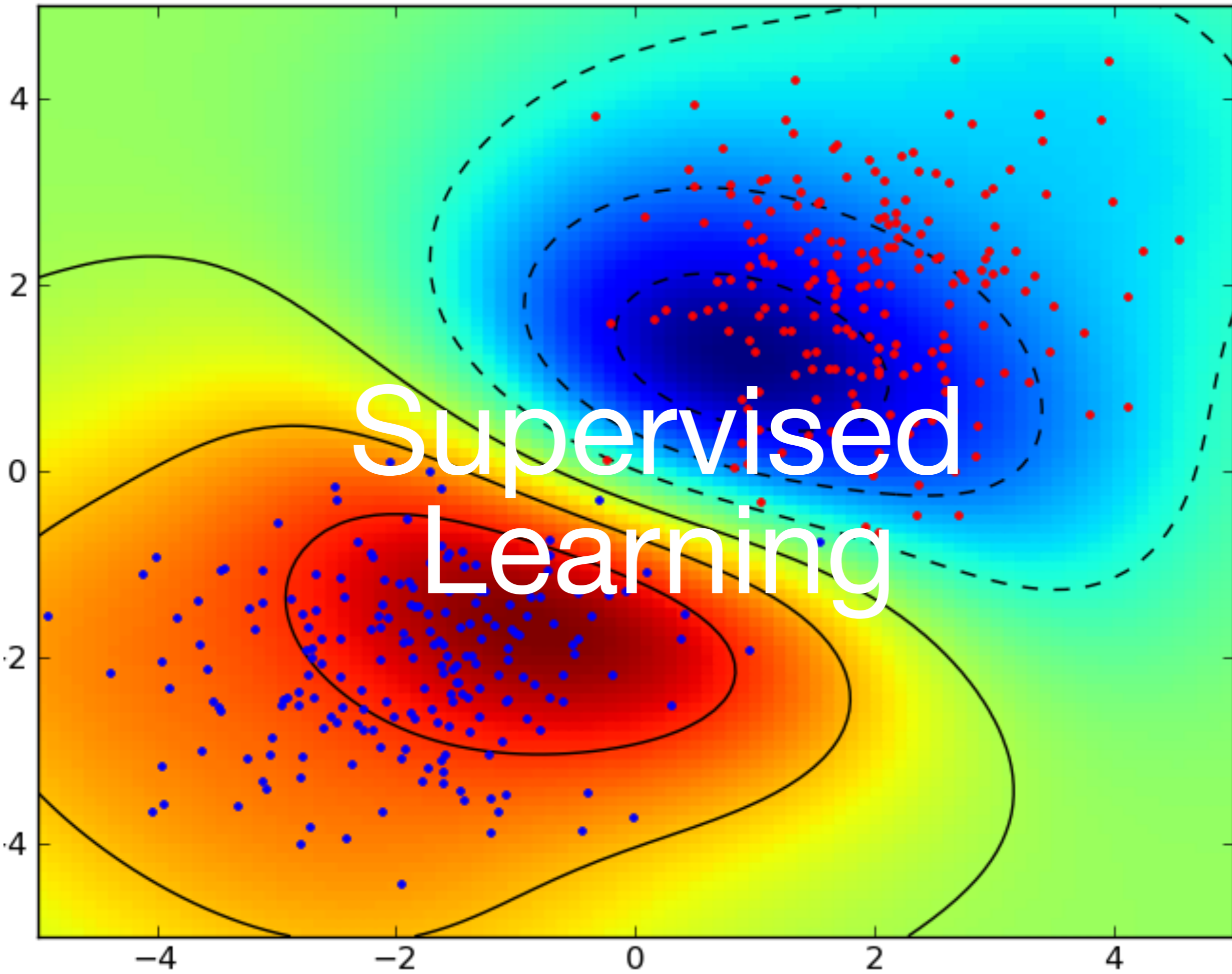
Definitions and Terminology

- **Example:** an object, instance of the data used.
- **Features:** the set of attributes, often represented as a vector, associated to an example (e.g., height and weight for gender prediction).
- **Labels:** in classification, category associated to an object (e.g., positive or negative in binary classification); in regression real value.
- **Training data:** data used for training learning algorithm (often labeled data).

Definitions and Terminology (cont'd.)

- **Test data:** data used for testing learning algorithm (unlabeled data).
- **Unsupervised learning:** no labeled data.
- **Supervised learning:** uses labeled data.
- **Weakly or semi-supervised learning:** intermediate scenarios.
- **Reinforcement learning:** rewards from sequence of action.

Supervised Learning



Supervised Learning

- **Binary classification**

Given x find y in $\{-1, 1\}$

- **Multicategory classification**

Given x find y in $\{1, \dots, k\}$

- **Regression**

Given x find y in \mathbb{R} (or \mathbb{R}^d)

- **Sequence annotation**

Given sequence $x_1 \dots x_l$ find $y_1 \dots y_l$

- **Hierarchical Categorization (Ontology)**

Given x find a point in the hierarchy of y (e.g. a tree)

- **Prediction**

Given x_t and $y_{t-1} \dots y_1$ find y_t

often with loss

$$l(y, f(x))$$

Binary Classification

+Alex Search Images Maps Play YouTube News

Google

Gmail

COMPOSE

Inbox (7,180)

Important

Sent Mail

Drafts (61)

- Southwest Airlines
- DiscountMags.com
- support, Alex (3)
- American Airlines AAdv...
- Taesup, Alex, Taesup (3)

+Alex Search Images Maps Play YouTube News

Google

in:spam

Gmail

COMPOSE

Inbox (7,180)

Important

Sent Mail

Drafts (61)

All Mail

Circles

[Gmail]

Done (1,006)

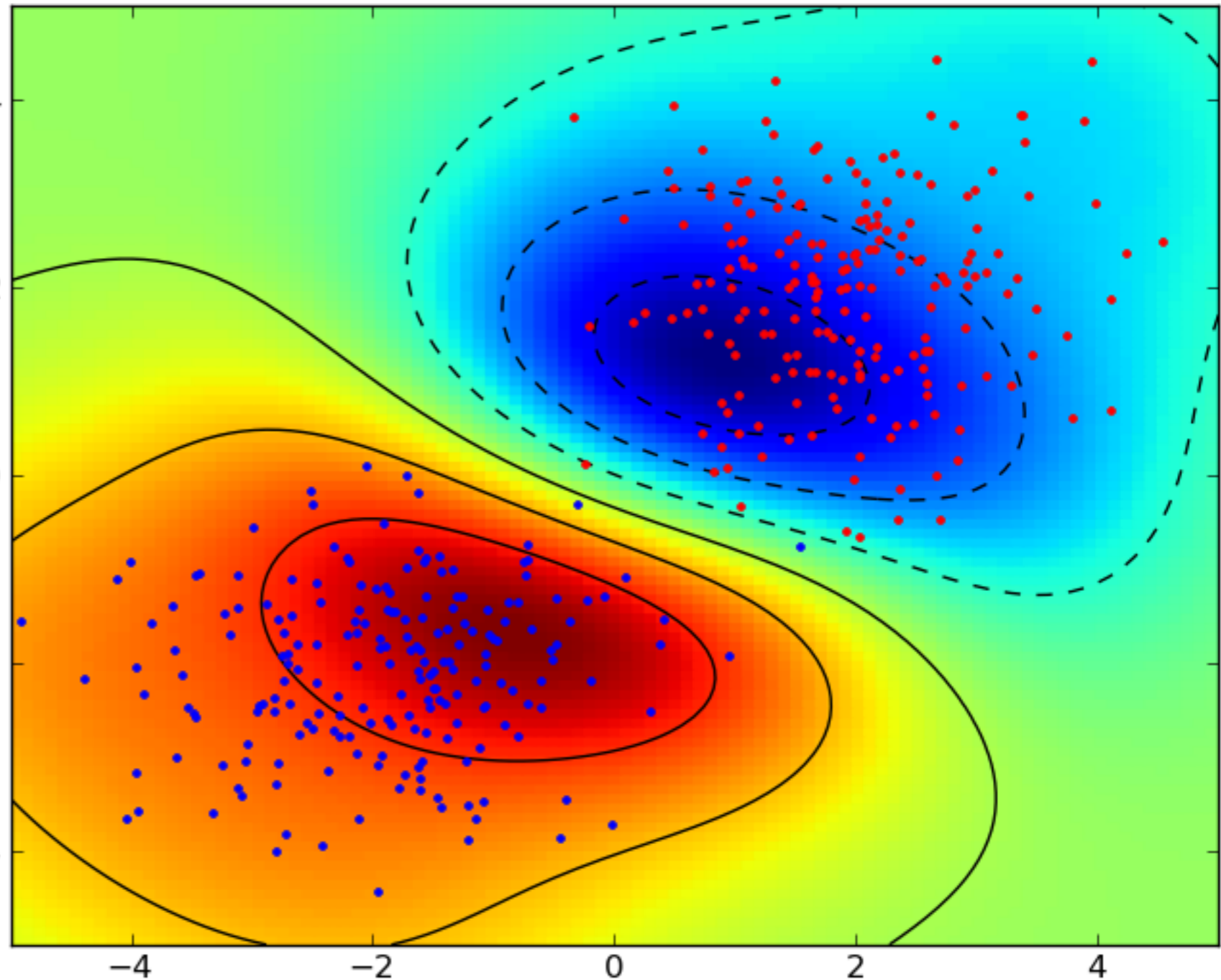
[Imap]/Drafts

[Imap]/Sent

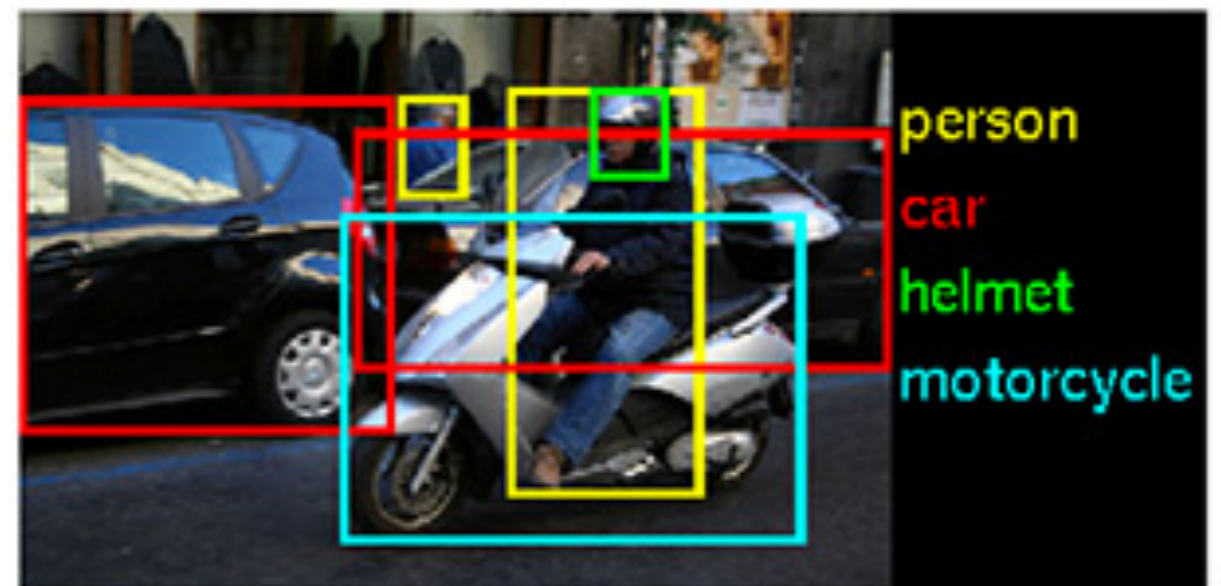
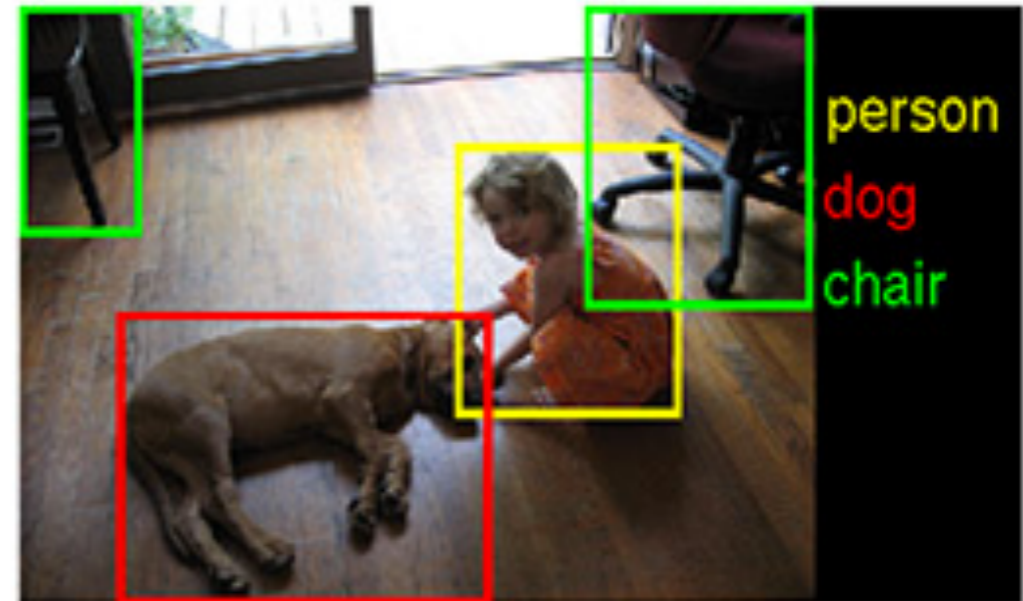
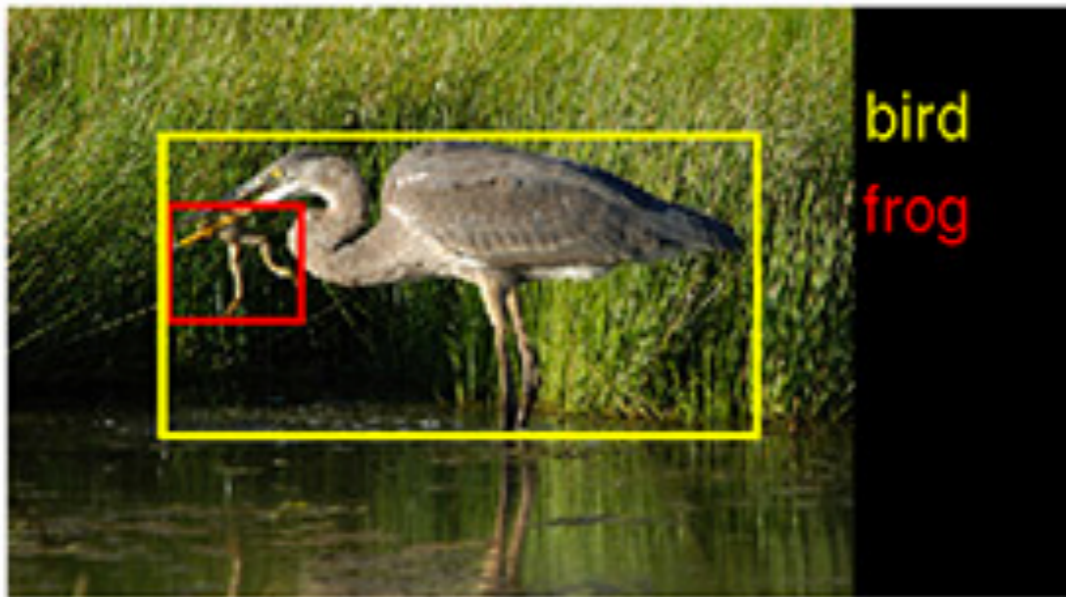
alex.smola@yah...

- maee
- Dear Valued Customers,
- garjeti
- Steven Cooke
- paper18
- First-Class Mail Service
- garjeti
- Candy.Li
- Ronan Morgan
- RE/MAX®
- newsletter
- CJCR editor
- garieti (2)

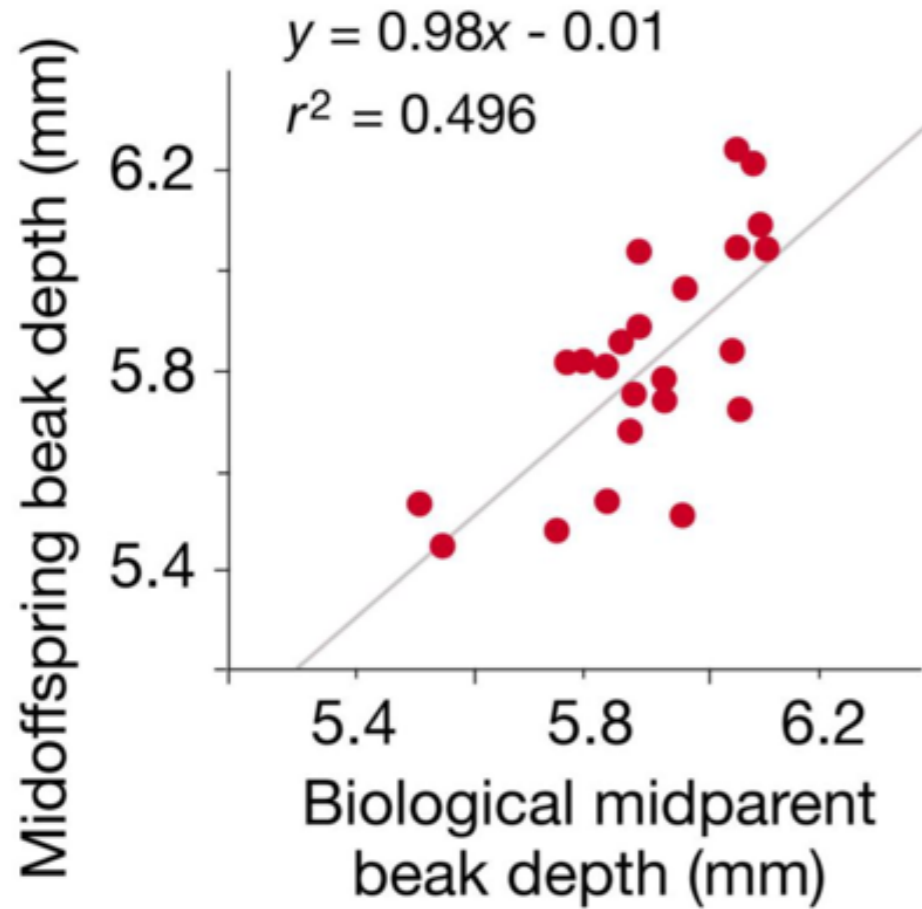
Search people...



Multiclass Classification + Annotation



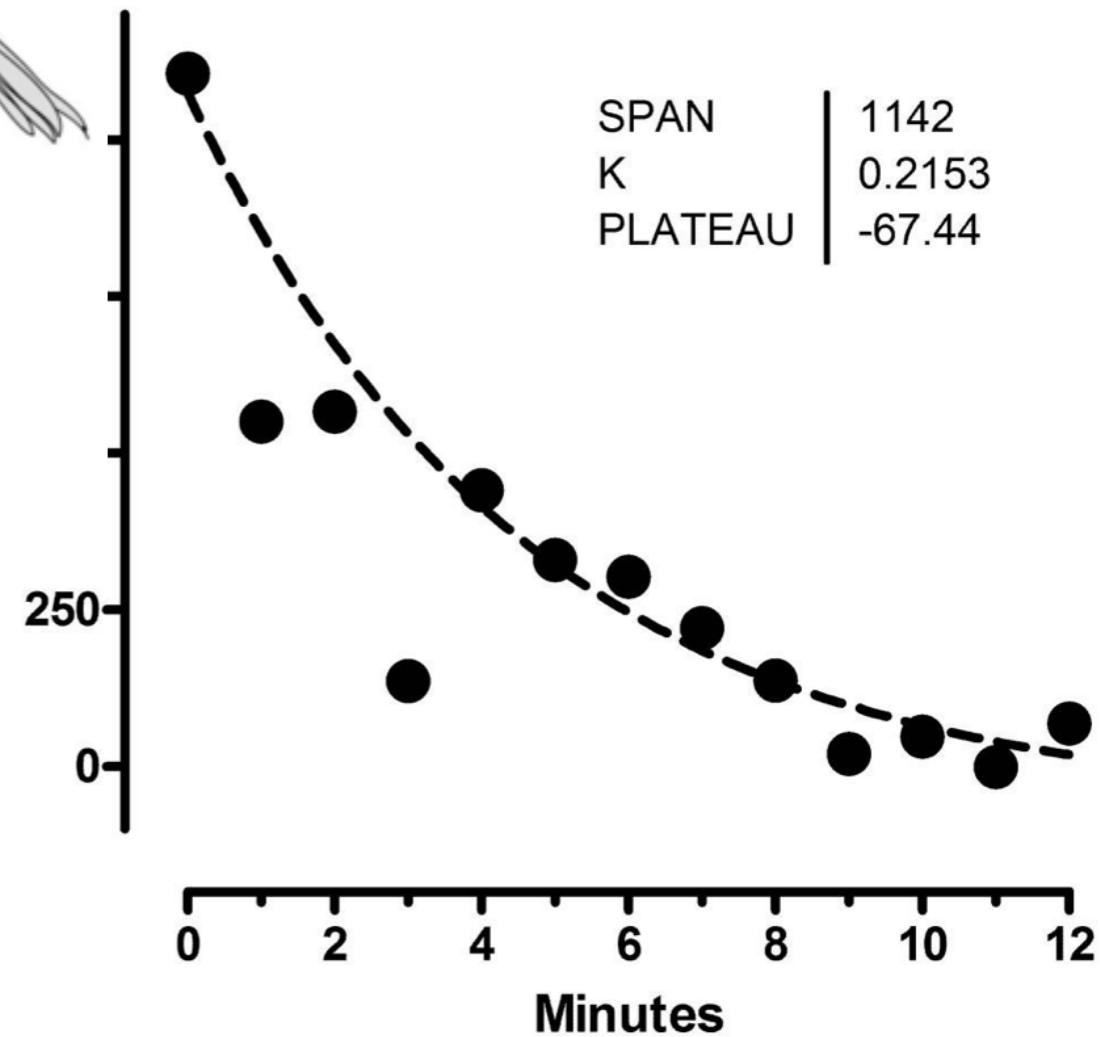
Regression



Copyright © 2004 Pearson Prentice Hall, Inc.

