Introduction to Recognition

Computer Vision

CS 543 / ECE 549

University of Illinois

Many Slides from D. Hoiem, L. Lazebnik.

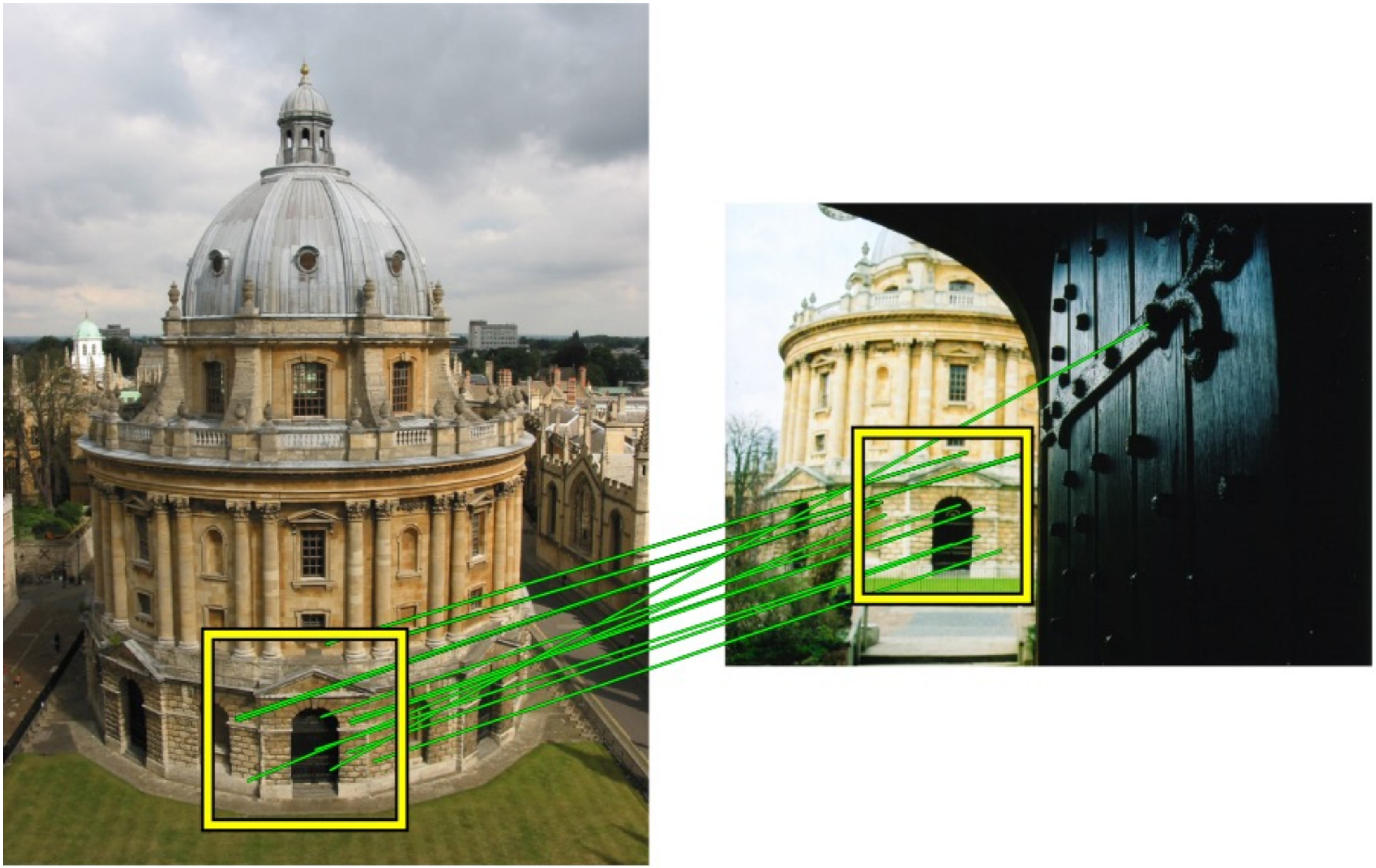
Outline

- Overview
  - Task descriptions
  - Basic approach
- Classifiers
- Features
- Basic Machine Learning Concepts
- Convolutional neural networks (CNNs)

Recognition as 3D Matching

http://www.robots.ox.ac.uk/~vgg/research/oxbuildings/index.html

Recognition as 3D Matching



Recognizing solid objects by alignment with an image. Huttenlocher and Ullman IJCV 1990.

Recognition as 3D Matching

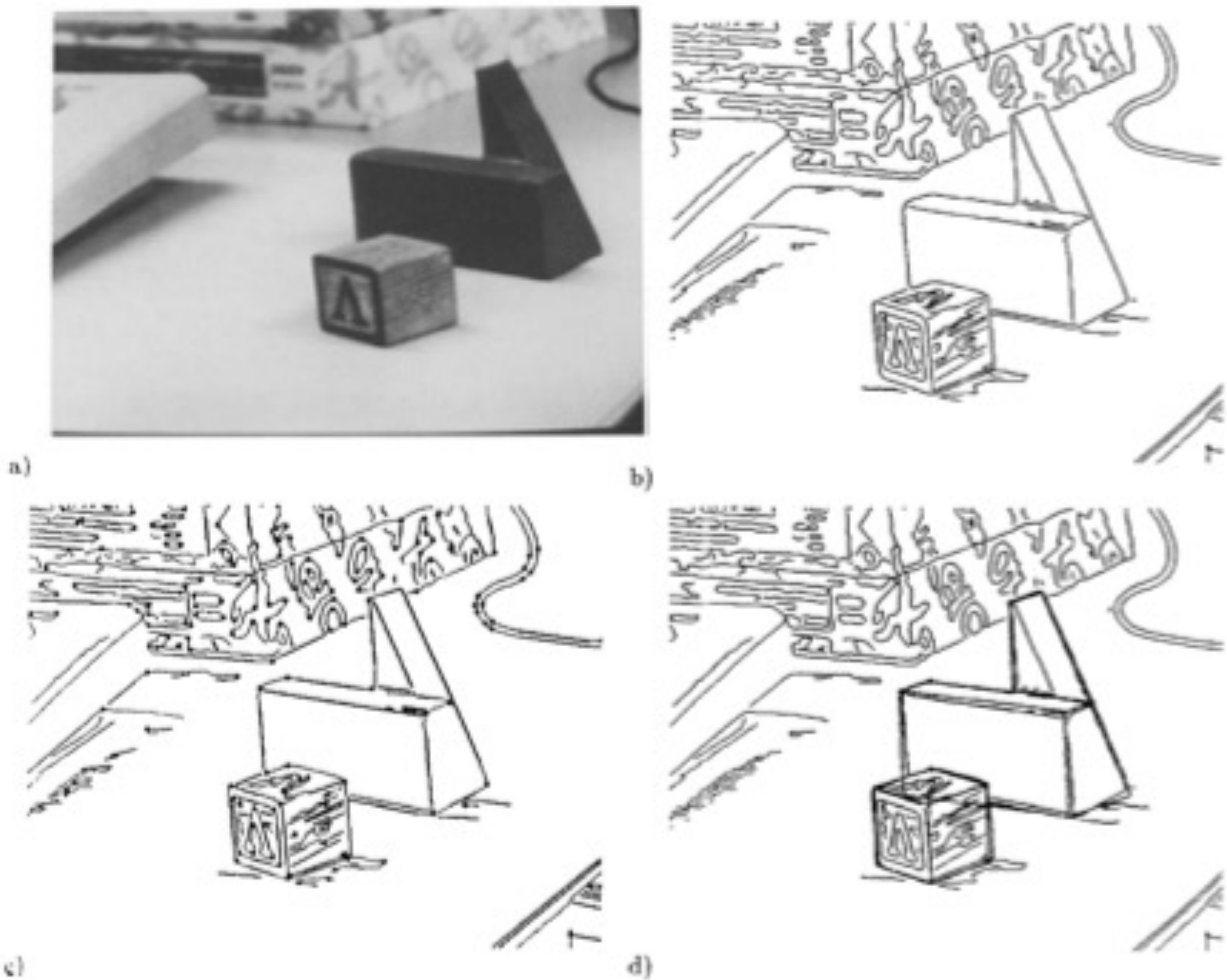


Fig. 8. The output of the recognizer: (a) grey-level image input, (b) Canny edges, (c) edge segments, (d) recovered instances.

