Machine Learning

Lecture 1: introduction

Jan Chorowski
Instytut Informatyki
Wydział Matematyki i Informatyki Uniwersytet Wrocławski

2018-2019

Slides taken from presentations by: A. Ng, P. Vincent, Y. LeCun, W. Zaremba, R. Salakhutdinov, L. Fei-Fei, A. Karpathy, argmax.ai ML course

Practical Information

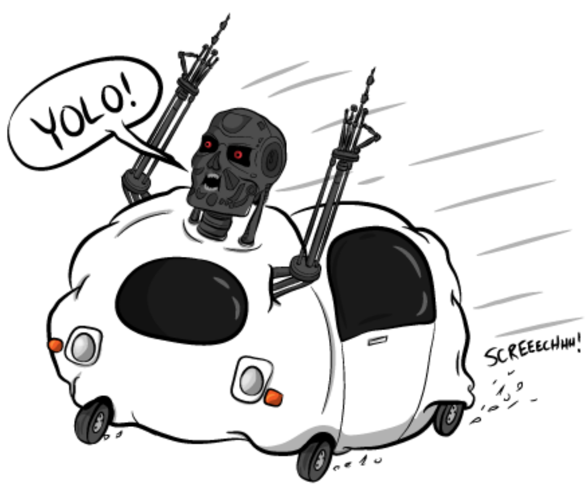
- Course Materials:
  - SKOS (anouncements, course rules etc.): https://skos.ii.uni.wroc.pl/course/view.php?id=224
  - Github (lecture notes, assignment notebooks): https://github.com/janchorowski/ml_uwr
  - USOS: grades
- On-line Resources:
  - http://cs229.stanford.edu/
  - https://argmax.ai/ml-course/
- Textbook
  - Kevin Murphy "Machine Learning"
- Extra reading
  - Bishop, Pattern Recognition and Machine Learning (PRML)

ML INTUITIONS

Where is machine learning?

Everywhere!

- web search ☺
- and ads ☹
- recommendations
- Self-driving cars



http://theoatmeal.com/blog/google_self_driving_car

Supervised Learning

$$I \rightarrow O$$

- Classification: assign data to classes
- Regression: compute numerical outcomes
- Structure prediction

ML Example: classification

- Assign small images to one of ten categories

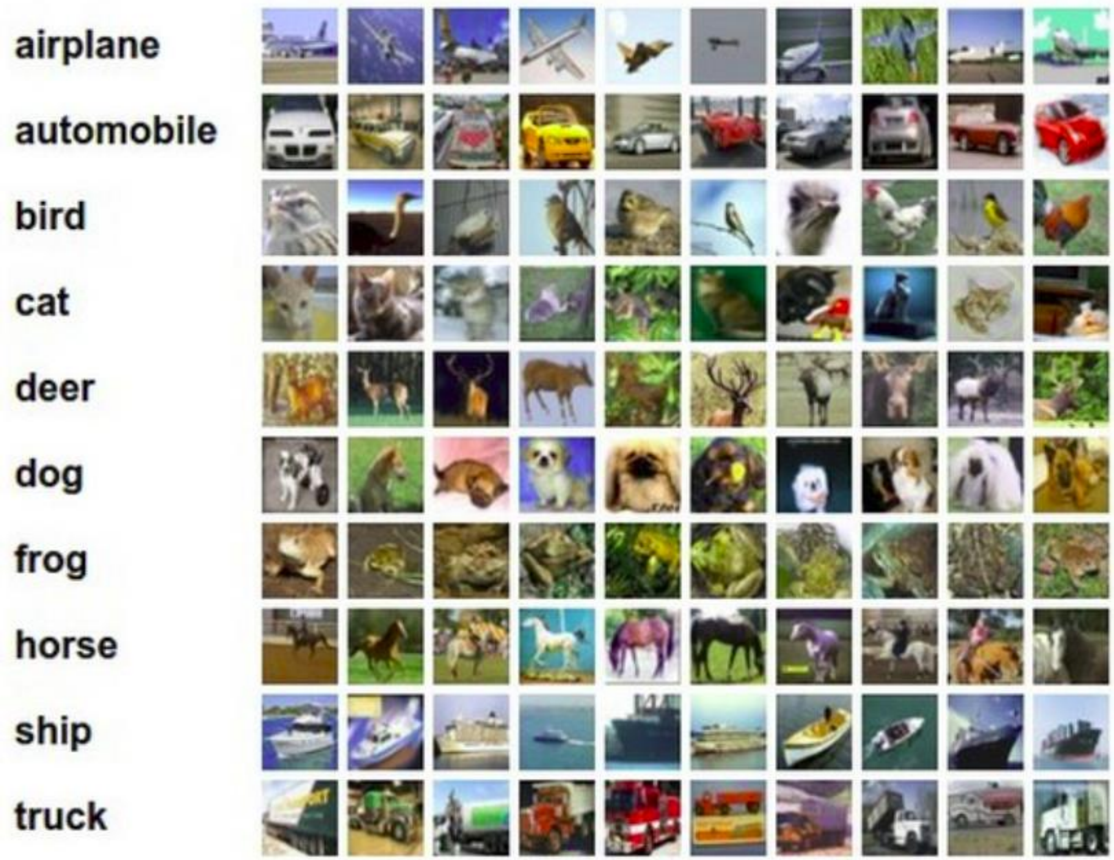
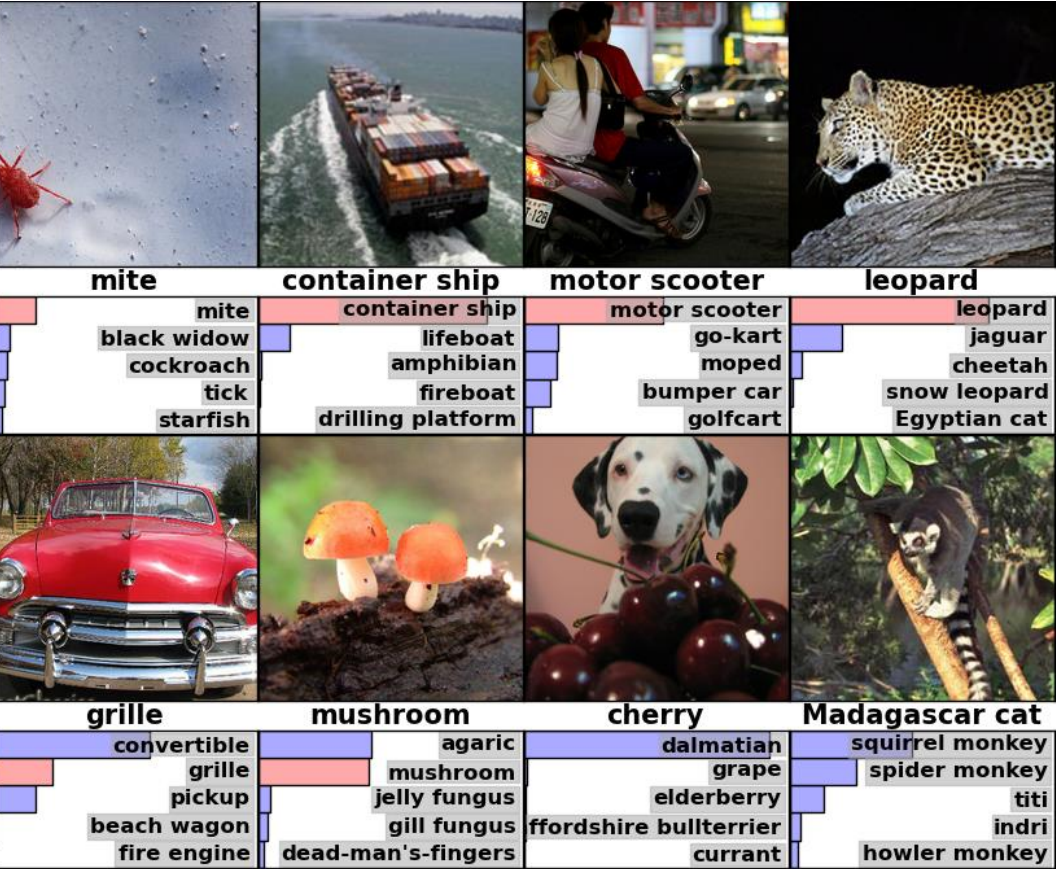
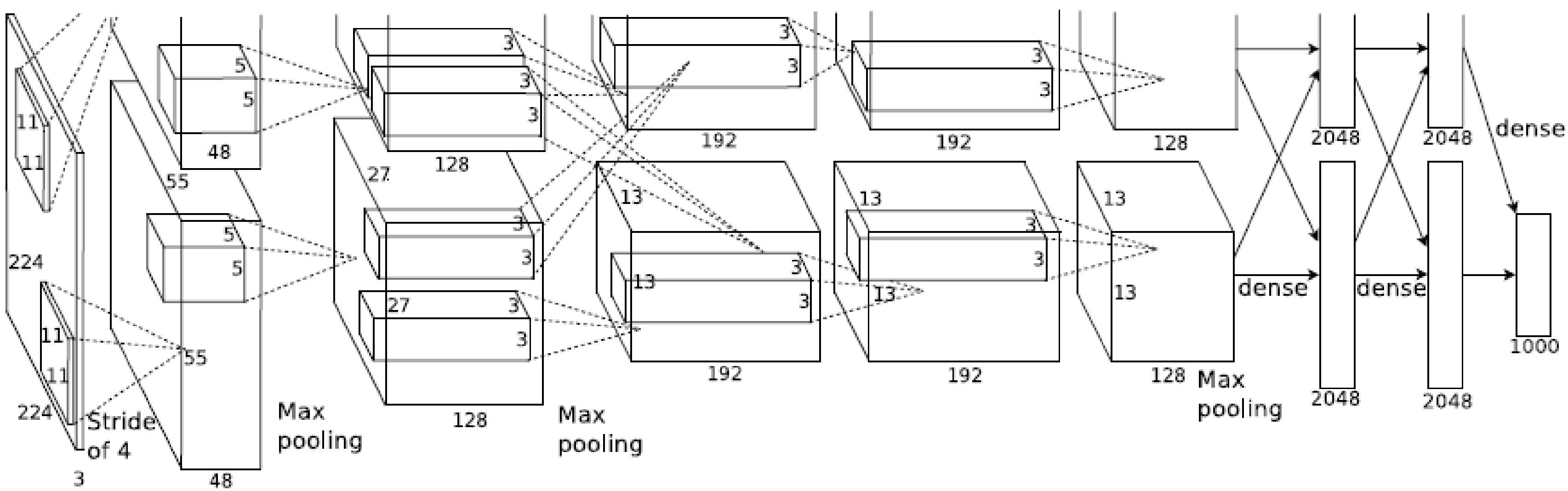
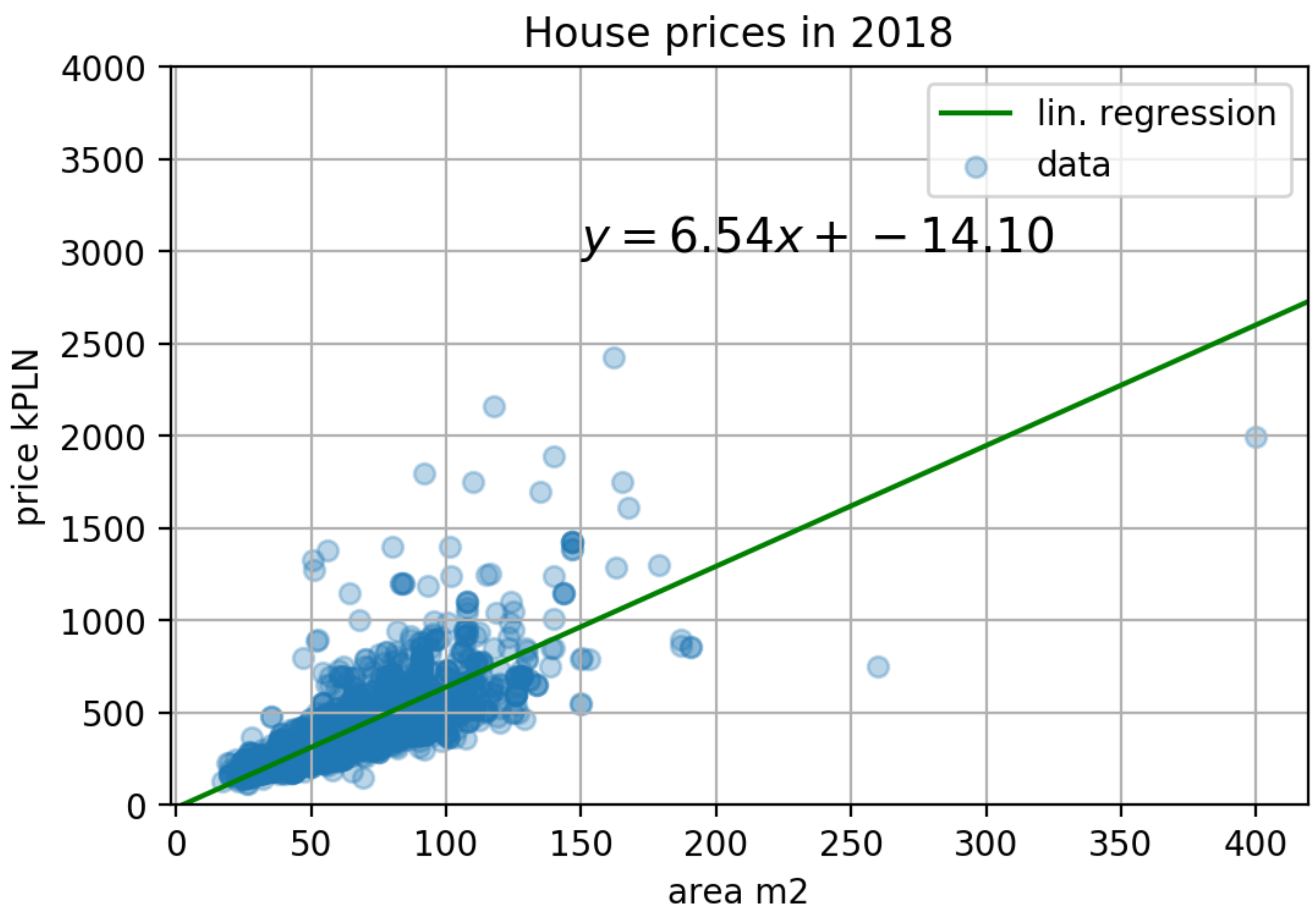


