Feature engineering

- **Outline:**

  1. Introduction
  2. Feature engineering
  3. Features in visual pattern recognition
  4. Shape-based feature descriptors

Feature engineering

- **Outline:**

  1. Introduction

  2. Feature engineering

  3. Features in visual pattern recognition

  4. Shape-based feature descriptors

Introduction to feature & feature engineering

- Feature:
  - an individual measurable property or characteristic of a data example
  - describes the example
  - features are usually numeric
- Feature engineering: transfer raw data into feature vector

Data → feature vector → ML model

Curse of dimensionality

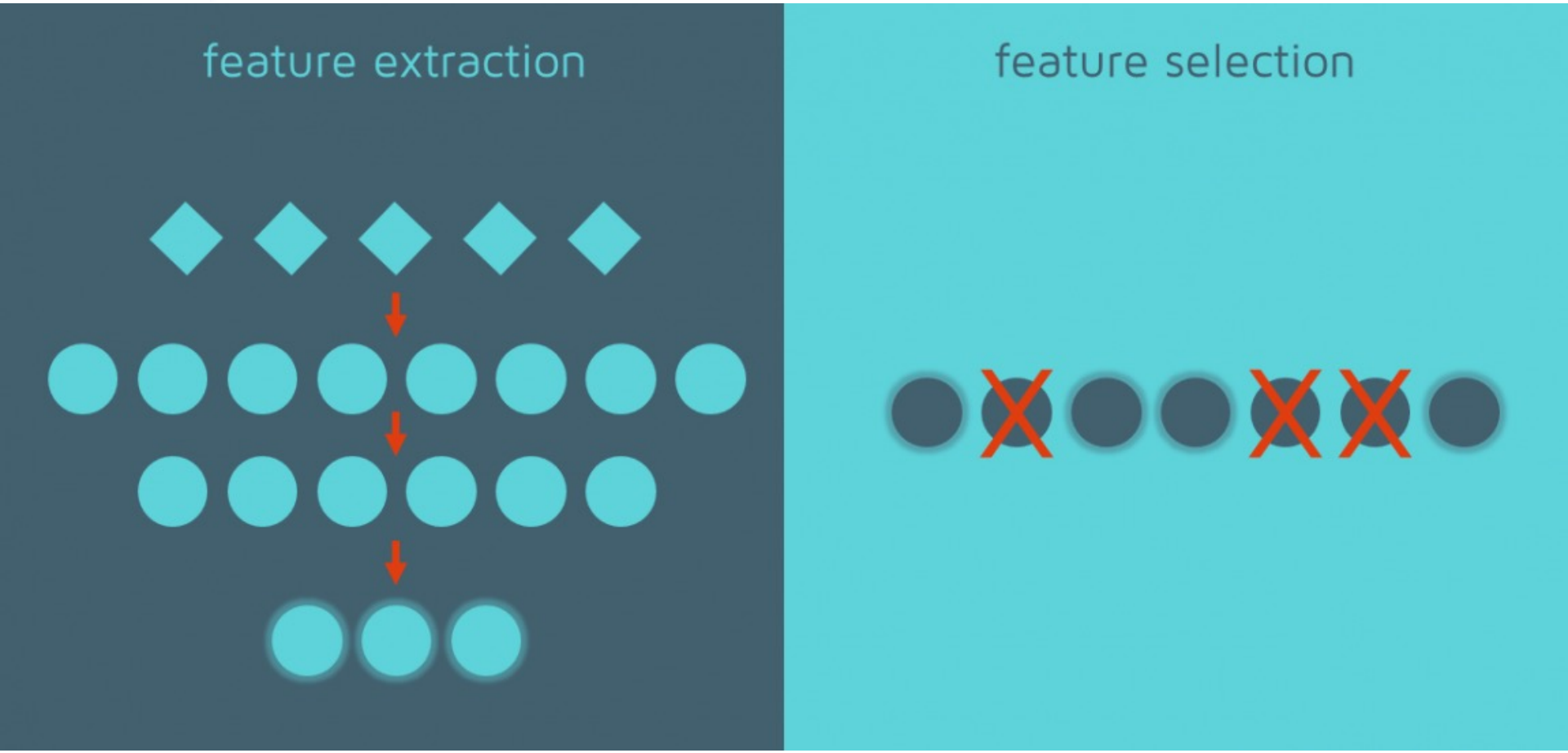
- Dimensionality: the number of features in feature vector. Ex: house (price, size, # bedrooms, location,…)
- Curse of dimensionality:
  - Performance of algorithms degradates as the dimensionality increases exponentially
  - Complexity of data increases without increasing the amount of useful information
  - Data becomes sparse

→ Solution???

