

# Image Analysis

## 1. A First Look at Image Classification

Lars Schmidt-Thieme

Information Systems and Machine Learning Lab (ISMLL)  
Institute for Business Economics and Information Systems  
& Institute for Computer Science  
University of Hildesheim  
<http://www.ismll.uni-hildesheim.de>

---

Lars Schmidt-Thieme, Information Systems and Machine Learning Lab (ISMLL), Institute BW/WI & Institute for Computer Science, University of Hildesheim  
Course on Image Analysis, winter term 2008

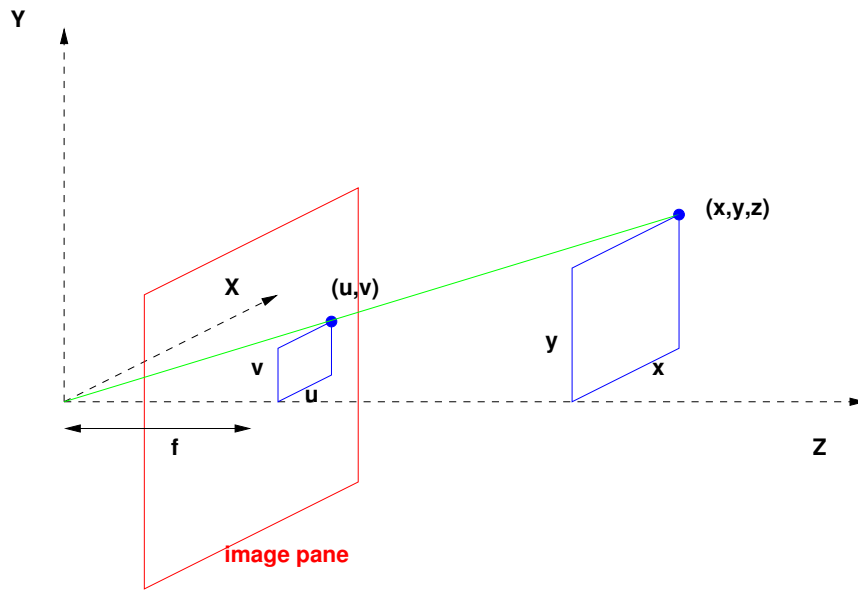
1/35

### 1. Digital Images

### 2. Image Interpolation

### 3. Image Classification

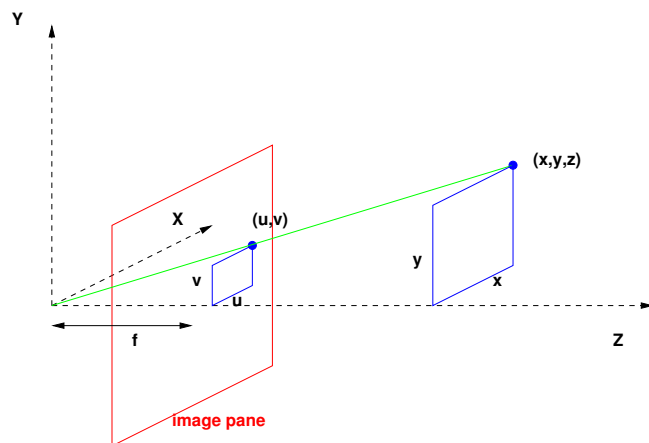
## Perspective Projection



Lars Schmidt-Thieme, Information Systems and Machine Learning Lab (ISMLL), Institute BW/WI & Institute for Computer Science, University of Hildesheim  
Course on Image Analysis, winter term 2008

1/35

## Perspective Projection



Two-dimensional images often are projections from 3D scenes:

$$\begin{aligned}
 X &= (x, y, z) && \text{co-ordinates in 3D scene} \\
 u &= (u, v) && \text{co-ordinates in 2D image pane} \\
 u &= \frac{x f}{z}, \quad v = \frac{y f}{z} && \text{transformation}
 \end{aligned}$$

$f$  is called **focal length**.

## Image Functions

Images are described by **image functions**, that for each co-ordinate tuple provide an **intensity value** (also: **brightness value**).

**2D Continuous image function:**

$$x : X \times Y \rightarrow I$$

where

- $X, Y$  are intervalls in  $\mathbb{R}$  defining the ranges of the co-ordinate values and
- $I$  defines the intensity of the points.

**discrete / digital / raster 2d image function:**

$$u : U \times V \rightarrow J$$

where

- $U := \{0, 1, 2, \dots, n - 1\}$  and  $V := \{0, 1, 2, \dots, m - 1\}$  are discrete co-ordinate ranges and
- $J$  defines discrete intensities of pixels.

## Intensities

Different types of images are modelled by different intensity ranges:

**Binary images:**

$$I := \{0, 1\}$$

i.e., each pixel can be either white (1) or black (0).

**Gray-level images:**

$$I := [0, I_{\max}] \quad \text{or} \quad J := \{0, 1, \dots, I_{\max}\}$$

i.e., each pixel can have a scalar grey-level intensity value between white ( $I_{\max}$ ) and black (0).