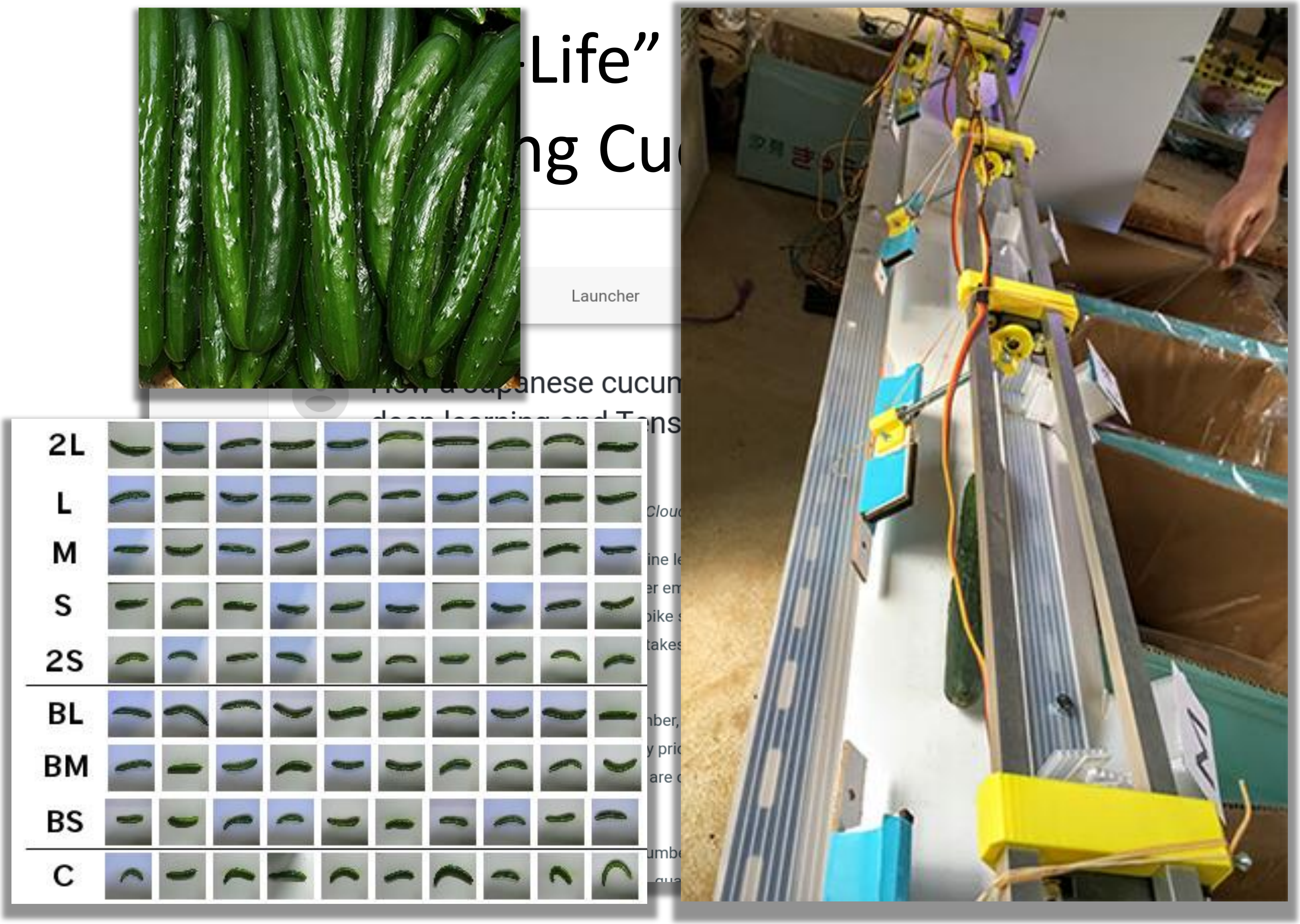
Image recognition





ML Example: Regression

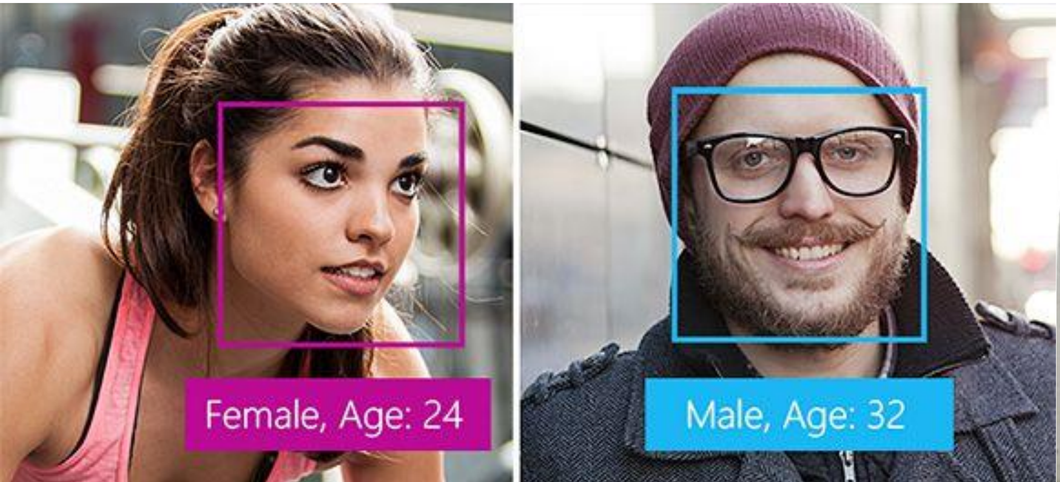


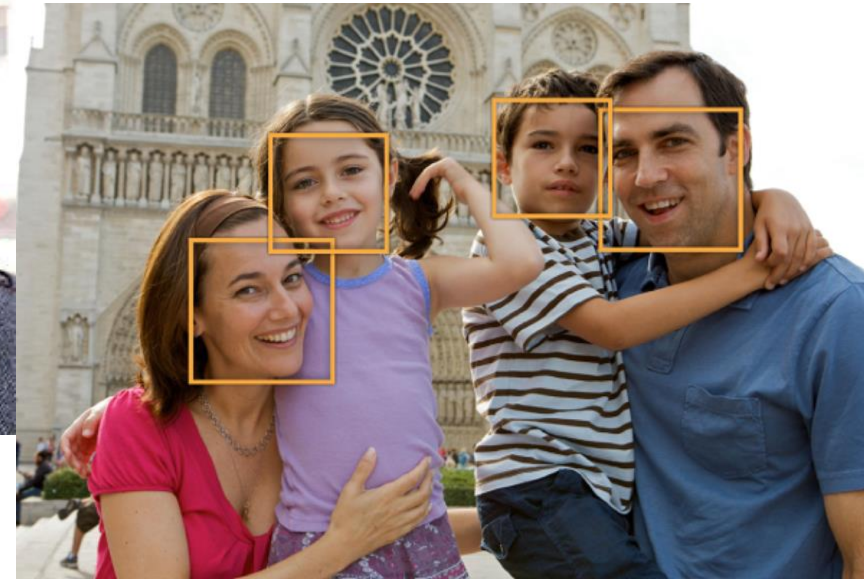


Source: https://cloud.google.com/blog/big-data/2016/08/how-a-japanese-cucumber-farmer-is-using-deep-learning-and-tensorflow

ML example: face detection

classify each image patch as face/non-face

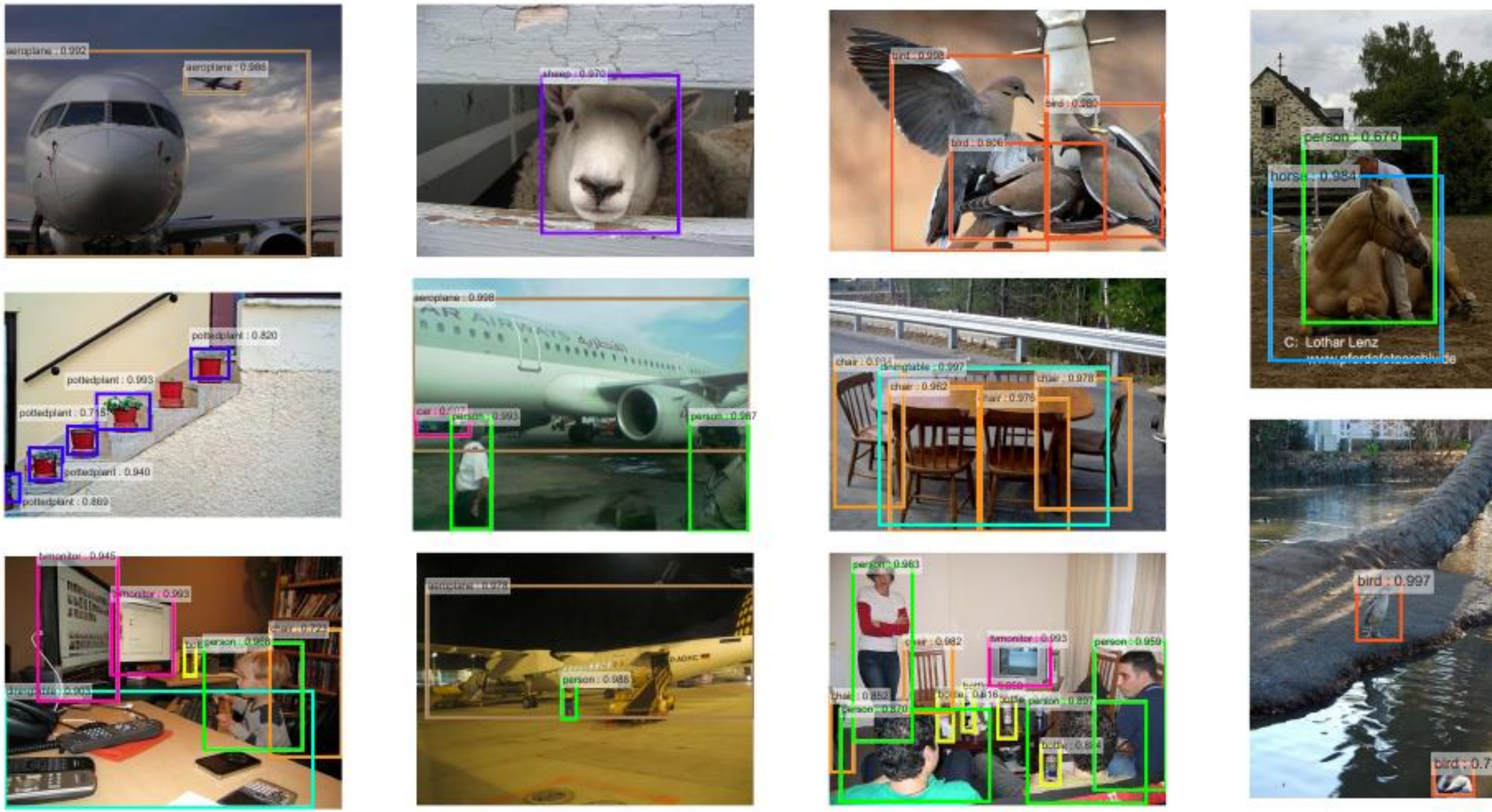




Source: Microsoft and Apple face detection API documentations, wykop.pl

Object Detection

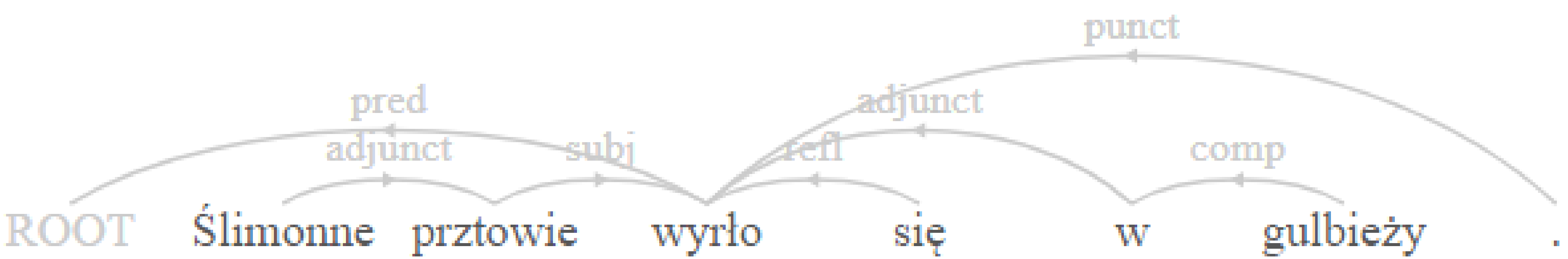
https://www.youtube.com/watch?v=WZmSMkK9VuA



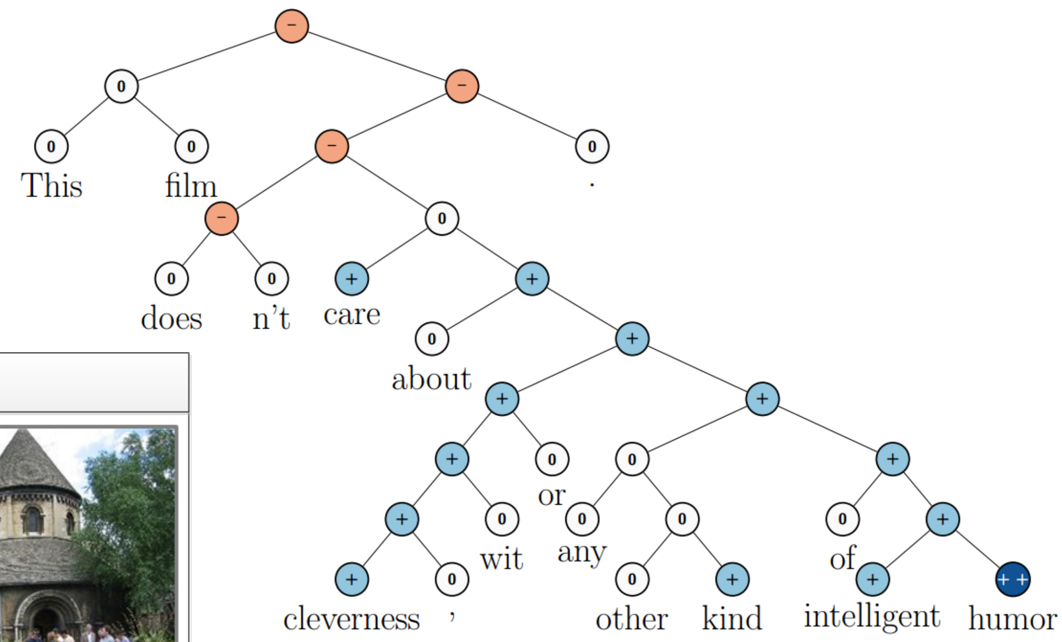
(Ren, He, Girshick and Sun, „Faster R-CNN", 2015)