“Instance” Recognition

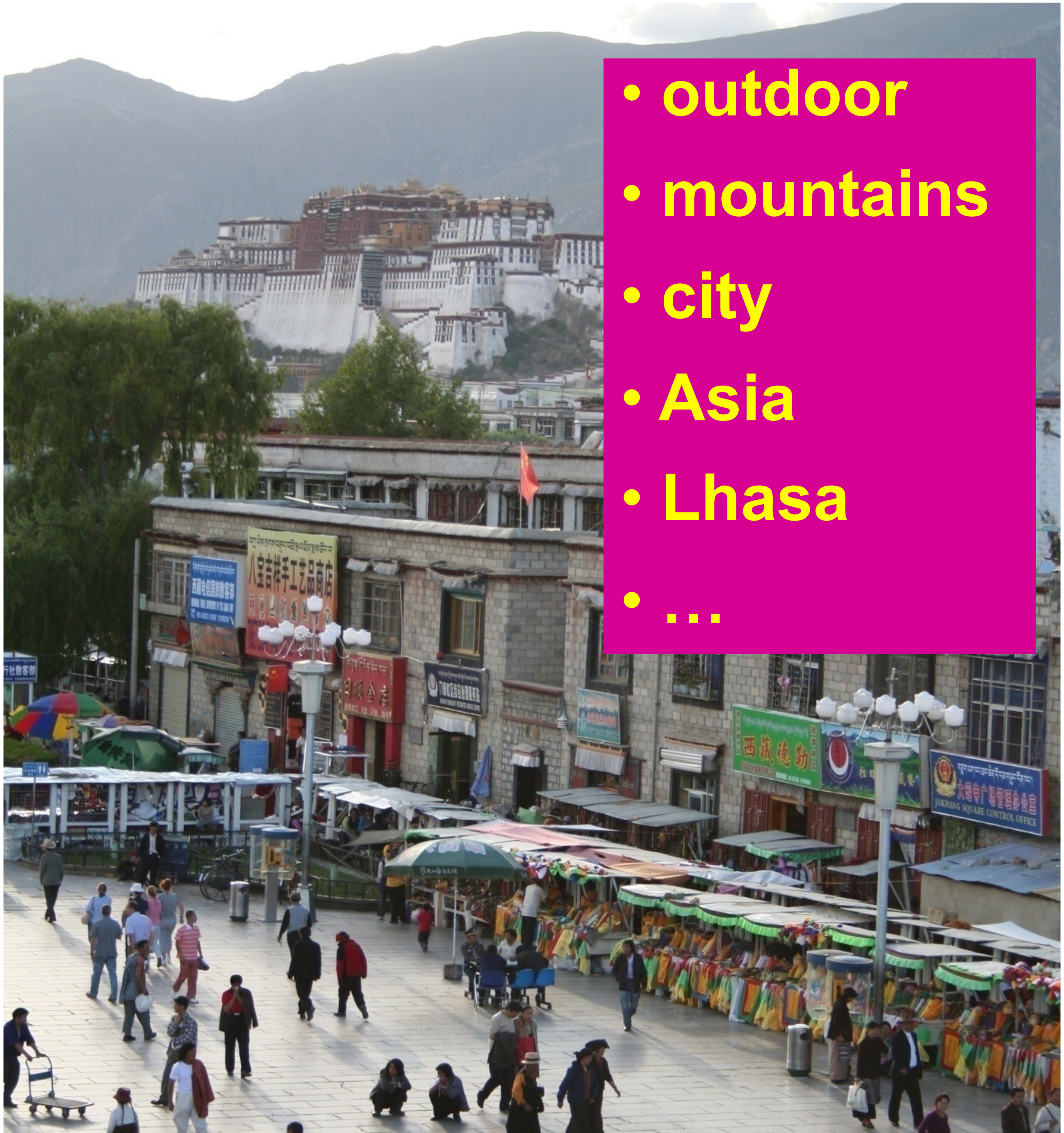
“Category-level” Recognition

Recognizing solid objects by alignment with an image. Huttenlocher and Ullman IJCV 1990.

Common recognition tasks

Adapted from
Fei-Fei Li

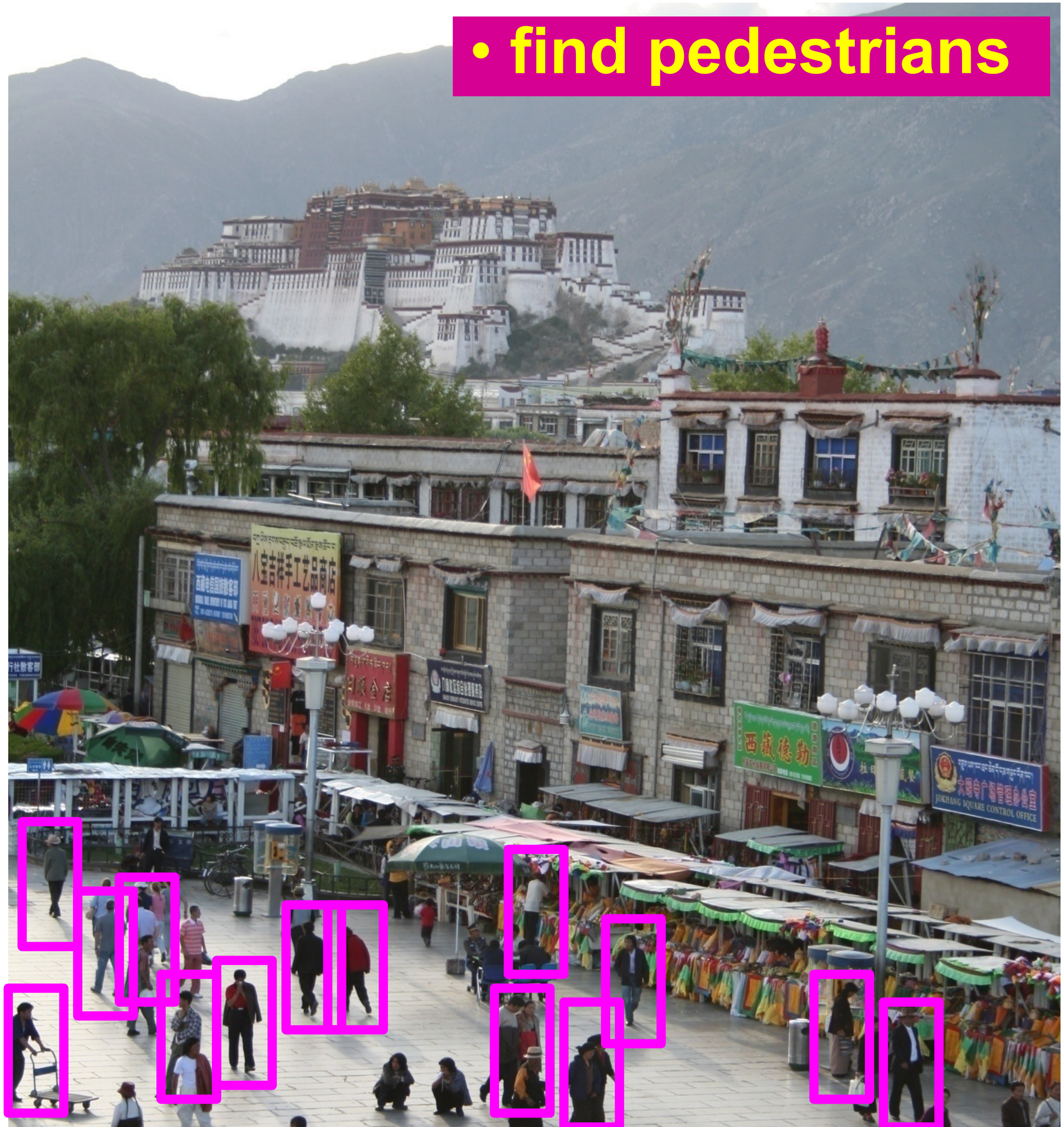
Image classification and tagging



- outdoor
- mountains
- city
- Asia
- Lhasa
- …

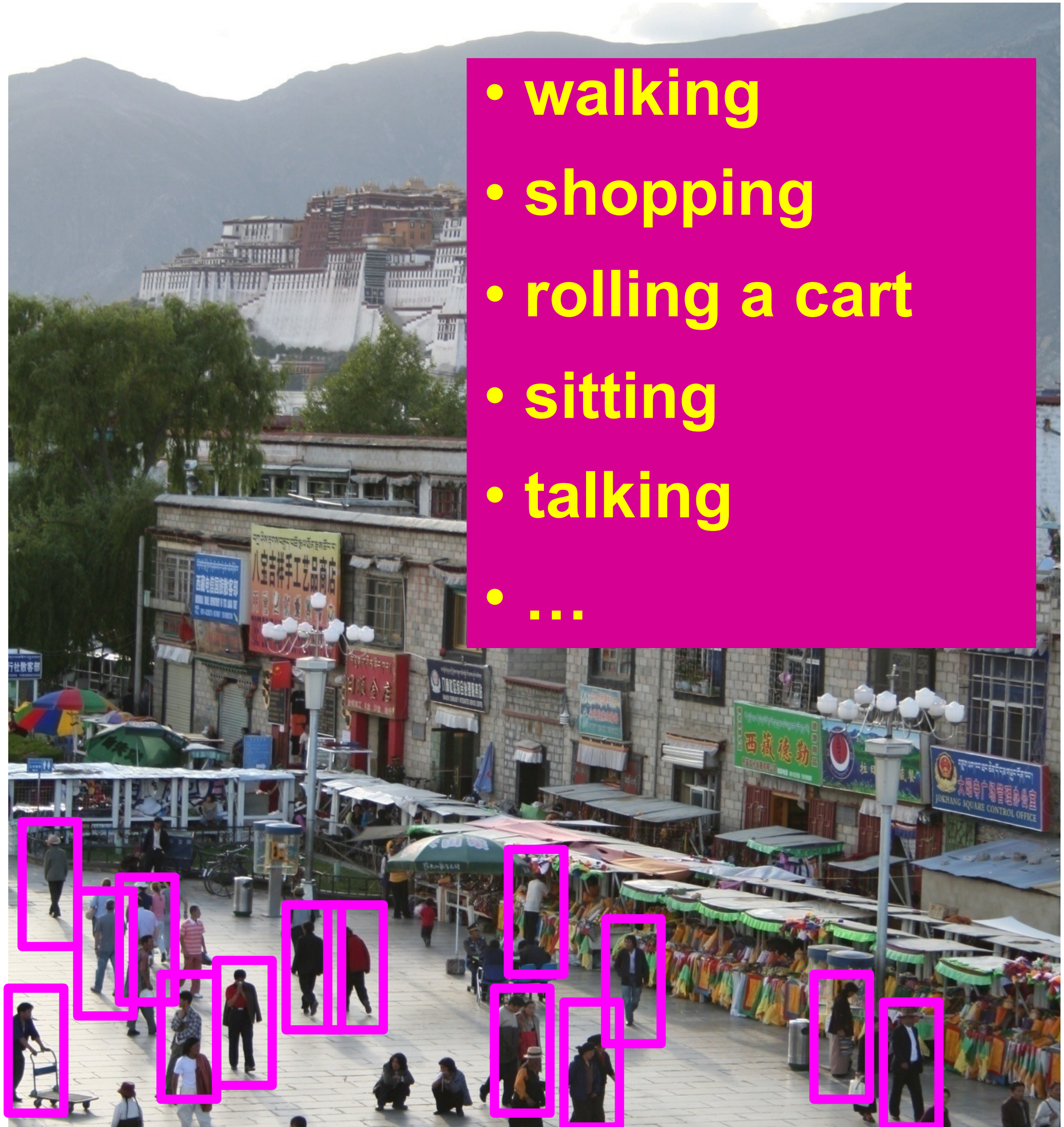
Adapted from Fei-Fei Li

Object detection



Adapted from
Fei-Fei Li

Activity recognition



- walking
- shopping
- rolling a cart
- sitting
- talking
- …

Adapted from
Fei-Fei Li

Semantic segmentation

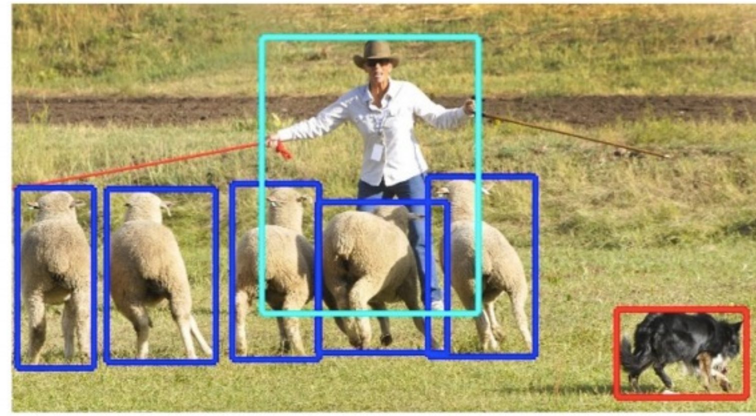
Adapted from
Fei-Fei Li

Semantic segmentation

Adapted from
Fei-Fei Li

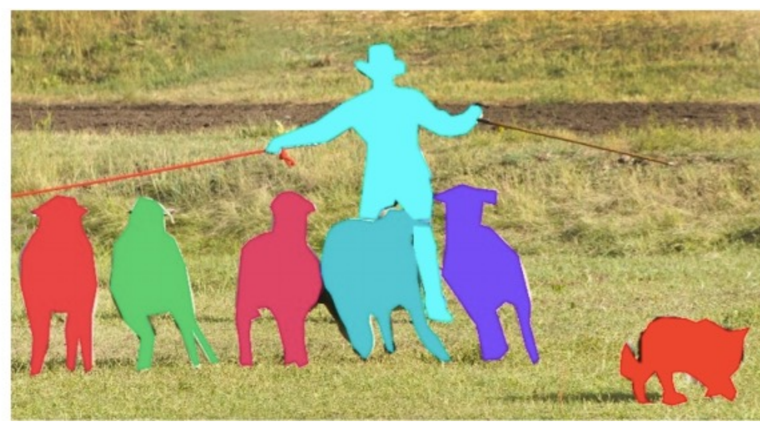
Detection, semantic segmentation, instance segmentation