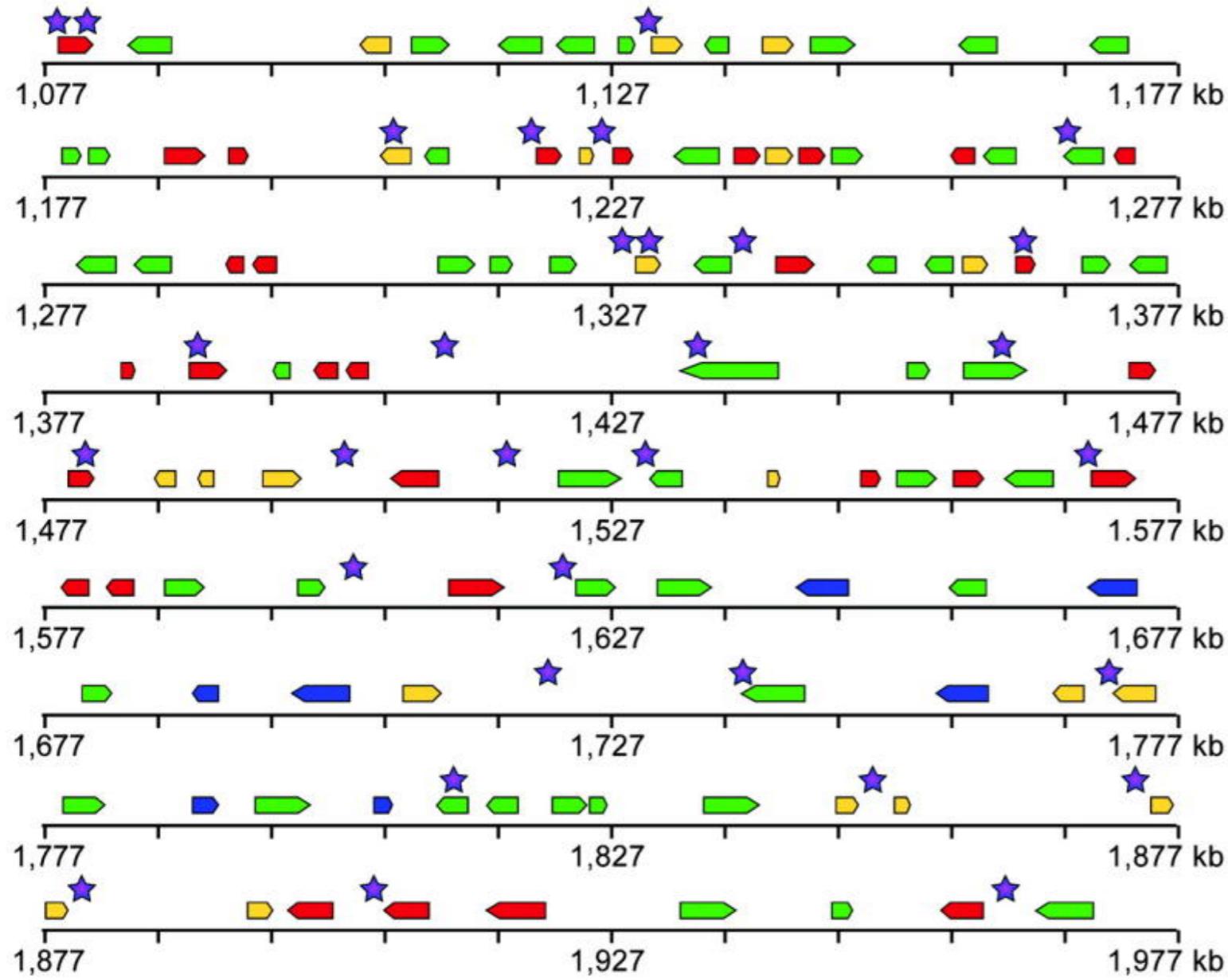
linear

nonlinear



SPAN	1142
K	0.2153
PLATEAU	-67.44

Sequence Annotation



given sequence

gene finding
speech recognition
activity segmentation
named entities

Ontology

dmoz open directory project

In partnership with AOL Search.

[about dmoz](#) | [dmoz blog](#) | [suggest URL](#) | [help](#) | [link](#) | [editor login](#)

webpages

Search [advanced](#)

Arts

[Movies](#), [Television](#), [Music](#)...

Games

[Video Games](#), [RPGs](#), [Gambling](#)...

Kids and Teens

[Arts](#), [School Time](#), [Teen Life](#)...

Reference

[Maps](#), [Education](#), [Libraries](#)...

Shopping

[Clothing](#), [Food](#), [Gifts](#)...

World

[Català](#), [Dansk](#), [Deutsch](#), [Español](#), [Français](#), [Italiano](#), [日本語](#), [Nederlands](#), [Polski](#), [Русский](#), [Svenska](#)...

Business

[Jobs](#), [Real Estate](#), [Investing](#)...

Health

[Fitness](#), [Medicine](#), [Alternative](#)...

News

[Media](#), [Newspapers](#), [Weather](#)...

Regional

[US](#), [Canada](#), [UK](#), [Europe](#)...

Society

[People](#), [Religion](#), [Issues](#)...

Computers

[Internet](#), [Software](#), [Hardware](#)...

Home

[Family](#), [Consumers](#), [Cooking](#)...

Recreation

[Travel](#), [Food](#), [Outdoors](#), [Humor](#)...

Science

[Biology](#), [Psychology](#), [Physics](#)...

Sports

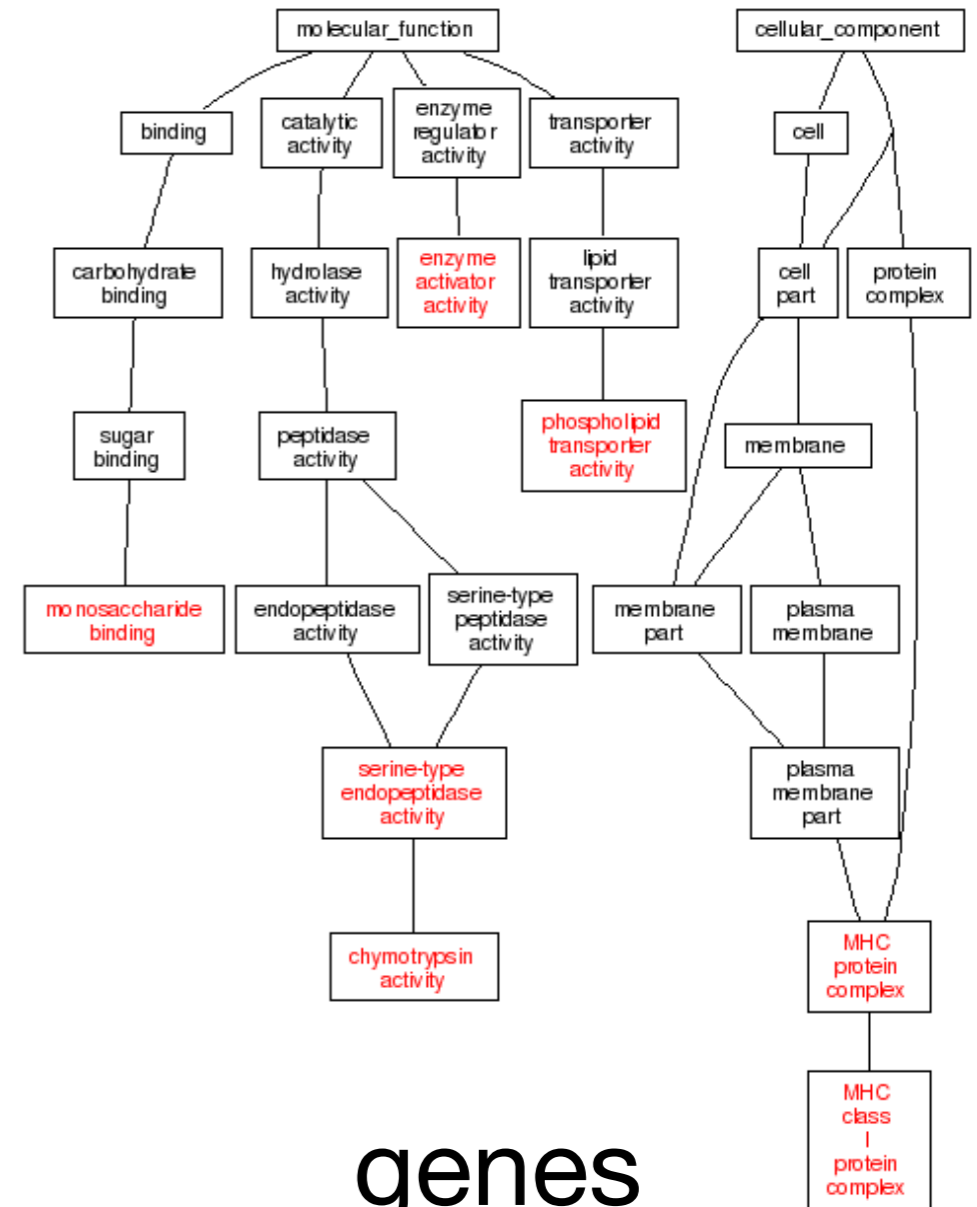
[Baseball](#), [Soccer](#), [Basketball](#)...

[Become an Editor](#) Help build the largest human-edited directory of the web



Copyright © 2013 Netscape

5,114,083 sites - 96,877 editors - over 1,014,849 categories



genes

Prediction



tomorrow's stock price

Unsupervised Learning

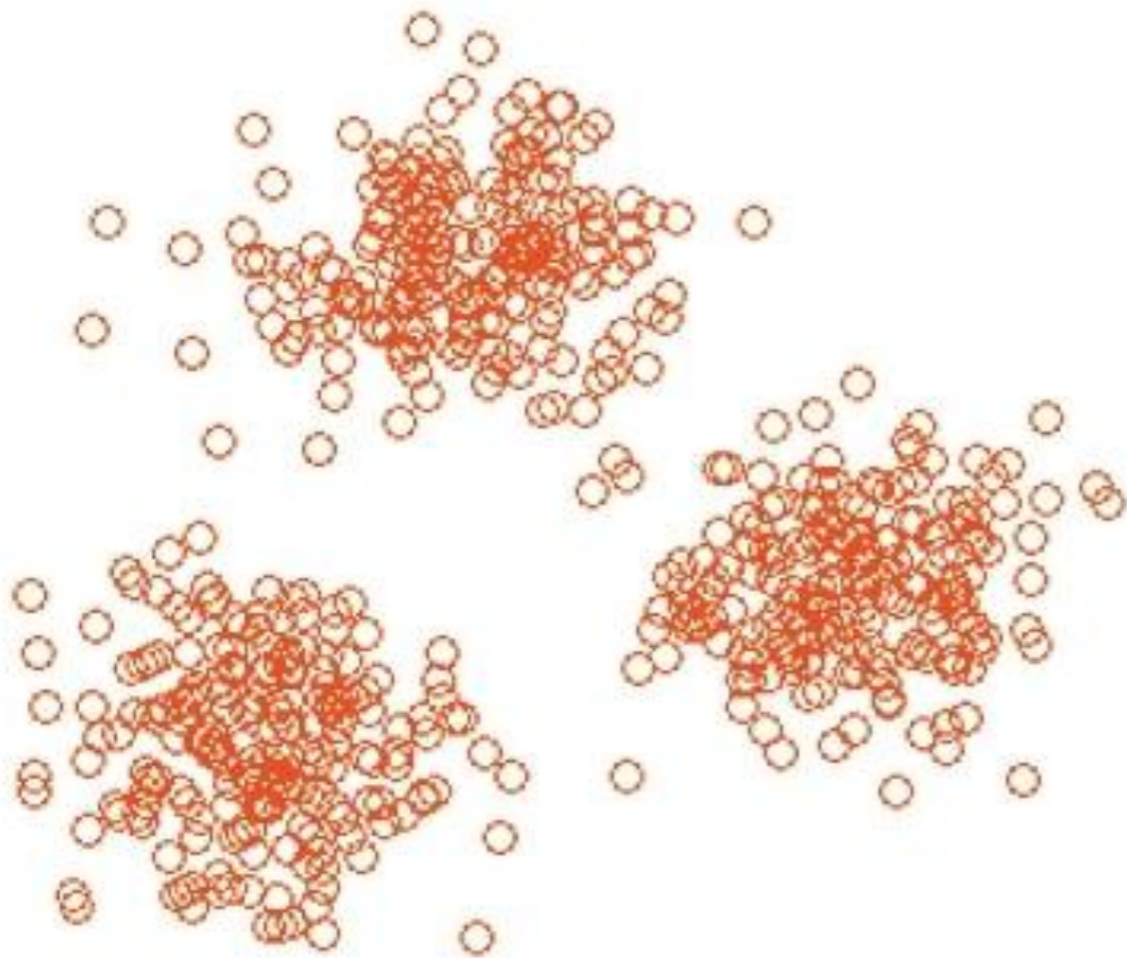


Unsupervised Learning

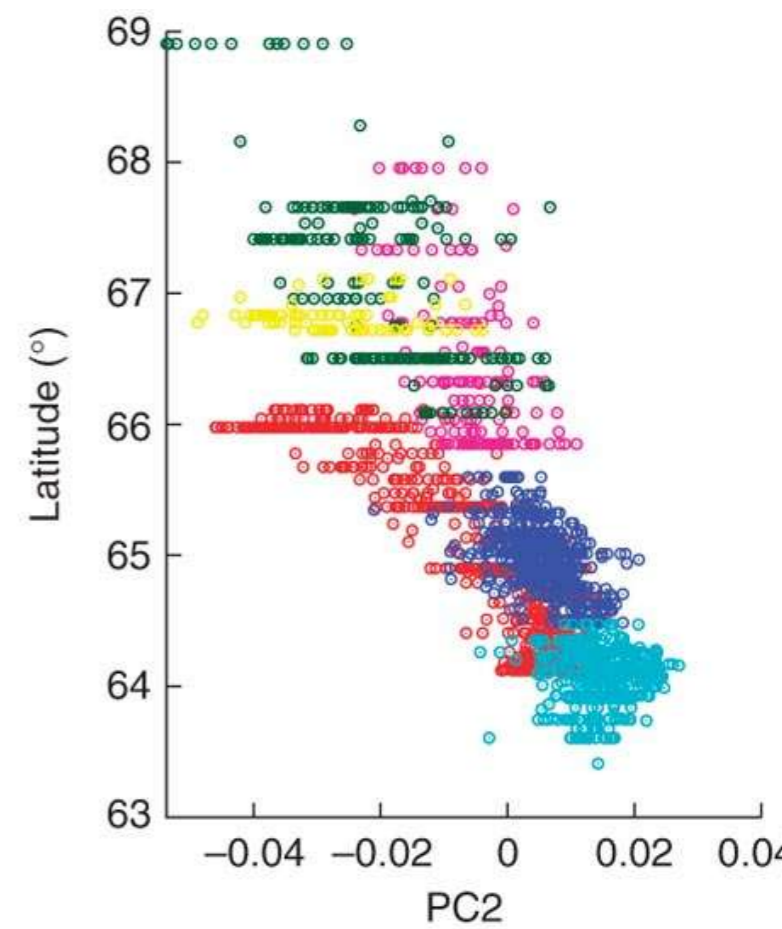
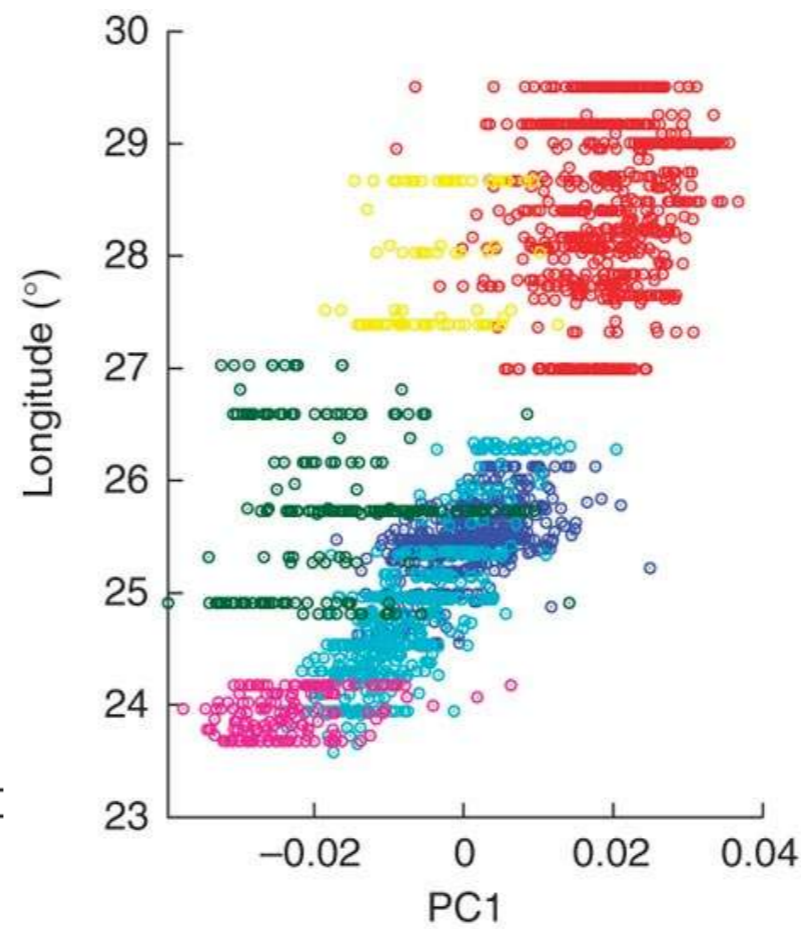
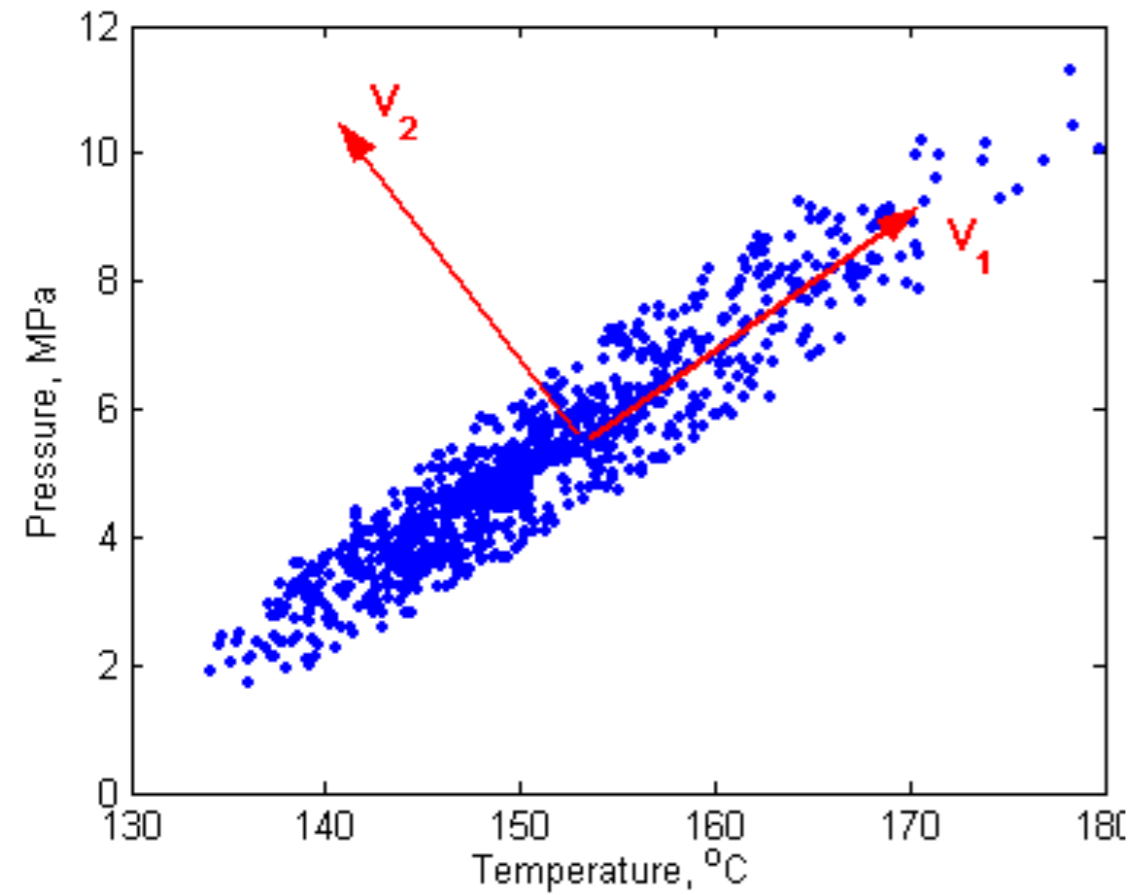
- Given data x , ask a good question ... about x or about model for x
- **Clustering**
Find a set of prototypes representing the data
- **Principal Components**
Find a subspace representing the data
- **Sequence Analysis**
Find a latent causal sequence for observations
 - Sequence Segmentation
 - Hidden Markov Model (discrete state)
 - Kalman Filter (continuous state)
- **Hierarchical representations**
- **Independent components / dictionary learning**
Find (small) set of factors for observation
- **Novelty detection**
Find the odd one out

Clustering

- Documents
- Users
- Webpages
- Diseases
- Pictures
- Vehicles
- ...