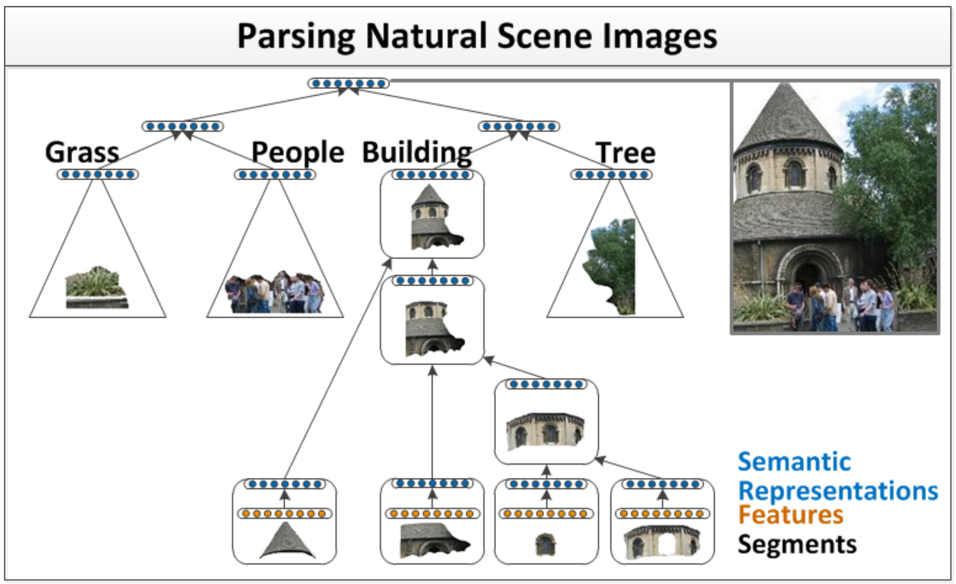
Natural Language Processing

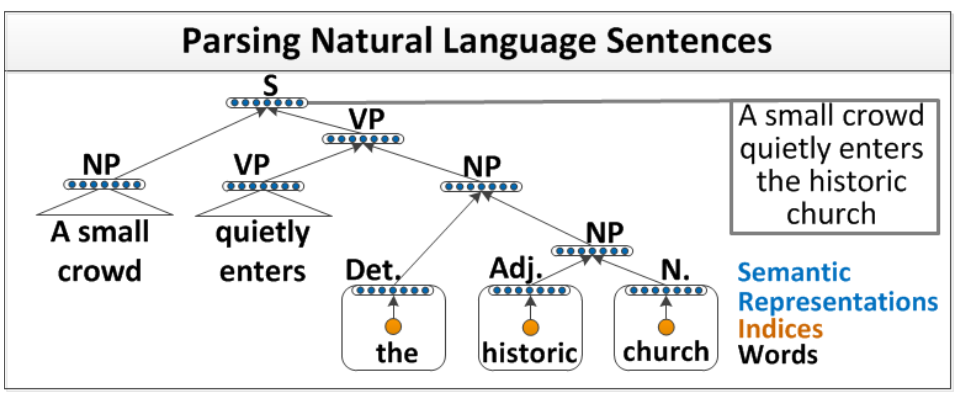
http://zapotoczny.pl/parser/



Parsing, scene understanding





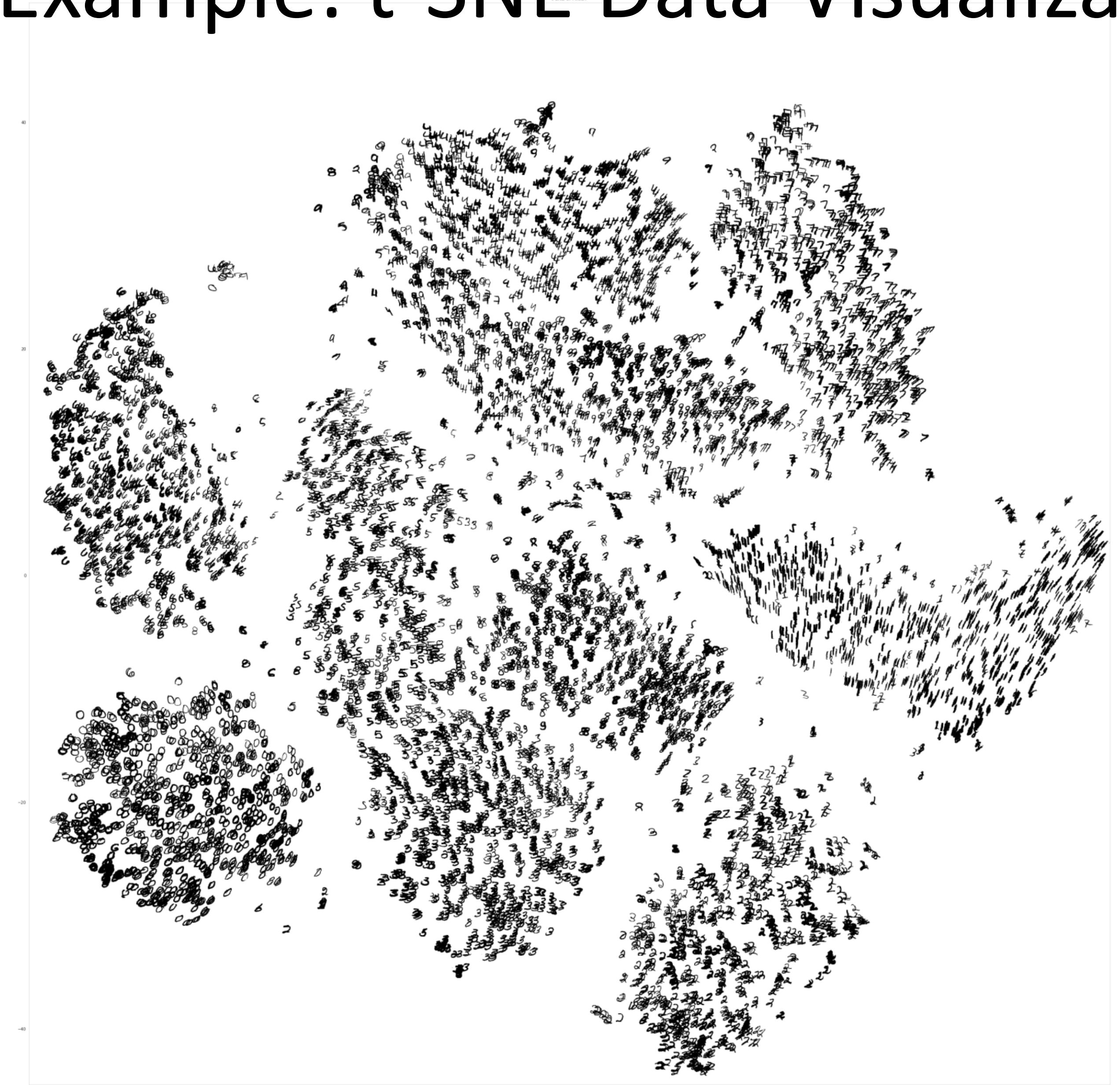


Unsupervised learning

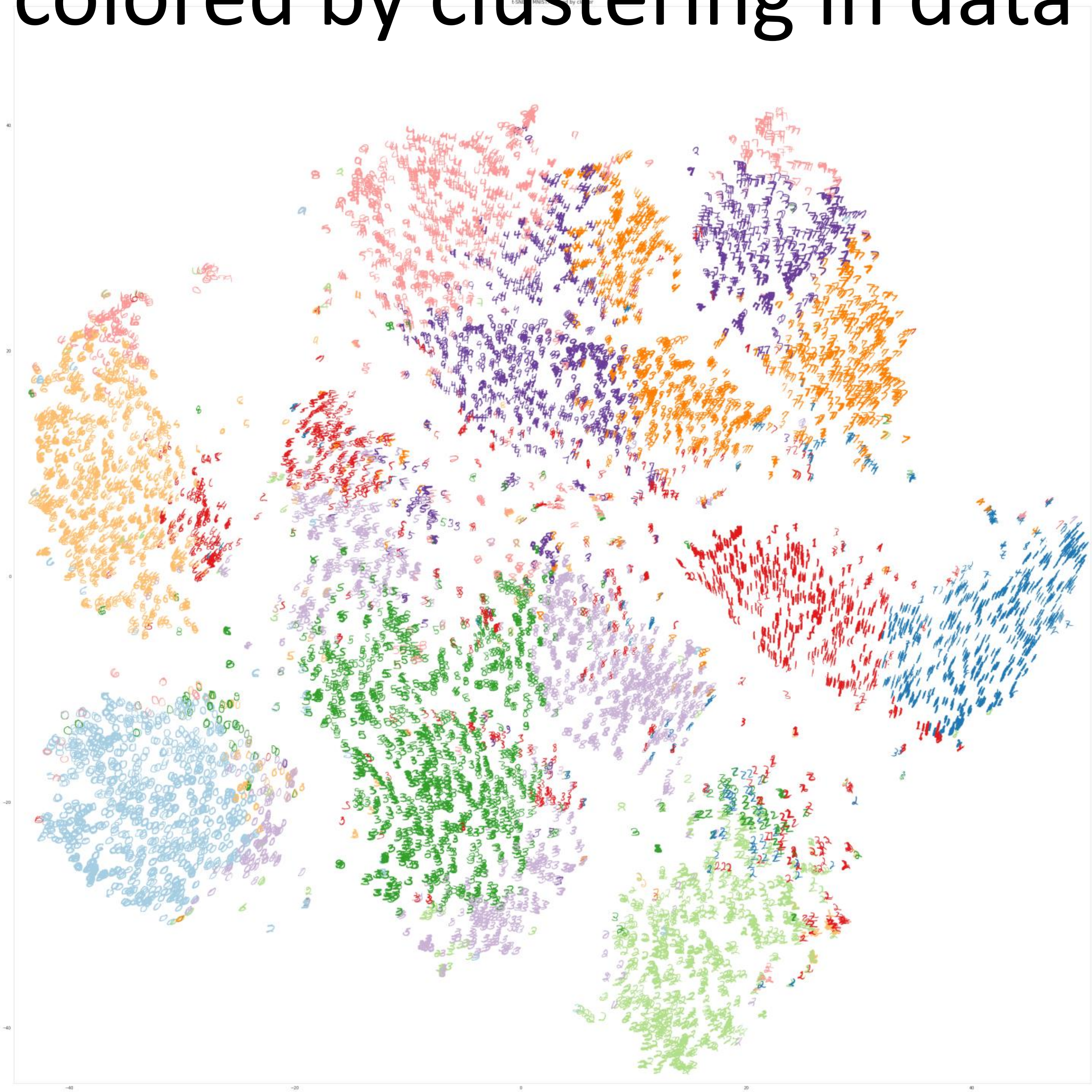
$$I \rightarrow ?$$

- Visualization: present the data in 2D
- Clustering: discover groups in data
- Generation: learn to generate fake data

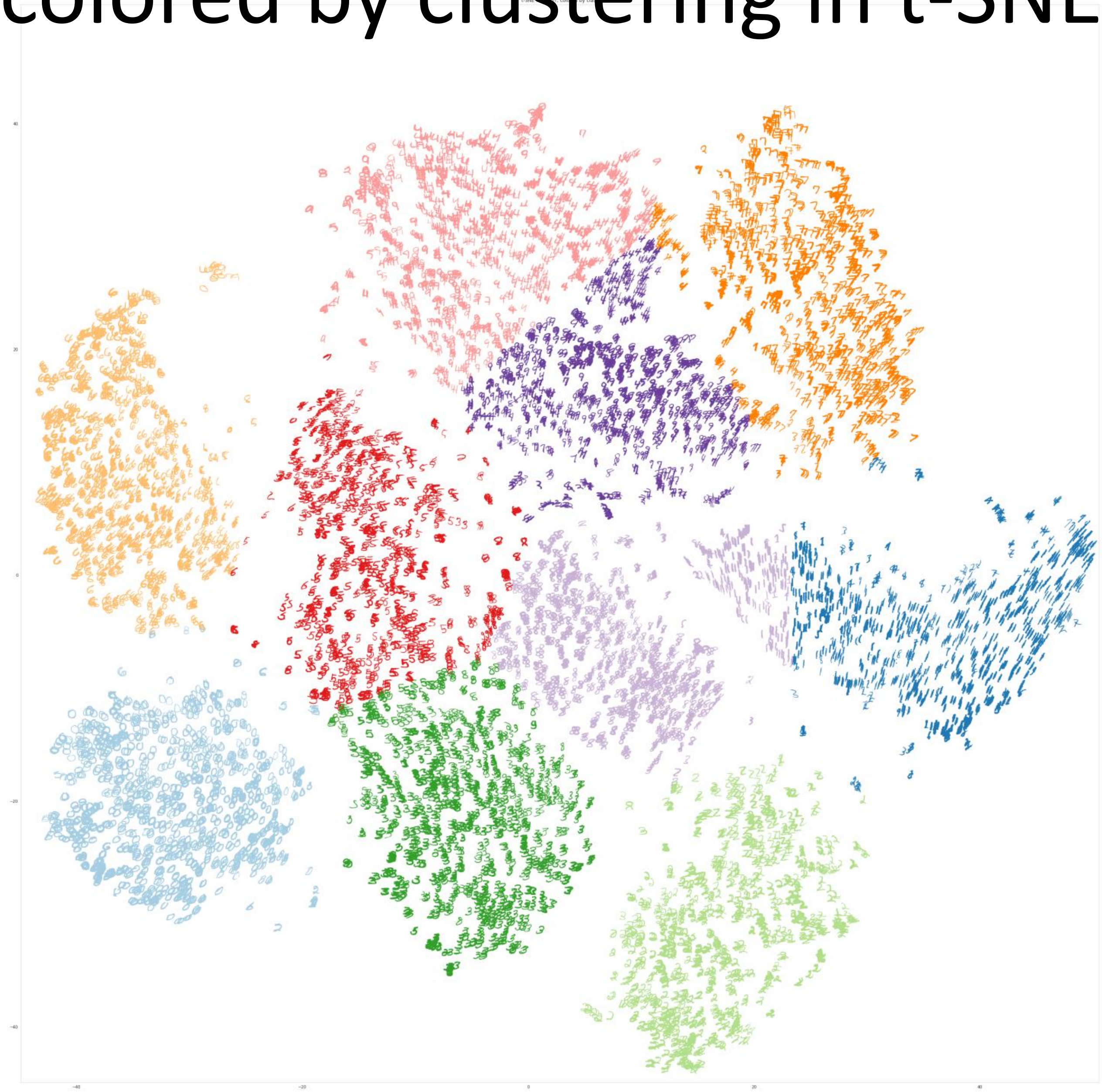
ML Example: t-SNE Data Visualization



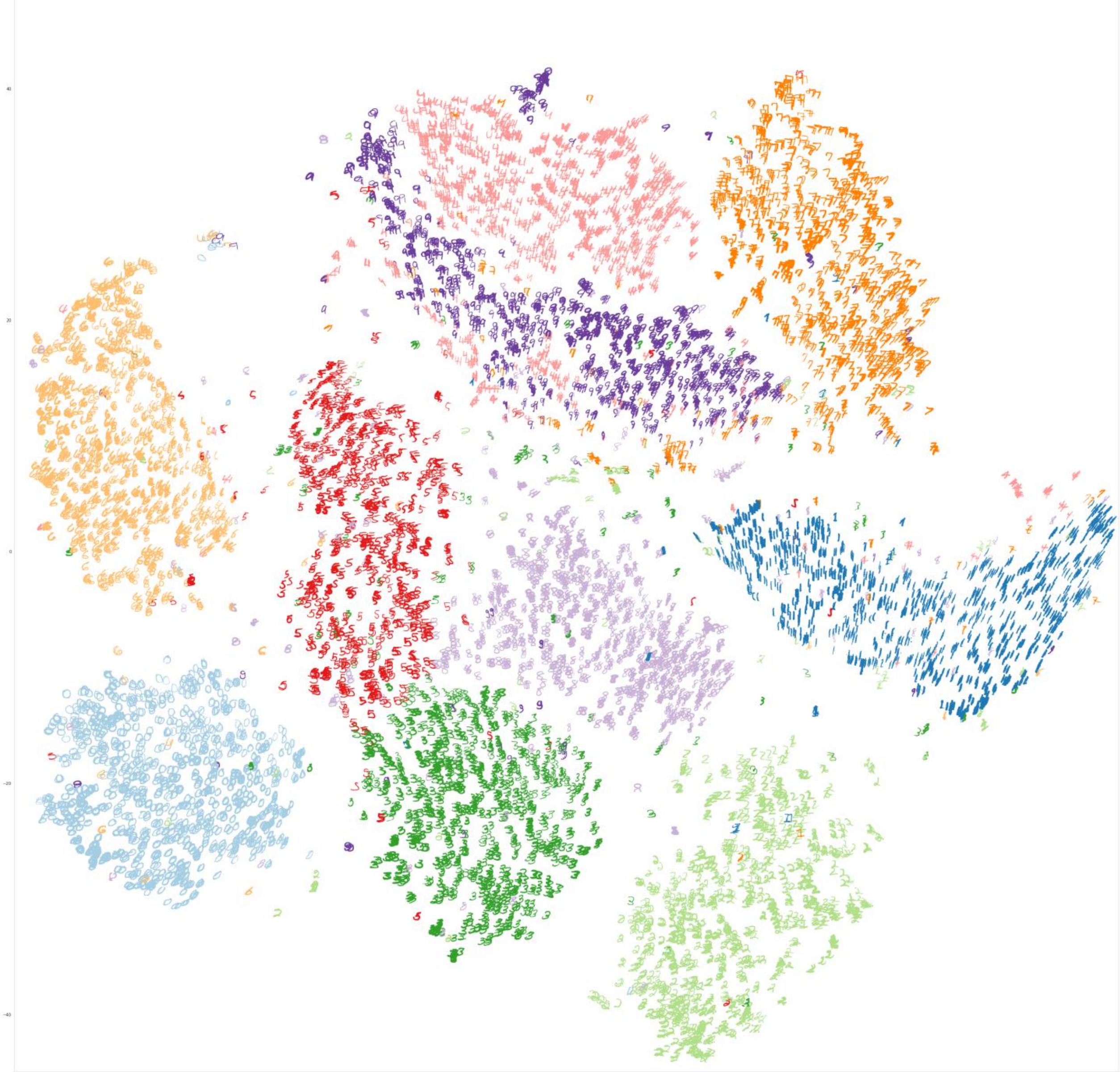
t-SNE colored by clustering in data space