image classification



object detection

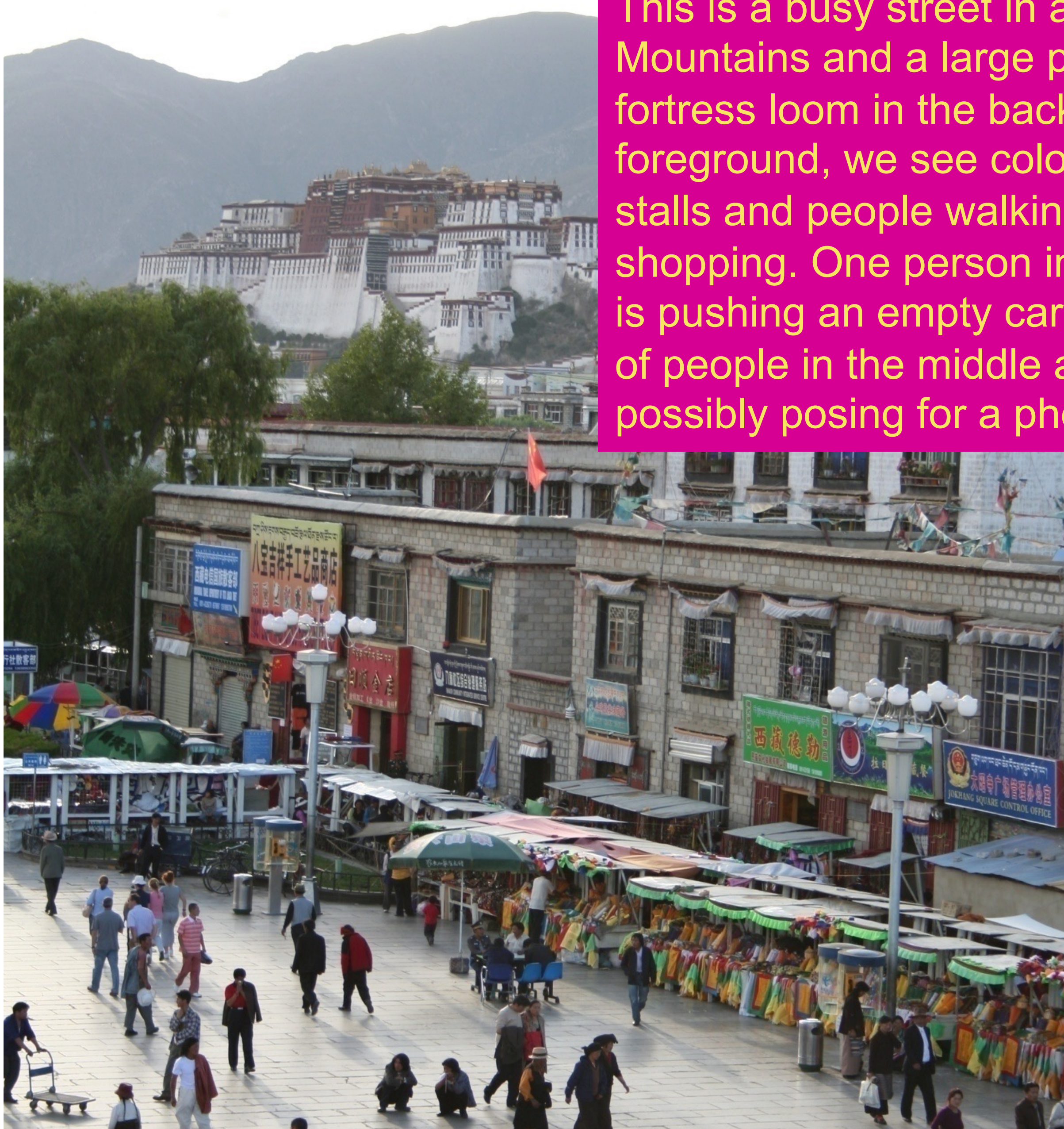
semantic segmentation



instance segmentation

Image source

Image description



This is a busy street in an Asian city. Mountains and a large palace or fortress loom in the background. In the foreground, we see colorful souvenir stalls and people walking around and shopping. One person in the lower left is pushing an empty cart, and a couple of people in the middle are sitting, possibly posing for a photograph.

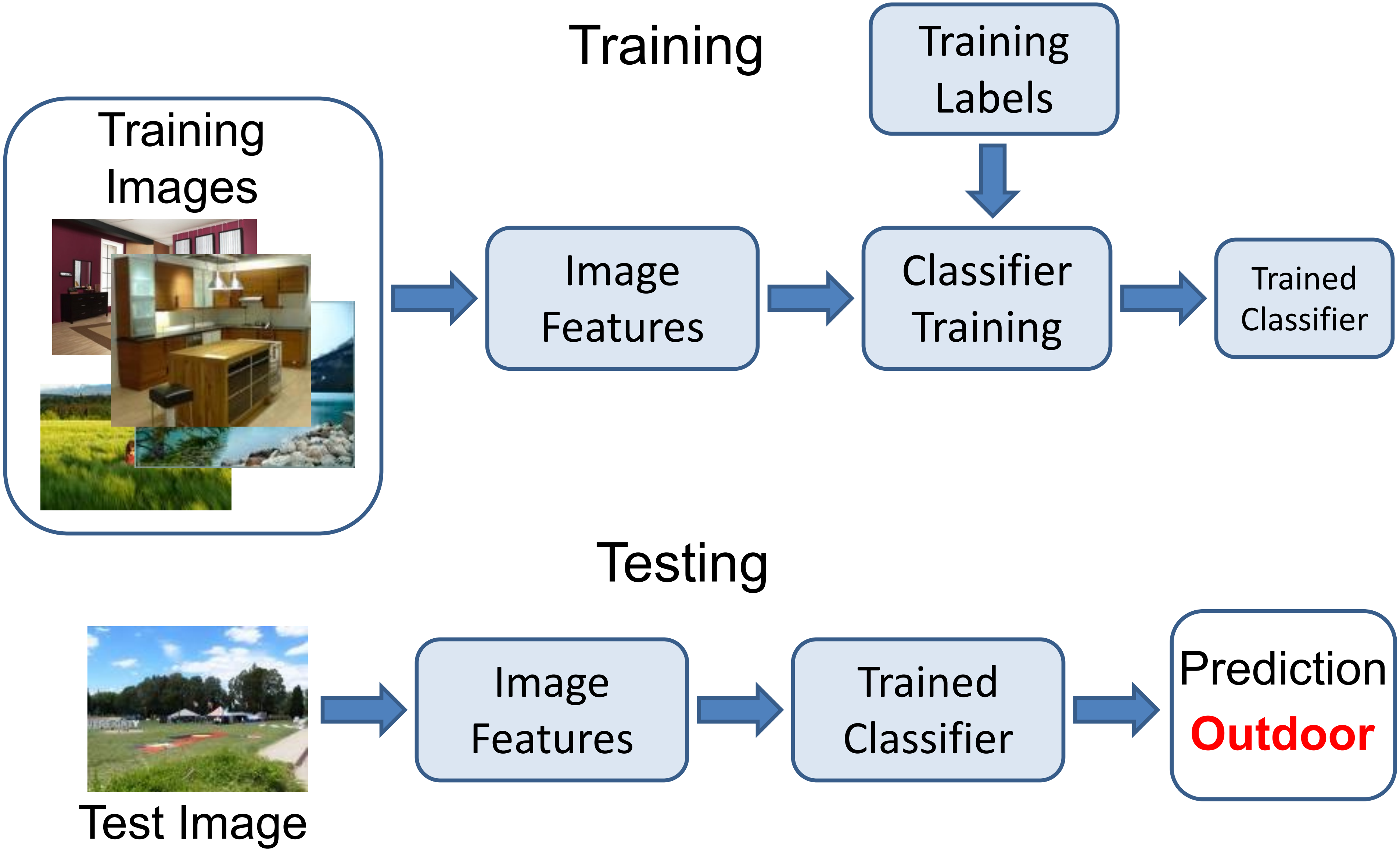
Adapted from Fei-Fei Li

Many vision problems involve categorization

- Image: *Classify* as indoor/outdoor, which room, what objects are there, etc.
- Object Detection: *classify* location (bounding box or region) as object or non-object
- Semantic Segmentation: *classify* pixel into an object, material, part, etc.
- Action Recognition: *classify* a frame or sequence into an action type