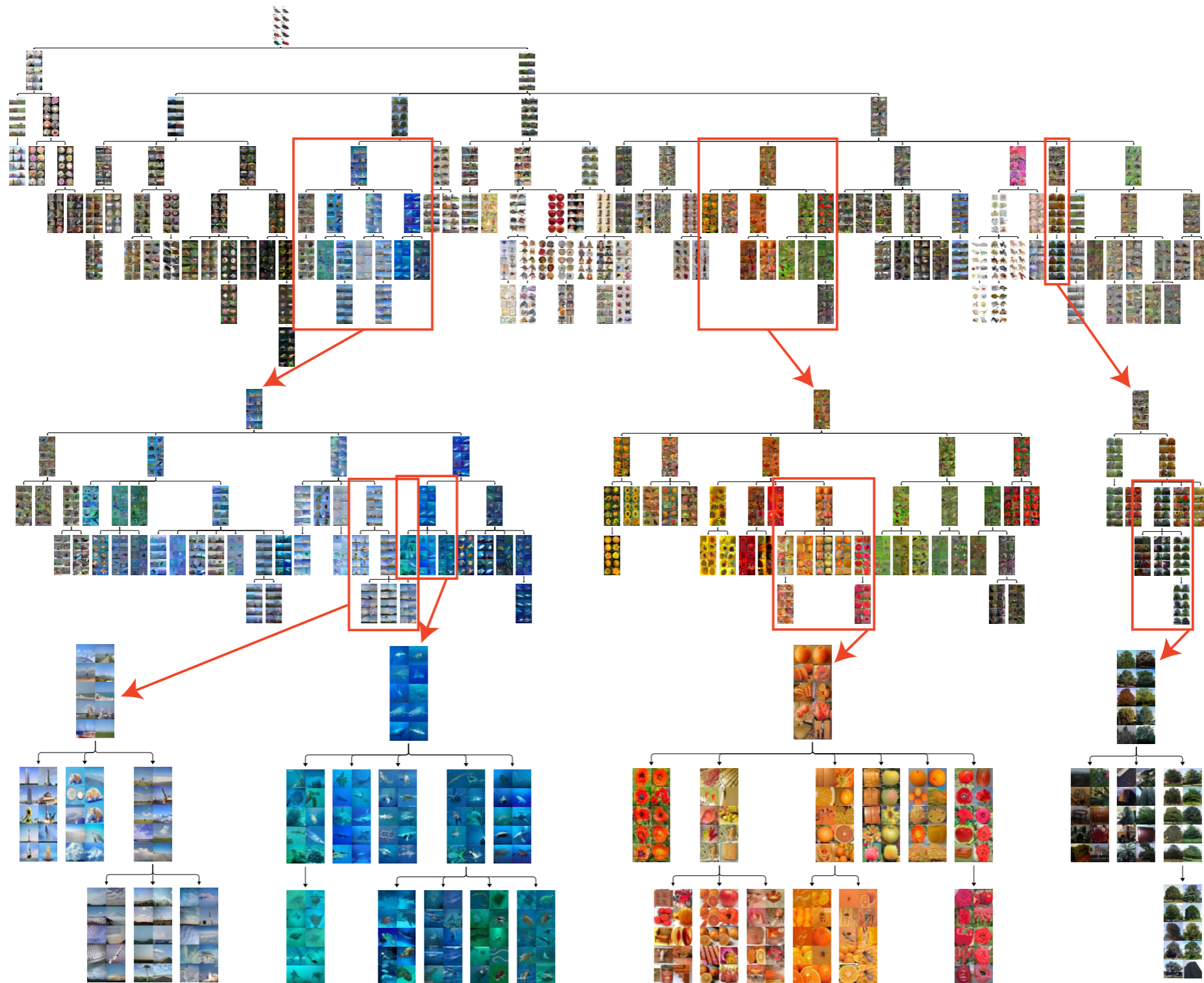


Principal Components

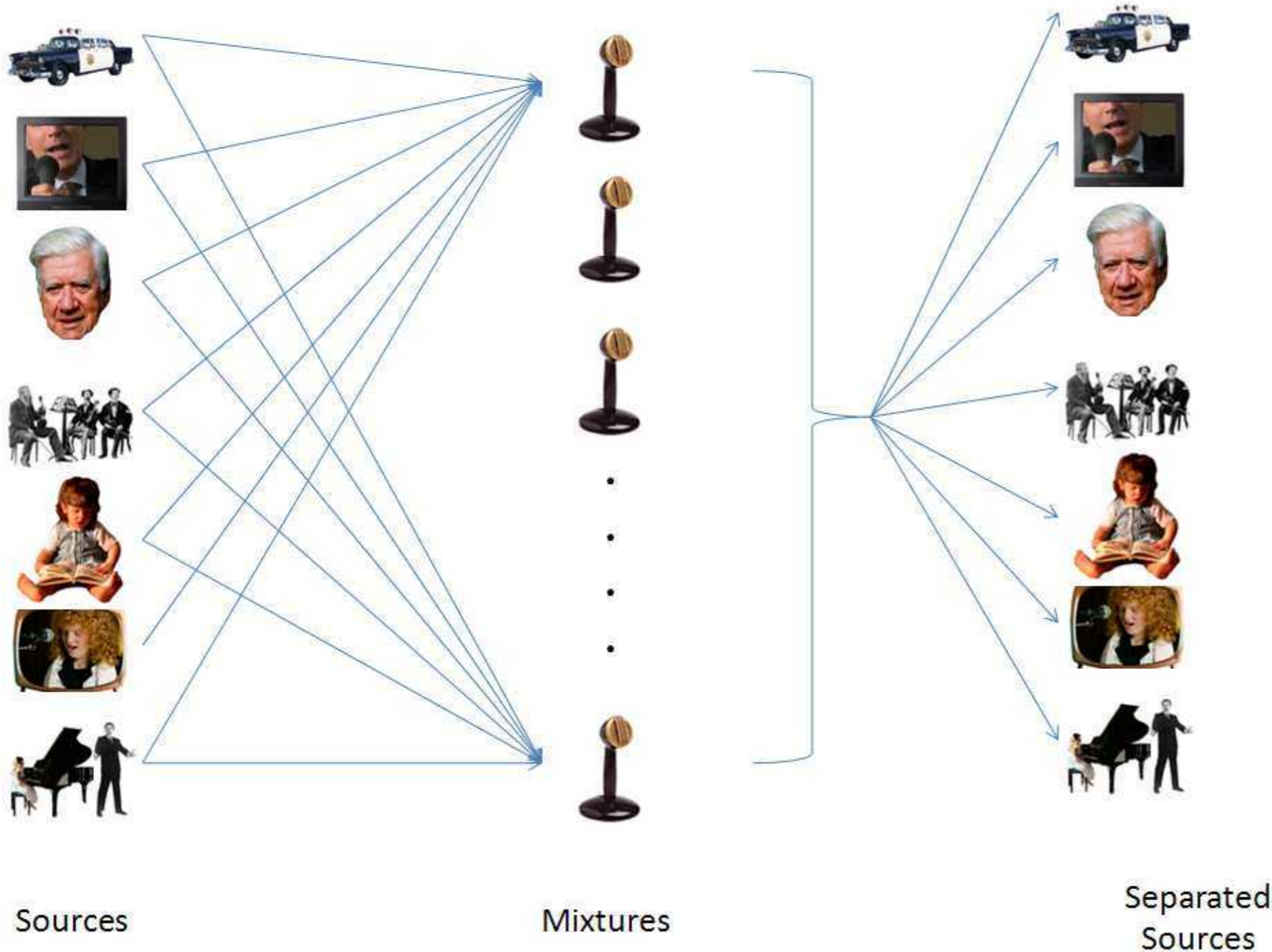


Variance component model to account for sample structure in genome-wide association studies, Nature Genetics 2010

Hierarchical Grouping

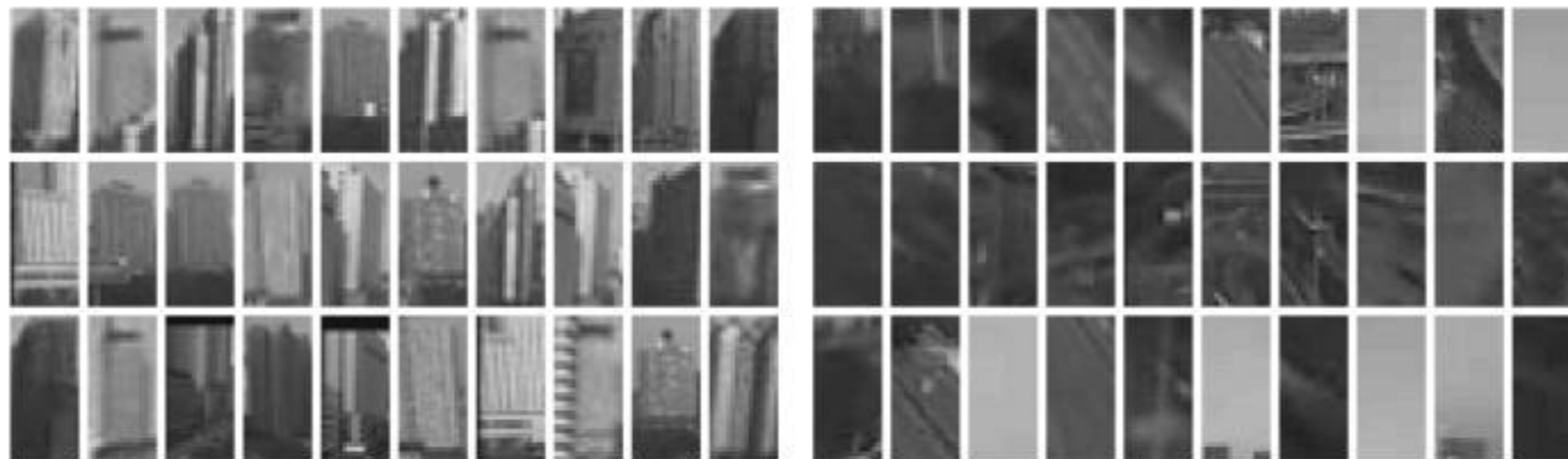


Independent Components



find them
automatically

Novelty detection



typical

atypical

Important challenges in ML

- How important is the actual learning algorithm and its tuning
- Simple versus complex algorithm
- Overfitting
- Model Selection
- Regularization

Your 1st Classifier: Nearest Neighbor Classifier

Concept Learning

- **Definition:** Acquire an operational definition of a general category of objects given *positive* and *negative* training examples.
- Also called *binary classification*, *binary supervised learning*

Concept Learning Example

	correct (complete, partial, guessing)	color (yes, no)	original (yes, no)	presentation (clear, unclear, cryptic)	binder (yes, no)	A+
1	complete	yes	yes	clear	no	yes
2	complete	no	yes	clear	no	yes
3	partial	yes	no	unclear	no	no
4	complete	yes	yes	clear	yes	yes

- **Instance Space X** : Set of all possible objects describable by attributes (often called *features*).
- **Concept c** : Subset of objects from X (c is unknown).
- **Target Function f** : Characteristic function indicating membership in c based on attributes (i.e. *label*) (f is unknown).
- **Training Data S** : Set of instances labeled with target function.

Concept Learning as Learning A Binary Function

- **Task**

- Learn (to imitate) a function $f: X \rightarrow \{+1, -1\}$

- **Training Examples**

- Learning algorithm is given the correct value of the function for particular inputs \rightarrow training examples

- An example is a pair (x, y) , where x is the input and $y = f(x)$ is the output of the target function applied to x .

- **Goal**

- Find a function

$$h: X \rightarrow \{+1, -1\}$$

that approximates

$$f: X \rightarrow \{+1, -1\}$$

as well as possible.

Supervised Learning

- **Task**

- Learn (to imitate) a function $f: X \rightarrow Y$

- **Training Examples**

- Learning algorithm is given the correct value of the function for particular inputs \rightarrow training examples

- An example is a pair $(x, f(x))$, where x is the input and $y=f(x)$ is

the output of the target function applied to x .

- **Goal**

- Find a function

$$h: X \rightarrow Y$$

that approximates

$$f: X \rightarrow Y$$

as well as possible.

Supervised / Inductive Learning

- Given
 - examples of a function $(x, f(x))$
- Predict function $f(x)$ for new examples x
 - Discrete $f(x)$: Classification
 - Continuous $f(x)$: Regression
 - $f(x) = \text{Probability}(x)$: Probability estimation

Image Classification: a core task in Computer Vision



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



cat

The problem: semantic gap



08	02	22	97	38	15	00	40	00	75	04	05	07	78	52	12	50	77	81	88
49	49	99	40	17	81	18	57	60	87	17	40	98	43	69	48	04	56	62	00
81	49	31	73	55	79	14	29	93	71	40	67	52	88	30	03	49	13	36	65
52	70	95	23	04	60	11	42	69	27	68	56	01	32	56	71	37	02	36	91
22	31	16	71	51	63	83	89	41	92	36	54	22	40	40	28	66	33	13	80
24	47	38	80	99	03	45	02	44	75	33	53	78	36	84	20	35	17	12	50
32	98	81	28	64	23	67	10	26	38	40	67	59	54	70	66	18	38	64	70
67	26	20	68	02	62	12	20	95	63	94	39	63	08	40	91	66	49	94	21
24	55	58	05	66	73	99	26	97	17	78	78	96	83	14	88	34	89	63	72
21	36	23	09	75	00	76	44	20	45	35	14	00	61	33	97	34	31	33	95
78	17	53	28	22	75	31	67	15	94	03	80	04	62	16	14	09	53	56	92
16	39	05	42	96	35	31	47	55	58	88	24	00	17	54	24	36	29	85	57
86	56	00	48	35	71	89	07	05	44	44	37	44	60	21	58	51	54	17	58
19	80	81	68	05	94	47	69	28	73	92	13	86	52	17	77	04	89	55	40
04	52	08	83	97	35	99	16	07	97	57	32	16	26	26	79	33	27	98	66
07	86	68	87	57	62	20	72	03	46	35	67	46	55	12	32	63	93	53	69
04	42	16	73	38	85	39	31	24	94	72	18	08	46	29	32	40	62	76	36
20	69	36	41	72	30	23	88	34	82	99	69	82	67	59	85	74	04	36	16
20	73	35	29	78	31	90	01	74	31	49	71	48	88	81	16	23	57	05	54
01	70	54	71	83	51	54	69	16	92	33	48	61	43	52	01	89	19	67	88

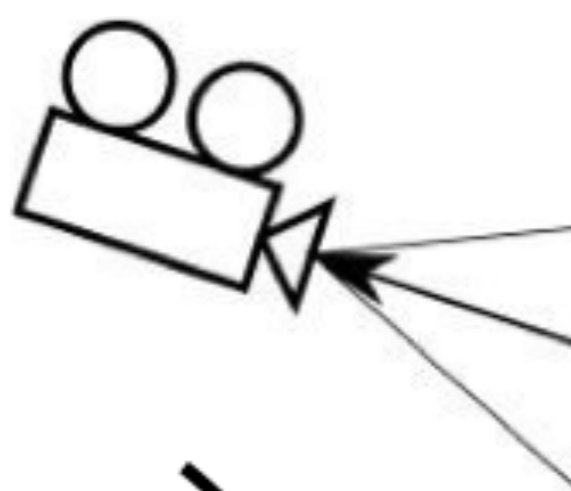
What the computer sees

Images are represented as 3D arrays of numbers, with integers between [0, 255].

E.g.
300 x 100 x 3

(3 for 3 color channels RGB)

Challenges: Viewpoint Variation



08	02	22	97	38	15	00	40	00	75	04	05	07	78	52	12	50	77	91	29
49	49	99	40	17	81	18	57	60	87	17	40	98	43	69	48	04	56	62	00
81	49	31	73	55	79	14	29	93	71	40	67	57	88	30	03	49	13	36	65
52	70	95	23	04	60	11	42	69	77	65	56	01	32	56	71	37	02	36	91
22	31	16	71	51	67	43	59	41	92	36	54	22	40	40	28	66	33	13	80
24	47	39	00	99	03	45	02	44	75	33	53	78	36	84	20	35	17	12	50
32	98	81	28	64	23	67	10	26	38	40	67	59	54	70	66	18	38	64	70
67	26	20	68	02	62	12	20	95	63	94	39	63	08	40	91	66	49	94	21
24	55	58	05	66	73	99	26	97	17	78	78	96	83	14	88	34	89	63	72
21	36	23	09	75	00	76	44	20	45	35	14	00	61	33	97	34	31	33	95
78	17	53	28	22	75	31	67	15	94	03	80	04	62	16	14	09	53	56	92
16	39	05	42	96	35	31	47	55	58	88	24	00	17	54	24	36	29	85	57
86	56	00	48	35	71	89	07	05	44	44	37	44	60	21	58	51	54	17	58
19	80	81	68	05	94	47	69	28	73	92	13	86	52	17	77	04	89	55	40
04	52	08	83	97	35	99	16	07	97	57	32	16	26	26	79	33	27	98	66
77	46	69	87	57	62	20	72	03	46	33	67	44	55	12	32	63	93	53	49
04	42	16	73	39	85	39	11	24	94	72	18	08	46	29	32	40	62	76	36
20	69	36	41	72	30	23	88	34	10	88	69	82	67	59	85	74	04	36	16
20	73	35	29	78	31	90	01	74	31	49	71	48	58	81	16	23	57	05	54
01	70	54	71	83	51	54	69	16	92	33	48	61	43	52	01	87	17	67	45

All pixels change when the camera moves!