## Intensities

binary image:



gray-level image (256 different levels)



Lars Schmidt-Thieme, Information Systems and Machine Learning Lab (ISMLL), Institute BW/WI & Institute for Computer Science, University of Hildesheim  
Course on Image Analysis, winter term 2008

4/35

Image Analysis / 1. Digital Images

## Intensities / Color

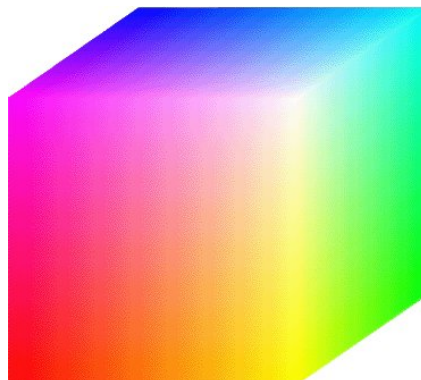
**Color images:**

$$I := [0, I_{\max}]^3 \quad \text{or} \quad J := \{0, 1, \dots, I_{\max}\}^3$$

i.e., each pixel is described by 3 different intensity values between full intensity ( $I_{\max}$ ) and no intensity (0) for three different color components, e.g., red, green, blue (**RGB color model**).

These different values are called **channels**.

RGB cube:



[from <http://imageprocessing.wordpress.com/2008/03/14/image-segmentation/>]

## Intensities / Color



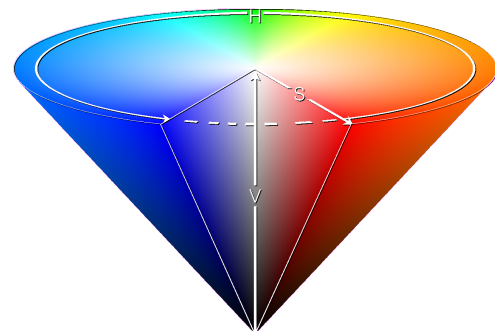
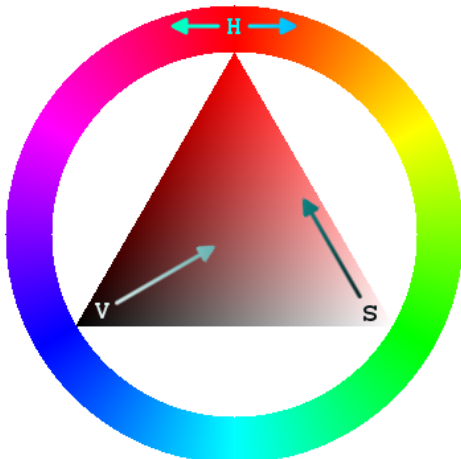
Lars Schmidt-Thieme, Information Systems and Machine Learning Lab (ISMLL), Institute BW/WI & Institute for Computer Science, University of Hildesheim  
Course on Image Analysis, winter term 2008

6/35

## Image Analysis / 1. Digital Images

## More Color Models

- **RGB (red, green, blue)** is an additive color model.
- **CMY (Cyan, Magenta, Yellow)** is the corresponding subtractive color model (used in printing; CMYK = CMY plus black).
- **HSV (Hue, Saturation, Value)**



[[http://en.wikipedia.org/wiki/HSL\\_and\\_HSV](http://en.wikipedia.org/wiki/HSL_and_HSV)]

## From RGB to HSV

Let  $(r, g, b)$  be the three components of a RGB color pixel and

$$\max := \max\{r, g, b\}$$

$$\min := \min\{r, g, b\}$$

$$h := \begin{cases} 0, & \text{if } \max = \min \\ 60^\circ \cdot \frac{g-b}{\max-\min} + 0^\circ, & \text{if } \max = r \\ 60^\circ \cdot \frac{b-r}{\max-\min} + 120^\circ, & \text{if } \max = g \\ 60^\circ \cdot \frac{r-g}{\max-\min} + 240^\circ, & \text{if } \max = b \end{cases}$$

$$s := \begin{cases} 0, & \text{if } \max = 0 \\ \frac{\max-\min}{\max}, & \text{else} \end{cases}$$

$$v := \max$$

- **Hue**  $h$  describes the dominant primary color (0-120 redish, 120-240 greenish, 240-360 blueish),
- **Saturation**  $s$  describes the relative intensity of the dominant primary color over the least dominant primary color,
- **Value**  $v$  describes the absolute intensity of the dominant primary color.

## Palette images / Indexed images

A **palette image** is a special way to store intensity values:

- the intensity value is an **index** into a lookup table,
- the **lookup table** stores the map from intensity indices to some intensity representation, e.g., RGB, HSV, gray-levels etc.

Many raster image formats such as TIFF, PNG and GIF can store images as palette images.

## From Color to Gray-levels to Binary

A color image can be converted into a gray-level image simply by averaging the intensities of the three channels:

$$X^{\text{gray}}(x, y) := \frac{X(x, y)_1 + X(x, y)_2 + X(x, y)_3}{3}$$

A gray-level image can be converted to a binary image by **thresholding**:

$$X^{\text{binary}}(x, y) := \begin{cases} 1, & \text{if } X(x, y) \geq I_{\text{threshold}} \\ 0, & \text{else} \end{cases}$$

with a given intensity threshold  $I_{\text{threshold}} \in I$ .