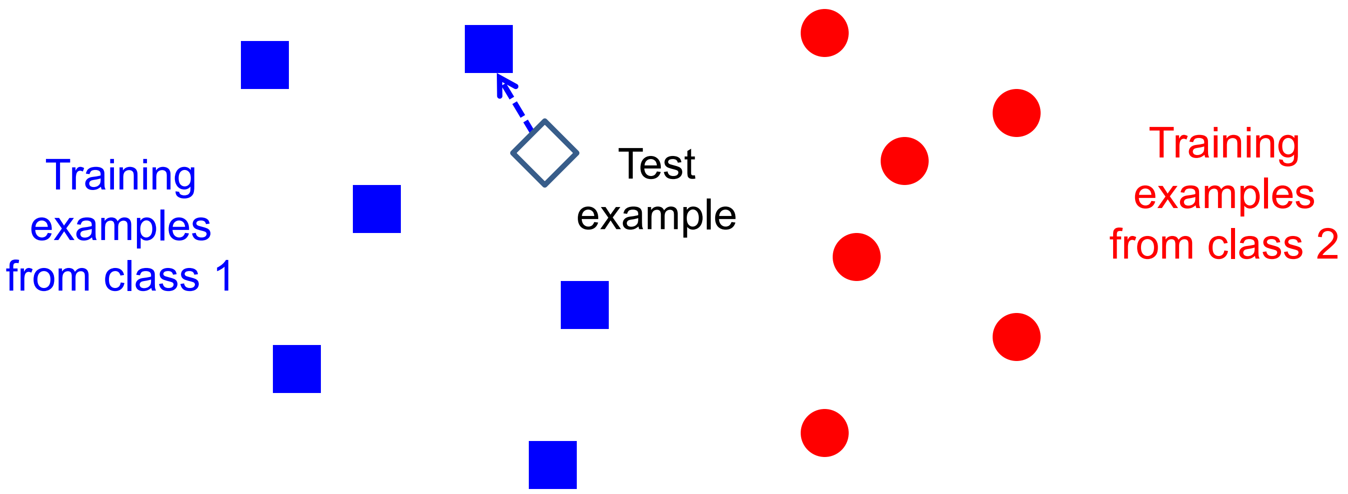
…

Basic Approach: Supervised Learning



- Do you know about the following? (Pick all)
  a) Nearest Neighbor Classifiers
  b) Support Vector Machines
  c) Kernelized Support Vector Machines
  d) Decision Tress
  e) Random Forests

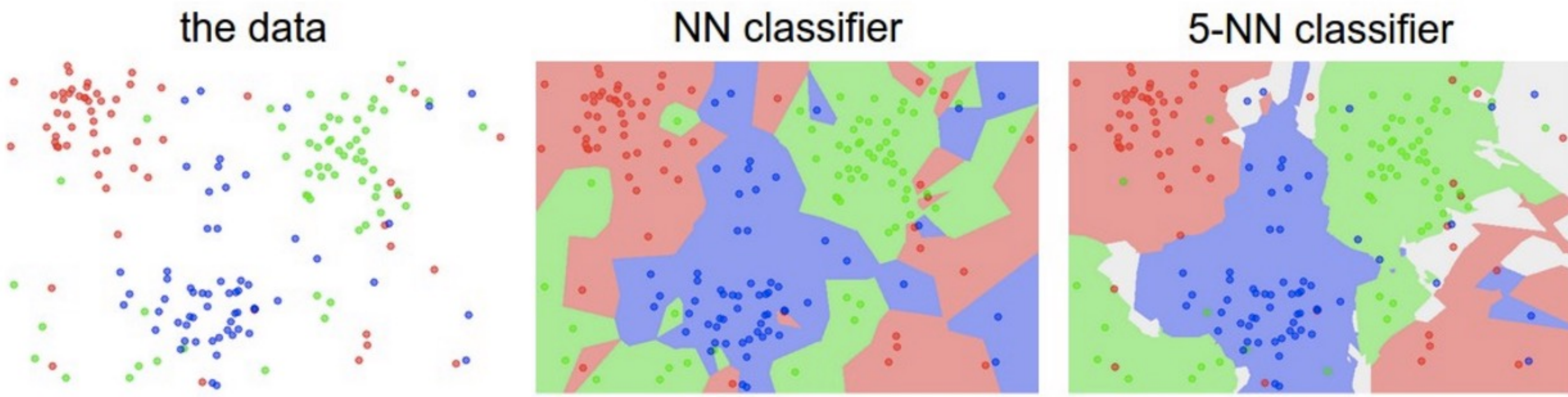
Classifiers: Nearest neighbor



f(**x**) = label of the training example nearest to **x**

- All we need is a distance or similarity function for our inputs
- No training required!

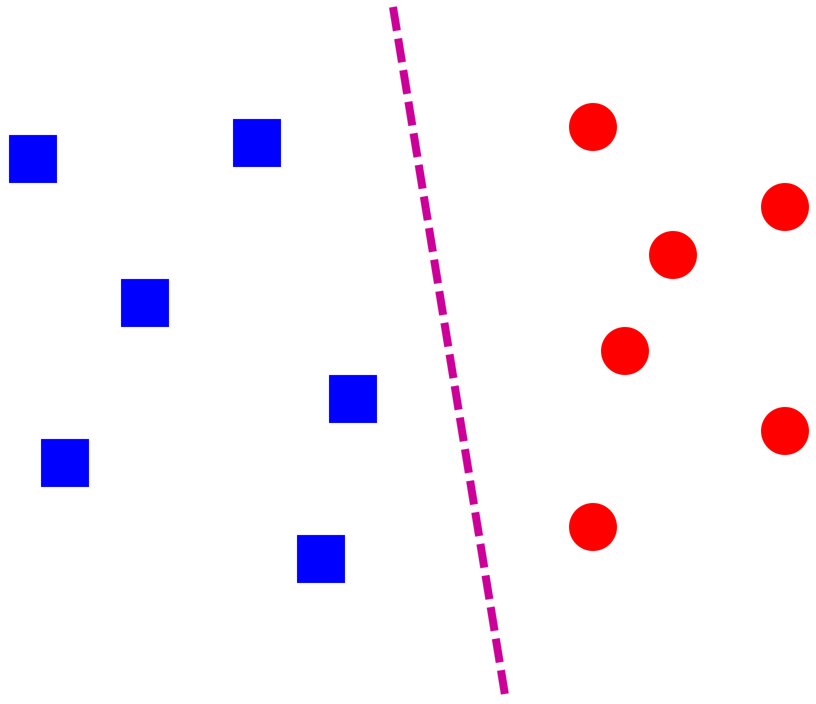
K-nearest neighbor classifier



- Which classifier is more robust to *outliers*?

Credit: Andrej Karpathy, http://cs231n.github.io/classification/

Linear classifiers

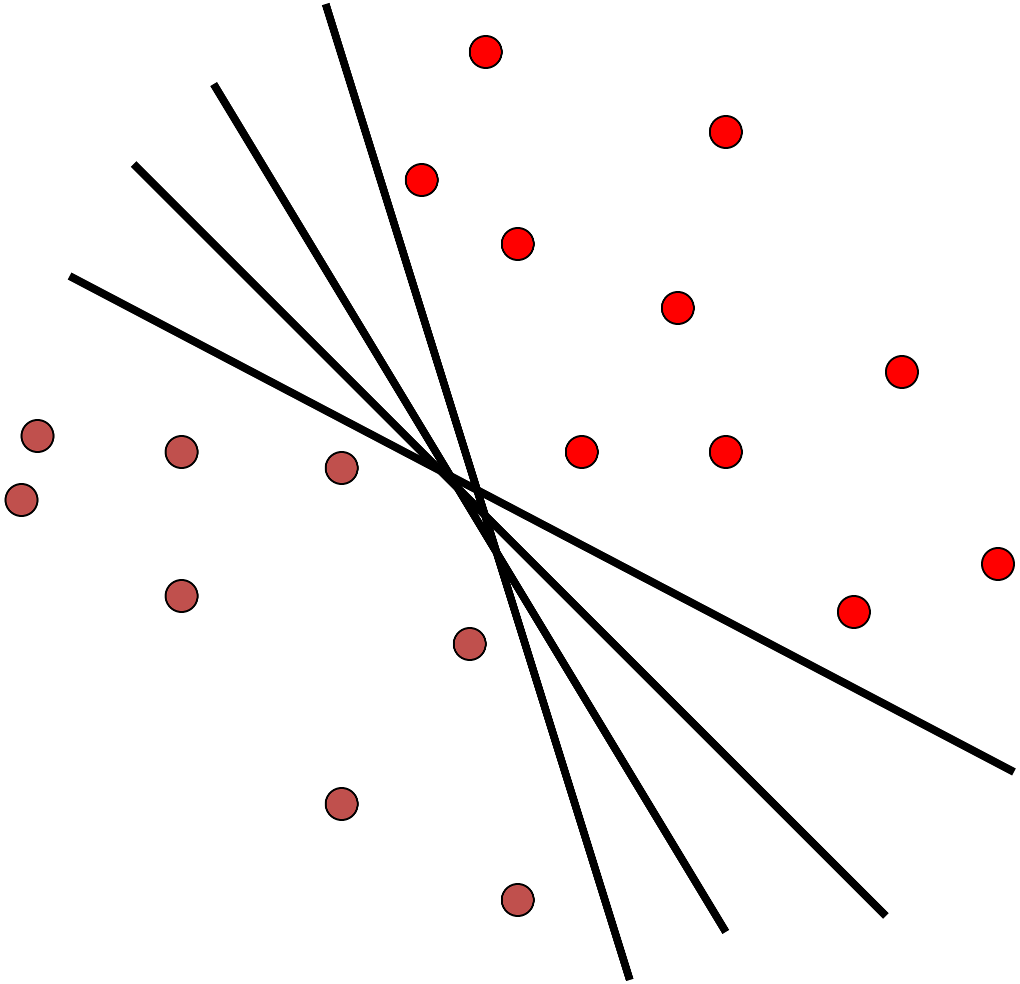


- Find a *linear function* to separate the classes:

$$f(\mathbf{x}) = \text{sign}(\mathbf{w} \cdot \mathbf{x} + b)$$

Linear classifiers

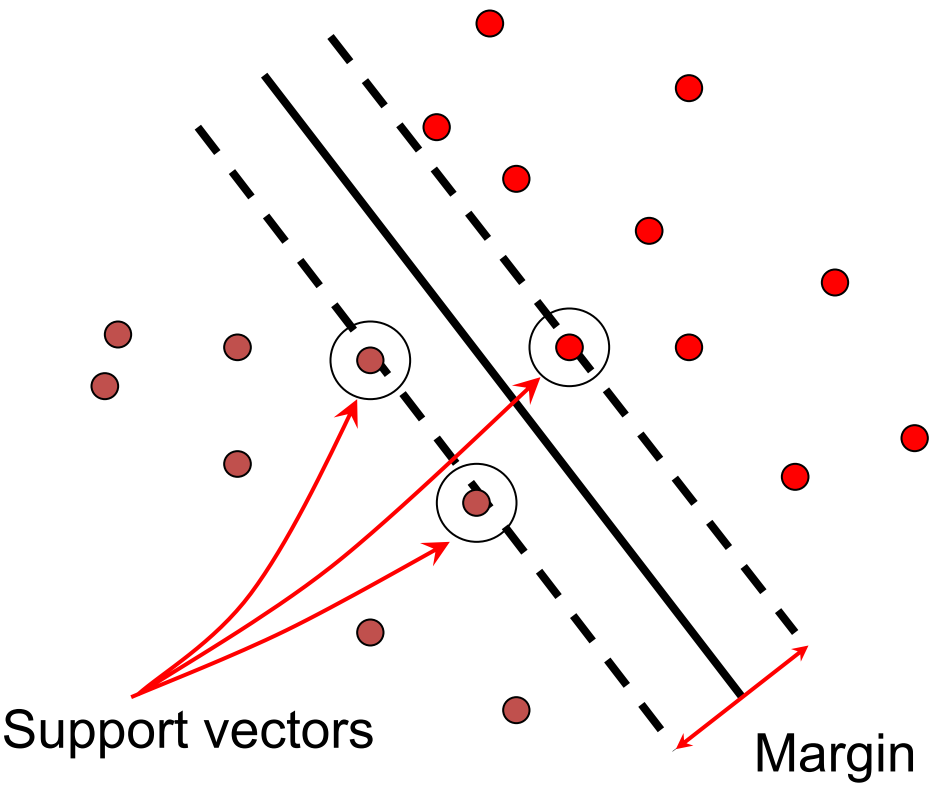
- When the data is linearly separable, there may be more than one separator (hyperplane)



Which separator is best?