Challenges: Illumination



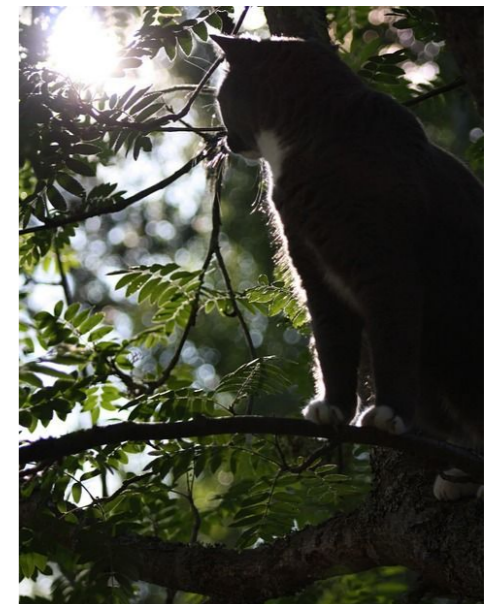
[This image is CC0 1.0 public domain](#)



[This image is CC0 1.0 public domain](#)

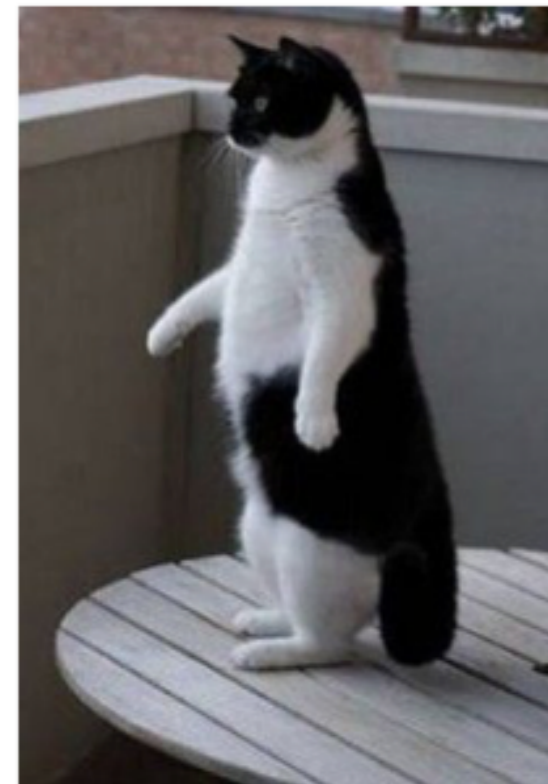


[This image is CC0 1.0 public domain](#)

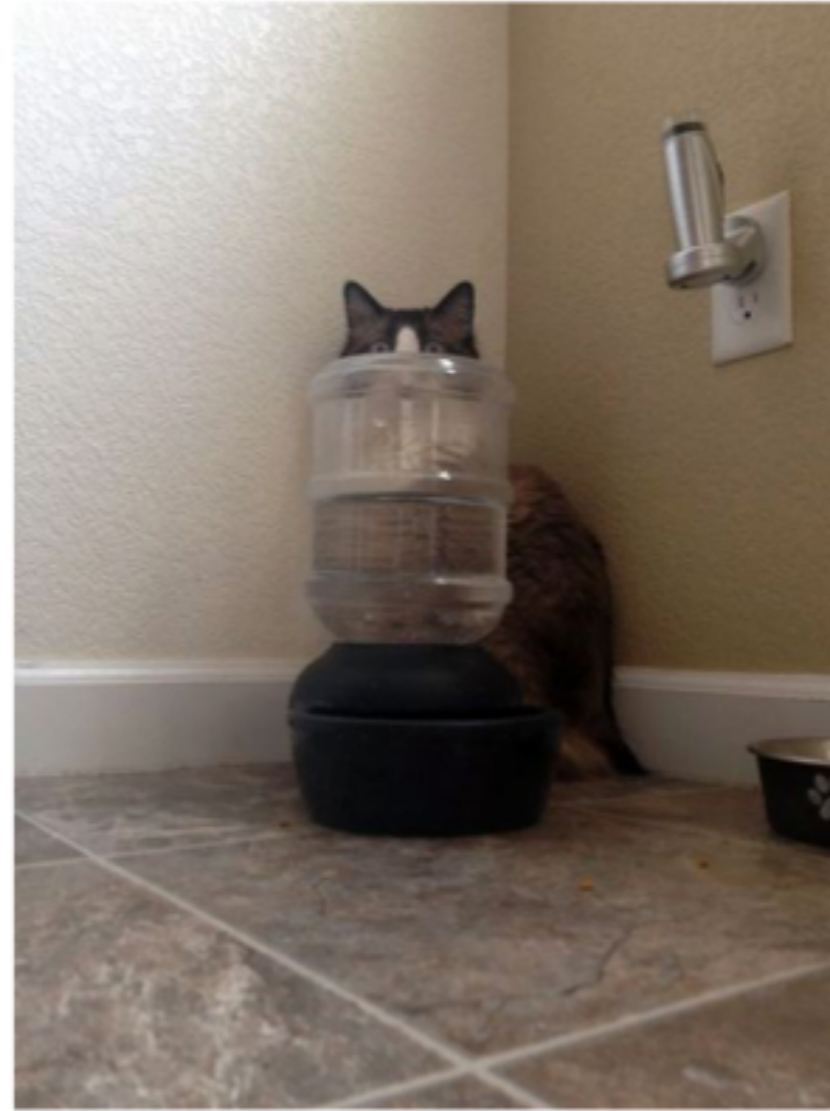


[This image is CC0 1.0 public domain](#)

Challenges: Deformation



Challenges: Occlusion



Challenges: Background clutter



Challenges: Intraclclass variation



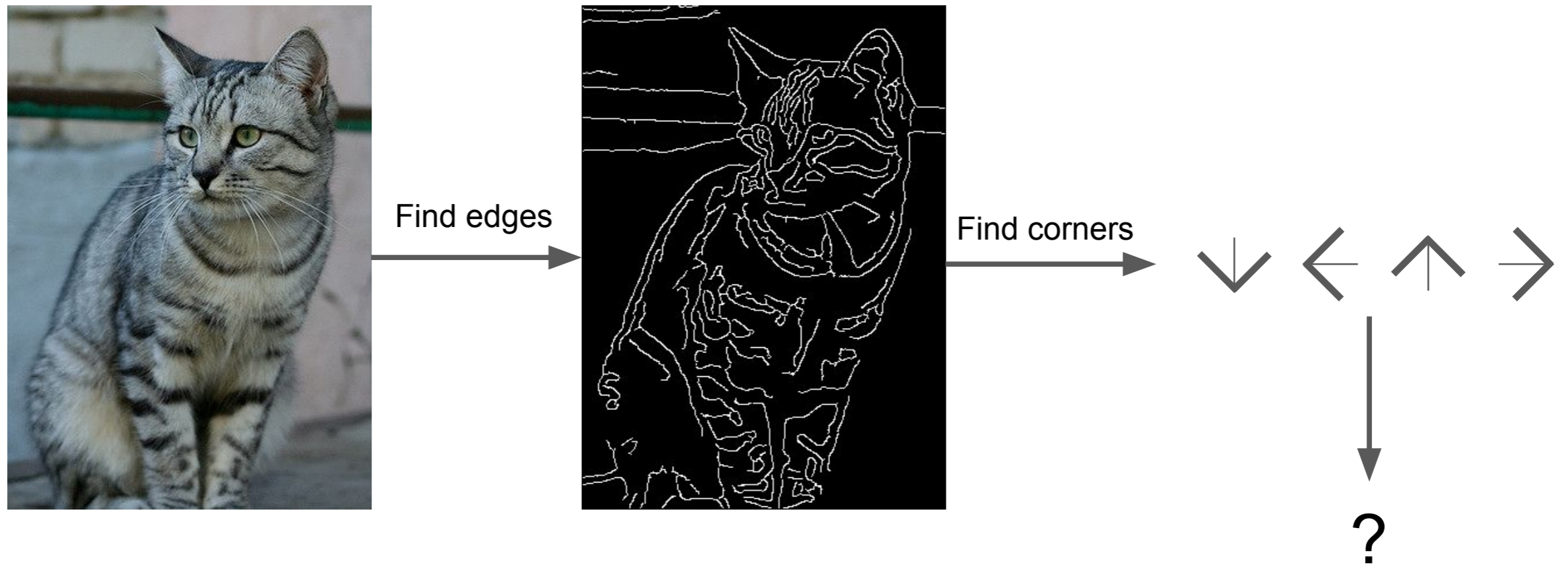
An image classifier

```
def classify_image(image):  
    # Some magic here?  
    return class_label
```

Unlike *e.g.* sorting a list of numbers,

no obvious way to hard-code the algorithm for recognizing a cat, or other classes.

Attempts have been made



Data-driven approach:

1. Collect a dataset of images and labels
2. Use Machine Learning to train an image classifier
3. Evaluate the classifier on a withheld set of test images

Example training set

```
def train(train_images, train_labels):  
    # build a model for images -> labels...  
    return model  
  
def predict(model, test_images):  
    # predict test_labels using the model...  
    return test_labels
```



First classifier: Nearest Neighbor Classifier

```
def train(train_images, train_labels):  
    # build a model for images -> labels...  
    return model  
  
def predict(model, test_images):  
    # predict test_labels using the model...  
    return test_labels
```

Memorize all training images and their labels

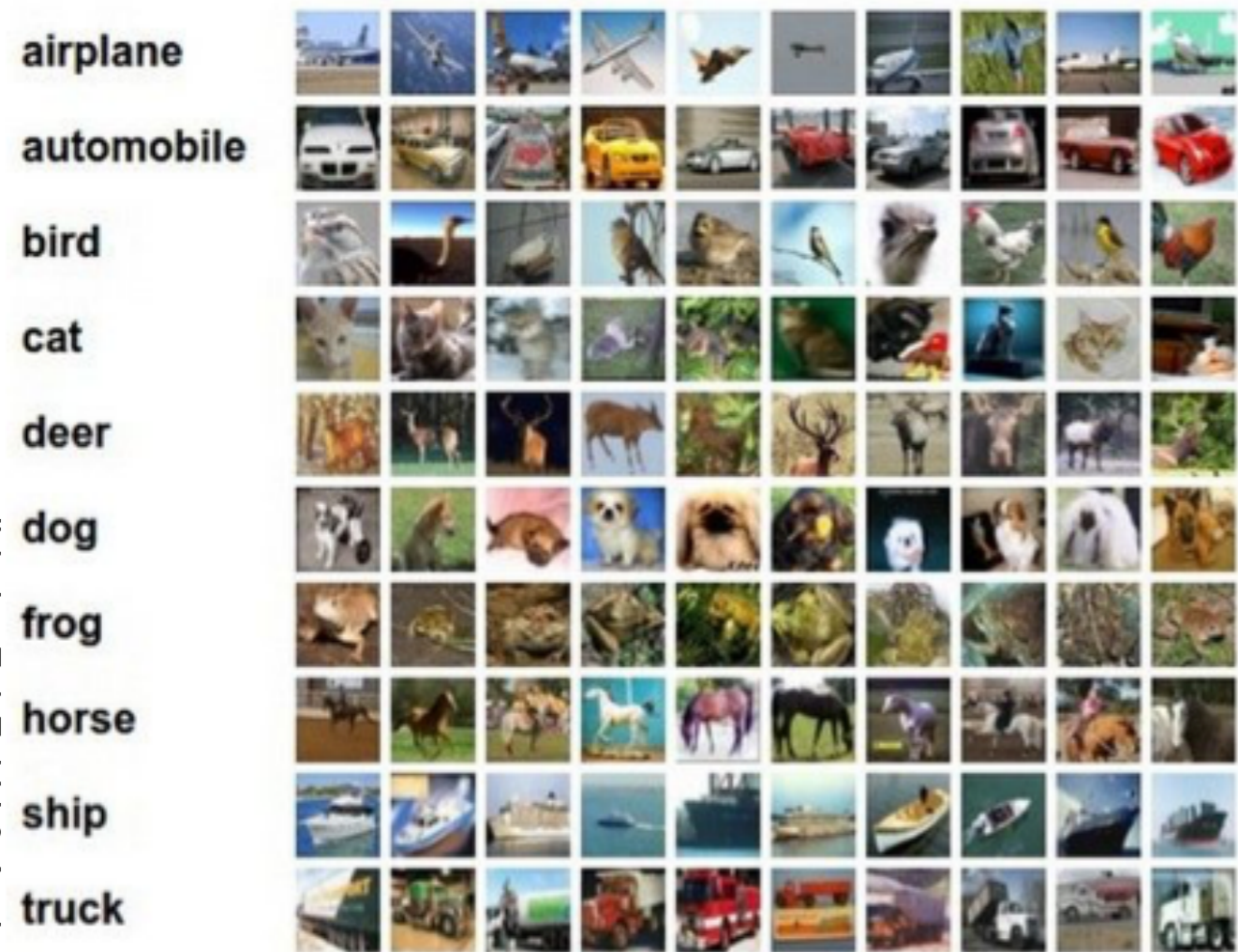
Predict the label of the most similar training image

Example dataset: **CIFAR-10**

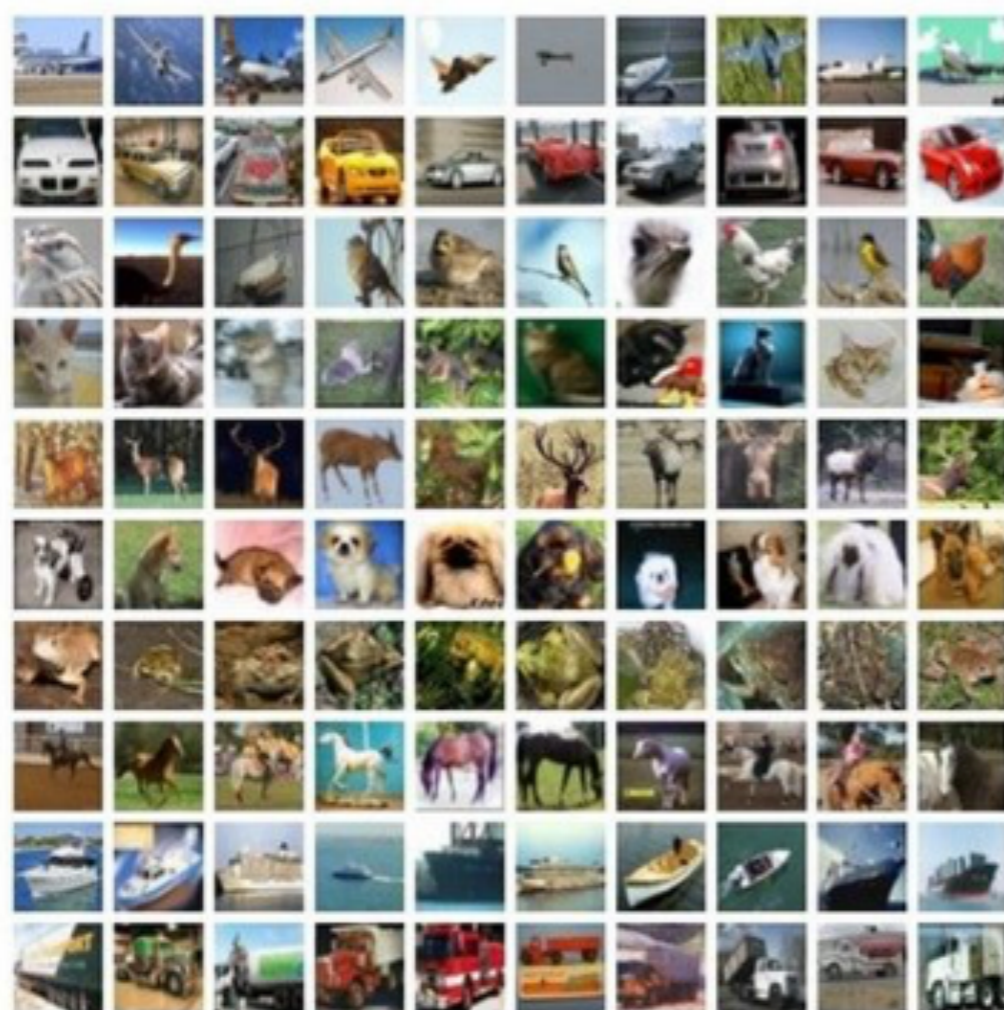
10 labels

50,000 training images, each image is tiny: 32x32

10,000 test images.



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



How do we compare the images? What is the **distance metric**?

L1 distance:
$$d_1(I_1, I_2) = \sum_p |I_1^p - I_2^p|$$

test image	training image	pixel-wise absolute value differences																																																
<table border="1"><tr><td>56</td><td>32</td><td>10</td><td>18</td></tr><tr><td>90</td><td>23</td><td>128</td><td>133</td></tr><tr><td>24</td><td>26</td><td>178</td><td>200</td></tr><tr><td>2</td><td>0</td><td>255</td><td>220</td></tr></table>	56	32	10	18	90	23	128	133	24	26	178	200	2	0	255	220	<table border="1"><tr><td>10</td><td>20</td><td>24</td><td>17</td></tr><tr><td>8</td><td>10</td><td>89</td><td>100</td></tr><tr><td>12</td><td>16</td><td>178</td><td>170</td></tr><tr><td>4</td><td>32</td><td>233</td><td>112</td></tr></table>	10	20	24	17	8	10	89	100	12	16	178	170	4	32	233	112	<table border="1"><tr><td>46</td><td>12</td><td>14</td><td>1</td></tr><tr><td>82</td><td>13</td><td>39</td><td>33</td></tr><tr><td>12</td><td>10</td><td>0</td><td>30</td></tr><tr><td>2</td><td>32</td><td>22</td><td>108</td></tr></table>	46	12	14	1	82	13	39	33	12	10	0	30	2	32	22	108
56	32	10	18																																															
90	23	128	133																																															
24	26	178	200																																															
2	0	255	220																																															
10	20	24	17																																															
8	10	89	100																																															
12	16	178	170																																															
4	32	233	112																																															
46	12	14	1																																															
82	13	39	33																																															
12	10	0	30																																															
2	32	22	108																																															

add → 456

Nearest Neighbor classifier

```
import numpy as np

class NearestNeighbor:
    def __init__(self):
        pass

    def train(self, X, y):
        """ X is N x D where each row is an example. Y is 1-dimension of size N """
        # the nearest neighbor classifier simply remembers all the training data
        self.Xtr = X
        self.ytr = y

    def predict(self, X):
        """ X is N x D where each row is an example we wish to predict label for """
        num_test = X.shape[0]
        # lets make sure that the output type matches the input type
        Ypred = np.zeros(num_test, dtype = self.ytr.dtype)

        # loop over all test rows
        for i in xrange(num_test):
            # find the nearest training image to the i'th test image
            # using the L1 distance (sum of absolute value differences)
            distances = np.sum(np.abs(self.Xtr - X[i,:]), axis = 1)
            min_index = np.argmin(distances) # get the index with smallest distance
            Ypred[i] = self.ytr[min_index] # predict the label of the nearest example

        return Ypred
```

Nearest Neighbor classifier

```
import numpy as np
```

```
class NearestNeighbor:
```

```
    def __init__(self):  
        pass
```

```
    def train(self, X, y):
```

```
        """ X is N x D where each row is an example. Y is 1-dimension of size N """  
        # the nearest neighbor classifier simply remembers all the training data  
        self.Xtr = X  
        self.ytr = y
```

```
    def predict(self, X):
```

```
        """ X is N x D where each row is an example we wish to predict label for """  
        num_test = X.shape[0]  
        # lets make sure that the output type matches the input type  
        Ypred = np.zeros(num_test, dtype = self.ytr.dtype)  
  
        # loop over all test rows  
        for i in xrange(num_test):  
            # find the nearest training image to the i'th test image  
            # using the L1 distance (sum of absolute value differences)  
            distances = np.sum(np.abs(self.Xtr - X[i,:]), axis = 1)  
            min_index = np.argmin(distances) # get the index with smallest distance  
            Ypred[i] = self.ytr[min_index] # predict the label of the nearest example
```

```
    return Ypred
```

memorize
training data

Nearest Neighbor classifier

```
import numpy as np

class NearestNeighbor:
    def __init__(self):
        pass

    def train(self, X, y):
        """ X is N x D where each row is an example. Y is 1-dimension of size N """
        # the nearest neighbor classifier simply remembers all the training data
        self.Xtr = X
        self.ytr = y

    def predict(self, X):
        """ X is N x D where each row is an example we wish to predict label for """
        num_test = X.shape[0]
        # lets make sure that the output type matches the input type
        Ypred = np.zeros(num_test, dtype = self.ytr.dtype)

        # loop over all test rows
        for i in xrange(num_test):
            # find the nearest training image to the i'th test image
            # using the L1 distance (sum of absolute value differences)
            distances = np.sum(np.abs(self.Xtr - X[i,:]), axis = 1)
            min_index = np.argmin(distances) # get the index with smallest distance
            Ypred[i] = self.ytr[min_index] # predict the label of the nearest example

        return Ypred
```

- for every test image:
- find closest train image with L1 distance
 - predict the label of nearest training image

```

import numpy as np

class NearestNeighbor:
    def __init__(self):
        pass

    def train(self, X, y):
        """ X is N x D where each row is an example. Y is 1-dimension of size N """
        # the nearest neighbor classifier simply remembers all the training data
        self.Xtr = X
        self.ytr = y

    def predict(self, X):
        """ X is N x D where each row is an example we wish to predict label for """
        num_test = X.shape[0]
        # lets make sure that the output type matches the input type
        Ypred = np.zeros(num_test, dtype = self.ytr.dtype)

        # loop over all test rows
        for i in xrange(num_test):
            # find the nearest training image to the i'th test image
            # using the L1 distance (sum of absolute value differences)
            distances = np.sum(np.abs(self.Xtr - X[i,:]), axis = 1)
            min_index = np.argmin(distances) # get the index with smallest distance
            Ypred[i] = self.ytr[min_index] # predict the label of the nearest example

        return Ypred

```

Nearest Neighbor classifier

Q: With N examples, how fast are training and prediction?

```

import numpy as np

class NearestNeighbor:
    def __init__(self):
        pass

    def train(self, X, y):
        """ X is N x D where each row is an example. Y is 1-dimension of size N """
        # the nearest neighbor classifier simply remembers all the training data
        self.Xtr = X
        self.ytr = y

    def predict(self, X):
        """ X is N x D where each row is an example we wish to predict label for """
        num_test = X.shape[0]
        # lets make sure that the output type matches the input type
        Ypred = np.zeros(num_test, dtype = self.ytr.dtype)

        # loop over all test rows
        for i in xrange(num_test):
            # find the nearest training image to the i'th test image
            # using the L1 distance (sum of absolute value differences)
            distances = np.sum(np.abs(self.Xtr - X[i,:]), axis = 1)
            min_index = np.argmin(distances) # get the index with smallest distance
            Ypred[i] = self.ytr[min_index] # predict the label of the nearest example

        return Ypred

```

Nearest Neighbor classifier

Q: With N examples, how fast are training and prediction?

A: train $O(1)$, predict $O(N)$

This is bad: we want classifiers that are **fast** at prediction; **slow** for training is ok.

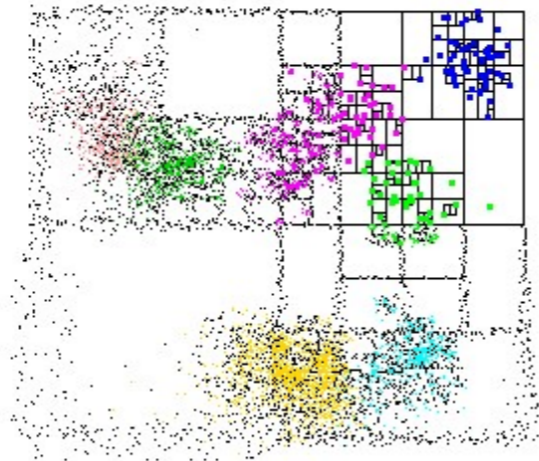
Aside: Approximate Nearest Neighbor find approximate nearest neighbors quickly

ANN: A Library for Approximate Nearest Neighbor Searching

David M. Mount and Sunil Arya

Version 1.1.2

Release Date: Jan 27, 2010



What is ANN?