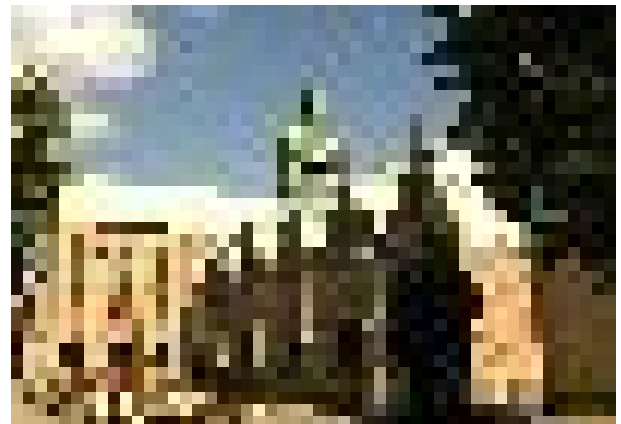
## Image Digitization

When a continuous image function is processed in a computer, it needs to be digitized, i.e., represented as raster image. This involves two aspects:

- **sampling**:  
the intensity of the continuous image is measured at **sampling points**, usually organized in a square **grid**.  
Sampling points usually are called **pixels** (or **image element** or **voxels** in 3D).  
Typical grid sizes are around  $512 \times 512$  ( $768 \times 576$  for PAL TV,  $640 \times 480$  for NTSC TV,  $1920 \times 1080$  for HDTV)
- **Quantization**:  
the intensity at each pixel is discretized in a given number of levels.  
A typical number of levels is 256 (8 bits; per channel).

A  $512 \times 512$  RGB color image with 256 levels / channel is 768 kB (uncompressed).

## Different Sampling / Spatial Resolution



## Different Quantization / Radiometric Resolution

64 intensity levels:



16 intensity levels:



4 intensity levels:



2 intensity levels:





## Image Resolution

How well the digitized image describes the continuous original image, can be described by several types of resolutions:

- **spatial resolution**: the distance between two pixels.
- **spectral resolution**: bandwidth (frequency spectrum) captured by the light sensor.
- **radiometric resolution**: number of different gray levels.
- **time resolution**: interval between two samples in time (for videos).

### 1. Digital Images

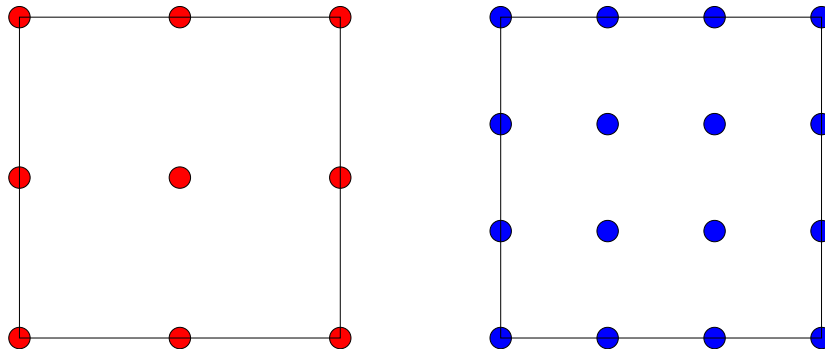
### 2. Image Interpolation

### 3. Image Classification

## Interpolation

Let  $f$  be a raster image of size  $n \times m$ .

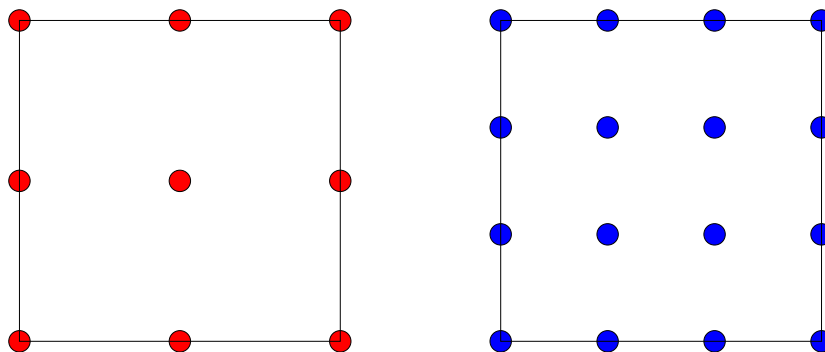
We look for a scaled representation  $f'$  of size  $n' \times m'$ .



The corresponding co-ordinates of pixel  $(u', v')$  in image  $f$  are

$$T^{-1}(x', y') := \begin{pmatrix} \frac{x'(n-1)}{(n'-1)} \\ \frac{y'(m-1)}{(m'-1)} \end{pmatrix}$$

## Interpolation



Example:  $n = m = 3$ ,  $n' = m' = 4$ .

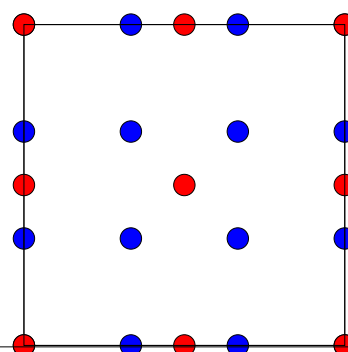
The  $f$ -co-ordinates of  $(2, 1)$  are

$$T^{-1}(2, 1) = (1.33, 0.66)$$

As this is not a gridpoint of  $f$ ,  
there is no measured intensity!