Feature extraction and feature selection

- Feature selection:
  - Filtering irrelevant or redundant features from dataset
  - Choosing a subset of the original features
- Feature extraction:
  - Creating a new smaller set of features
  - Getting useful features from existing data
- Feature need to be informative, discriminating and independent

Feature extraction vs. feature selection



Feature engineering

- **Outline:**

  1. Introduction
  2. **Feature engineering**
  3. Features in visual pattern recognition
  4. Shape-based feature descriptors

Feature engineering

- One-hot encoding
- Binning
- Normalization
- Standardization
- Dealing with missing feature
- Data imputation techniques

One-hot encoding

- Transform a categorical feature into several binary features
- Example: feature “color” has 3 values “red”, “yellow”, green”
  - “red” = 1, “yellow” = 2, “green” = 3
  - ?

$$red = [1, 0, 0]$$
$$yellow = [0, 1, 0]$$
$$green = [0, 0, 1]$$

Binning (bucketing)

- Transform a numerical feature into categorical feature
- Example: feature “age”
  - Put all ages between 0 and 5 years-old into one bin
  - Put ages from 6 to 10 years-old in the second bin
  - Put ages from 11 to 15 years-old in the third bin, and so on.

Normalization

- Converting an actual range of values of a numerical feature into a standard range of values, typically in the interval [-1, 1] or [0, 1].
- Example: natural range = [350, 1450]
  - Subtracting 350 from every value of the feature
  - Dividing the result by 1100 → normalized range = [0, 1].

Standardization

- Rescaling the feature values so that they have the properties of a standard normal distribution with $\mu = 0$ and $\sigma = 1$
- Formula:

$$\hat{x}^{(j)} = \frac{x^{(j)} - \mu^{(j)}}{\sigma^{(j)}}.$$

Standardization or normalization?

- Try two if have time ☺
- Rule of thumbs:

- unsupervised learning algorithms, in practice, more often benefit from standardization than from normalization;
- standardization is also preferred for a feature if the values this feature takes are distributed close to a normal distribution (so-called bell curve);
- again, standardization is preferred for a feature if it can sometimes have extremely high or low values (outliers); this is because normalization will "squeeze" the normal values into a very small range;
- in all other cases, normalization is preferable.

Dealing with missing features

- Removing the examples with missing features.
- Use data imputation technique

Data imputation techniques

- Technique 1: Replacing the missing value of a feature by an average value of this feature in the dataset
- Technique 2: Replacing the missing value by the same value outside the normal range of values.
- Technique 3: Replacing the missing value by a value in the middle of the range.

…etc…

Feature engineering

- **Outline:**

  1. Introduction
  2. Feature engineering
  3. **Features in visual pattern recognition**
  4. Shape-based feature descriptors