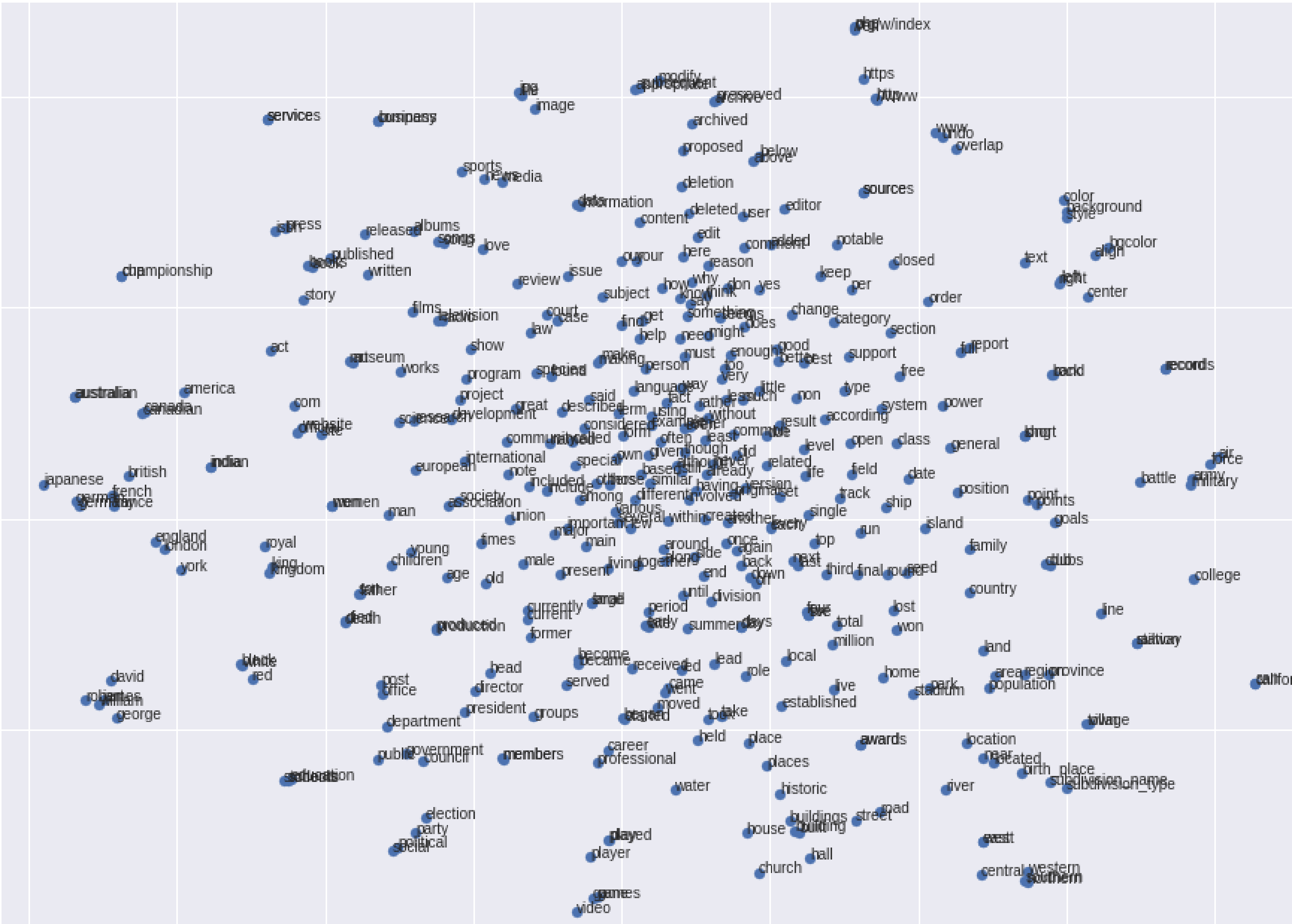
t-SNE colored by clustering in t-SNE space



t-SNE with real labels



Putting words into a vector space



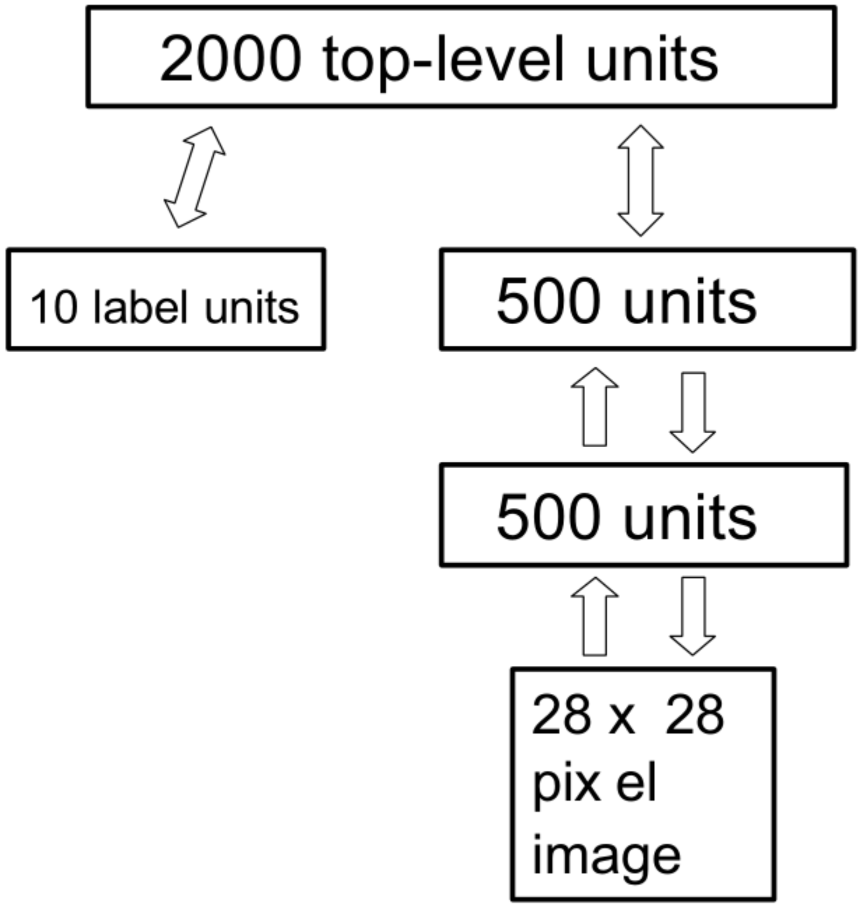
ML Example: Album cover generation

Image by Alec Radford

Other Examples

Combine aspects of supervised and unsupervised learning

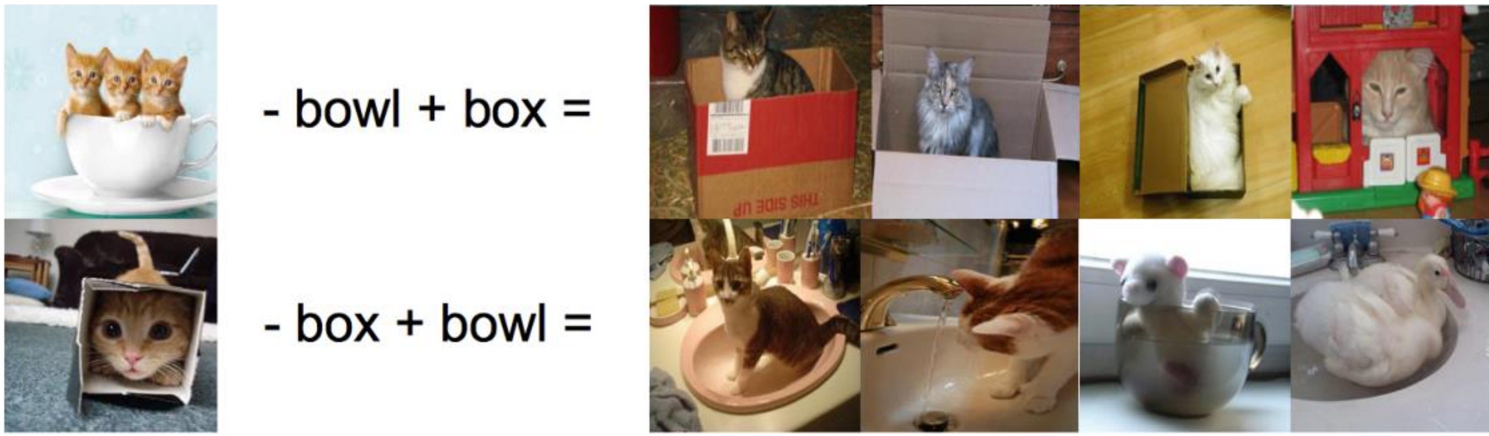
Conditional Digit Generation



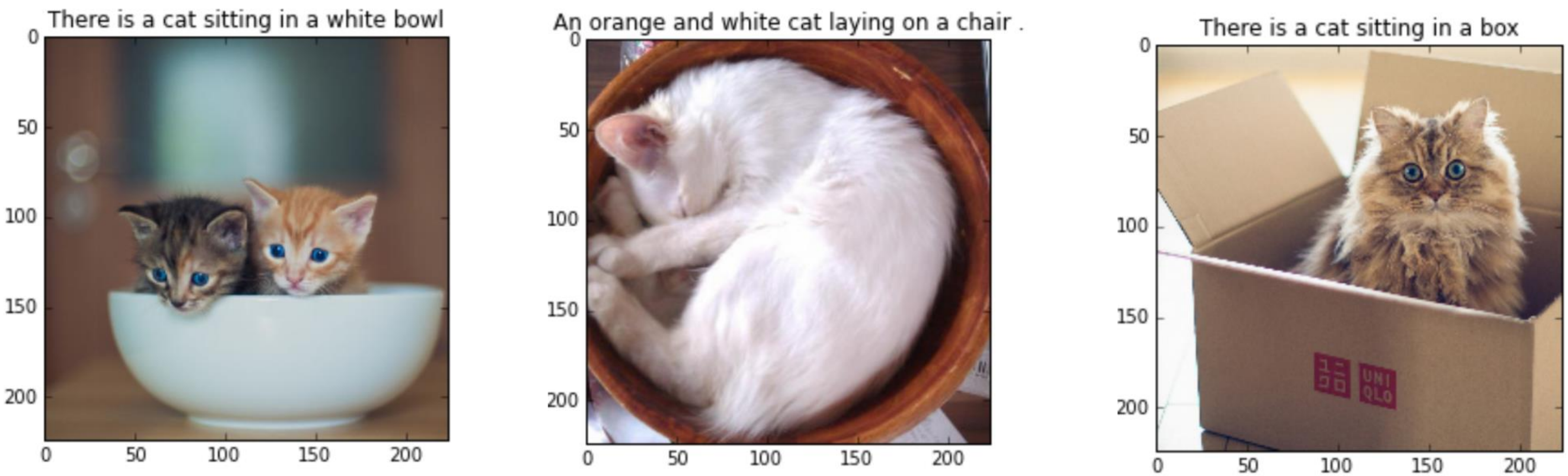
Demo: http://www.cs.toronto.edu/~hinton/adi/index.htm

Putting text and images into vector space

(Kiros, Salakhutdinov, Zemel, TACL 2015)

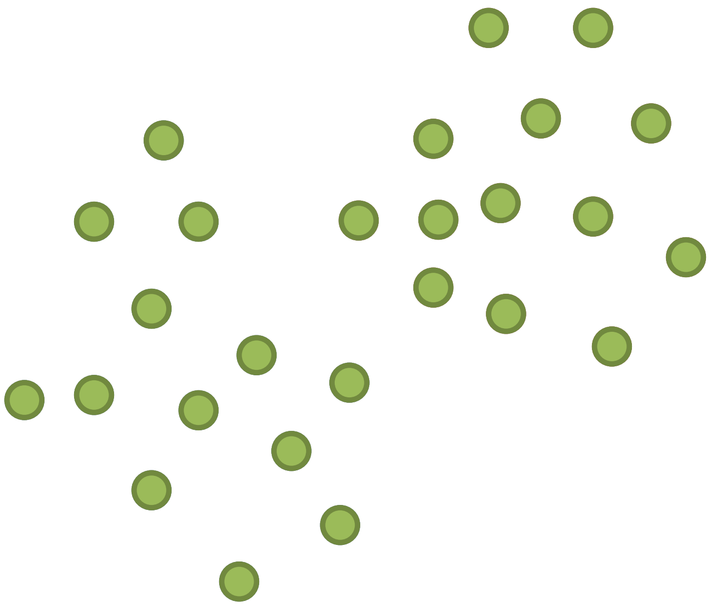


Caption generation (Xu et al ICML 2015)



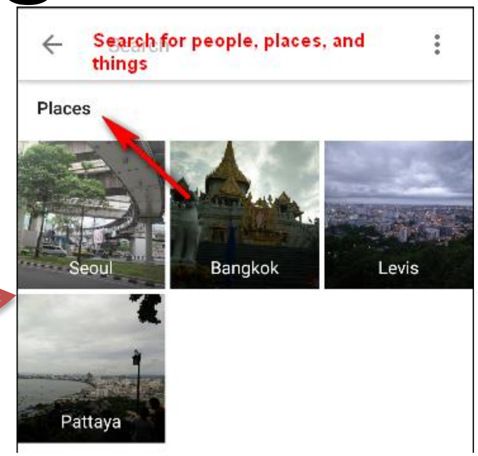
Summary: learning tasks taxonomy

- Supervised:
  - Learn an input-output relation, instant feedback
- Unsupervised:
  - Learn the density of data
  - Learn how to generate data
  - Group data into clusters
- Reinforcement:
  - Reward information given after a series of actions
  - Think of learning strategy in games