ANN is a library written in C++, which supports data structures and algorithms for both exact and approximate nearest neighbor searching in arbitrarily high dimensions.

In the nearest neighbor problem a set of data points in d -dimensional space is given. These points are preprocessed into a data structure, so that given any query point q , the nearest or generally k nearest points of P to q can be reported efficiently. The distance between two points can be defined in many ways. ANN assumes that distances are measured using any class of distance functions called Minkowski metrics. These include the well known Euclidean distance, Manhattan distance, and max distance.

Based on our own experience, ANN performs quite efficiently for point sets ranging in size from thousands to hundreds of thousands, and in dimensions as high as 20. (For applications in significantly higher dimensions, the results are rather spotty, but you might try it anyway.)

The library implements a number of different data structures, based on kd-trees and box-decomposition trees, and employs a couple of different search strategies.

The library also comes with test programs for measuring the quality of performance of ANN on any particular data sets, as well as programs for visualizing the structure of the geometric data structures.

FLANN - Fast Library for Approximate Nearest Neighbors

- Home
- News
- Publications
- Download
- Changelog
- Repository

What is FLANN?

FLANN is a library for performing fast approximate nearest neighbor searches in high dimensional spaces. It contains a collection of algorithms we found to work best for nearest neighbor search and a system for automatically choosing the best algorithm and optimum parameters depending on the dataset.

FLANN is written in C++ and contains bindings for the following languages: C, MATLAB and Python.

News

- (14 December 2012) Version 1.8.0 is out bringing incremental addition/removal of points to/from indexes
- (20 December 2011) Version 1.7.0 is out bringing two new index types and several other improvements.
- You can find binary installers for FLANN on the [Point Cloud Library](#) project page. Thanks to the PCL developers!
- Mac OS X users can install flann through MacPorts (thanks to Mark Moll for maintaining the Portfile)
- New release introducing an easier way to use custom distances, kd-tree implementation optimized for low dimensionality search and experimental MPI support
- New release introducing new C++ templated API, thread-safe search, save/load of indexes and more.
- The FLANN license was changed from LGPL to BSD.

How fast is it?

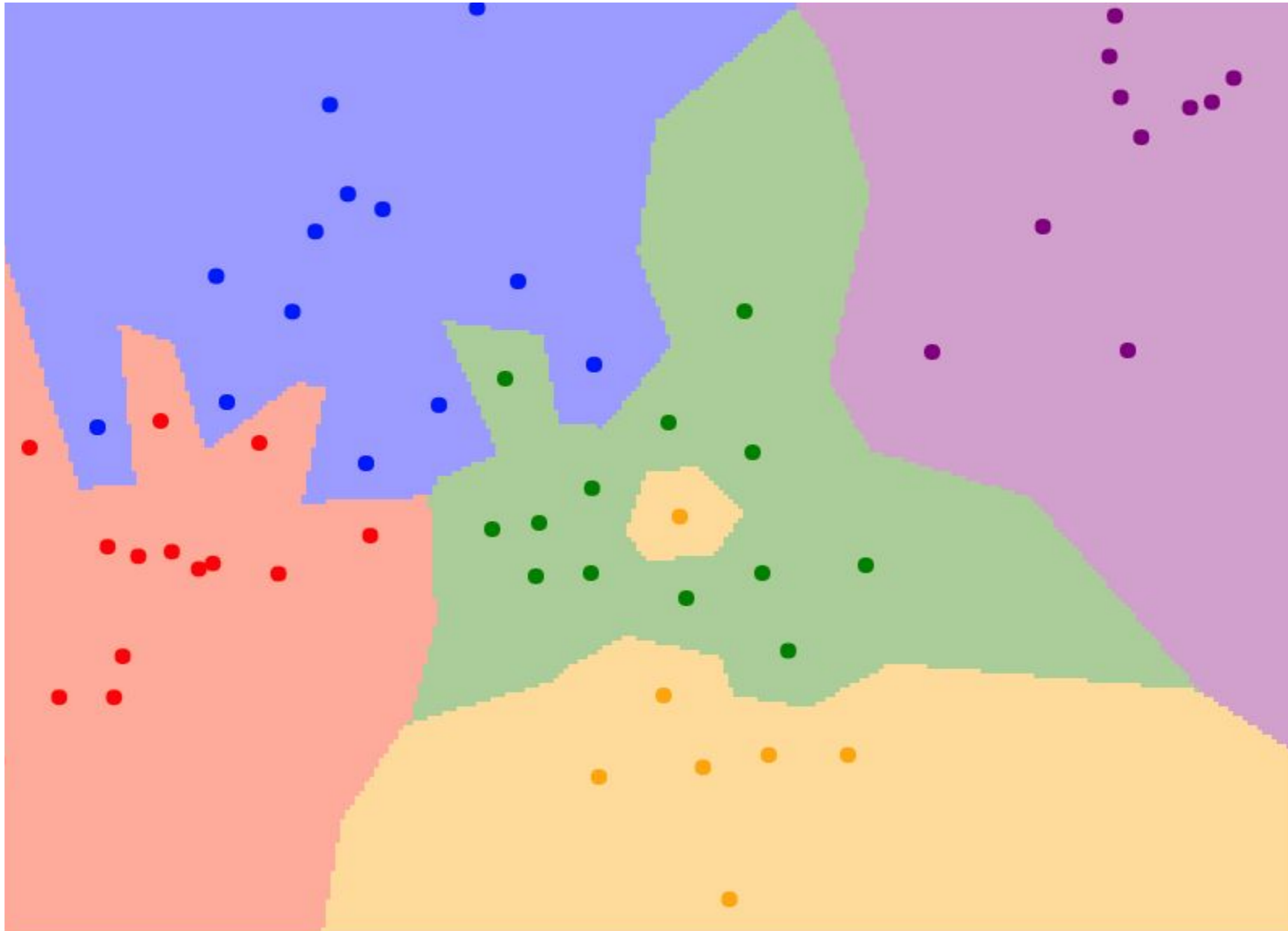
In our experiments we have found FLANN to be about one order of magnitude faster on many datasets (in query time), than previously available approximate nearest neighbor search software.

Publications

More information and experimental results can be found in the following papers:

- Marius Muja and David G. Lowe: "Scalable Nearest Neighbor Algorithms for High Dimensional Data". Pattern Analysis and Machine Intelligence (PAMI), Vol. 36, 2014. [\[PDF\]](#) [\[BibTeX\]](#)
- Marius Muja and David G. Lowe: "Fast Matching of Binary Features". Conference on Computer and Robot Vision (CRV) 2012. [\[PDF\]](#) [\[BibTeX\]](#)
- Marius Muja and David G. Lowe, "Fast Approximate Nearest Neighbors with Automatic Algorithm Configuration", in International Conference on Computer Vision Theory and Applications (VISAPP'09), 2009 [\[PDF\]](#) [\[BibTeX\]](#)

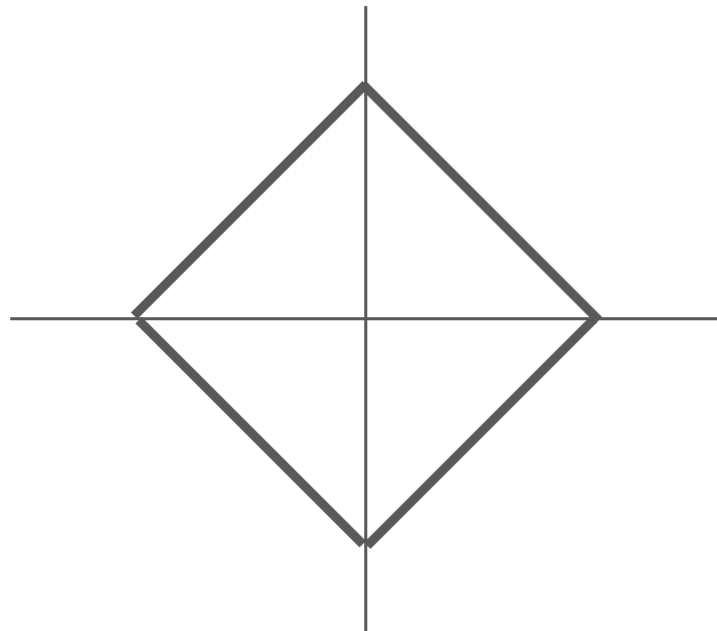
What does Nearest Neighbor classifier look like?



The choice of distance is a **hyperparameter**
common choices:

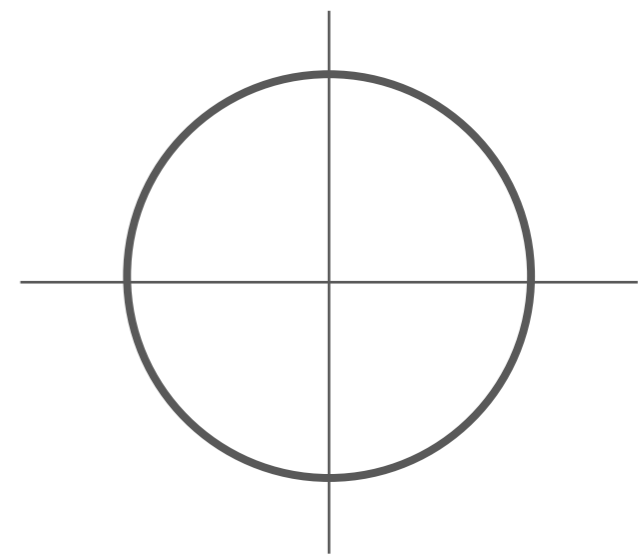
L1 (Manhattan) distance

$$d_1(I_1, I_2) = \sum_P |I_1^P - I_2^P|$$



L2 (Euclidean) distance

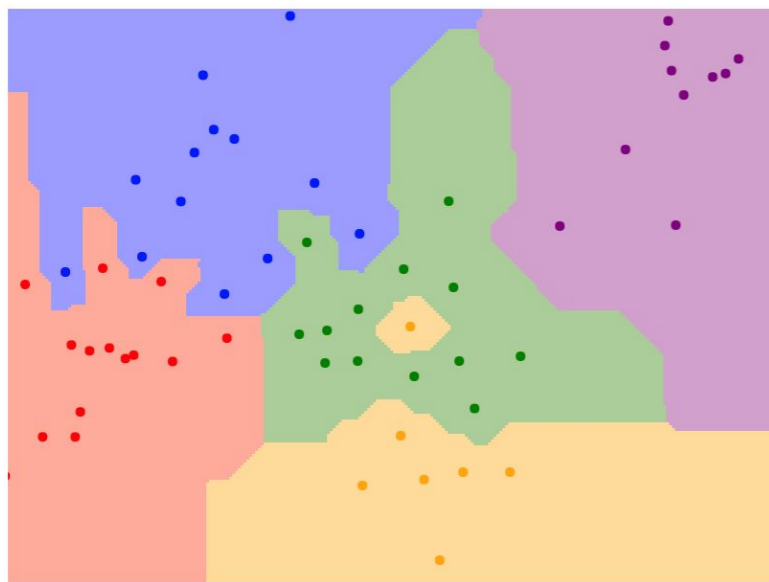
$$d_2(I_1, I_2) = \sqrt{\sum_P (I_1^P - I_2^P)^2}$$



The choice of distance is a **hyperparameter**
common choices:

L1 (Manhattan) distance

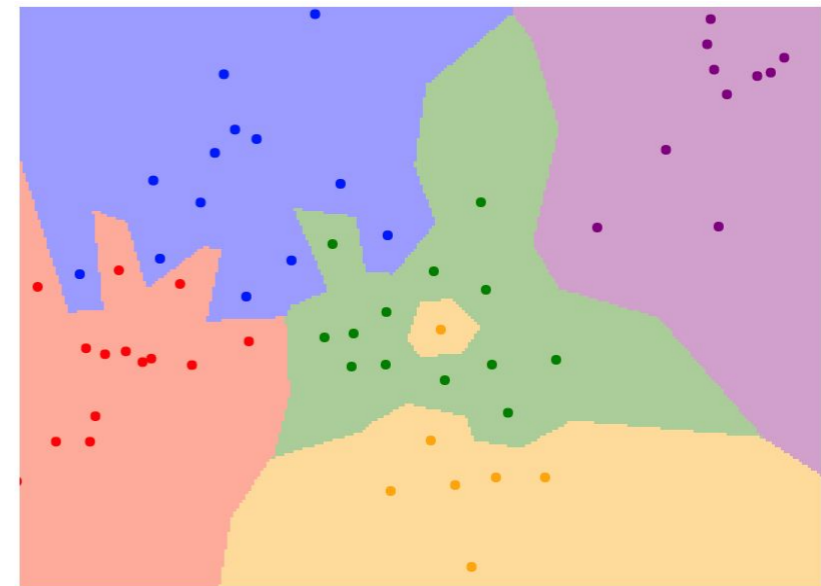
$$d_1(I_1, I_2) = \sum_P |I_1^P - I_2^P|$$



K = 1

L2 (Euclidean) distance

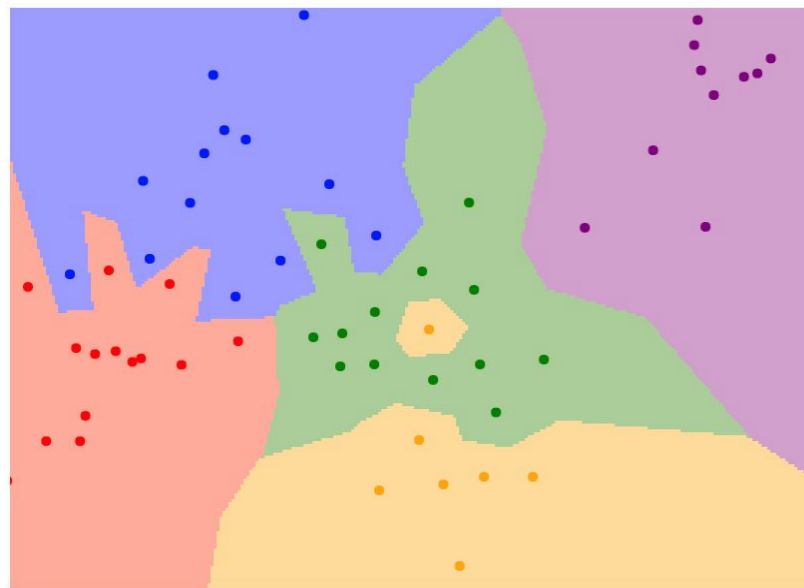
$$d_2(I_1, I_2) = \sqrt{\sum_P (I_1^P - I_2^P)^2}$$



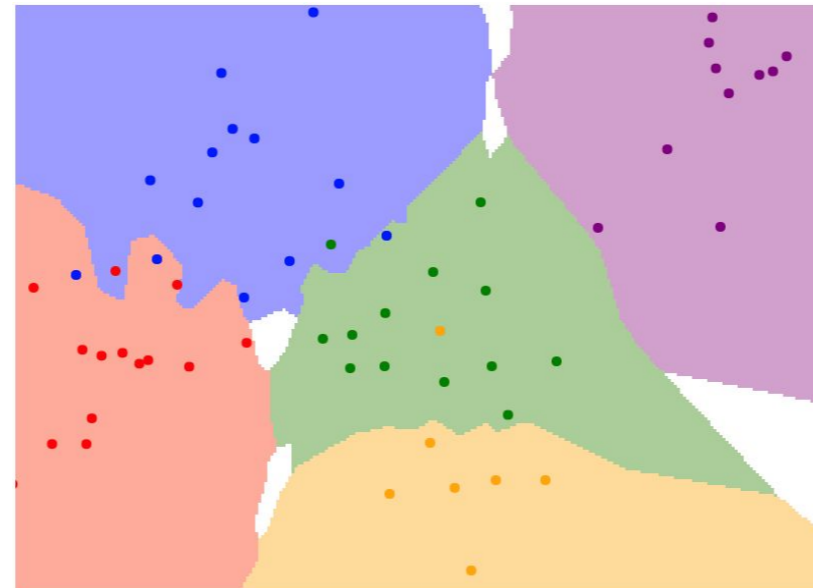
K = 1

k-Nearest Neighbor

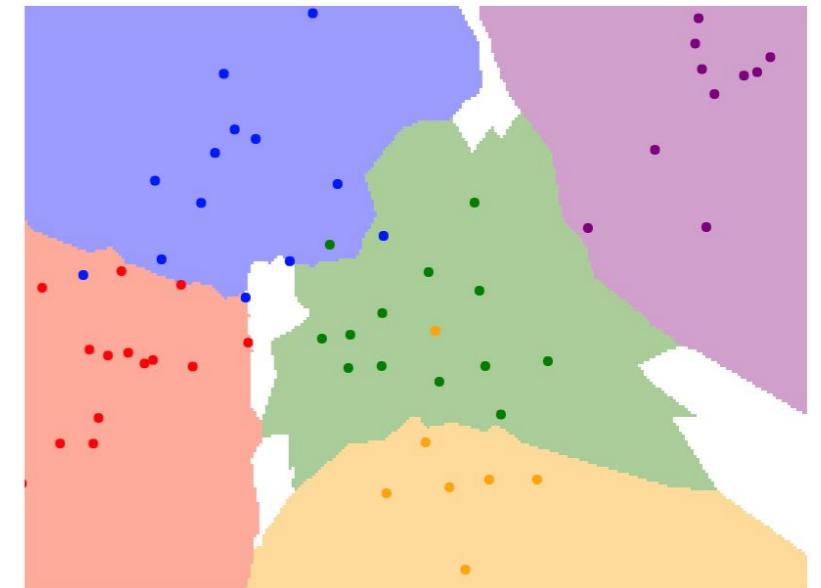
Instead of copying label from nearest neighbor, take **majority vote** from K closest points



$K = 1$



$K = 3$



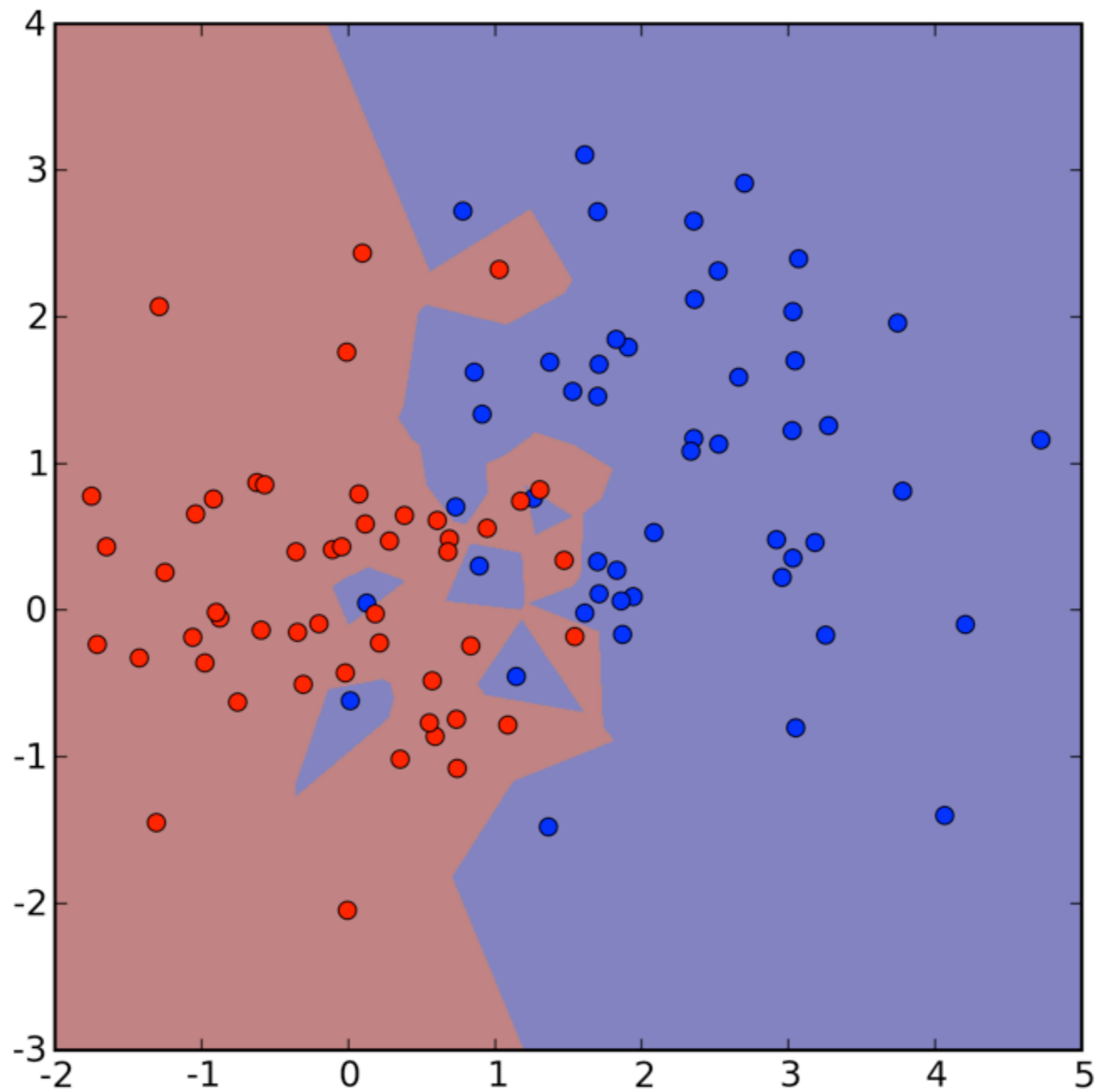
$K = 5$

K-Nearest Neighbor (kNN)