Support vector machines

- Find hyperplane that maximizes the *margin* between the positive and negative examples



$\mathbf{x}_i$ positive $(y_i = 1)$: $\quad \mathbf{x}_i \cdot \mathbf{w} + b \geq 1$

$\mathbf{x}_i$ negative $(y_i = -1)$: $\quad \mathbf{x}_i \cdot \mathbf{w} + b \leq -1$

For support vectors, $\quad \mathbf{x}_i \cdot \mathbf{w} + b = \pm 1$

Distance between point and hyperplane: $\quad \dfrac{|\,\mathbf{x}_i \cdot \mathbf{w} + b\,|}{\|\,\mathbf{w}\,\|}$

Therefore, the margin is $2 / \|\mathbf{w}\|$

C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery, 1998

Finding the maximum margin hyperplane

1. Maximize margin $2 / \|\mathbf{w}\|$
2. Correctly classify all training data:

$$\mathbf{x}_i \text{ positive } (y_i = 1): \quad \mathbf{x}_i \cdot \mathbf{w} + b \geq 1$$

$$\mathbf{x}_i \text{ negative } (y_i = -1): \quad \mathbf{x}_i \cdot \mathbf{w} + b \leq -1$$

- *Quadratic optimization problem*:
- $\min_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w}\|^2 \quad \text{subject to} \quad y_i(\mathbf{w} \cdot \mathbf{x}_i + b) \geq 1$

C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery, 1998

SVM parameter learning

- Separable data:

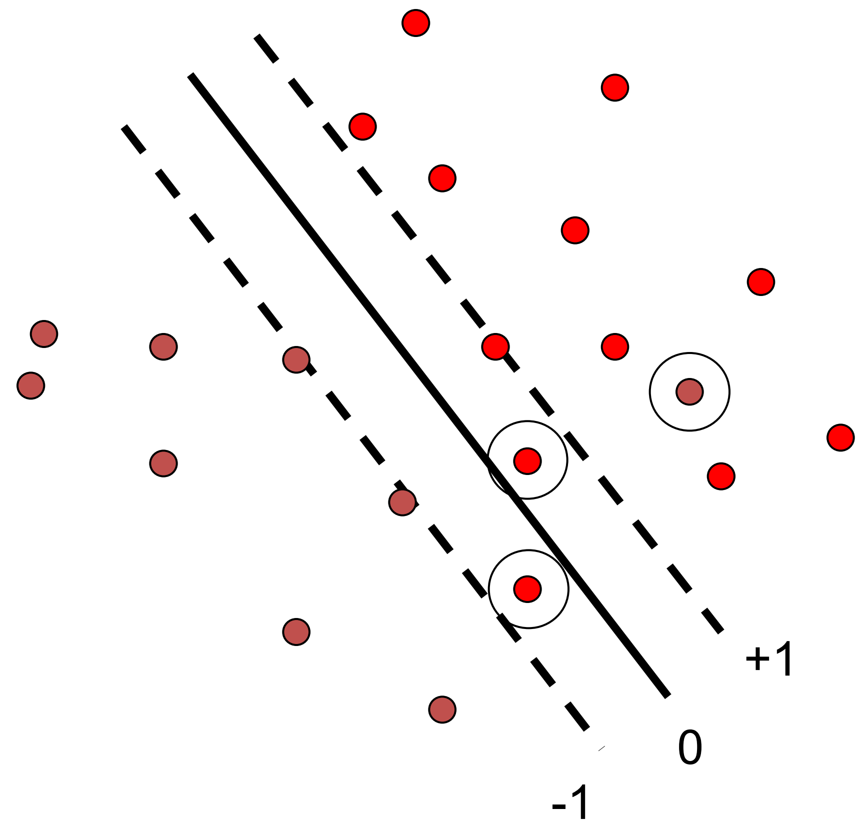
$$\underbrace{\min_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w}\|^2}_{\boxed{\text{Maximize margin}}} \quad \underbrace{\text{subject to} \quad y_i(\mathbf{w}\cdot\mathbf{x}_i + b) \geq 1}_{\boxed{\text{Classify training data correctly}}}$$

- Non-separable data:

$$\min_{\mathbf{w},b} \quad \underbrace{\frac{1}{2}\|\mathbf{w}\|^2}_{\boxed{\text{Maximize margin}}} + C \underbrace{\sum_{i=1}^{n} \max\left(0, 1 - y_i(\mathbf{w}\cdot\mathbf{x}_i + b)\right)}_{\boxed{\text{Minimize classification mistakes}}}$$

SVM parameter learning

$$\min_{\mathbf{w},b} \quad \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n}\max\left(0, 1 - y_i(\mathbf{w}\cdot\mathbf{x}_i + b)\right)$$



- Demo: http://cs.stanford.edu/people/karpathy/svmjs/demo

Nonlinear SVMs

- **General idea:** the original input space can always be mapped to some higher-dimensional feature space where the training set is separable

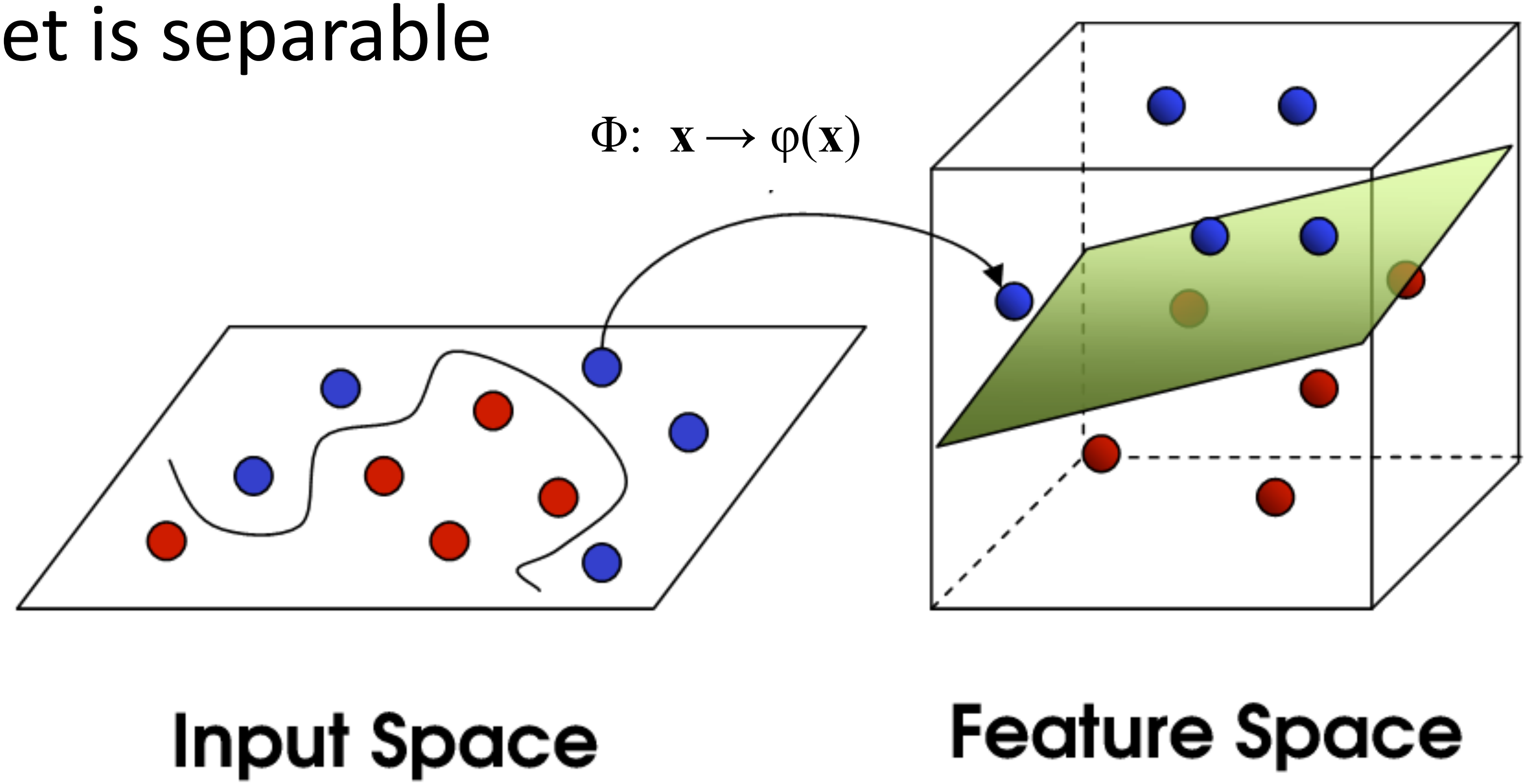
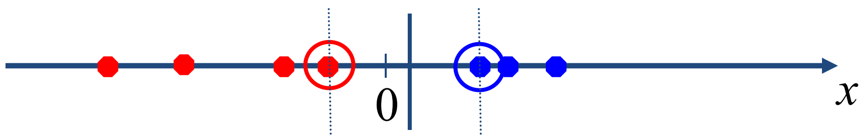