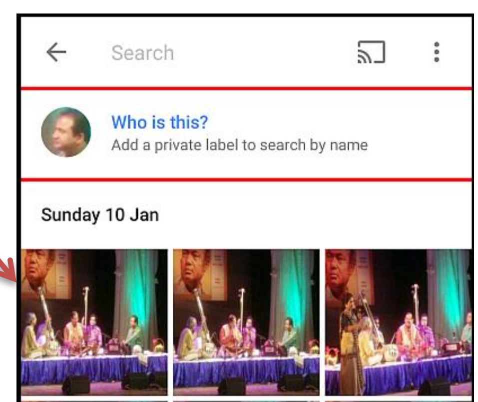
ML Example: Google Photos app

Algos and Data



Sup. learning

Unsup. learning



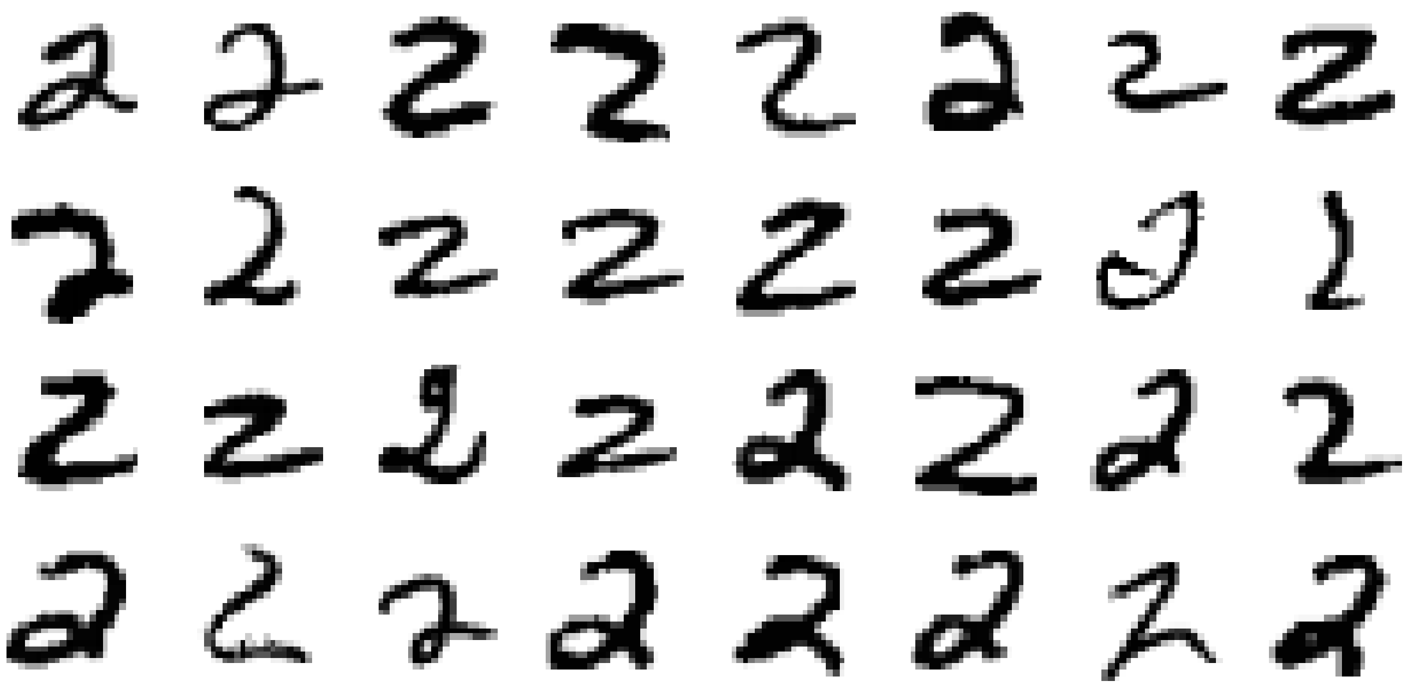
SIMPLEST ML ALGORITHM: K-NEAREST NEIGHBORS

ML approach to problems

1. Take **data** and a **learning algorithm**
2. The algorithm discovers **patterns** in the data and produces a **model**
3. Query model to ask questions about **new** data
   - „Is there any face in the image?”
   - “Is this review favorable?”
   - „What is the object in the image?”
   - “What caption is most likely given this image?”

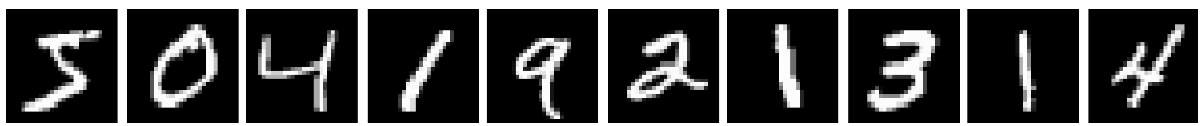
Example: Digit Classification

- Task: recognize handwritten digits
- Input: images 28 x 28 pixel values ($[0,1]^{784}$)
- Output: {0,1,…,9}

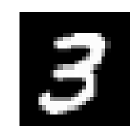


The 1-NN algorithm

- Collect examples – pairs of (input, output):



5 0 4 1 9 2 1 3 1 4

- To classify a new instance :
  - Find the most similar element of the train set 
  - Return its label

K-NN use: recommendations

- Items are similar if users rate them similarly
- Training (offline):
  – Represent each item by its scores
  – Compute the distance between scores

	5	2	5	...	
	3	1	5	...	

- Recommendation generation (online):
  – Find nearest neighbors for each item recently browsed/put into the cart

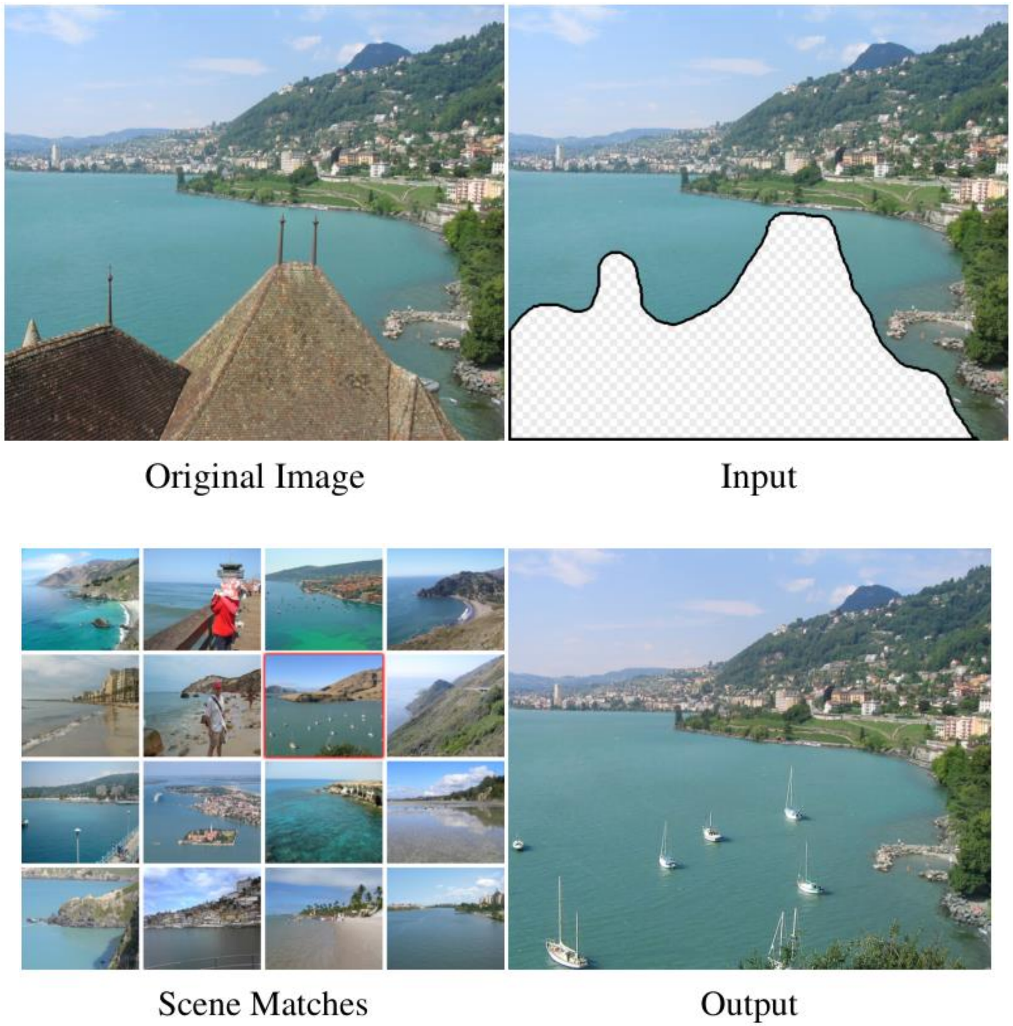
Amazon patented this:

US007113917B2

(12) **United States Patent**
Jacobi et al.

(10) **Patent No.: US 7,113,917 B2**
(45) **Date of Patent: *Sep. 26, 2006**

K-NN use: inpainting



1. Find similar photos
   - Use Gist descriptors
2. Match and copy

Works poorly with 1000 images…

But is good with >1000000

Hays, Efros „Scene Completion Using Millions of Photographs”

Alternative realities

Input

Alternative inpainting results

K-NN use: dialog system

A system Adrian Łańcucki coded based on Reddit.
Real (cherry-picked) conversation from Festiwal nauki:

User: hi

Bot: Hi.

User: i like you

Bot: Your in if u want.

User: lmao

Bot: Ayy lmao.

User: ayyy

Bot: Who are you?

User: im a nurse

Bot: Gott im Himmel. You have become the host.

User: and what are you ?

Bot: I am just a talking robot. But I like my job.

User: thats great

K-NN DETAILS

K-NN task

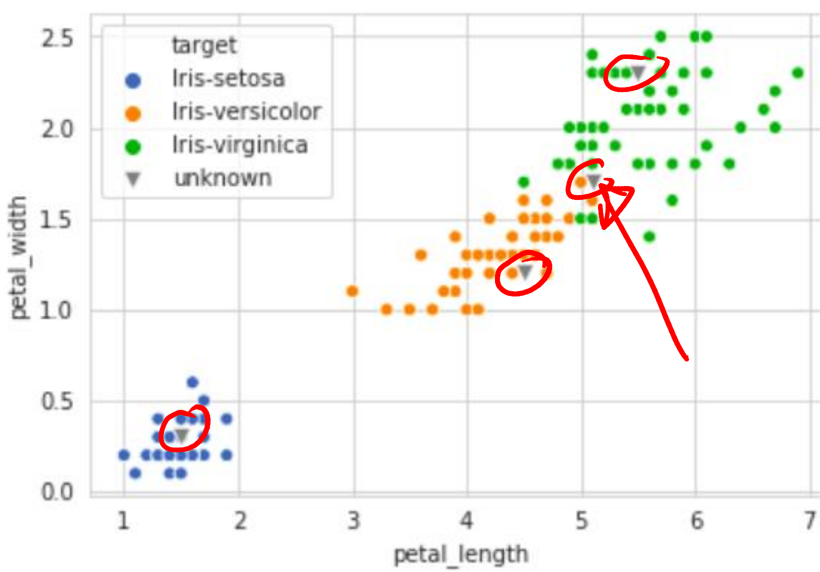
K-NN is a ML method that takes

a set of pairs $\mathcal{D} = \{(x_i\ y_i)\}$

and produces

predictions for new data $y \approx f(x, \mathcal{D})$

petal_length	petal_width	target
1.4	0.2	Iris-setosa
1.4	0.2	Iris-setosa
1.3	0.2	Iris-setosa
1.5	0.2	Iris-setosa
1.4	0.2	Iris-setosa
1.7	0.4	Iris-setosa



K-NN for classification

Let $\mathcal{N}_K(x, \mathcal{D})$ be the K nearest neighbors of a vector x.

Then:

$$p(y = c | x, \mathcal{D}, K) = \frac{1}{K} \sum_{i \in \mathcal{N}_K(x,\mathcal{D})} \mathbb{I}(y_i = c)$$

Decision rule is

$$y = \arg\max_c p(y = c | x, \mathcal{D}, K)$$

Where $\mathbb{I}(\cdot)$ is the indicator function

$$\mathbb{I}(e) = \begin{cases} 1 \text{ if } e \text{ is true} \\ 0 \text{ if } e \text{ is false} \end{cases}$$

K-NN for regression

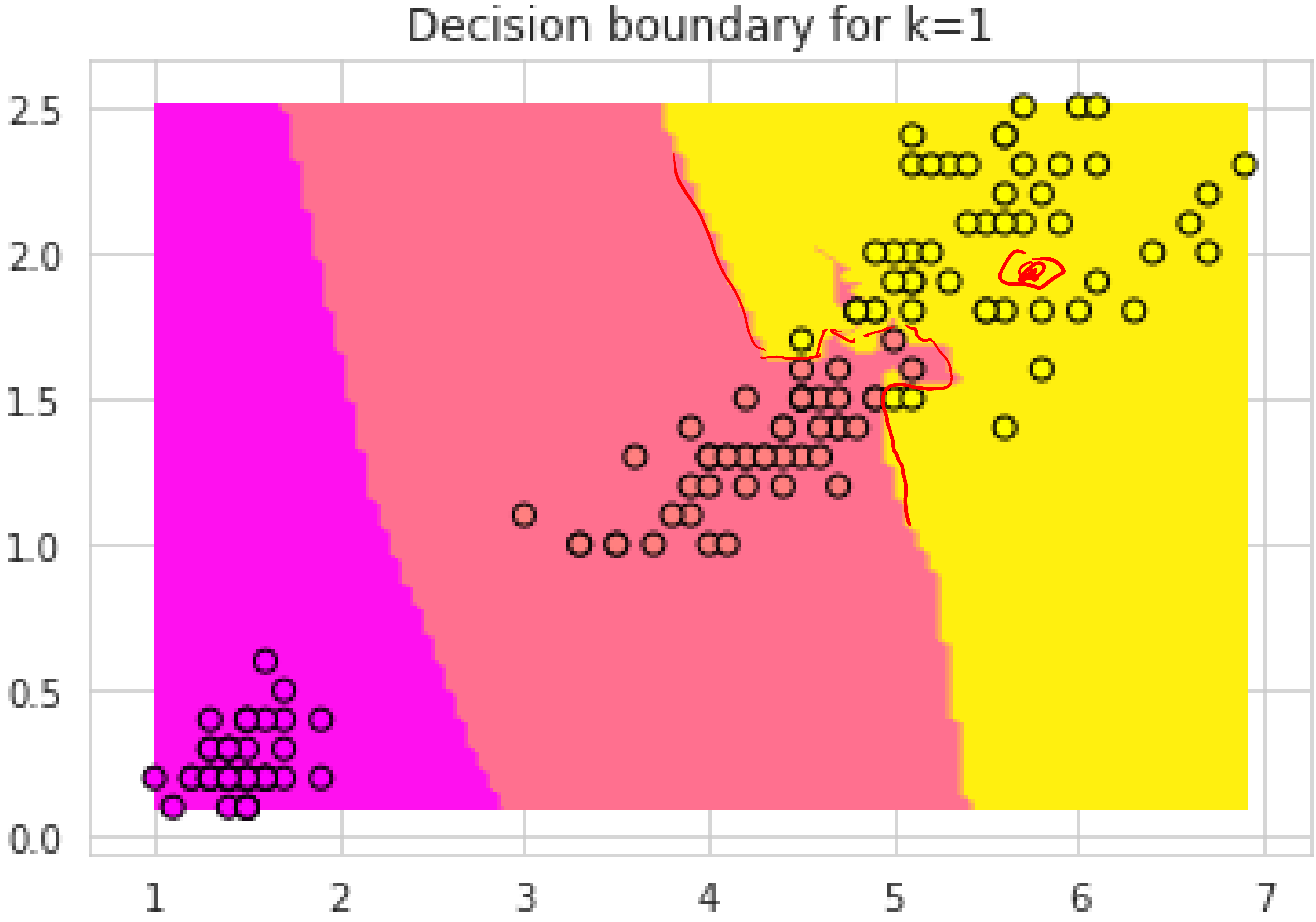
Let $\mathcal{N}_K(x, \mathcal{D})$ be the K nearest neighbors of a vector x.