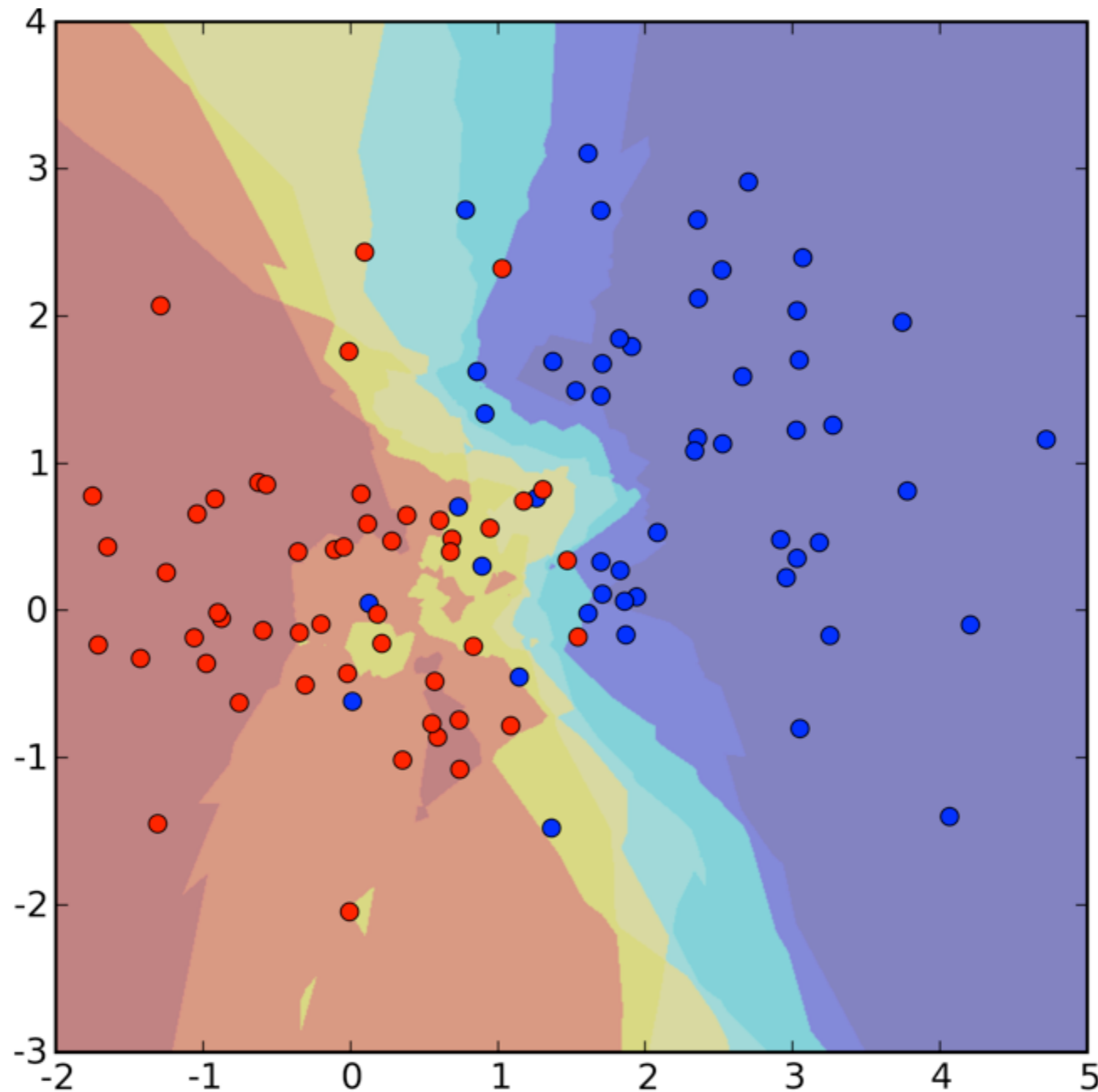
- Given: Training data $\{(x_1, y_1), \dots, (x_n, y_n)\}$
 - Attribute vectors: $x_i \in X$
 - Labels: $y_i \in Y$
- Parameter:
 - Similarity function: $K : X \times X \rightarrow R$
 - Number of nearest neighbors to consider: k
- Prediction rule
 - New example x'
 - K-nearest neighbors: k train examples with largest $K(x_i, x')$

$$h(\vec{x}') = \arg \max_{y \in Y} \left\{ \sum_{i \in knn(\vec{x}')} 1_{[y_i=y]} \right\}$$

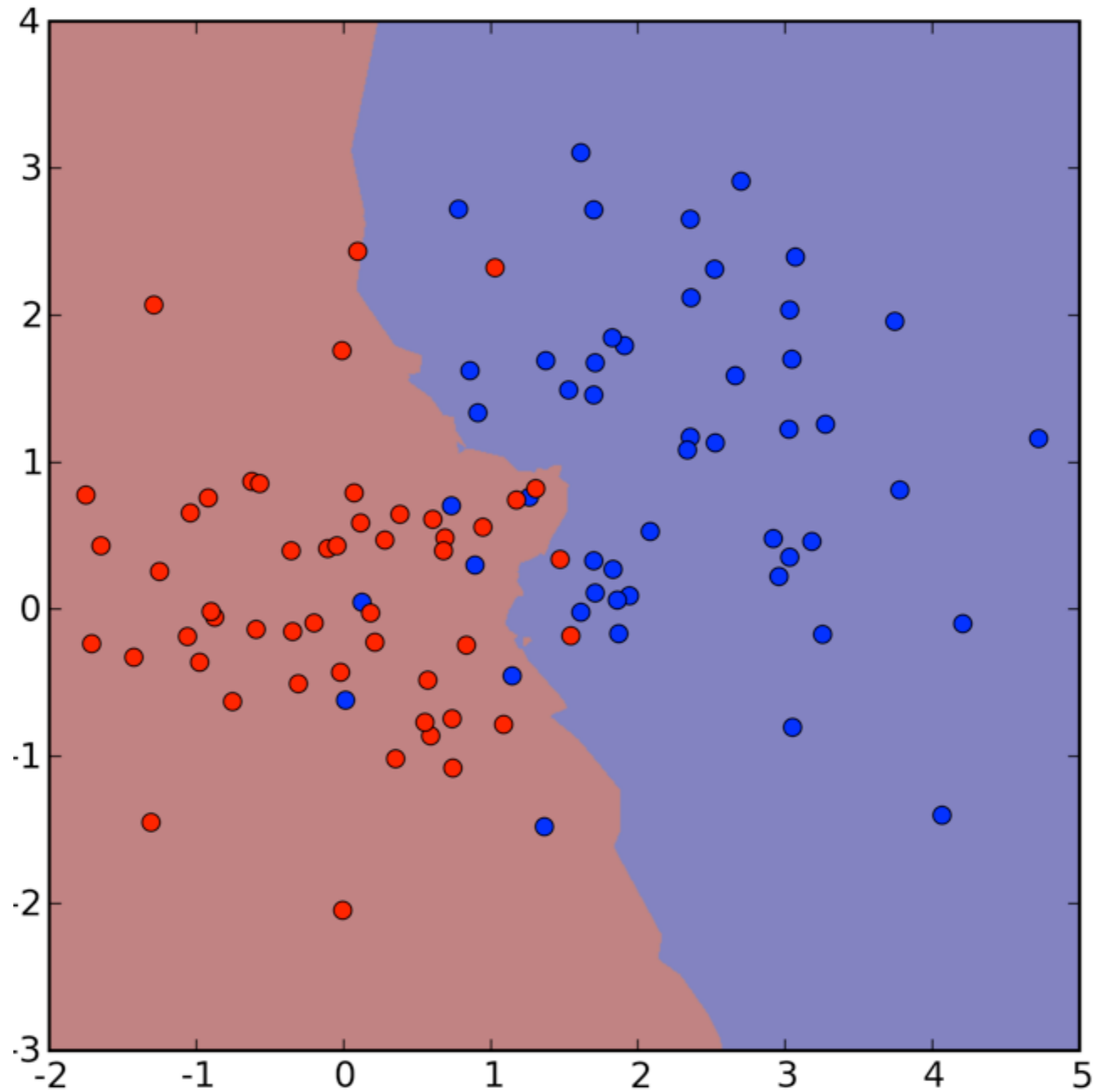
1-Nearest Neighbor



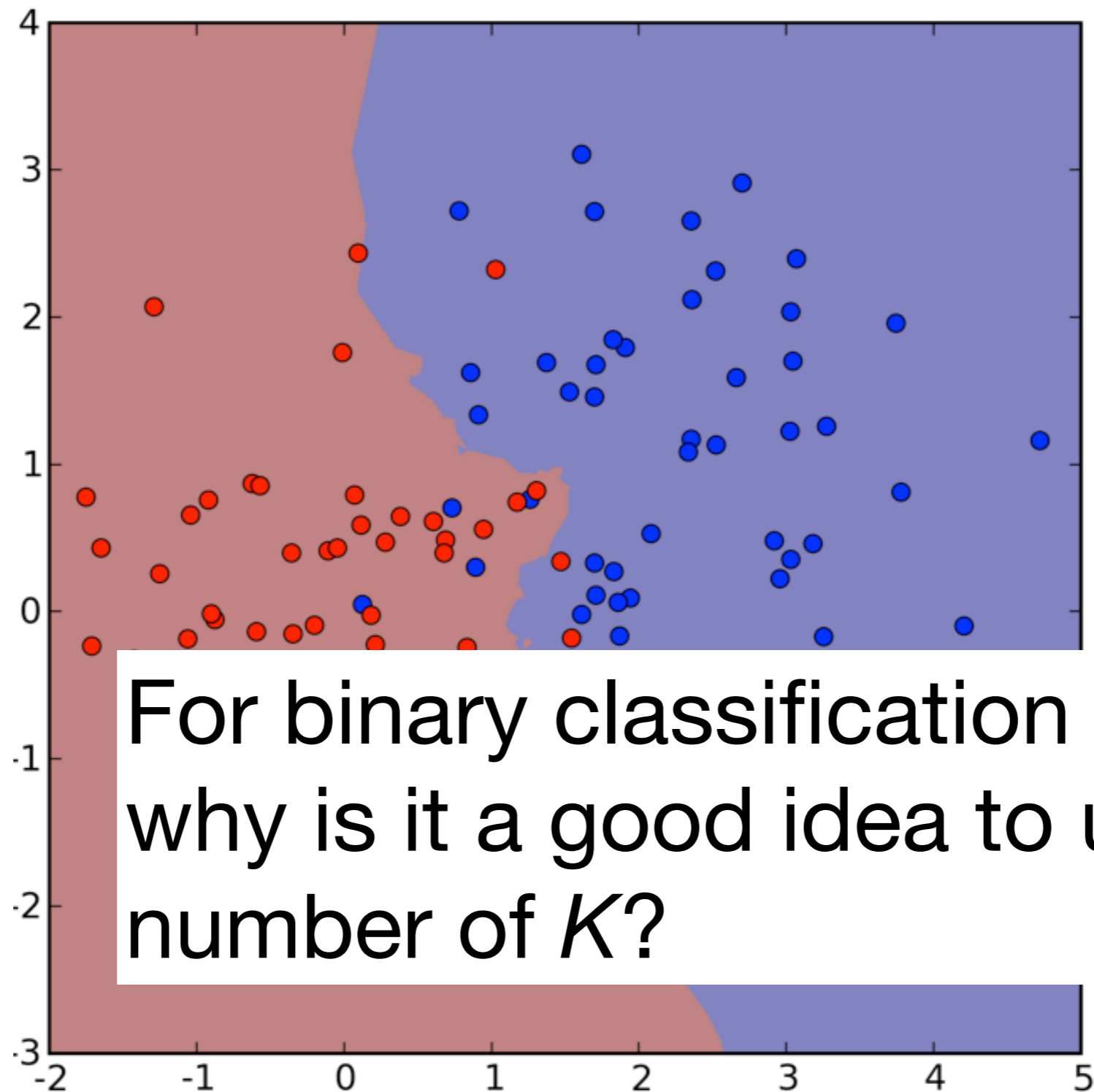
4-Nearest Neighbors



4-Nearest Neighbors Sign



4-Nearest Neighbors Sign



For binary classification problems, why is it a good idea to use an odd number of K ?

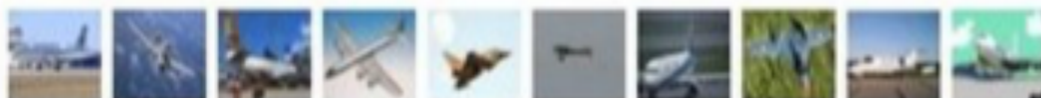
Example dataset: **CIFAR-10**

10 labels

50,000 training images

10,000 test images.

airplane



automobile



bird



cat



deer



dog



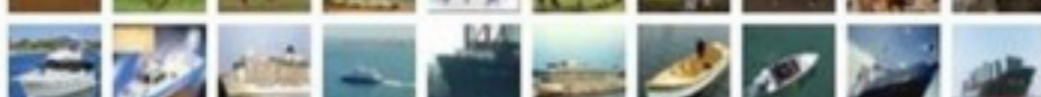
frog



horse



ship



truck



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



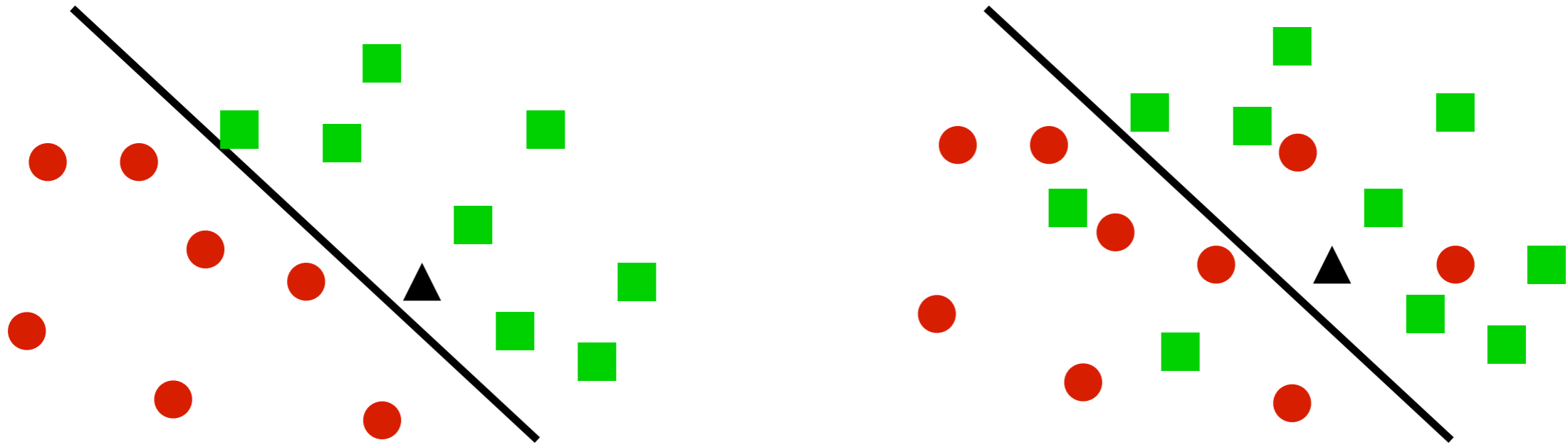
What is the best **distance** to use?

What is the best value of **k** to use?

i.e. how do we set the **hyperparameters**?

We will talk about this later!

If we get more data



- 1 Nearest Neighbor
 - Converges to perfect solution if clear separation
 - Twice the minimal error rate $2p(1-p)$ for noisy problems
- k-Nearest Neighbor
 - Converges to perfect solution if clear separation **(but needs more data)**
 - Converges to minimal error $\min(p, 1-p)$ for noisy problems if k increases

Demo

<http://vision.stanford.edu/teaching/cs231n-demos/knn/>

Weighted K-Nearest Neighbor

- Given: Training data $\{(x_1, y_1), \dots, (x_n, y_n)\}$
 - Attribute vectors: $x_i \in X$
 - Target attribute $y_i \in Y$
- Parameter:
 - Similarity function: $K : X \times X \rightarrow R$
 - Number of nearest neighbors to consider: k
- Prediction rule
 - New example x'
 - K-nearest neighbors: k train examples with largest $K(x_i, x')$

$$h(\vec{x}') = \arg \max_{y \in Y} \left\{ \sum_{i \in knn(\vec{x}')} 1_{[y_i=y]} K(\vec{x}_i, \vec{x}') \right\}$$

More Nearest Neighbors in Visual Data

Where in the World? [Hays & Efros, CVPR 2008]

A nearest neighbor recognition example



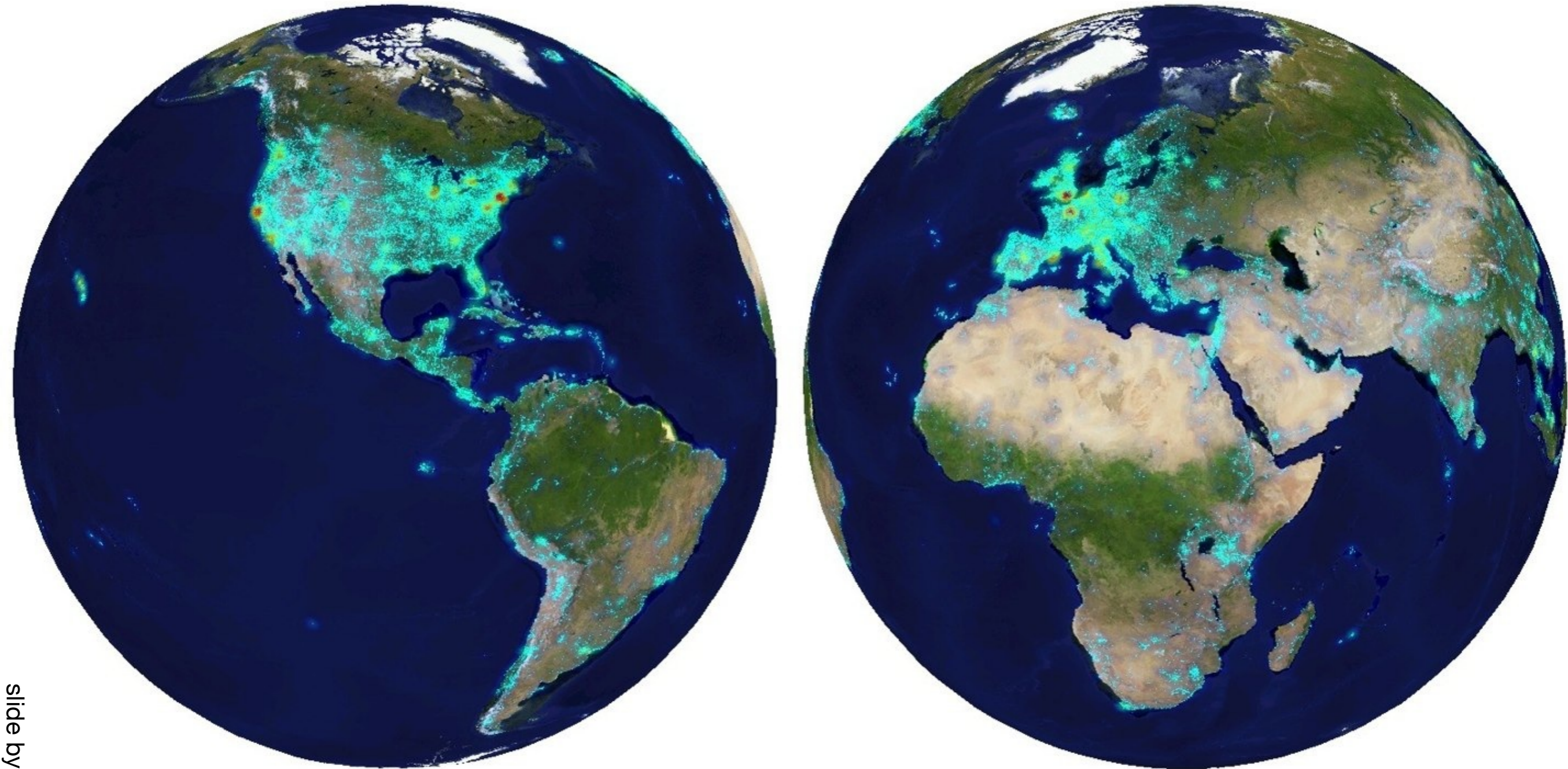
Where in the World? [Hays & Efros, CVPR 2008]