$\rightsquigarrow$  the intensity has to be

**interpolated.**



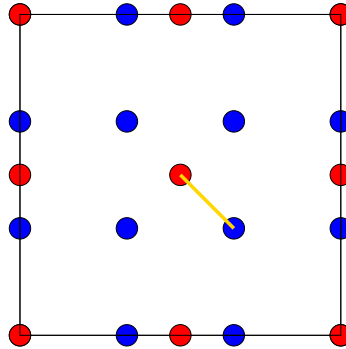
## Nearest-Neighbor Interpolation

The intensity could be estimated by the intensity of its **nearest neighbor**:

$$f'(x', y') := f(\text{round}(T^{-1}(x', y')))$$

Example:

$$f'(2, 1) := f(\text{round}(T^{-1}(x, y))) = f(\text{round}(1.33, 0.66)) = f(1, 1)$$



## Bi-linear Interpolation

A better estimation uses all four neighbors and the distances:

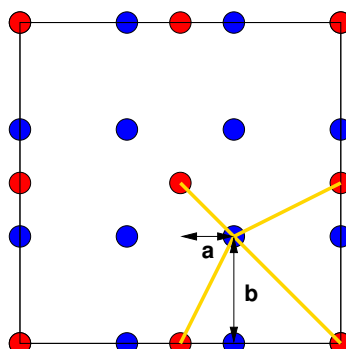
$$f'(x', y') := (1 - a)(1 - b)f(x, y) + a(1 - b)f(x + 1, y) \\ + (1 - a)b f(x, y + 1) + ab f(x + 1, y + 1)$$

with

$$(x, y) := \text{floor}(T^{-1}(x', y')) \\ a := T^{-1}(x', y')_1 - x \\ b := T^{-1}(x', y')_2 - y$$

Example:

$$f'(2, 1) := 0.66 \cdot 0.33 \cdot f(1, 0) + 0.33 \cdot 0.33 \cdot f(2, 0) + 0.66 \cdot 0.66 \cdot f(1, 1) + 0.33 \cdot 0.66 \cdot f(2, 1)$$



## More Interpolation Methods

There are more complex interpolation methods:

- **bi-cubic interpolation**: uses 16 neighboring points and a bi-cubic polynomial surface.
- **area interpolation**: uses all the points covered by a target pixel.

Nearest-neighbor interpolation may introduce step-like appearances, esp. for straight lines.

Linear interpolation can cause blur due to the averaging.

## Interpolation / Moire Effect

The next two slides show an original image (bricks.jpg,  $622 \times 756$ ) and downsamples of size  $205 \times 250$  by four different methods:

- nearest neighbor interpolation (upper left)
- bi-linear interpolation (upper right)
- bi-cubic interpolation (lower left)
- area interpolation (lower right)



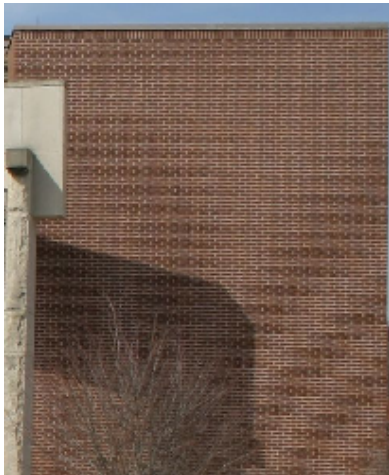
[<http://en.wikipedia.org/wiki/Moire>]

Lars Schmidt-Thieme, Information Systems and Machine Learning Lab (ISMLL), Institute BW/WI & Institute for Computer Science, University of Hildesheim  
Course on Image Analysis, winter term 2008

20/35

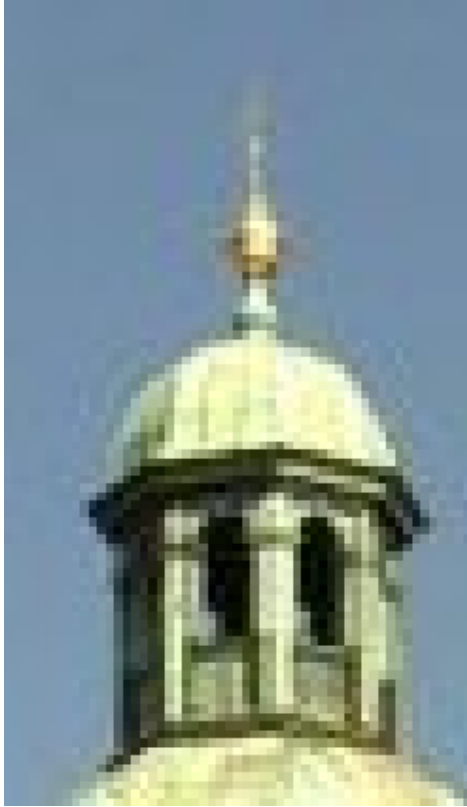
## Image Analysis / 2. Image Interpolation

### Interpolation / Example



Interpolation / Example 2 (1/2)

nearest neighbor:



bi-linear:



Interpolation / Example 2 (2/2)

bi-linear:



bi-cubic:



# 1. Digital Images

# 2. Image Interpolation

# 3. Image Classification

## A First Look at Image Classification

Given

- **images** and
- some (global) **annotation**,  
e.g., if the image shows a person or not,

try to learn the annotated concept,  
so that the annotation can be done  
automatically in future.