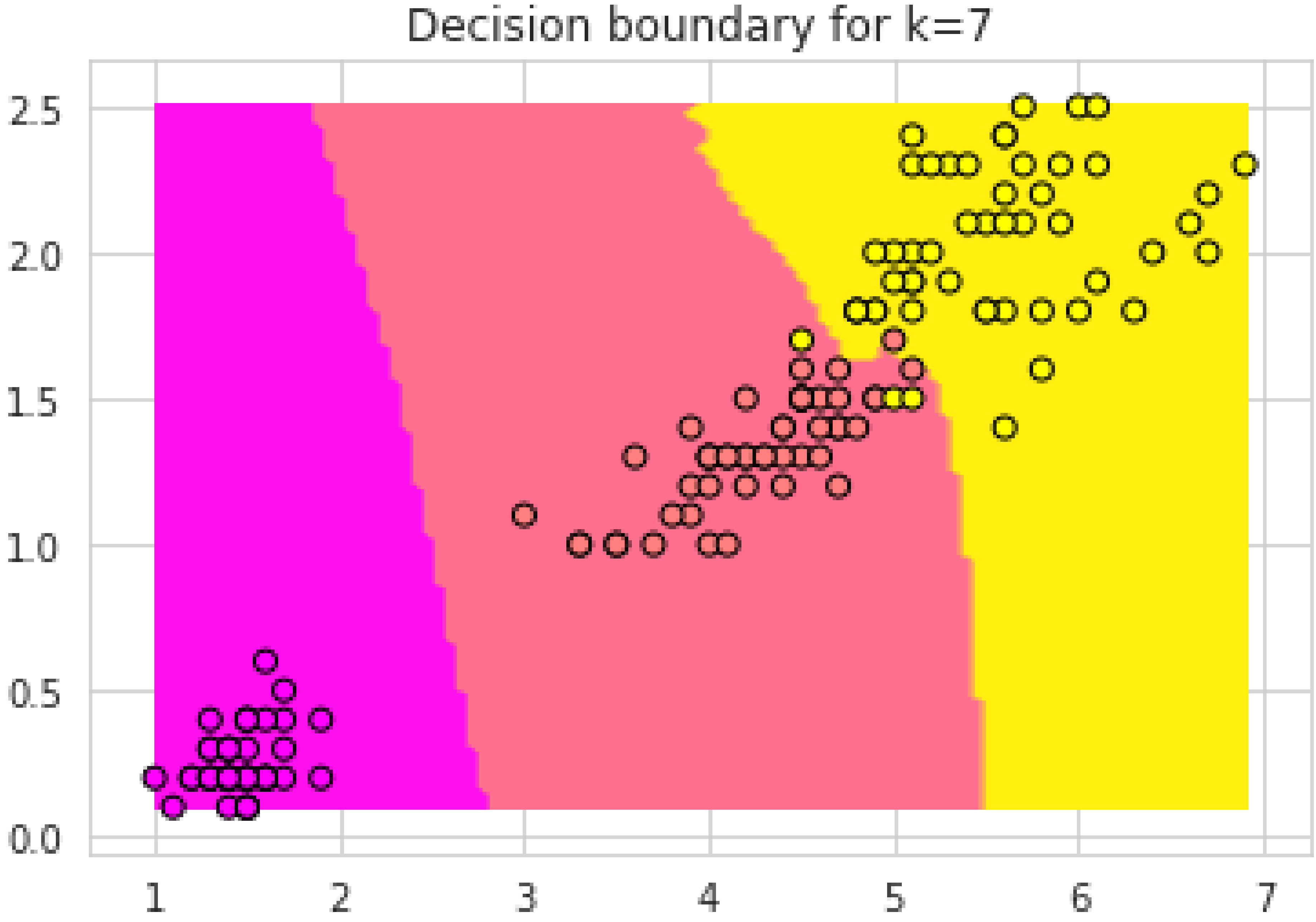
Then:

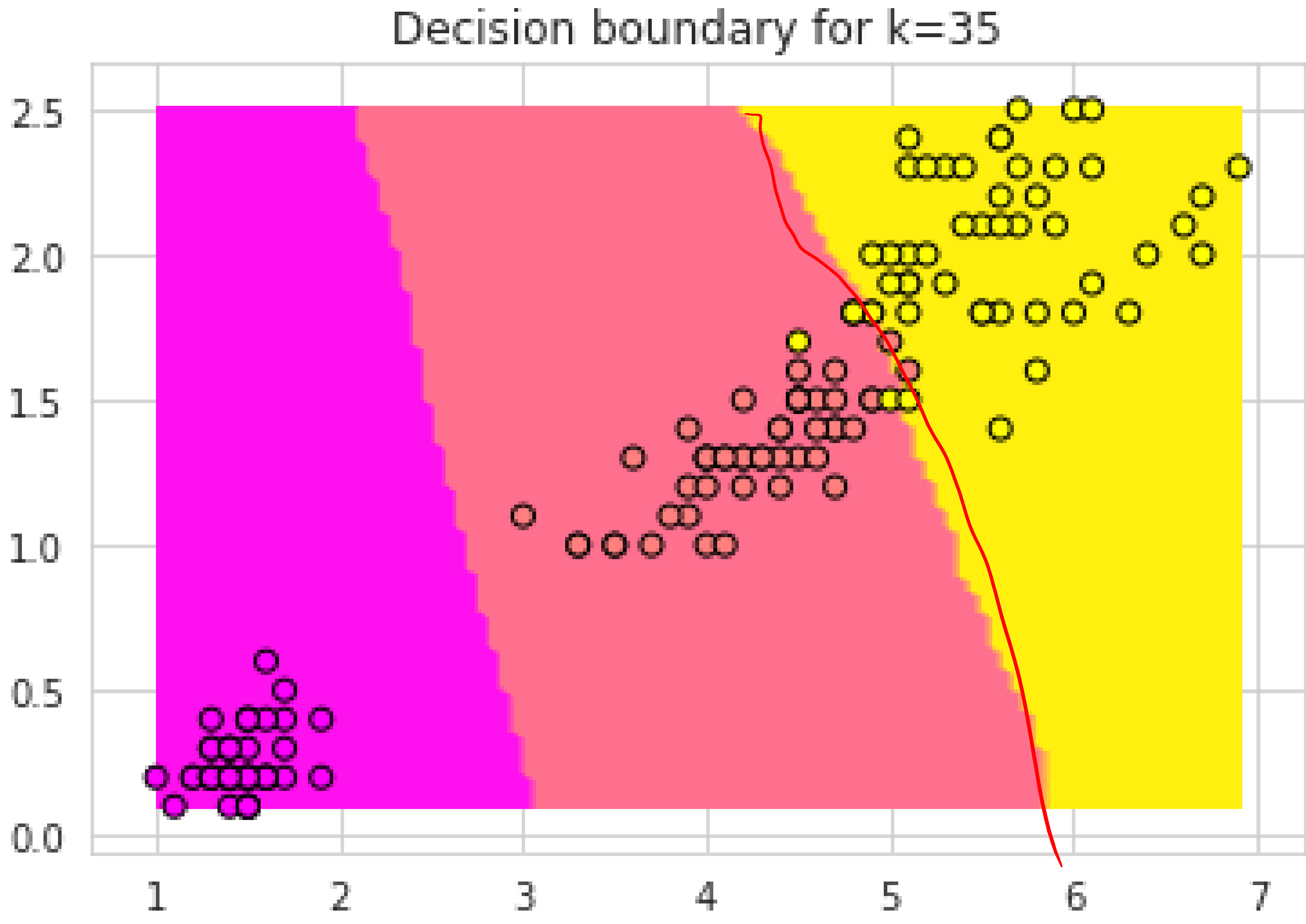
$$y = \frac{1}{C} \sum_{i \in \mathcal{N}_K(x,\mathcal{D})} \frac{1}{d(x_i, x)} y_i$$

where $C = \sum_{i \in \mathcal{N}_K(x,\mathcal{D})} \frac{1}{d(x_i,x)}$

K-NN: choosing K


Decision boundary for k=1
2.5
2.0
1.5
1.0
0.5
0.0
1
2
3
4
5
6
7


Decision boundary for k=7
2.5
2.0
1.5
1.0
0.5
0.0
1
2
3
4
5
6
7


Decision boundary for k=35
2.5
2.0
1.5
1.0
0.5
0.0
1
2
3
4
5
6
7

K-NN: choosing K

How many neighbors to use?

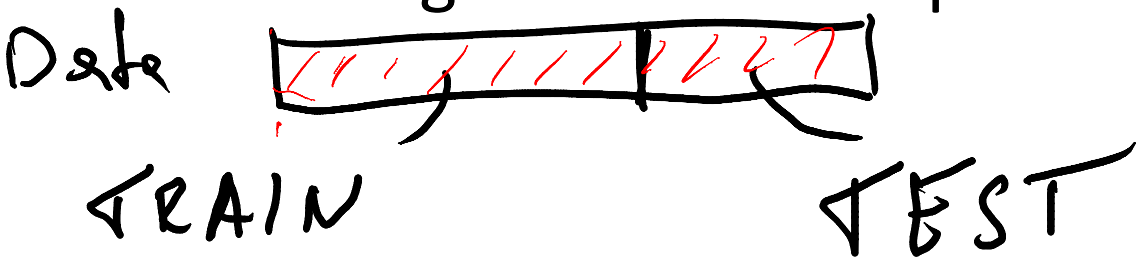
KNN solves a task

Pick K to maximize performance o it!

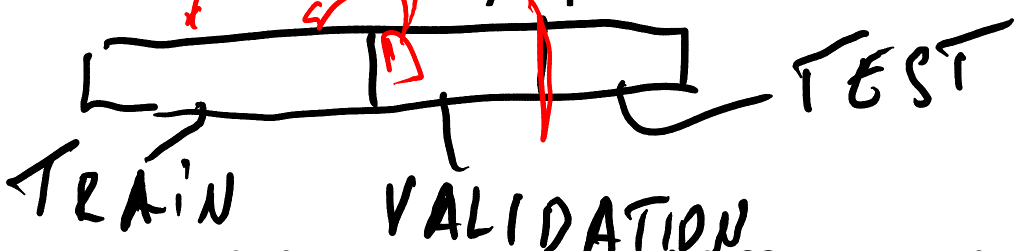
Measuring performance: Hold-out set

Large data case!!!

- Split the training data into two parts:



- Train only on training, then test on testing.
- Often we do a three-way split:



- Then:
  - Train many models on training (different algos, parameters)
  - Use validation to choose best model
  - Test on testing

Cross-validation

Small data case!!

- Hold-out set makes inefficient data use
- Idea:
  - Divide the data into k sets (~5,10)

  For i=1..k

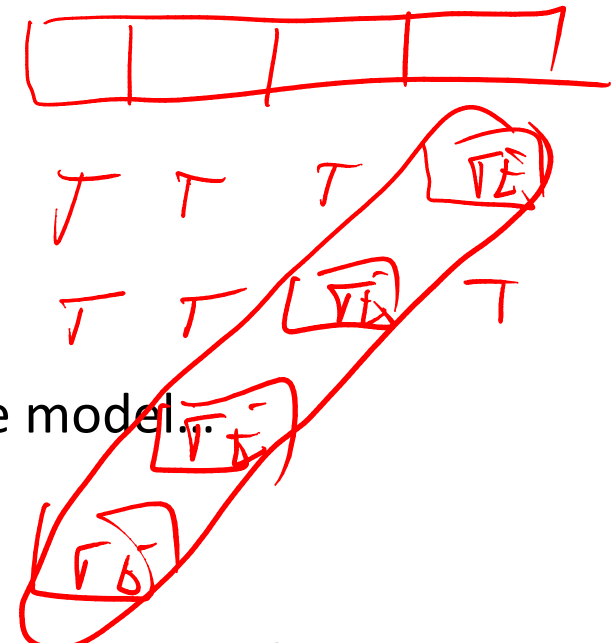
  Train on all but the i-th set

  may further split to choose the model…

  Test on the i-th set

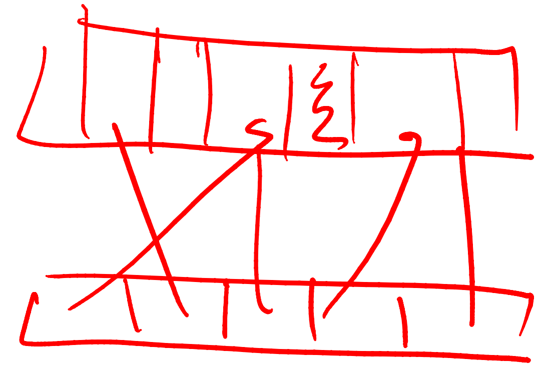
  Finally:

  take the answers on the testing sets and use them to compute the performance measures



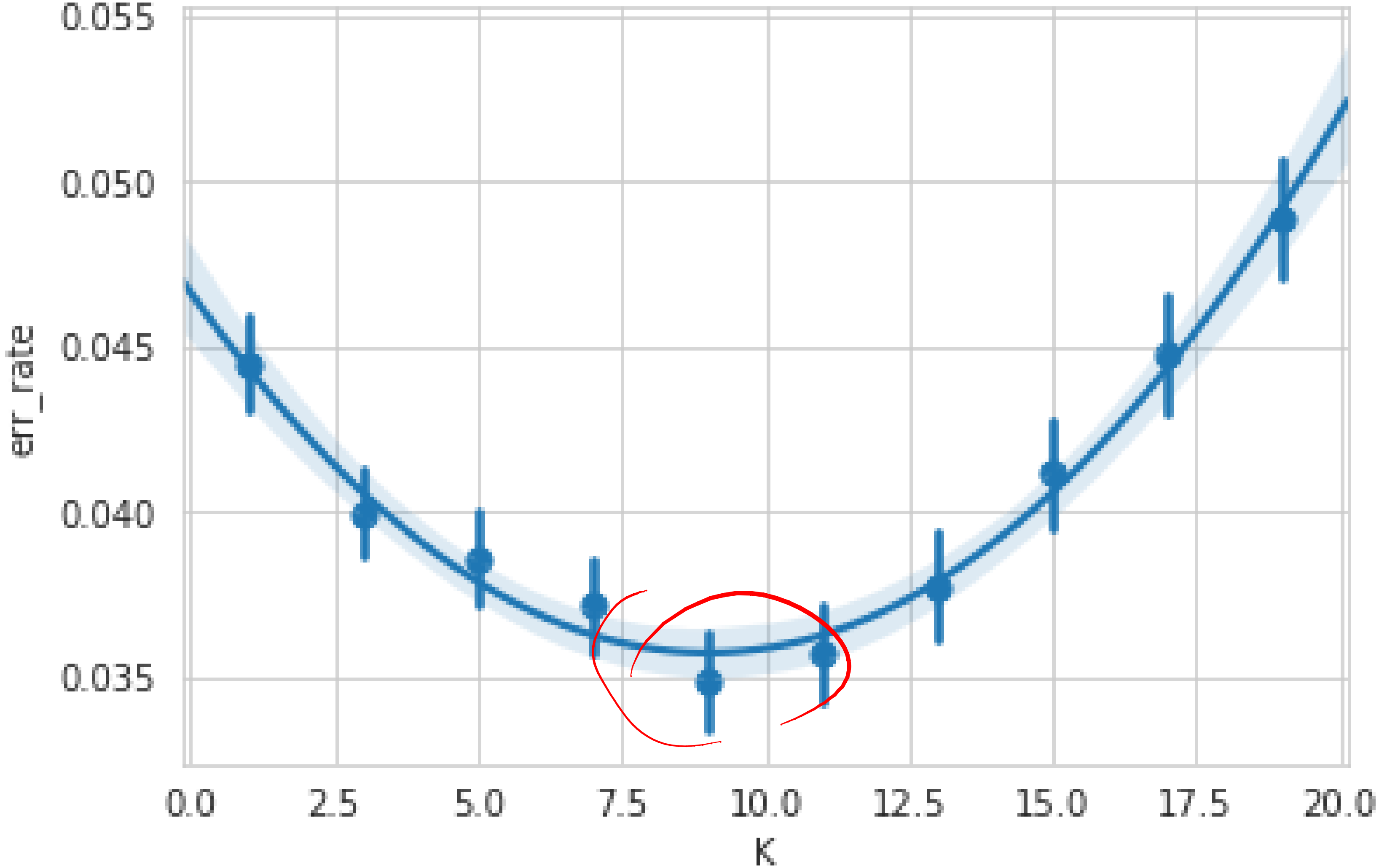
- Extreme case: leave-one-out (jackknife) – always use all but one sample to train!