Where in the World? [Hays & Efros, CVPR 2008]

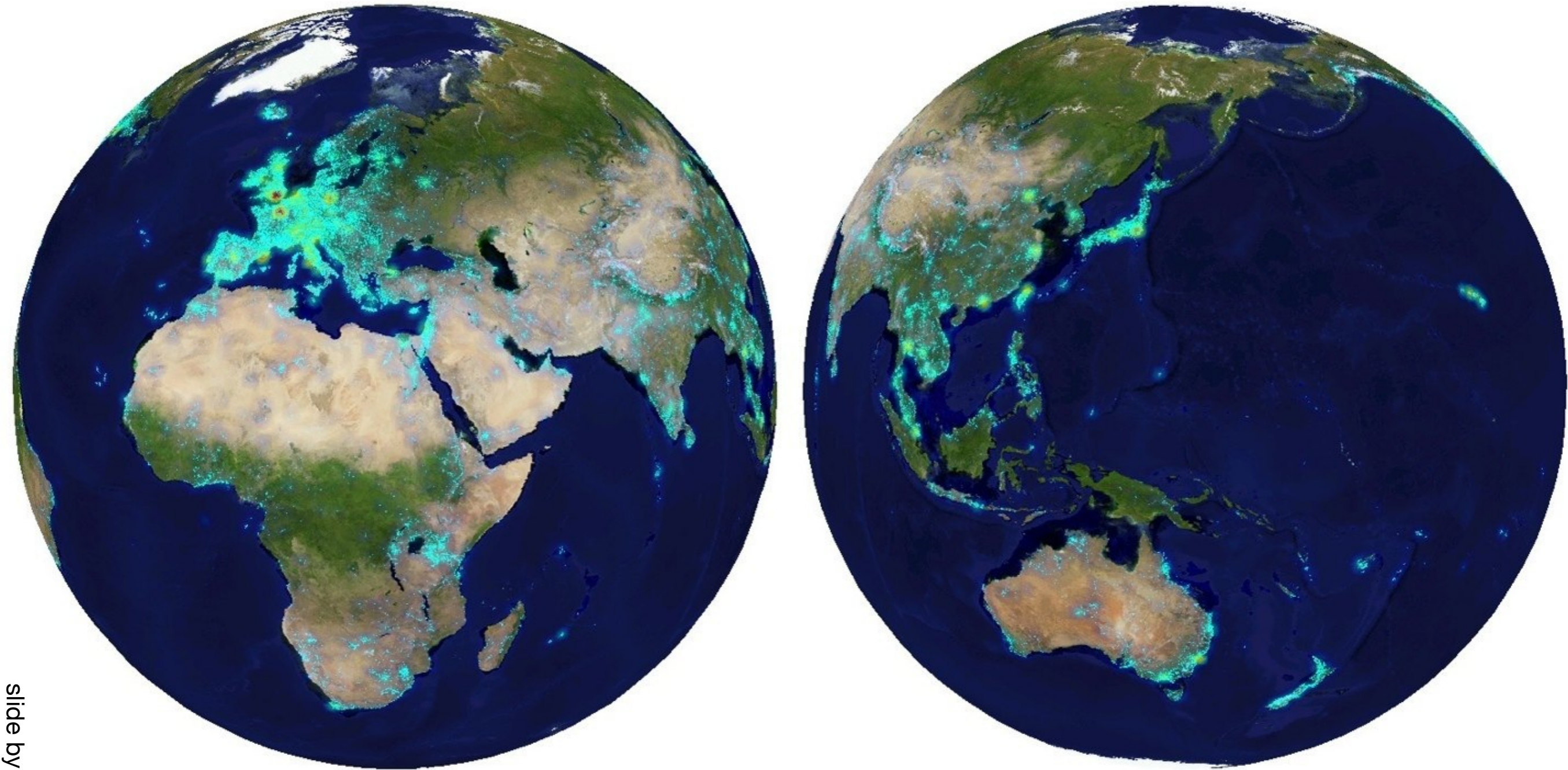


6+ million geotagged photos
by 109,788 photographers



Annotated by Flickr users

6+ million geotagged photos
by 109,788 photographers



Annotated by Flickr users



Scene Matches



Madrid



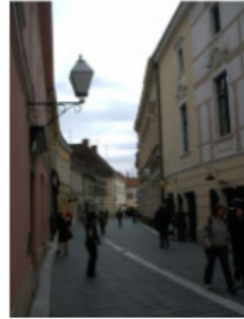
england



France



Paris



Croatia



heidelberg



Macau



Malta



Cairo



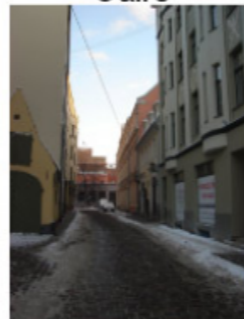
Italy



Italy



Italy



Latvia



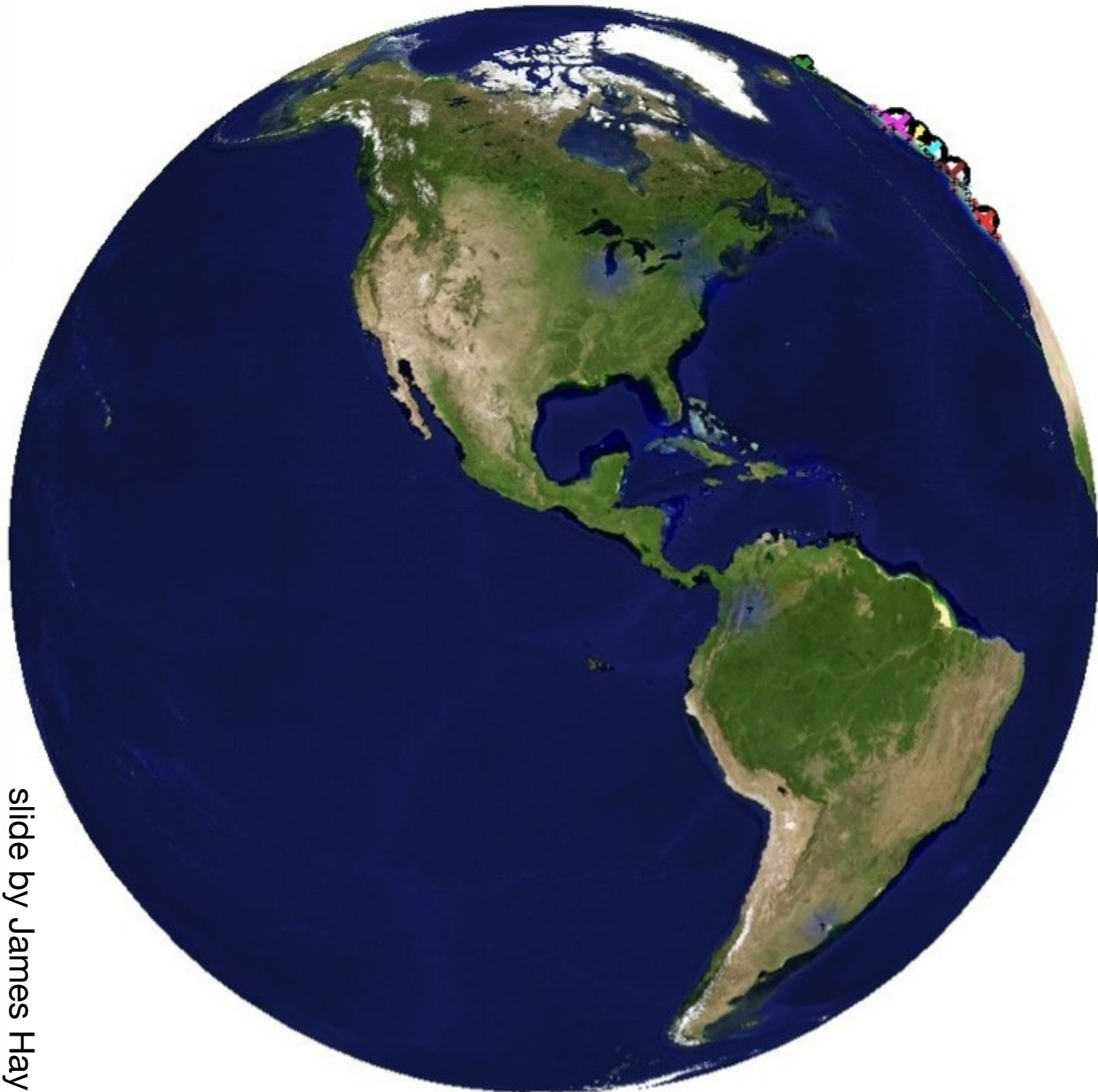
europa



Barcelona



Austria



Scene Matches



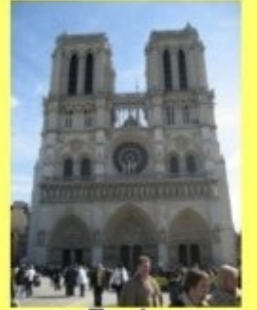
Paris



Paris



Paris



Paris



Paris



Paris



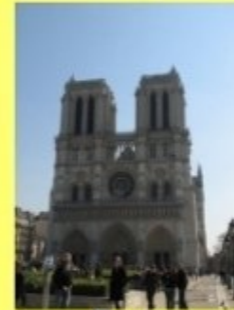
Paris



Madrid



Rome



Paris



Cuba



Paris



Paris



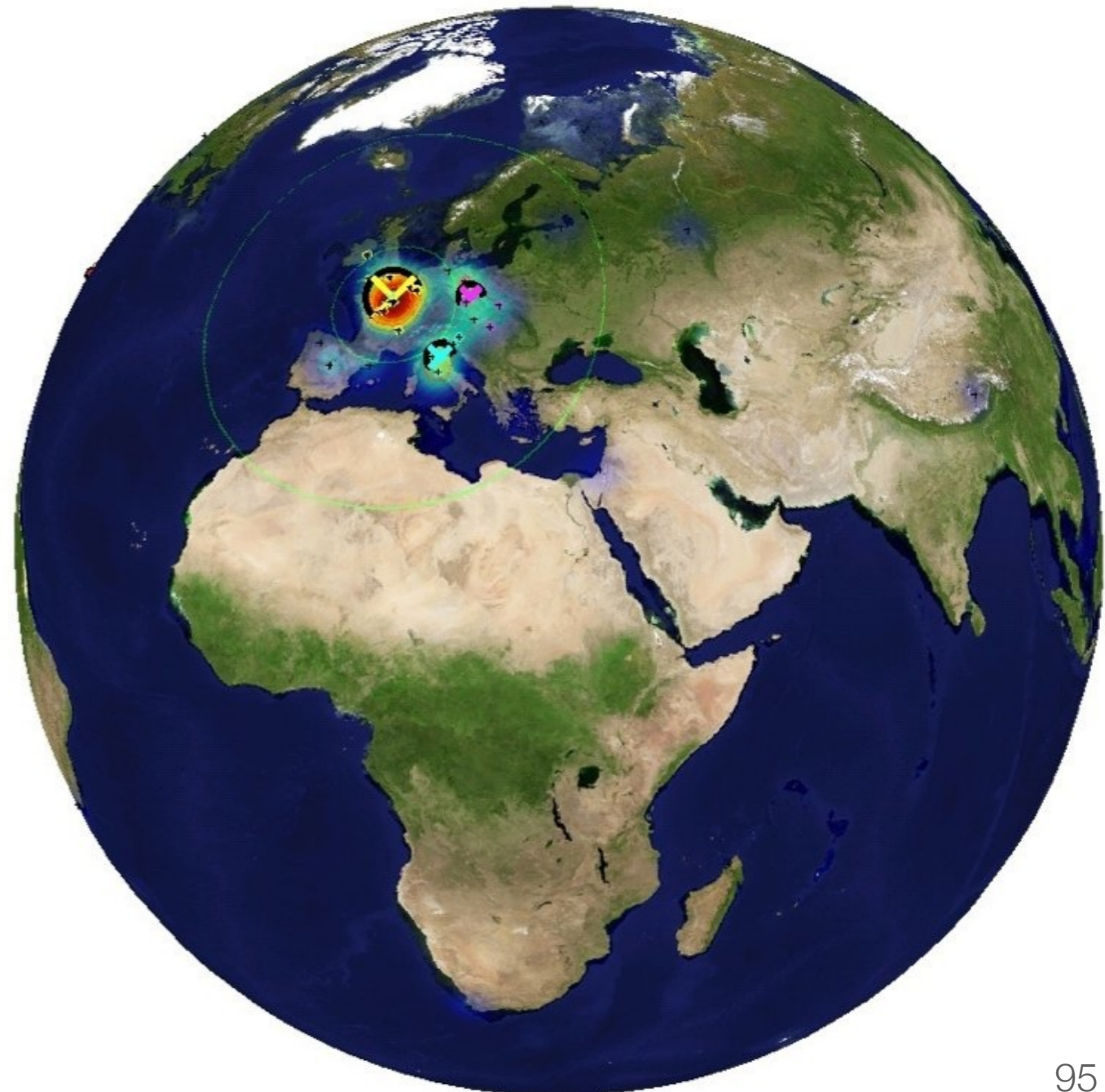
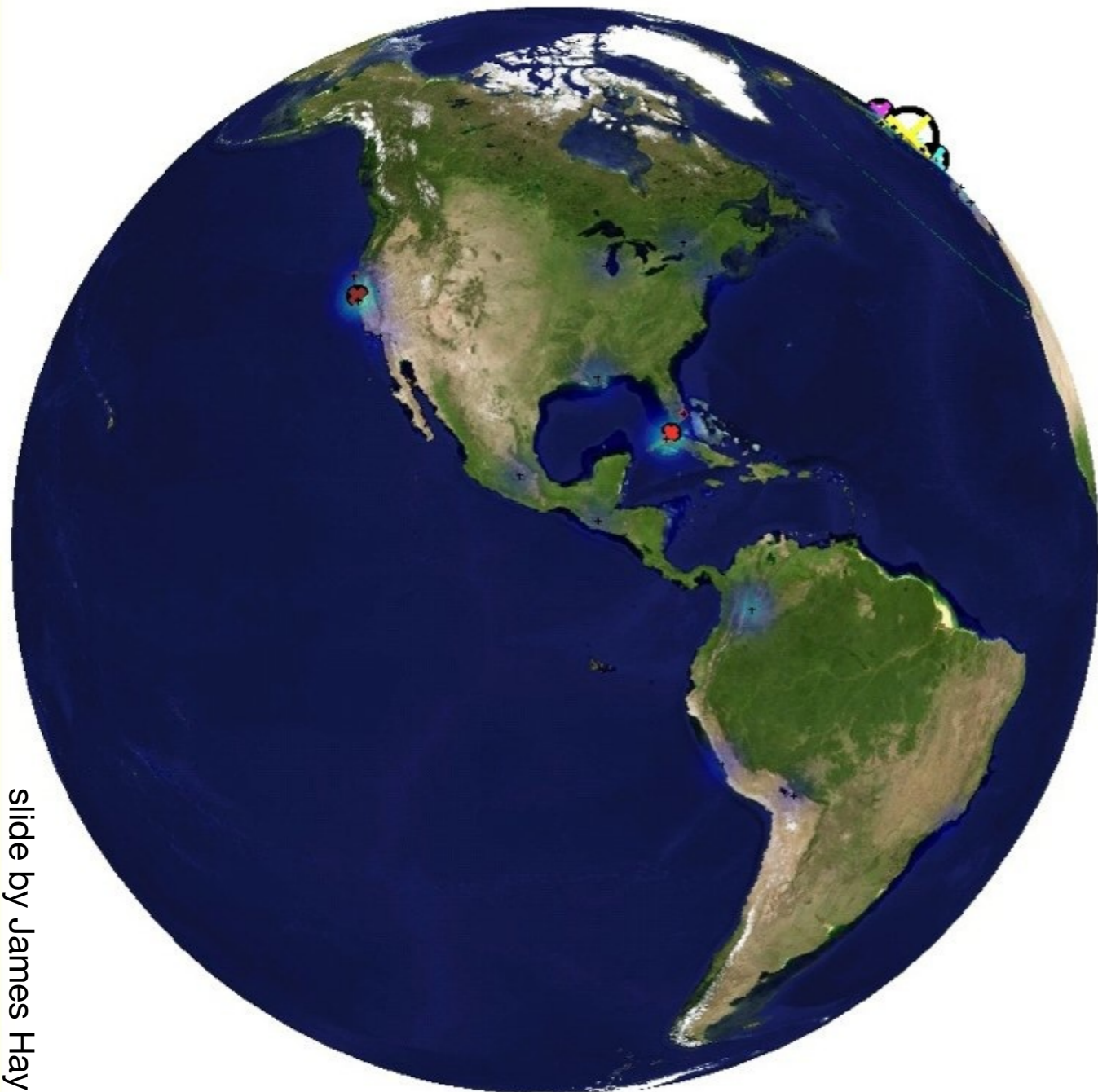
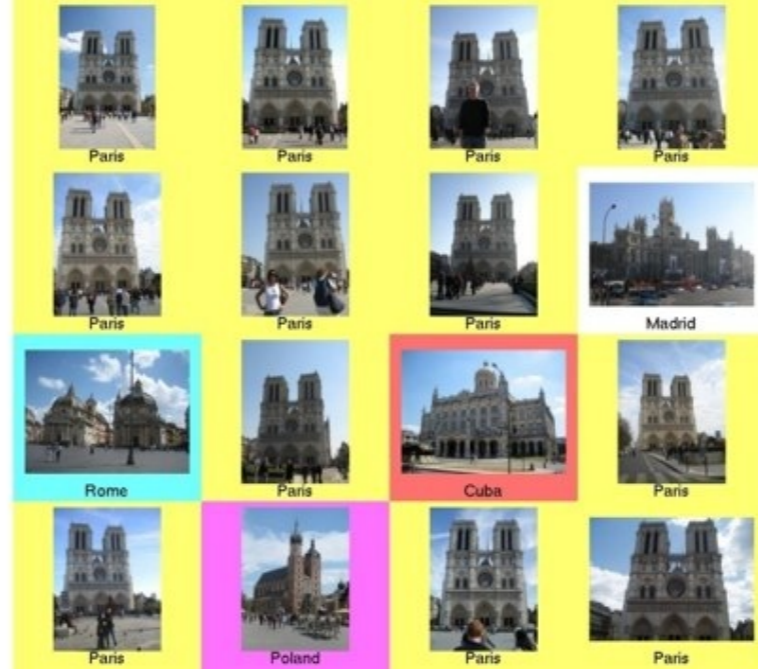
Poland