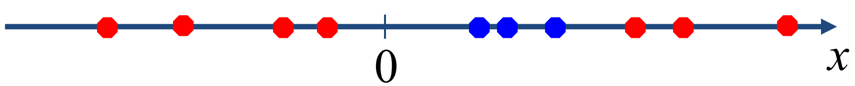
Image source

Nonlinear SVMs

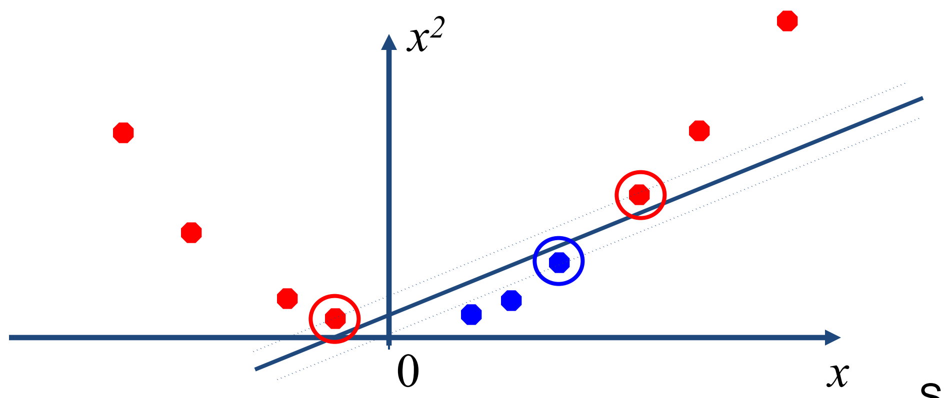
- Linearly separable dataset in 1D:



- Non-separable dataset in 1D:



- We can map the data to a *higher-dimensional space*:



Slide credit: Andrew Moore

The kernel trick

- **General idea:** the original input space can always be mapped to some higher-dimensional feature space where the training set is separable

- **The kernel trick:** instead of explicitly computing the lifting transformation $\boldsymbol{\varphi}(\mathbf{x})$, define a kernel function K such that

$$K(\mathbf{x}, \mathbf{y}) = \boldsymbol{\varphi}(\mathbf{x}) \cdot \boldsymbol{\varphi}(\mathbf{y})$$

- (to be valid, the kernel function must satisfy *Mercer's condition*)

The kernel trick

- Linear SVM decision function:

$$\mathbf{w}\cdot\mathbf{x}+b=\sum_i \alpha_i y_i \mathbf{x}_i\cdot\mathbf{x}\ +b$$

learned weight

Support vector

C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery, 1998

The kernel trick

- Linear SVM decision function:

$$\mathbf{w} \cdot \mathbf{x} + b = \sum_i \alpha_i y_i \mathbf{x}_i \cdot \mathbf{x} + b$$

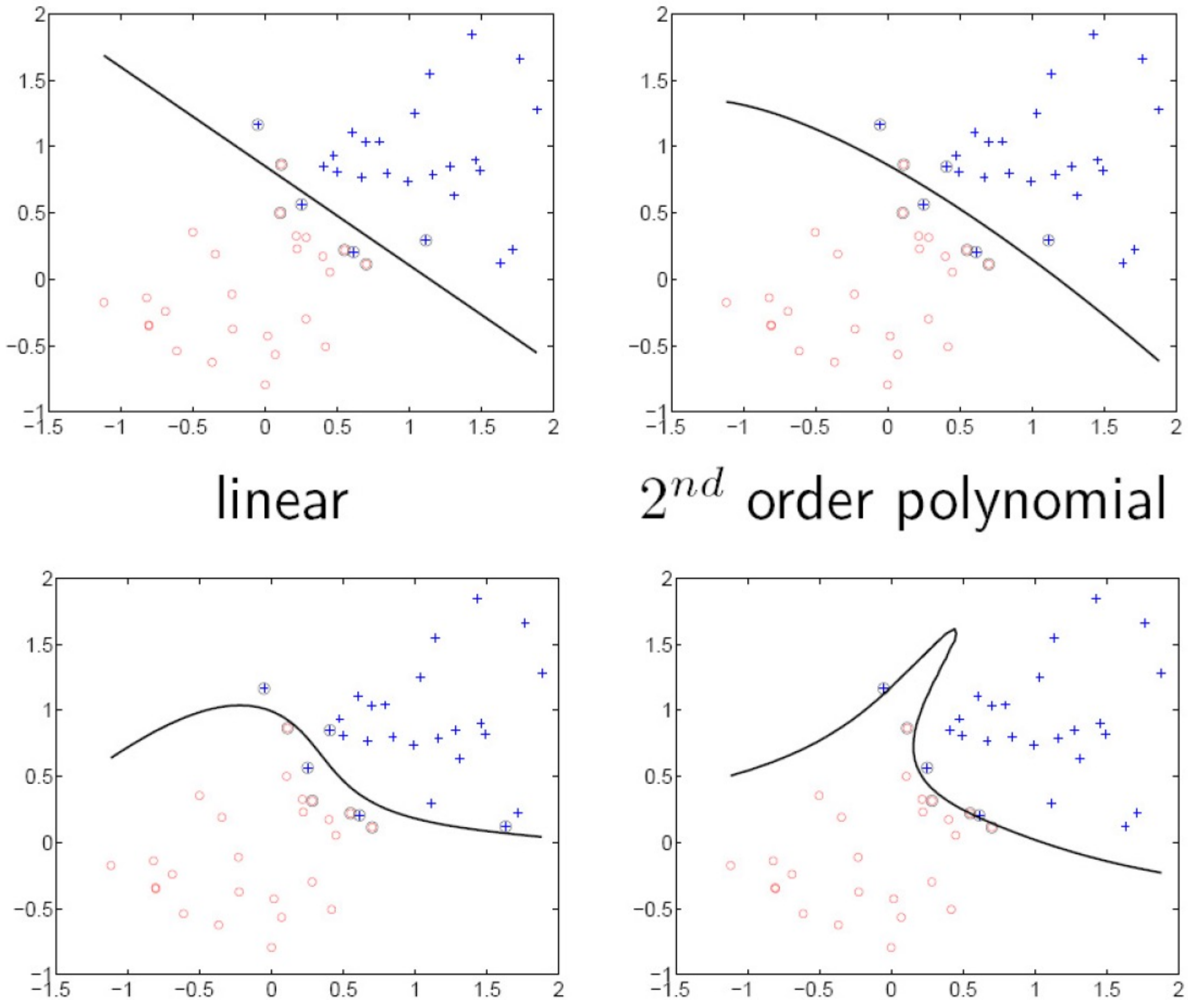
- Kernel SVM decision function:

$$\sum_i \alpha_i y_i \varphi(\mathbf{x}_i) \cdot \varphi(\mathbf{x}) + b = \sum_i \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) + b$$

- This gives a nonlinear decision boundary in the original feature space

C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery, 1998

Polynomial kernel: $K(\mathbf{x},\mathbf{y}) = (c + \mathbf{x}\cdot\mathbf{y})^d$



linear

2^{nd} order polynomial

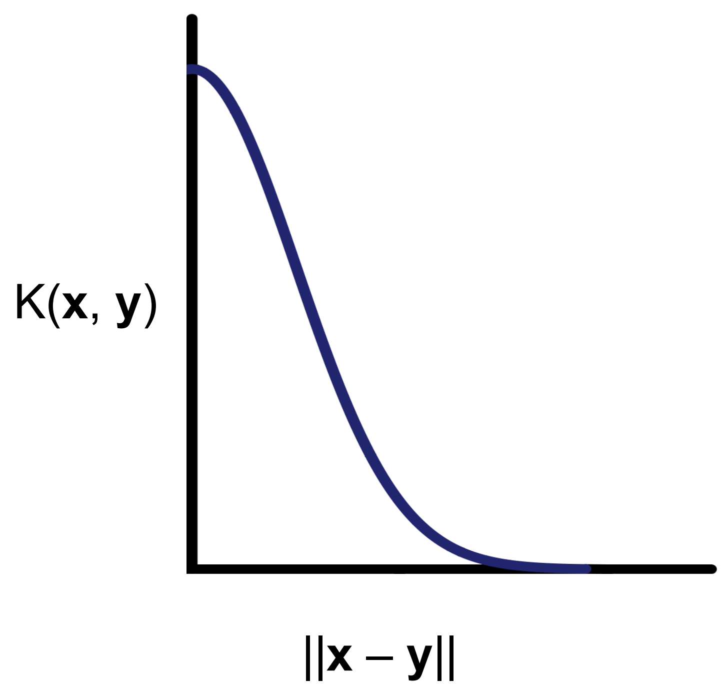
4^{th} order polynomial

8^{th} order polynomial

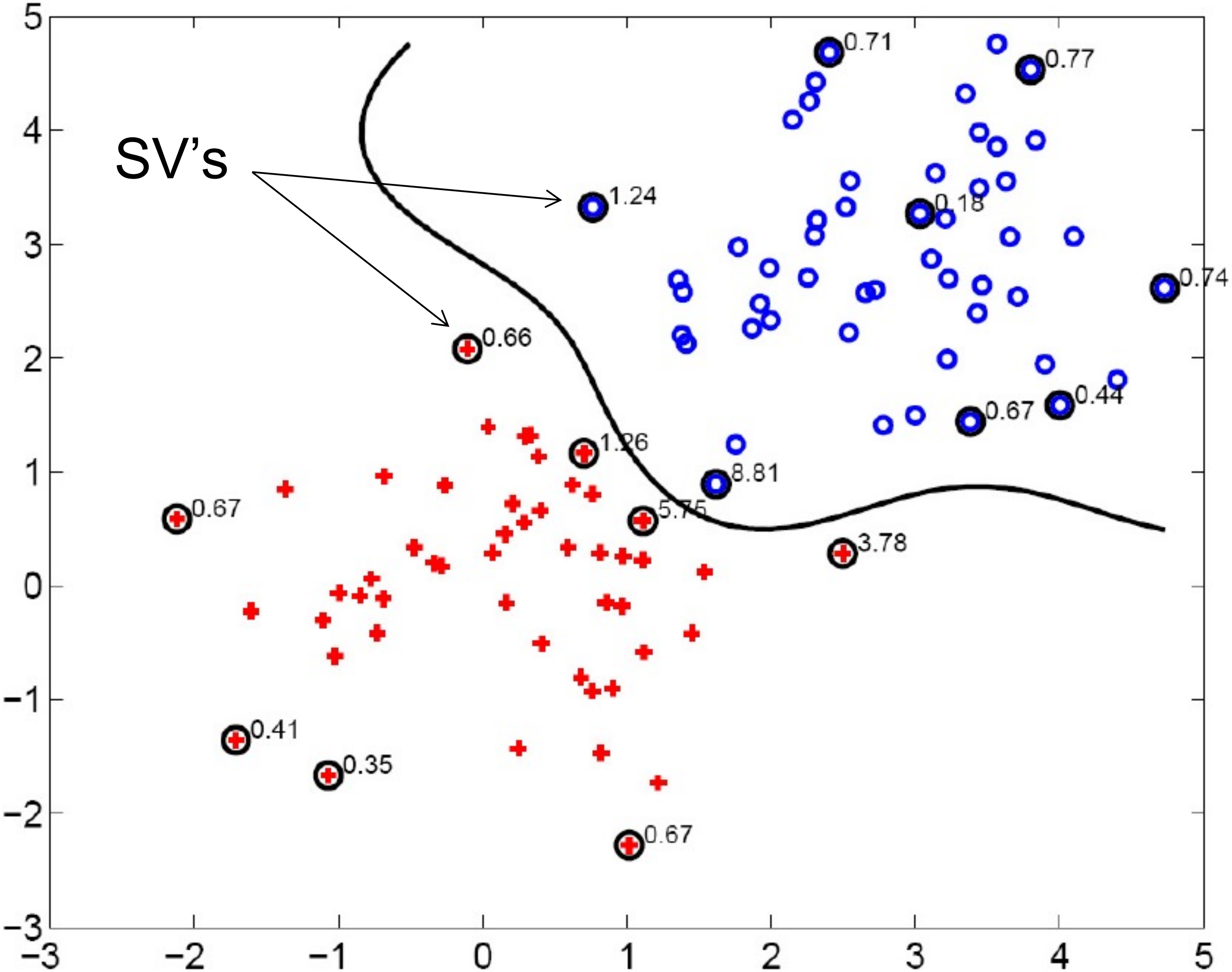
Gaussian kernel

- Also known as the radial basis function (RBF) kernel:

$$K(\mathbf{x}, \mathbf{y}) = \exp\left(-\frac{1}{\sigma^2} \|\mathbf{x} - \mathbf{y}\|^2 \right)$$



Gaussian kernel



Outline

- Overview
  - Task descriptions
  - Basic approach
- Classifiers
- Features
- Basic Machine Learning Concepts
- Convolutional neural networks (CNNs)

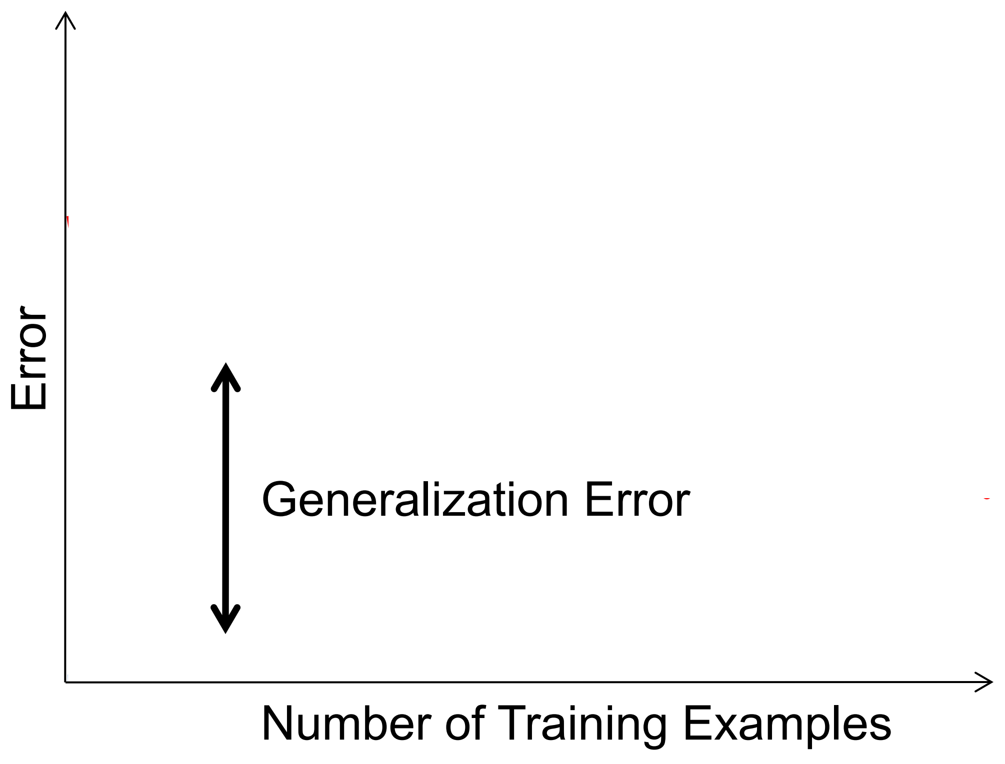
Digit Classification Case Study

The MNIST DATABASE of handwritten digits
Yann LeCun & Corinna Cortes

- Has a training set of 60 K examples (6K examples for each digit), and a test set of 10K examples.
- Each digit is a 28 x 28 pixel grey level image. The digit itself occupies the central 20 x 20 pixels, and the center of mass lies at the center of the box.

Generalization Error

Fixed classifier



Bias-Variance Trade-off

$$E(MSE) = noise^2 + bias^2 + variance$$

Unavoidable error

Error due to incorrect assumptions

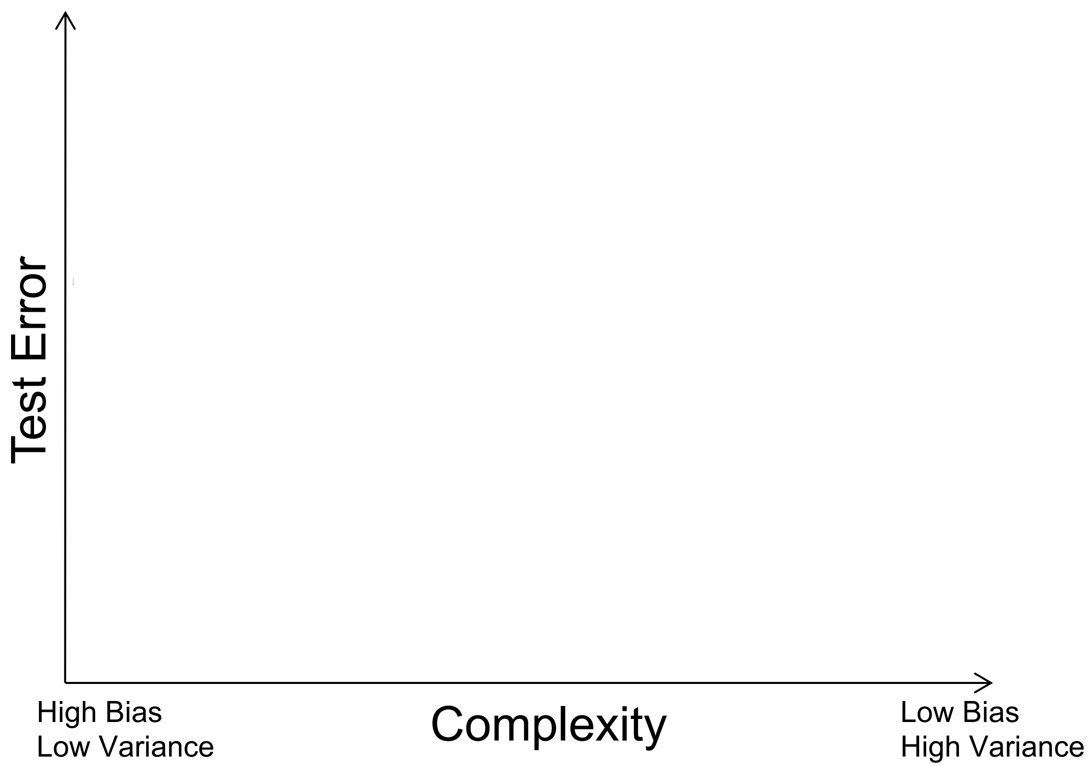
Error due to variance parameter estimates from training samples

See the following for explanation of bias-variance (also Bishop's "Neural Networks" book):

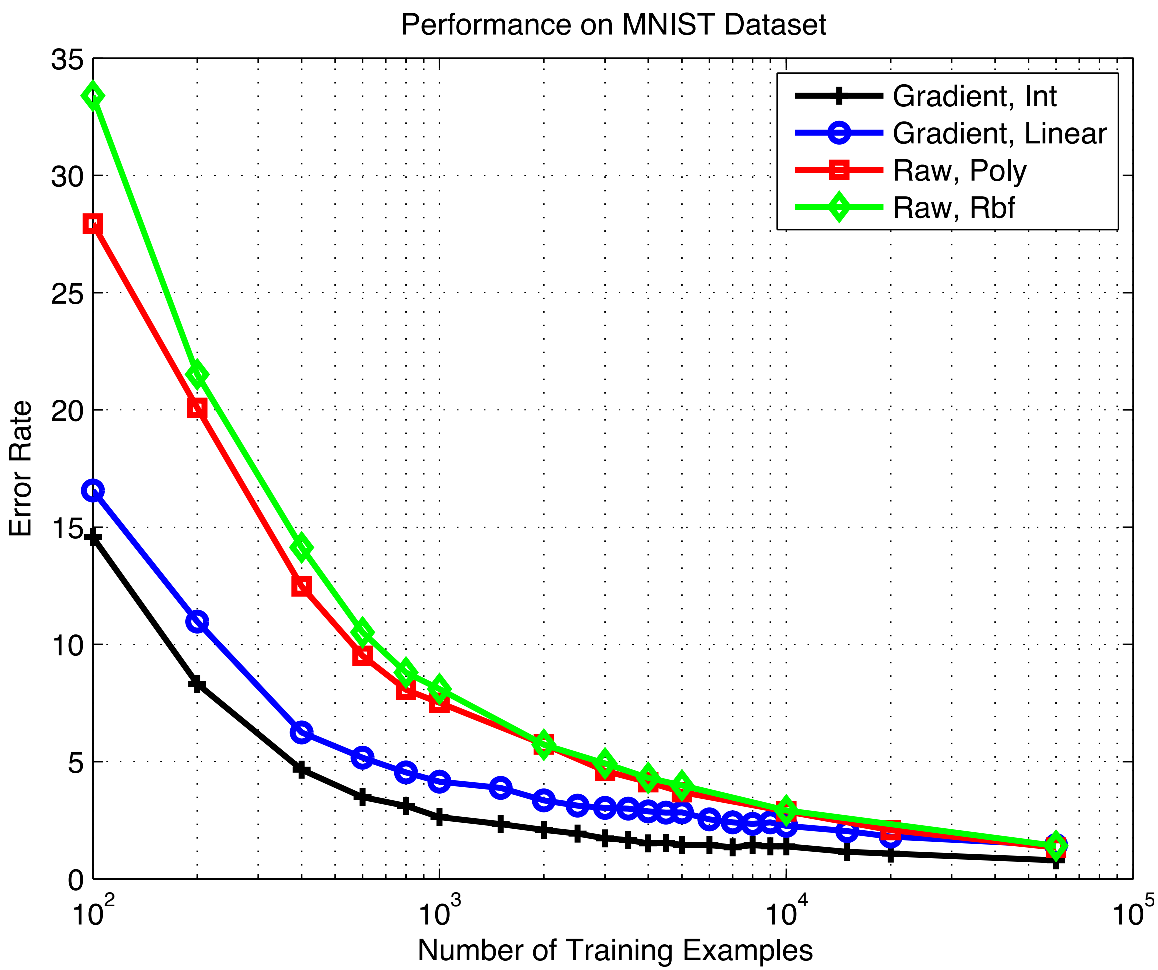
- http://www.inf.ed.ac.uk/teaching/courses/mlsc/Notes/Lecture4/BiasVariance.pdf

Bias and Variance

$$\text{Error} = \text{noise}^2 + \text{bias}^2 + \text{variance}$$



Back to the case study



What are the right features?