Useful for

- image retrieval  
(search by keyword/tag).
- many applications  
(e.g., sort tomato plants).

image	person?
	no
	yes
	no
	yes
	?

## How to Learn? / Training Data

To predict if an entity belongs to a specific class, a machine needs the following components:

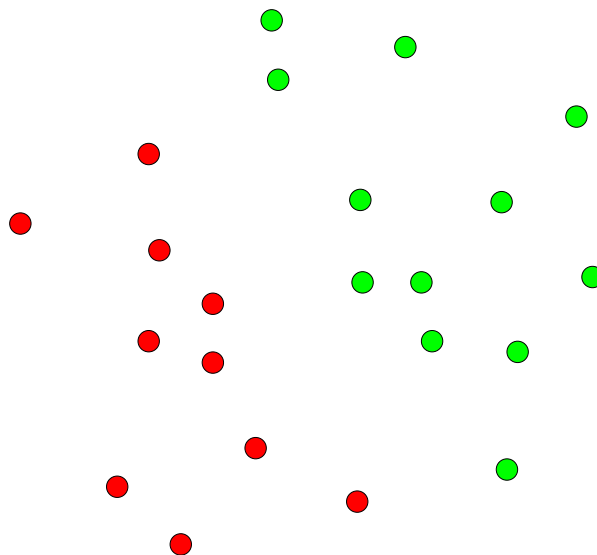
1. **training data**  $(x^j, y^j)_{j=1, \dots, N}$  consisting of  $N$  pairs  $(x, y) \in X \times Y$  where

- $x = (x_i)_{i=1, \dots, n}$  are the **predictor variables** that describe an entity and
- $y$  is the observed class label, also called **target variable**, here for simplicity:

$y = 1$ : entity belongs to the class,

$y = 0$ : entity does not belong to the class.

## How to Learn? / Training Data



Entities are described by two predictor variables (horizontal & vertical axis).

Classes are depicted by colors: green = 1, red = 0.



## How to Learn? / Models

To predict if an entity belongs to a specific class, a machine needs the following components:

2. a **model**

$$\hat{y} : X \rightarrow Y$$

that describes how the target variable  $Y$  depends on the predictor variables  $X$ .

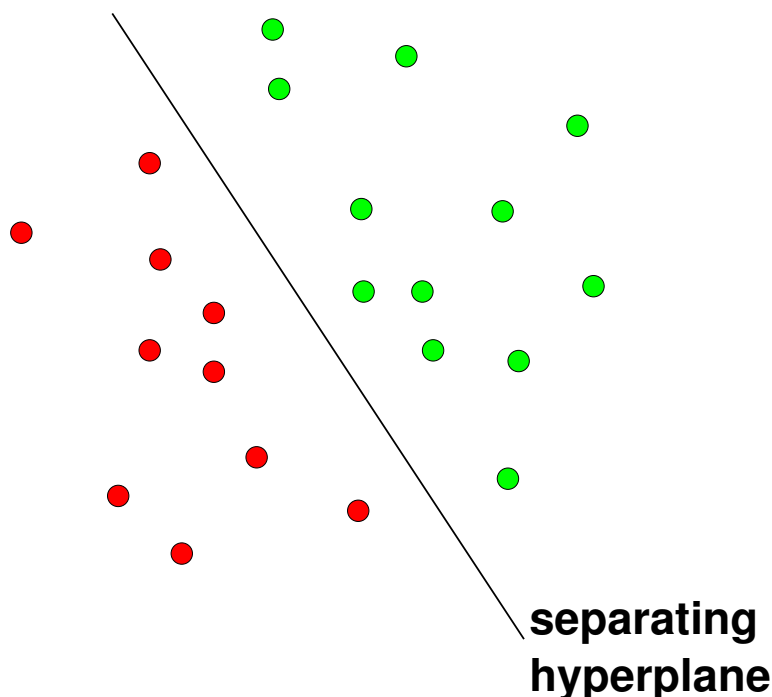
Example: the **linear support vector machine** (a specific model) decides if an entity  $x$  belongs to a class or not via