Bootstrap



- Small data case!!
- Sample with replacement m samples
  - About 37% will not be selected
- Train on the selected samples
- Test on the remaining ones
- Optionally repeat.

Bootstrap for choosing K



K-NN: Distance Measures

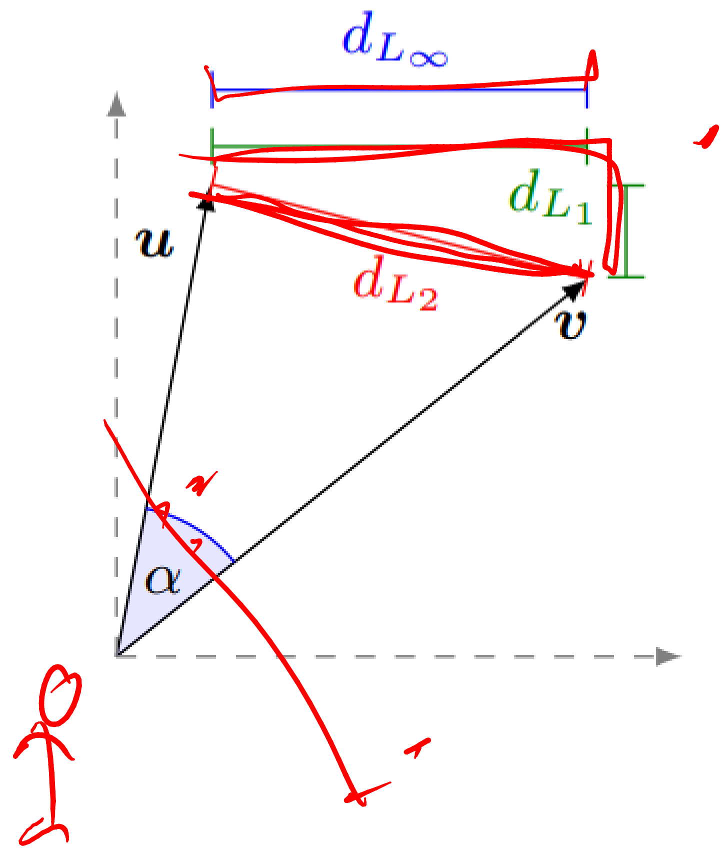
- Euclidean distance (L_2 norm) $\sqrt{\sum_i (u_i - v_i)^2}$
- L_1 norm $\sum_i |u_i - v_i|$
- L_∞ norm $\max_i |u_i - v_i|$
- Angle

$$\cos\alpha = \frac{u^T v}{\|u\|\|v\|}$$

- Mahalanobis
(Euclidean after a projection)

$$\sqrt{(u - v)^T \Sigma^{-1} (u - v)}$$

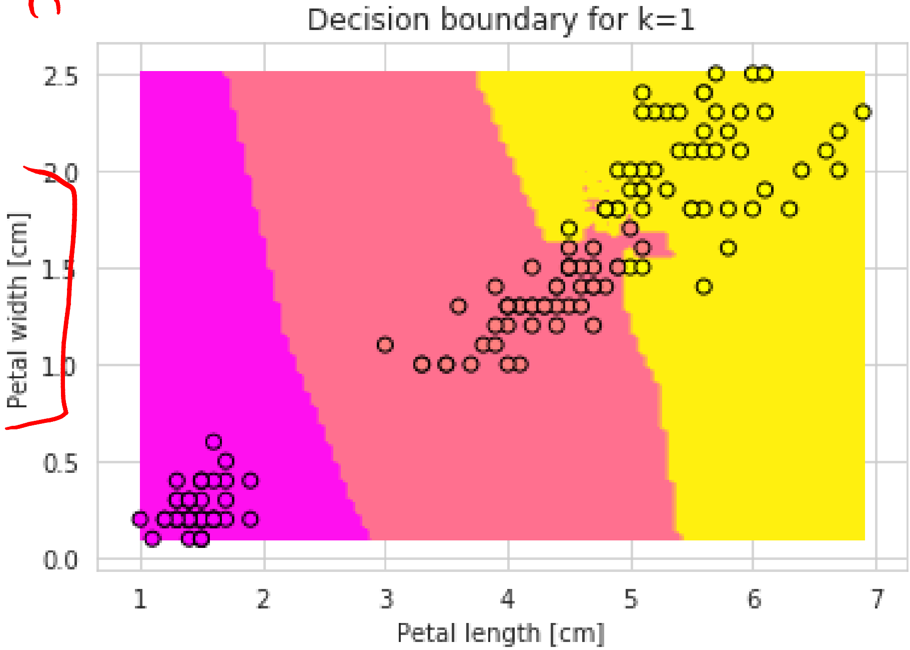
- Hamming
(number of differring coordinates)
Edit distance
Jaccard distance
...



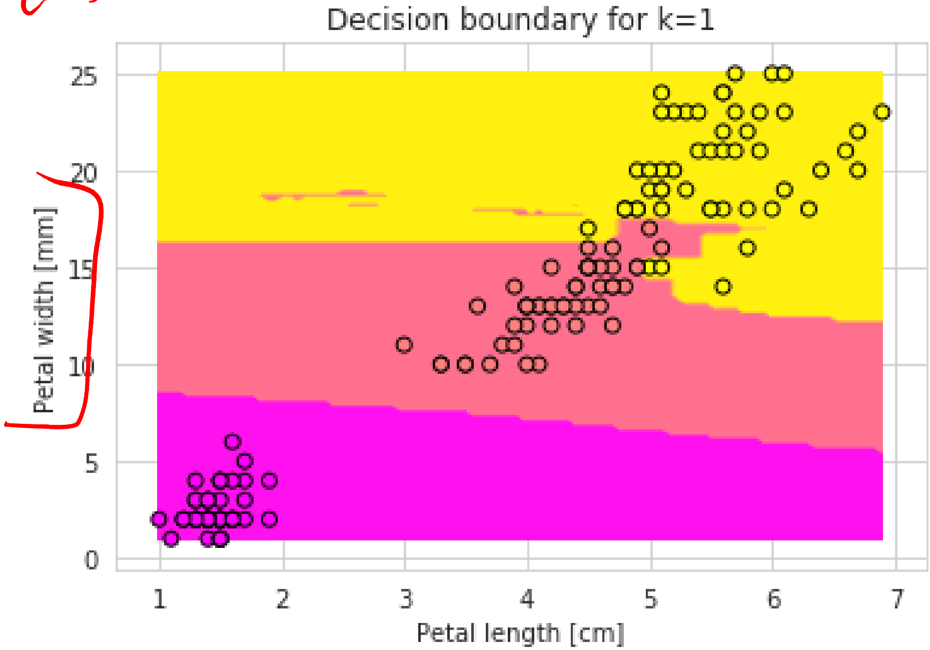
K-NN: data scaling issues

What has changed?

2, 5



2 5



Data Normalization

Map each input to [−1,1] range

Or scale each input to have mean 0, variance 1

$$x_{new} = \frac{x - \mu}{\sigma}$$

NB:

This corresponds to a Mahalanobis distance

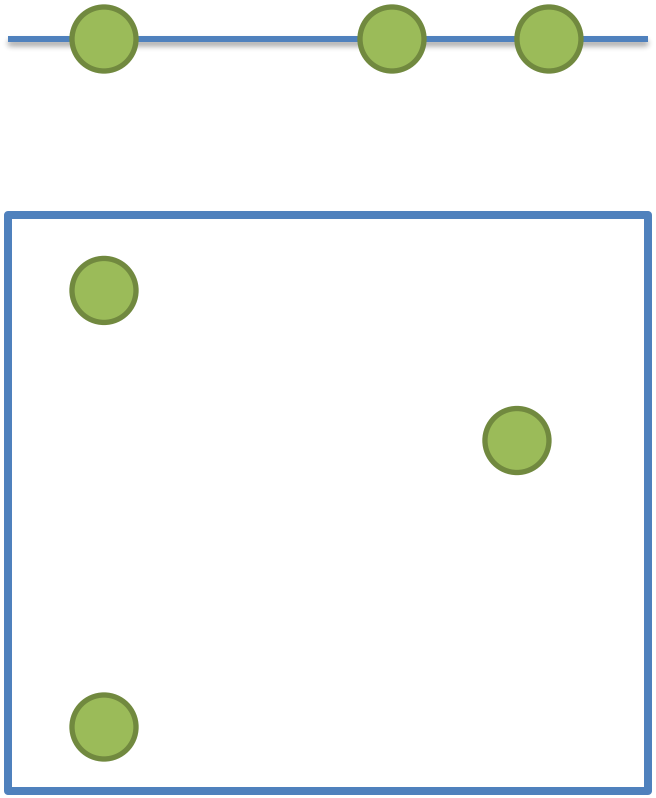
$$\sqrt{(u - v)^T \Sigma^{-1} (u - v)}$$

with $\Sigma = \begin{bmatrix} \sigma_1^2 & 0 & 0 \\ 0 & \sigma_2^2 & 0 \\ 0 & 0 & \sigma_D^2 \end{bmatrix}$

KNN requires lots of data

- Consider recognizing postal codes (5 digits)
- Will this work if we treat each postal code as a single image (10^5 combinations)?
- If we treat each digit separately, how to accommodate for:
  - Uniform style (slant, thickness) of digits in one code.
  - Segmentation, joins, overlaps of digits.

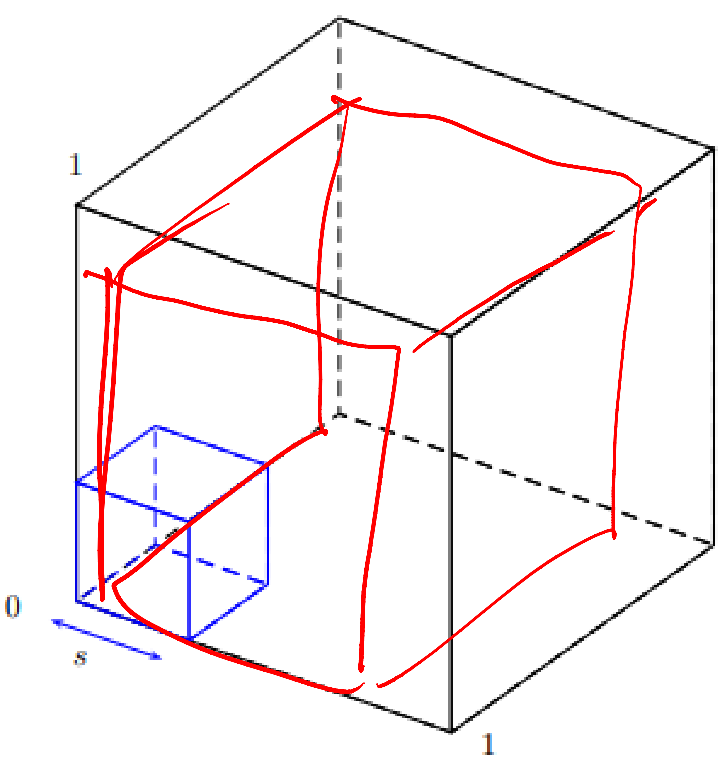
Curse of dimensionality

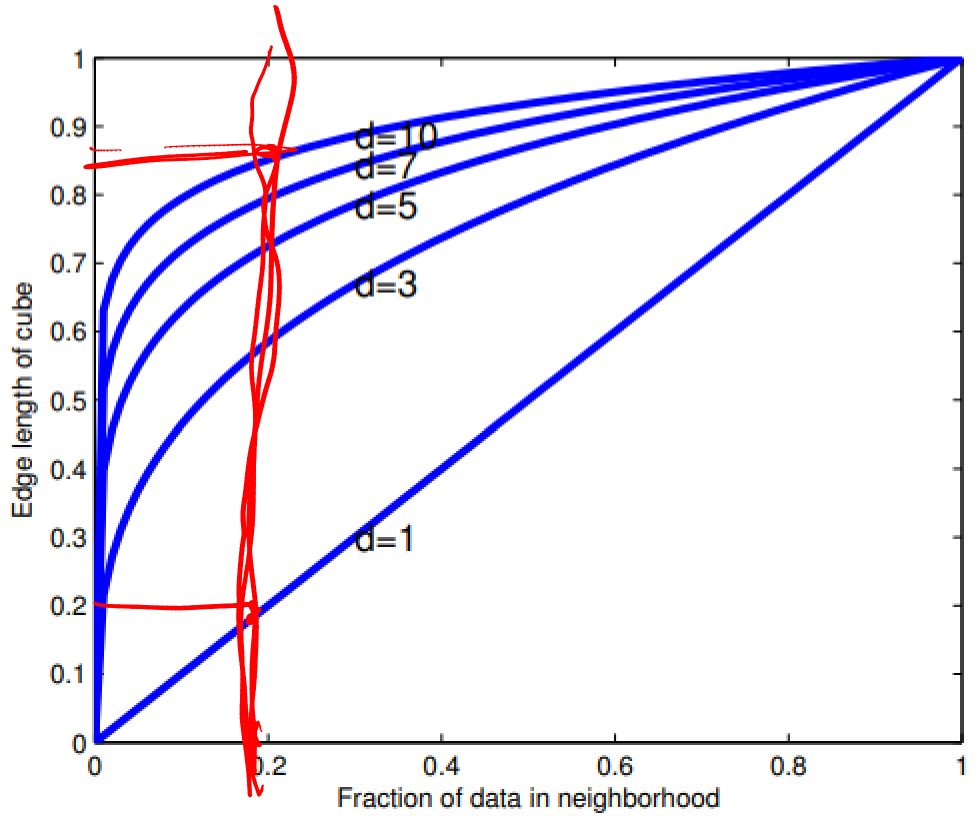


In highly dimensional spaces, things are dissimilar (far away from each other):

- Let N be the number of points.
- The average distance between points grows with data dimensionality.
- In other words, there are fewer neighbors within a radius from each point.