Paris



Paris



Scene Matches



Philippines



Houston



Thailand



Houston



Maldives



Philippines



New Zealand



Bermuda



Palau



Mexico2



Brazil



Mendoza



Brazil



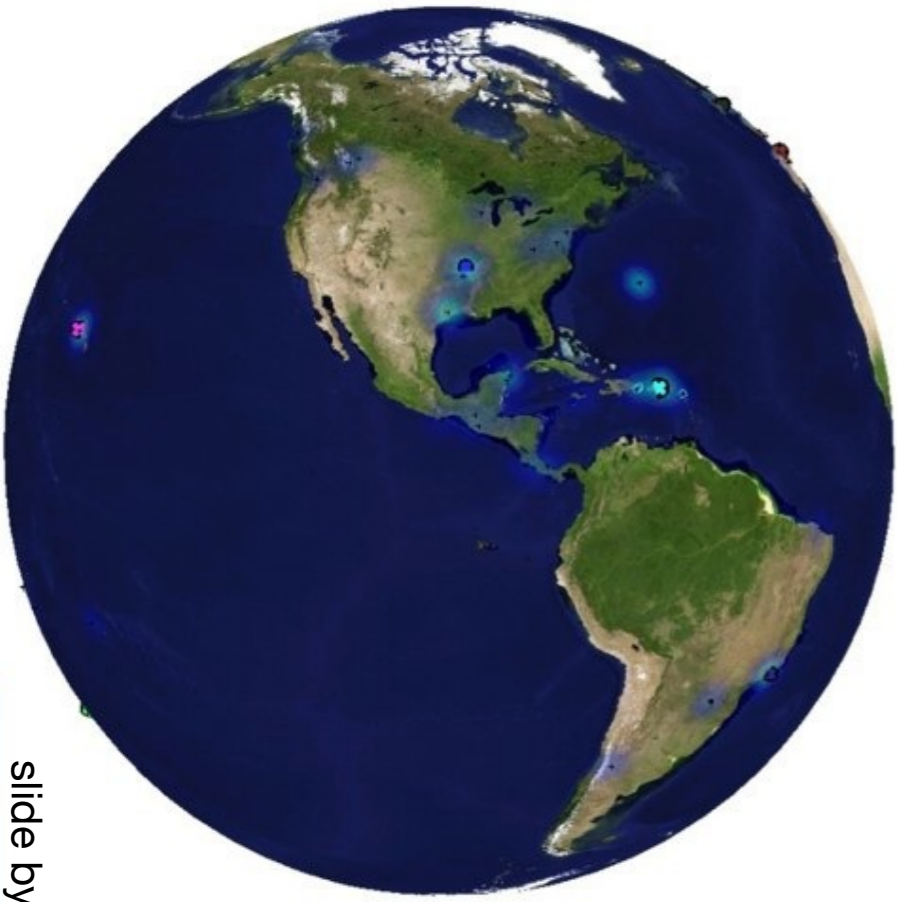
Thailand



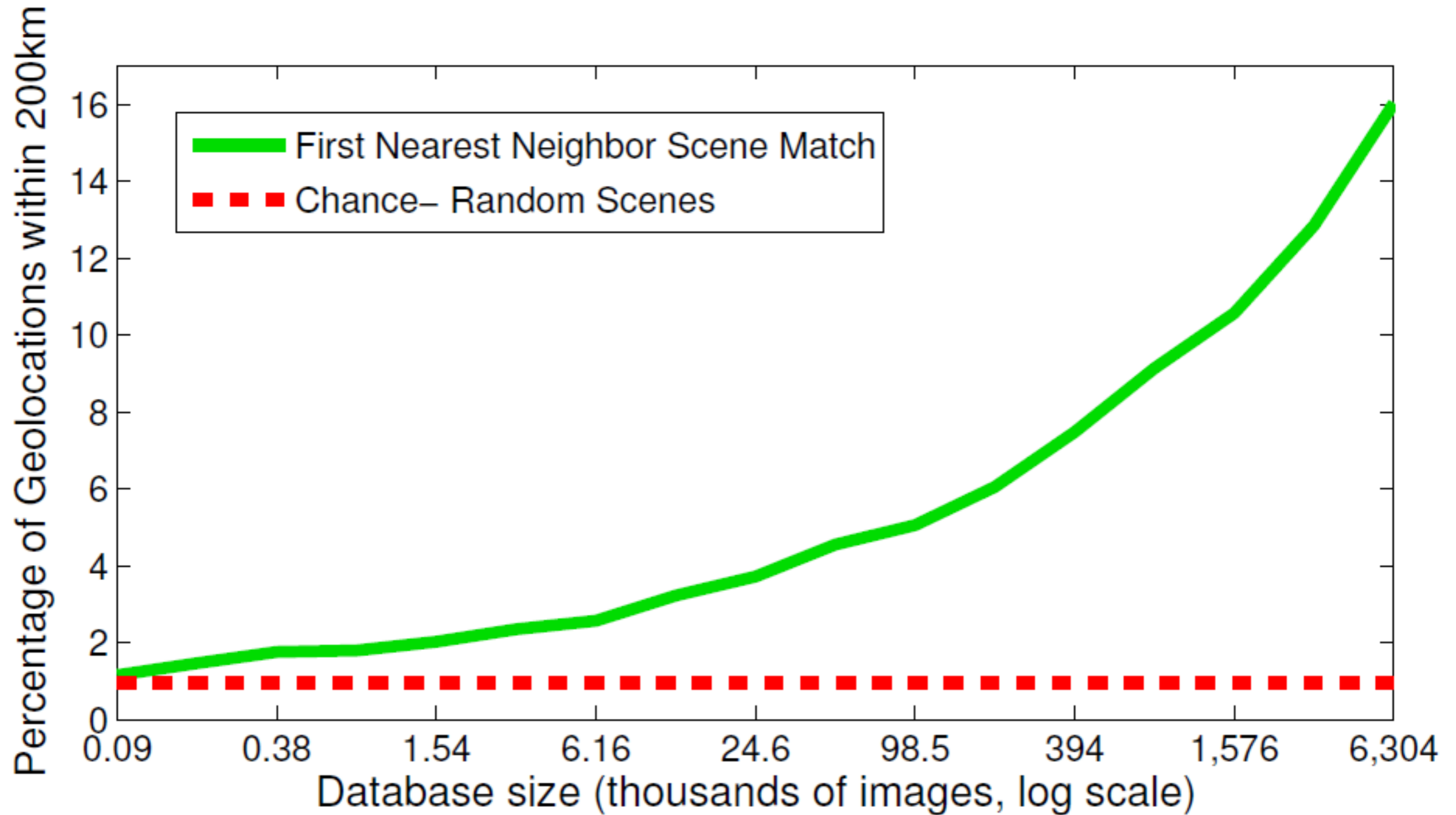
Arkansas



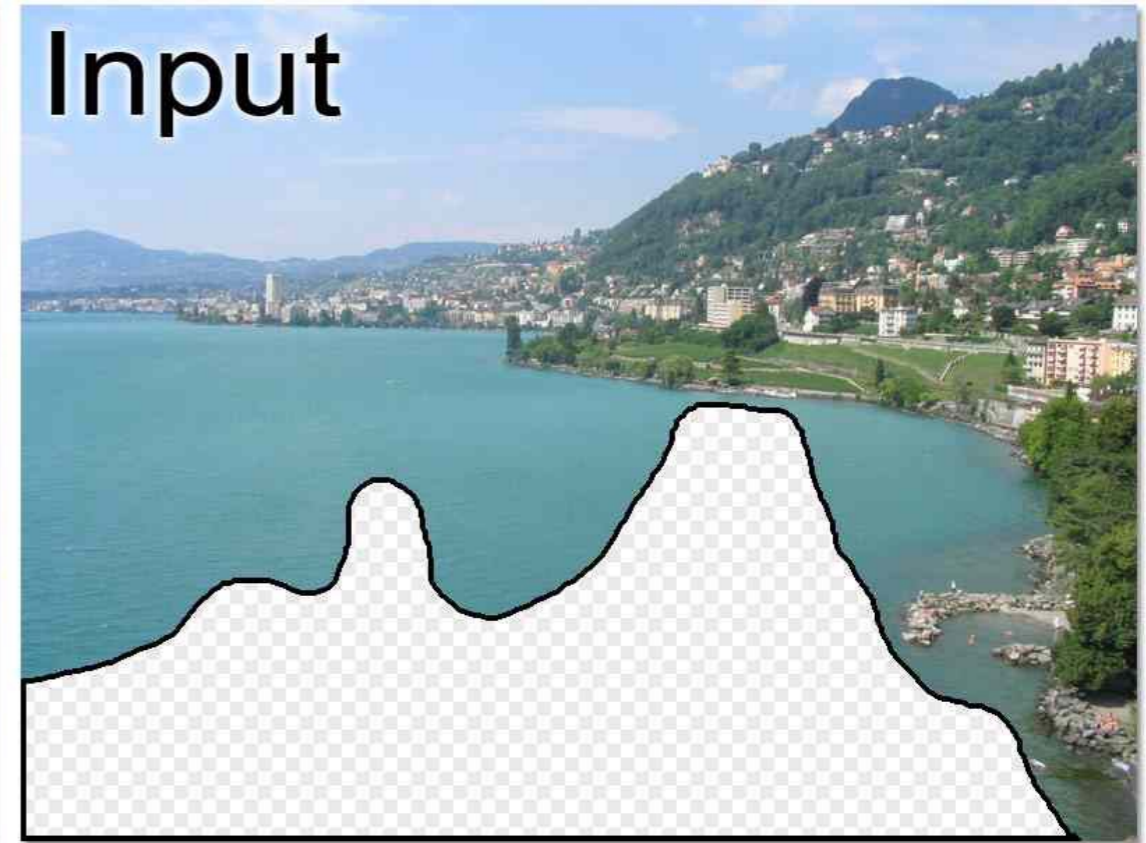
Hawaii

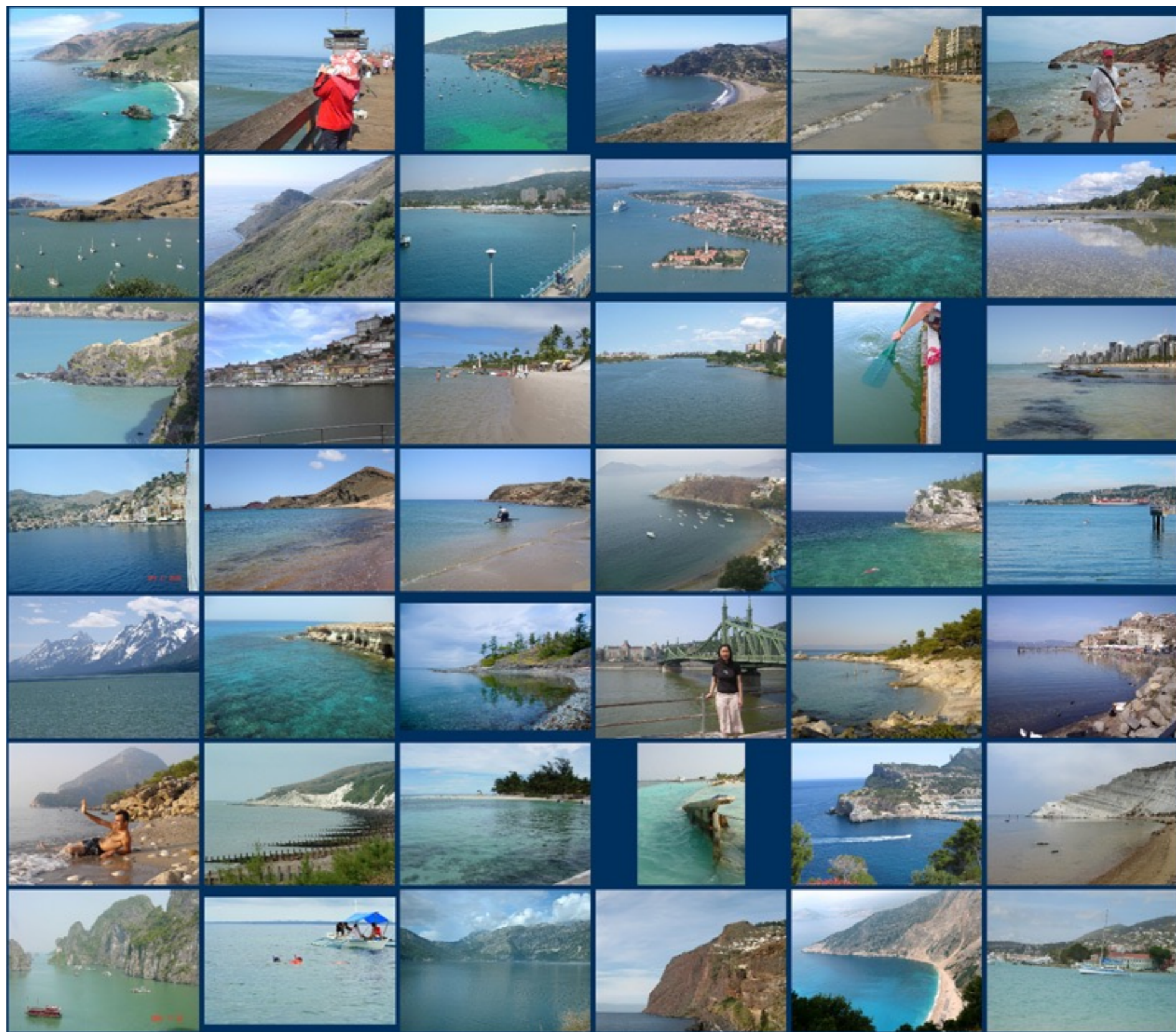
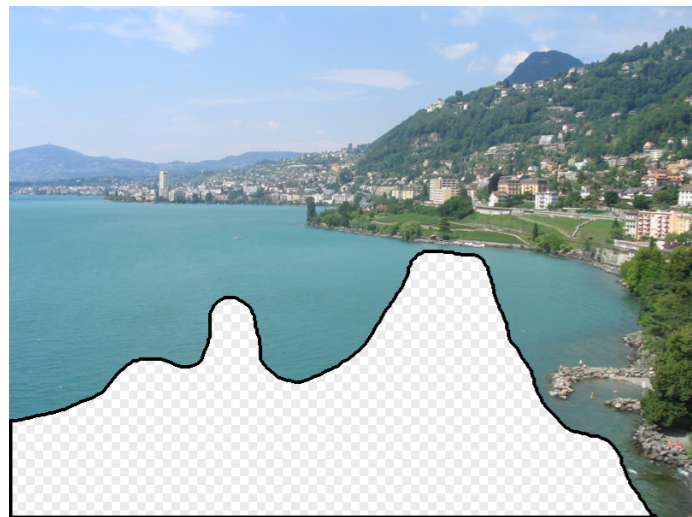


The Importance of Data



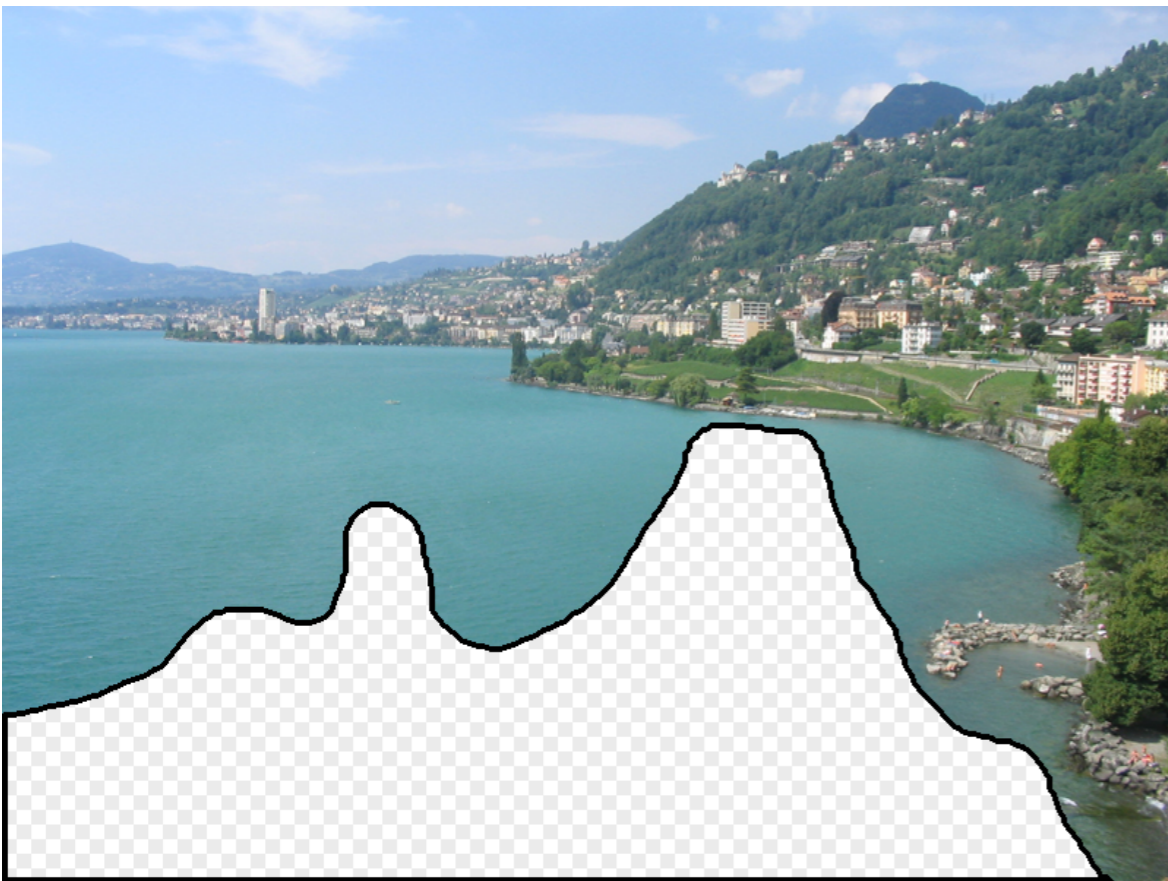
Scene Completion [Hays & Efros, SIGGRAPH07]





... 200 total

Context Matching





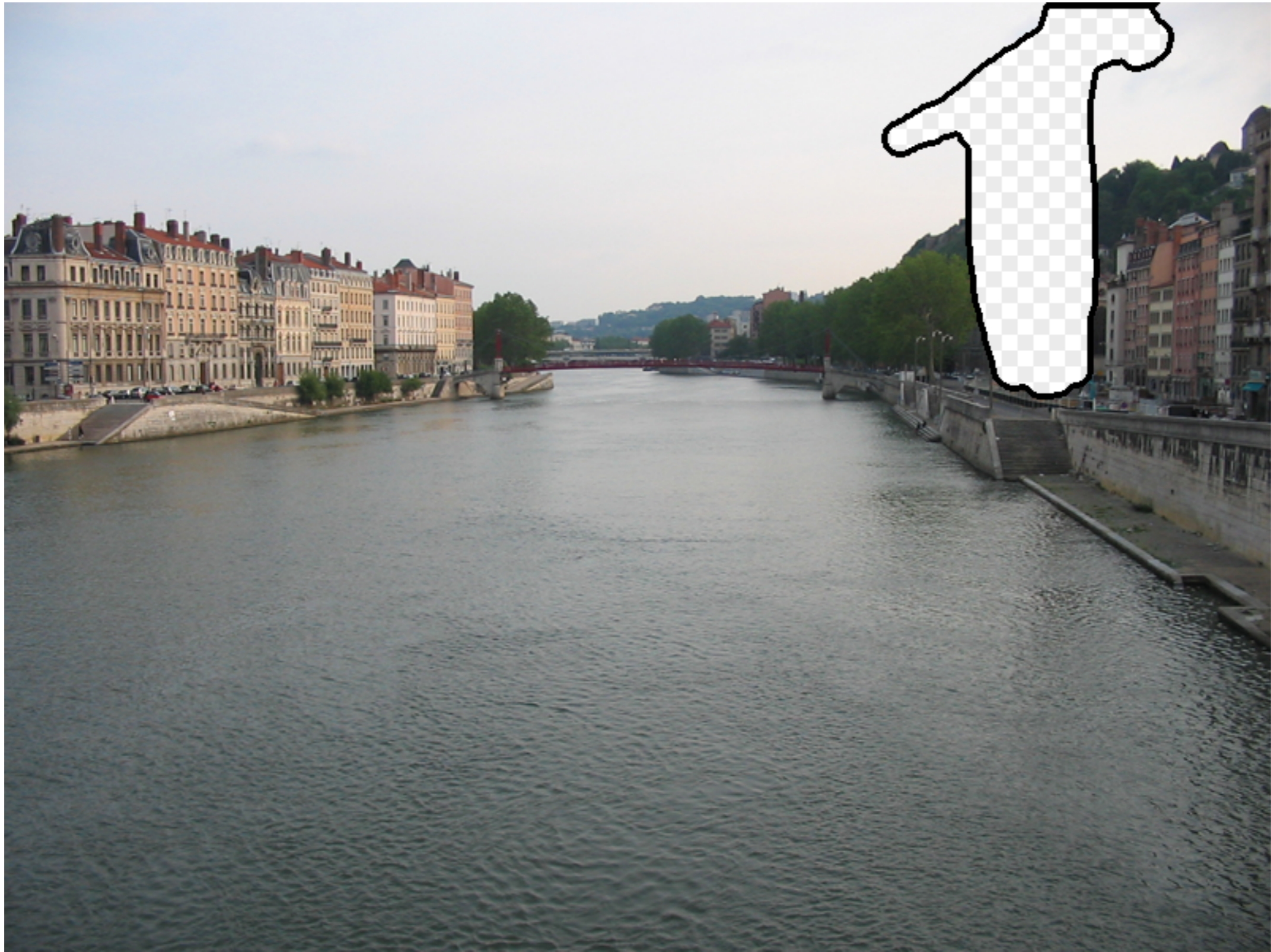
Graph cut + Poisson blending











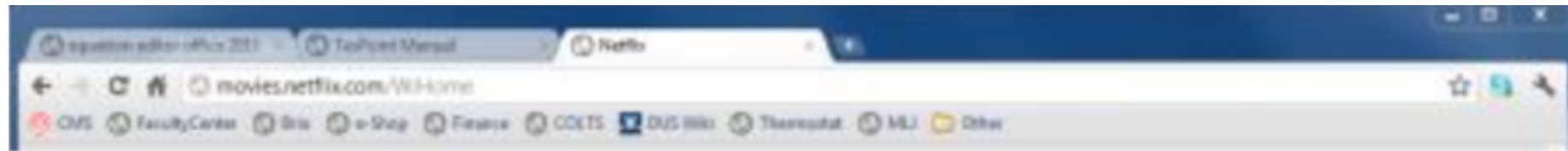


Weighted K-NN for Regression

- Given: Training data $\{(x_1, y_1), \dots, (x_n, y_n)\}$
 - Attribute vectors: $x_i \in X$
 - Target attribute $y_i \in \mathcal{R}$
- Parameter:
 - Similarity function: $K : X \times X \rightarrow \mathcal{R}$
 - Number of nearest neighbors to consider: k
- Prediction rule
 - New example x'
 - K-nearest neighbors: k train examples with largest $K(x_i, x')$

$$h(\vec{x}') = \frac{\sum_{i \in knn(\vec{x}')} y_i K(\vec{x}_i, \vec{x}')}{\sum_{i \in knn(\vec{x}')} K(\vec{x}_i, \vec{x}')}$$

Collaborative Filtering



Rating Matrix	m_1	m_2	m_3	m_4	m_5	m_6
u_1		1	5		3	5
u_2		5	1	1	3	1
u_3		2	4		1	5
u	?	1	4	?	?	?



Overview of Nearest Neighbors

- Very simple method
- Retain all training data
 - Can be slow in testing
 - Finding NN in high dimensions is slow
- Metrics are very important
- Good baseline

Next Class:

Kernel Regression,
Distance Metrics,
Curse of Dimensionality