$$\hat{y}(x) = 1 : \iff \sum_{i=1}^n a_i x_i \geq 0$$

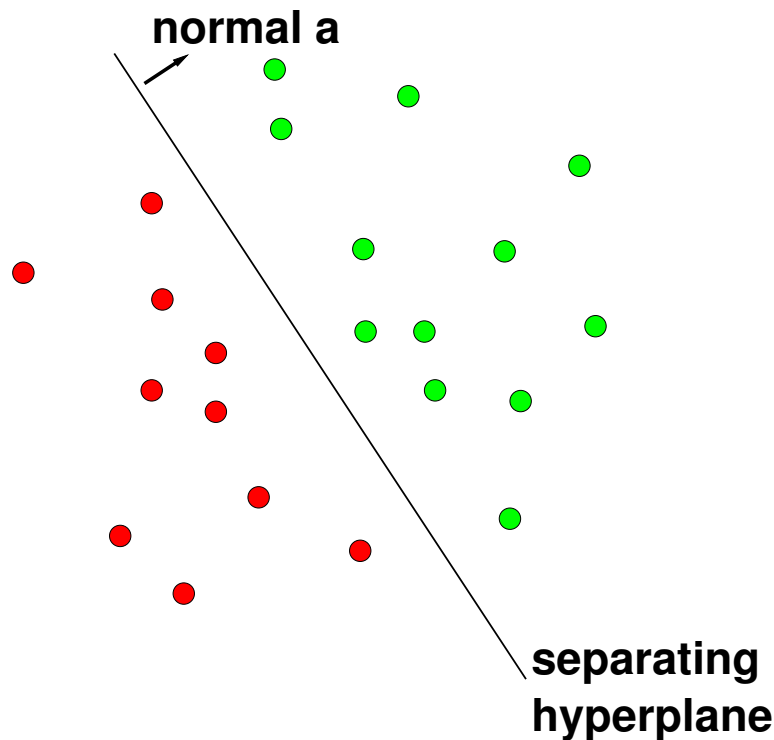
where

- $a = (a_i)_{i=1, \dots, n}$  are **parameters** of the model that need to be learned and
- $\hat{y}$  denotes the **class label predicted by the model**.

## How to Learn? / Models / SVM



## How to Learn? / Models / SVM



The points on the hyperplane are described by  $\langle a, x \rangle = 0$ .

For points on the side pointed to by the normal,  $\langle a, x \rangle > 0$ .

## How to Learn? / Error Measures &amp; Learning Algorithms

To predict if an entity belongs to a specific class, a machine needs the following components:

3. a **learning algorithm** that estimates the parameters  $a$  from the training data, i.e., chooses the parameters such that the **error** of the classifier is small.

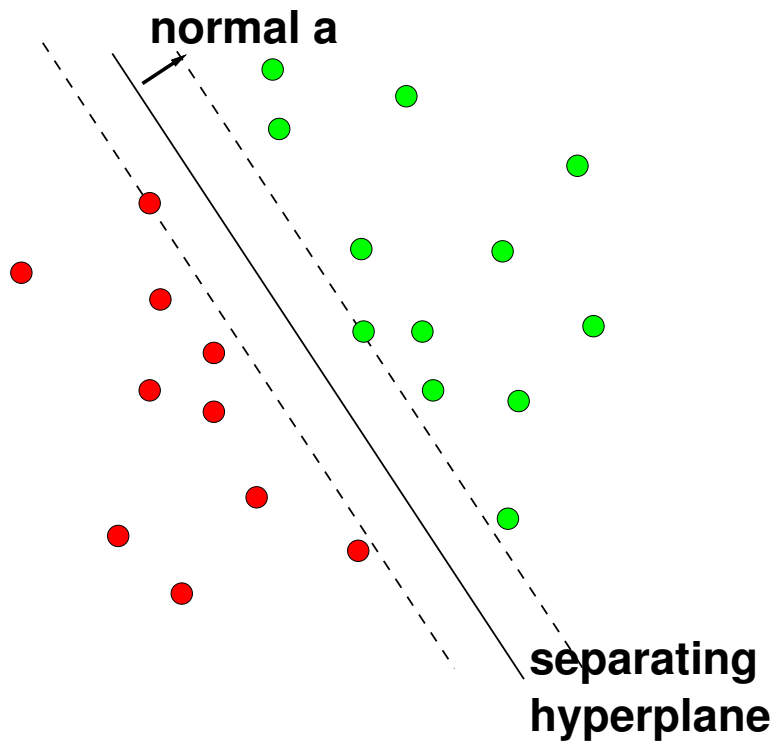
Example: for a binary classification problem, **accuracy** is a simple error measure:

$$\text{accuracy}(\hat{y}, (x^j, y^j)_{j=1, \dots, N}) := \frac{1}{N} \sum_{j=1}^N I(\hat{y}(x^j) = y^j)$$

where

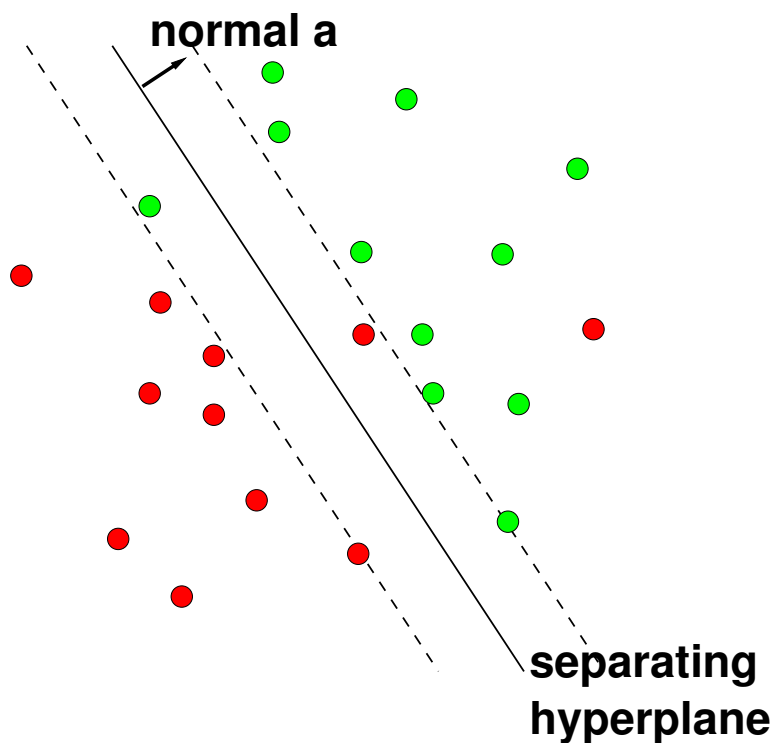
$$I(x) := \begin{cases} 1, & \text{if } x \text{ is true} \\ 0, & \text{else} \end{cases}$$

## How to Learn? / Models / SVM



Among all possible hyperplanes, the SVM searches for the one with the **maximal margin**.