Depend on what you want to know!

- Object: shape
  - Local shape info, shading, shadows, texture
- Scene : geometric layout
  - linear perspective, gradients, line segments
- Material properties: albedo, feel, hardness
  - Color, texture
- Action: motion
  - Optical flow, tracked points

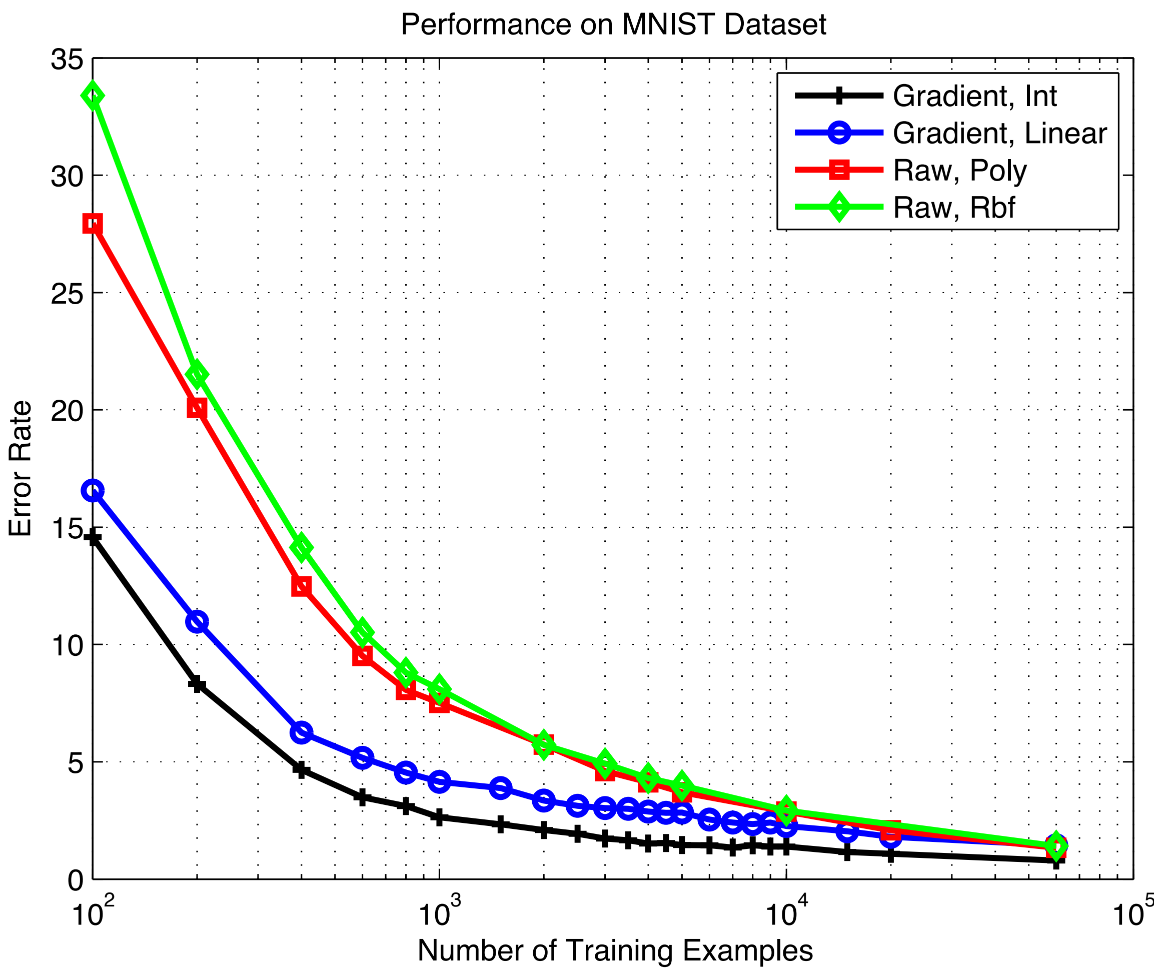
Stuff vs Objects

- recognizing cloth fabric vs recognizing cups

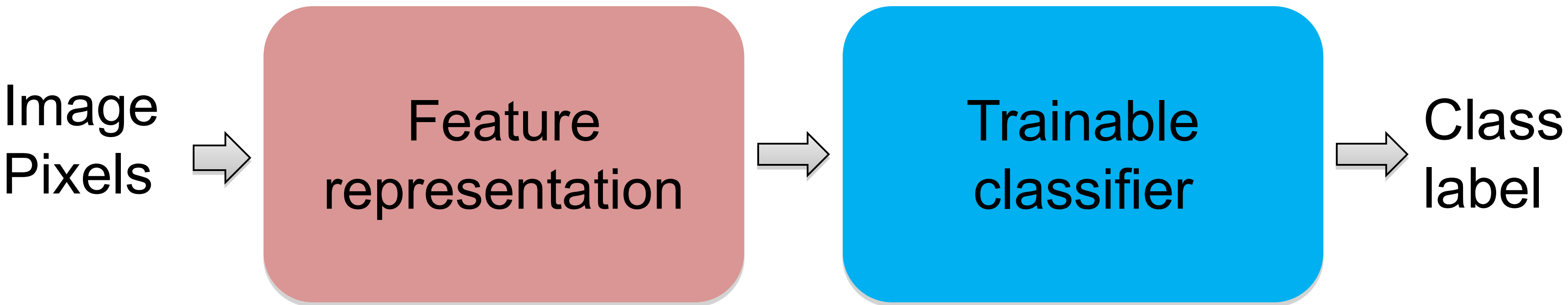
Feature Design Process

1. Start with a model
2. Look at errors on development set
3. Think of features that can improve performance
4. Develop new model, test whether new features help.
5. If not happy, go to step 1.
6. “Ablations”: Simplify system, prune out features that don’t help anymore in presence of other features.

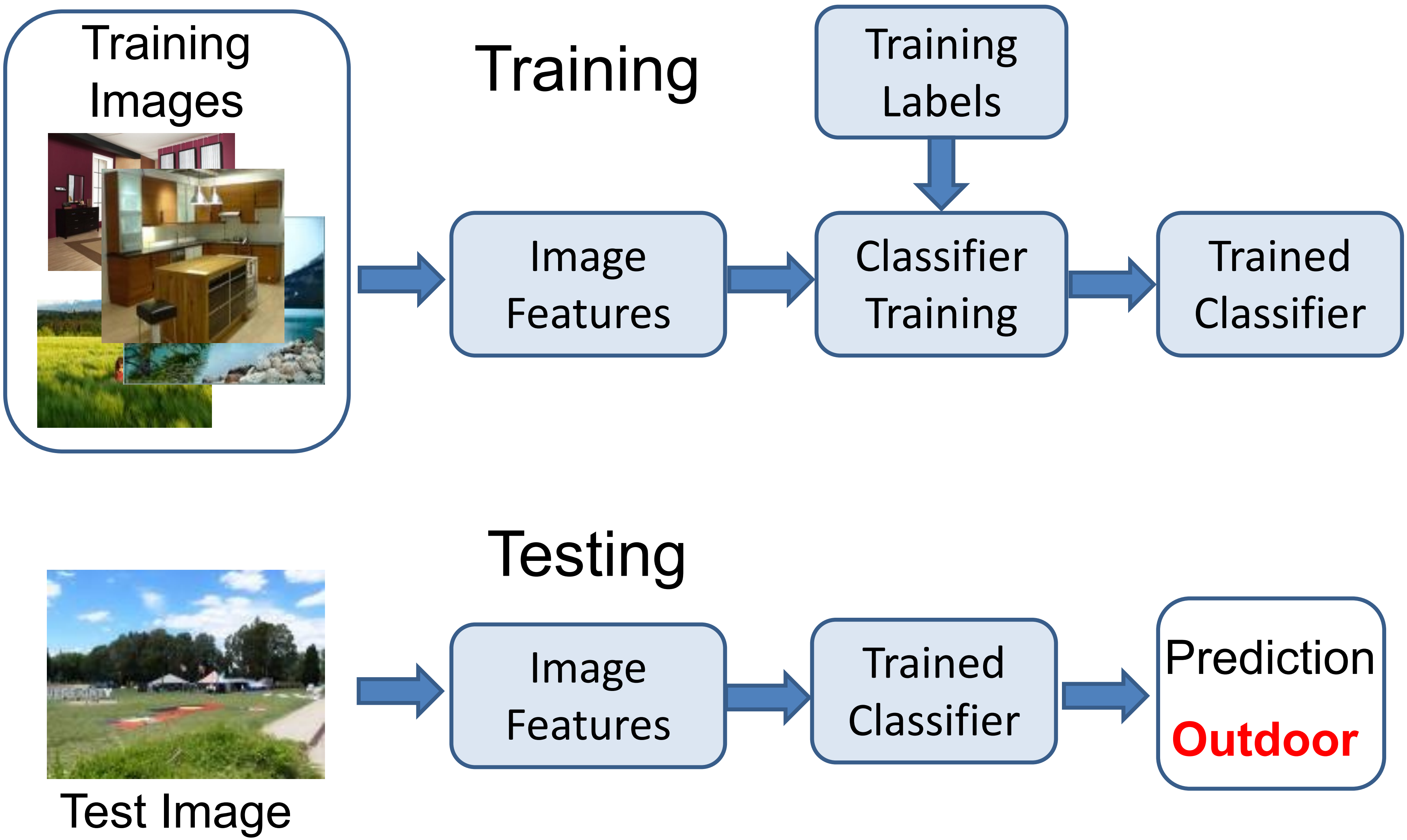
Features vs Classifiers



“Classic” recognition pipeline



Categorization involves **features** and a classifier



New training setup with moderate sized datasets