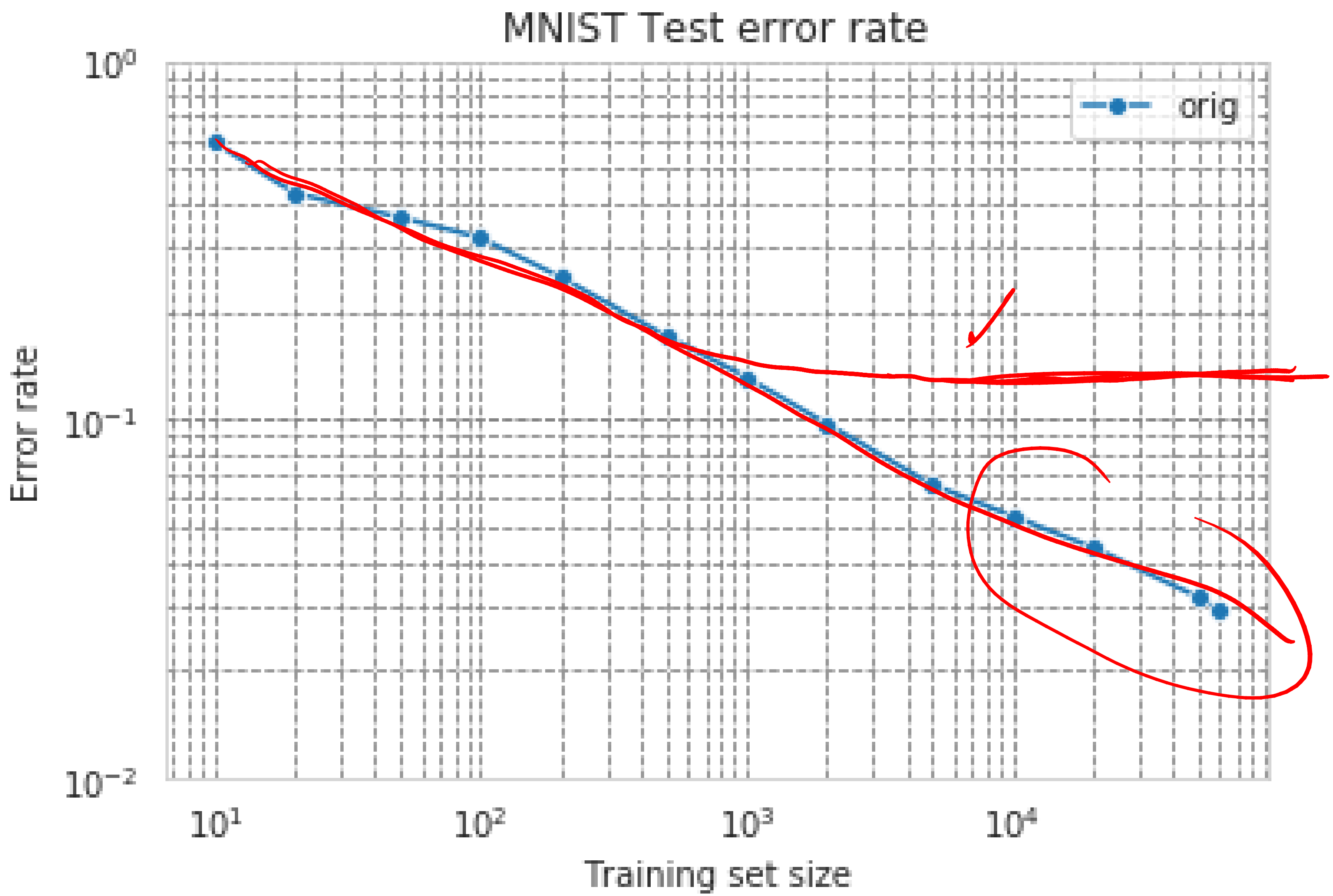
Curse of dimensionality



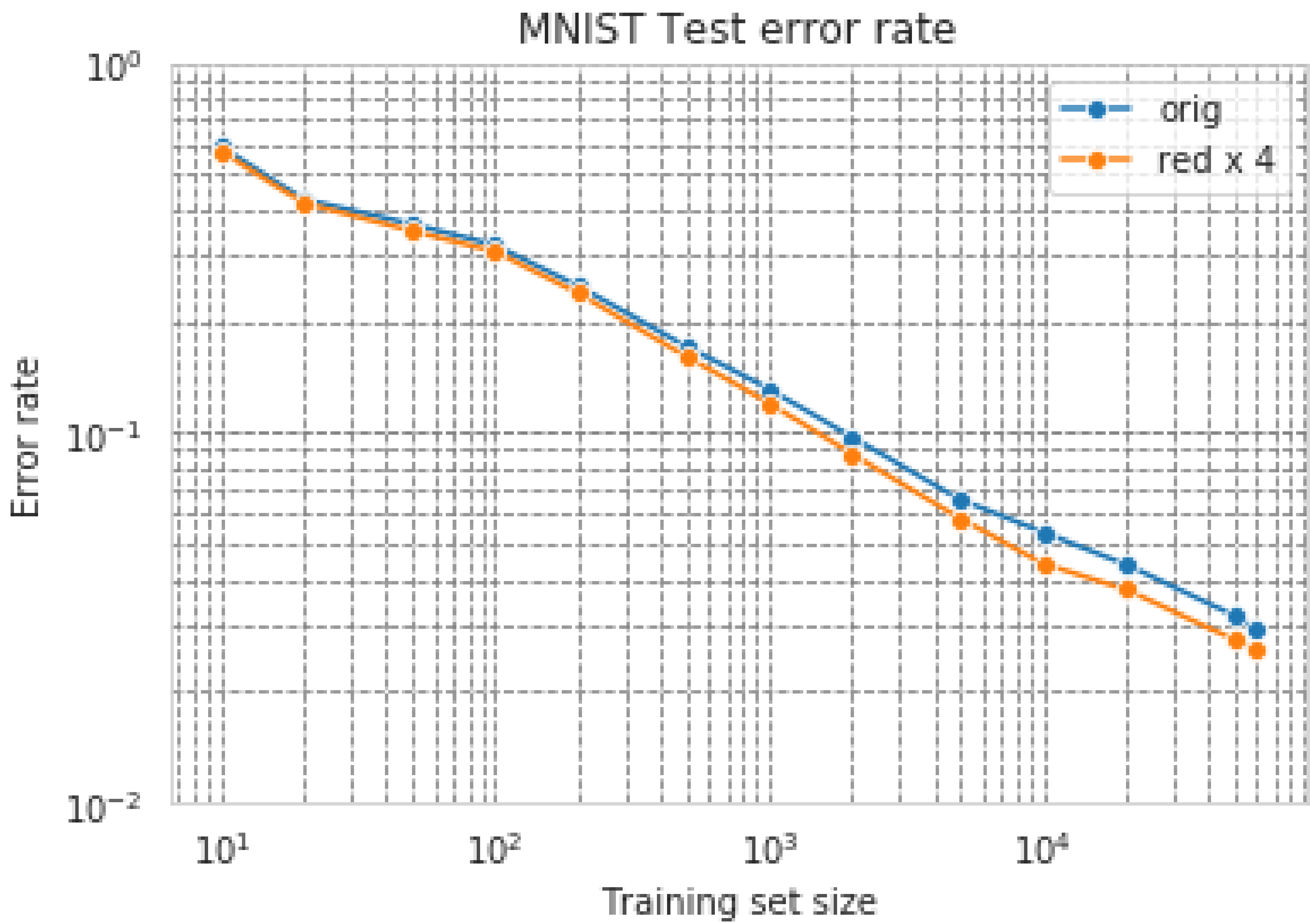


Img source: K. Murphy, Machine Learning fig. 1.16

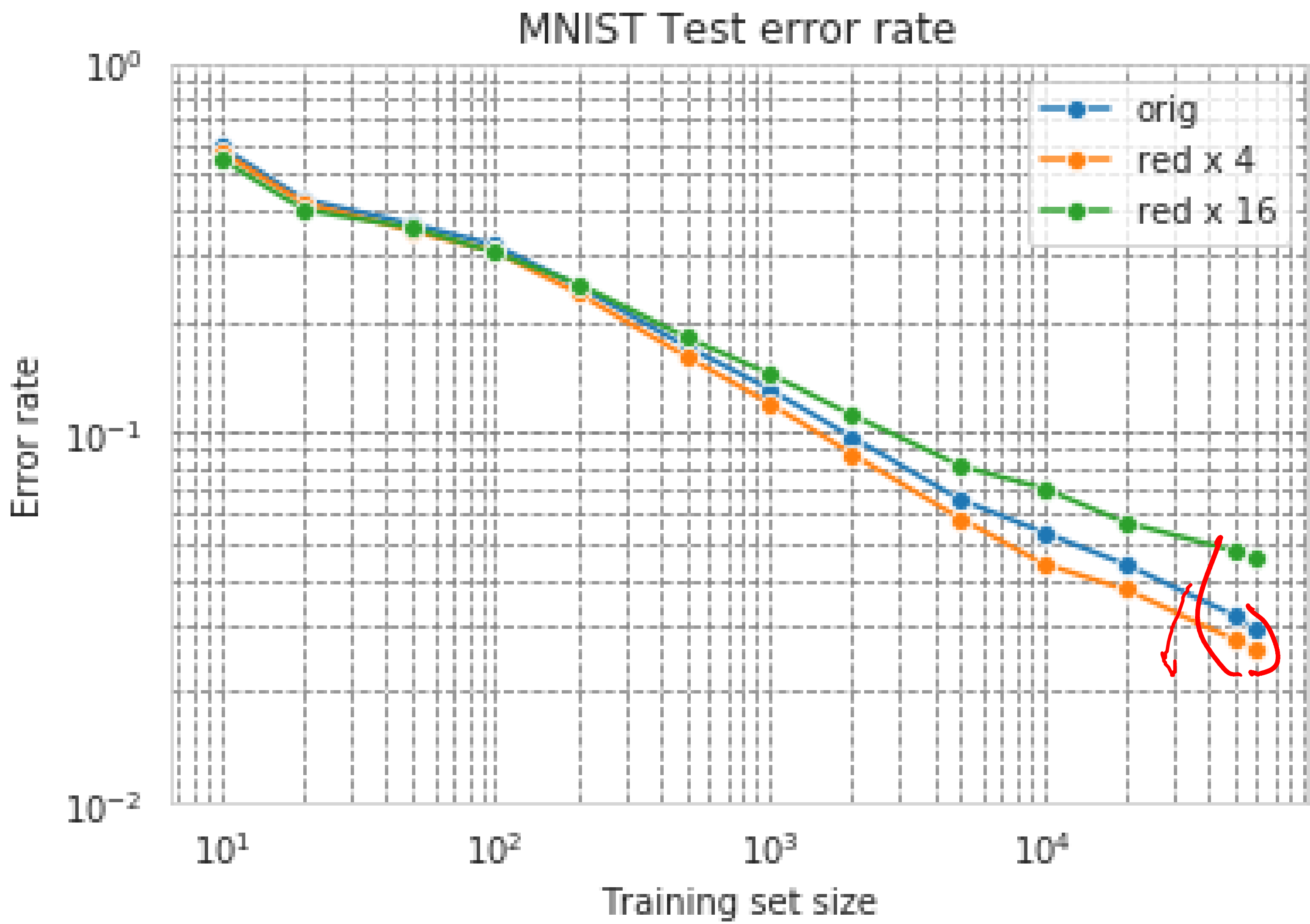
K-NN, scaling with amount of data



K-NN, scaling with amount & dimensionality of data

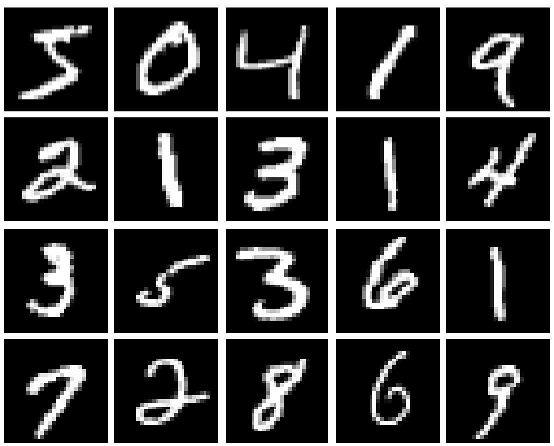


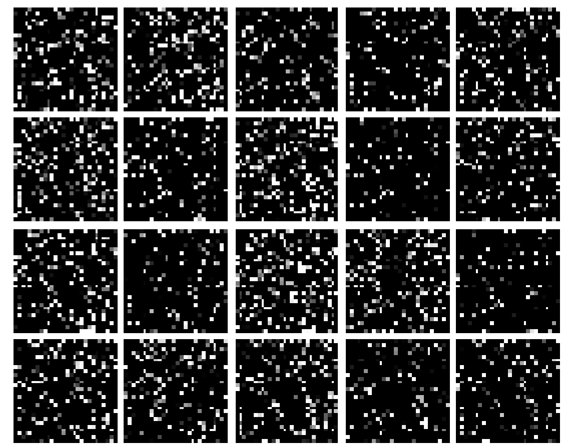
K-NN, scaling with amount & dimensionality of data



K-NN, use domain knowledge?

Which dataset is easier for K-NN?

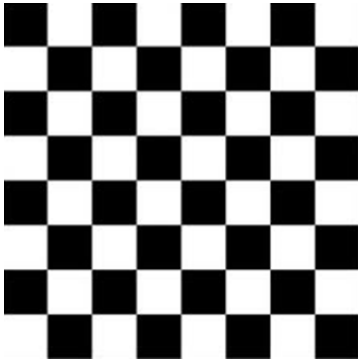


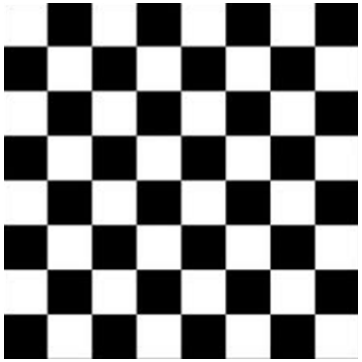


K-NN – what makes images similar?

Quite similar

Not quite similar

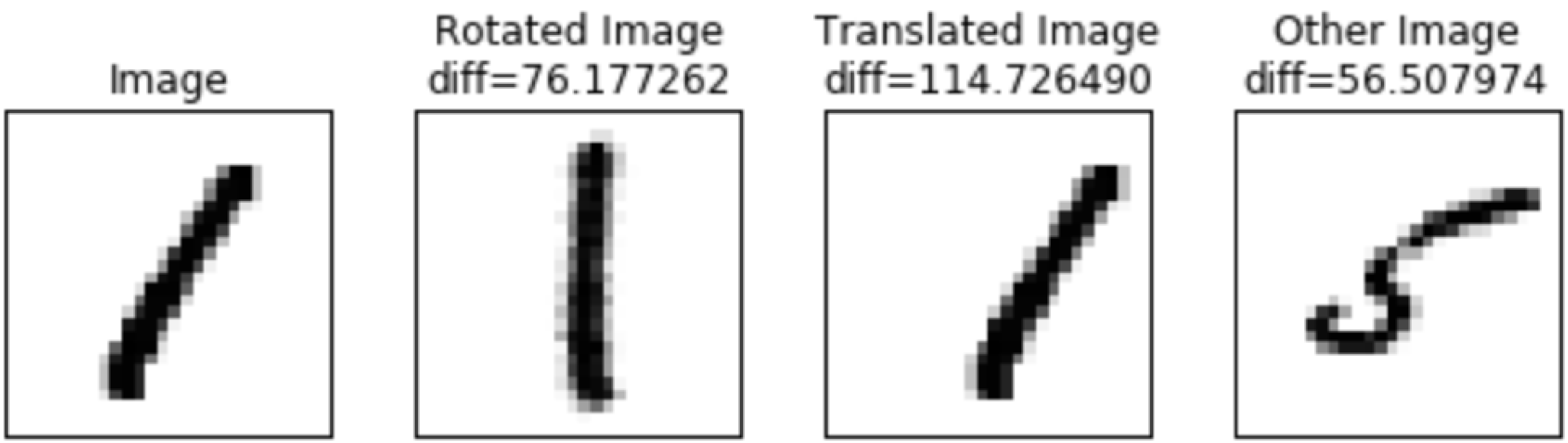




Opposite

K-NN – what makes images similar?

Pixel-wise difference can be large for visually similar images



K-NN domain knowledge

Can only embed through proper distance measure selection.

e.g. minimum over small geometric distortions