## How to Learn? / Models / SVM



## How to Describe an Image? (1/3)

To apply machine learning techniques, entities (usually) have to be described by predictor variables having numerical values.

As entities are compared via their predictor variables, all entities need to have the same predictor variables.

What can we do in the case of images?



$480 \times 320$  pixels a 3 channels.



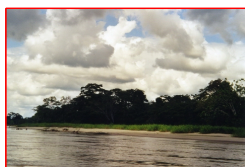
$320 \times 480$  pixels a 3 channels.

## How to Describe an Image? (2/3)

Idea:

1. Resize image to resolution  $k \times k$ .
2. Compare pixelwise.

original image:



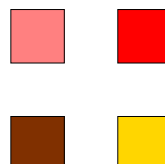
resize to resolution  $1 \times 1$ :








original image:



resize to resolution  $2 \times 2$ :



## How to Describe an Image? (3/3)

image	features			person?
	237	245	245	no
	238	222	209	yes
	64	65	61	no
	162	157	159	yes
	114	138	144	?

Lars Schmidt-Thieme, Information Systems and Machine Learning Lab (ISMLL), Institute BW/WI & Institute for Computer Science, University of Hildesheim  
Course on Image Analysis, winter term 2008

31/35

## A Trivial Classifier as Baseline

Assume you do not have any information about the image to classify, but still you have to make a decision.

You only have information about the class frequencies, e.g., you know that 67% of all images in the training data belong to the class persons.

Which class, person or not person, would you predict?

## A Trivial Classifier as Baseline

Assume you do not have any information about the image to classify, but still you have to make a decision.

You only have information about the class frequencies, e.g., you know that 67% of all images in the training data belong to the class persons.

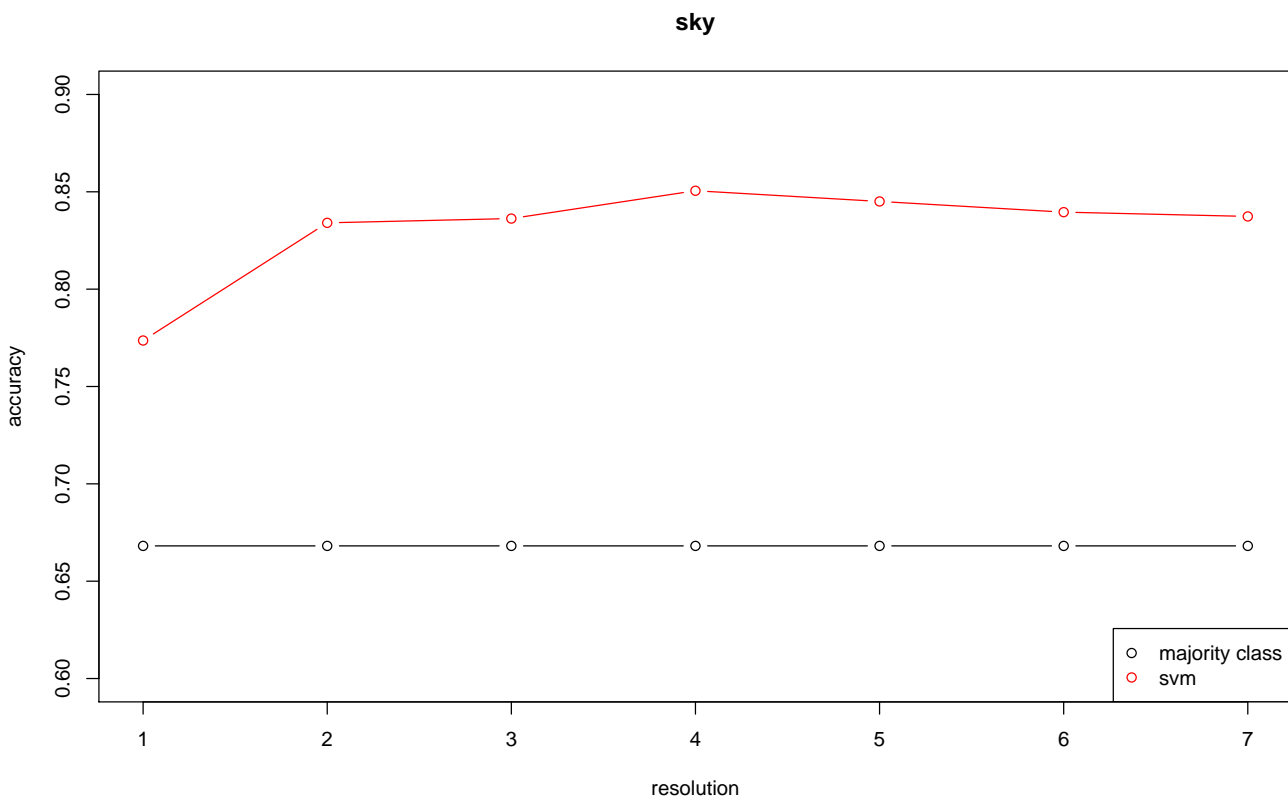
Which class, person or not person, would you predict?

Person, as it is the **majority class**:

$$\hat{y}^{\text{majority}} := \operatorname{argmax}_{y \in Y} \hat{p}(y), \quad \hat{p}(y) := |\{j \in \{1, \dots, N\} \mid y^j = y\}| / N$$

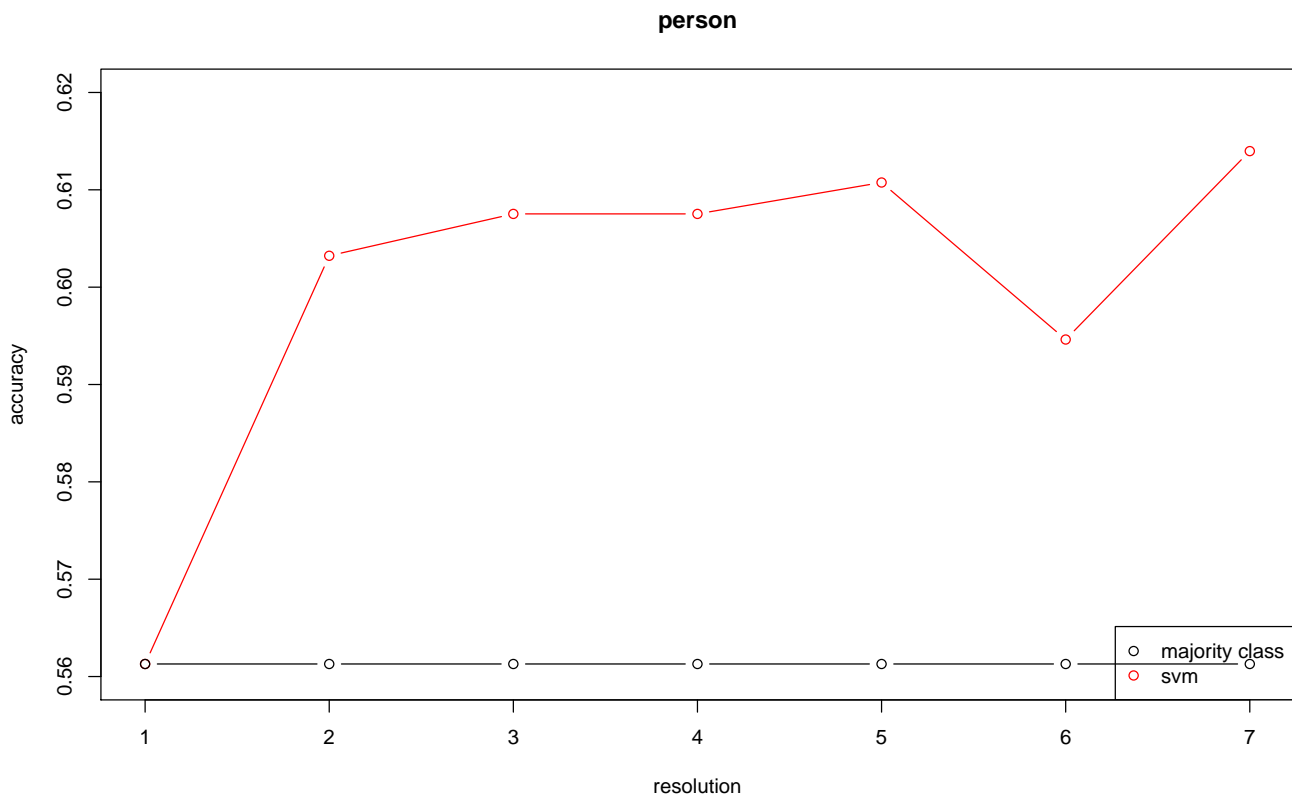
with expected accuracy  $\hat{p}(\hat{y}^{\text{majority}}) = 0.67$ .

## Image Classification / Example



[Data from ImageClef 2008, vcdt, random 50:50 split, hyperparameter  $C$  optimized on train]

## Image Classification / Example



[Data from ImageClef 2008, vcdt, random 50:50 split, hyperparameter  $C$  optimized on train]

Lars Schmidt-Thieme, Information Systems and Machine Learning Lab (ISMLL), Institute BW/WI & Institute for Computer Science, University of Hildesheim  
Course on Image Analysis, winter term 2008

34/35

## Summary (1/2)

- Digital / raster / discrete images can be understood as discretizations of a continuous image function
  - with a given spatial resolution (width and height in pixels) and
  - with a given intensity resolution (binary, gray-level, color; number of intensity values).
- Color images are represented by several intensity values of so-called primary colors per pixel (channels), e.g., red–green–blue.
- If images are resized, the intensity values of the new grid have to be interpolated from intensity values of pixels nearby in the old grid (nearest-neighbor; bi-linear; bi-cubic; area).

## Summary (2/2)

- In image classification a model such as a Support Vector Machine (SVM) is build from training data. The model can be applied to test data to predict unknown class labels.
- Renderings in small grids ( $1 \times 1$  to  $10 \times 10$ ) can be used as primitive features for image classification.