Image feature extraction

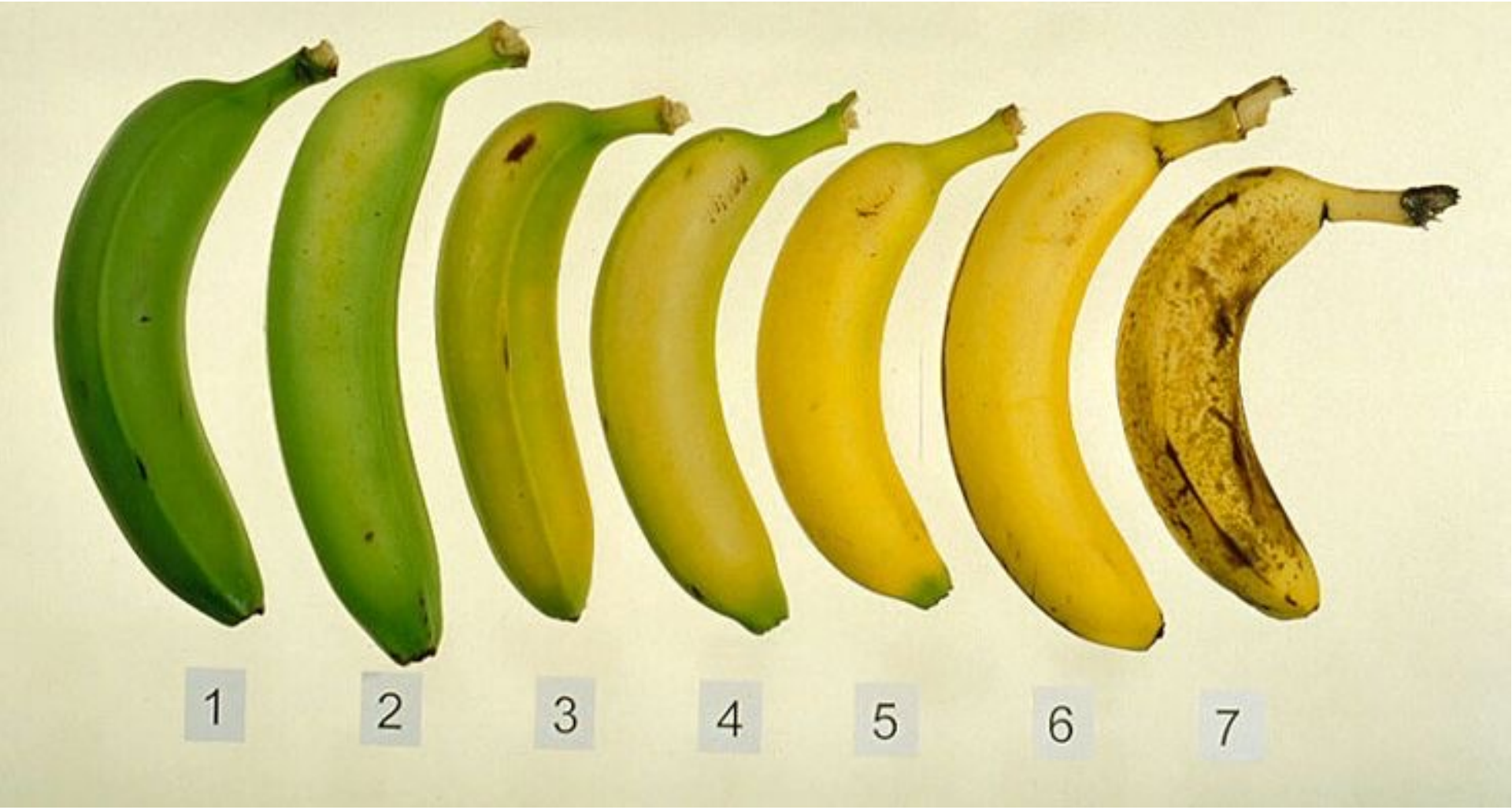
- Purpose:
  - To reduce the dimensionality of input image
  - To transform each input image into a corresponding multi-dimension feature vector
  - To perform the predefined classification tasks with sufficient accuracy without using the entire input image
- Requirements:
  - Features should extract the most suitable characteristics from the input image

An example of feature extraction

	S_1	S_2	S_3	S_4	S_5	S_6	S_7
Moment	0.3461	0.0738	0.0093	0.0042	2.61E-05	0.0011	4.85E-06
logHu	-1.0610	-2.6069	-4.6771	-5.4646	-10.5523	-6.8463	-12.2370

Visual features

- Color-based features

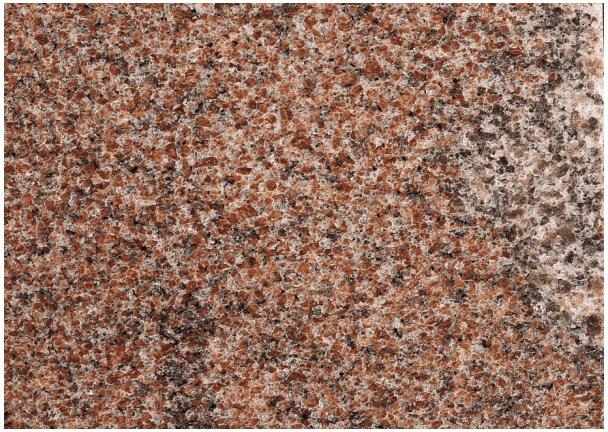


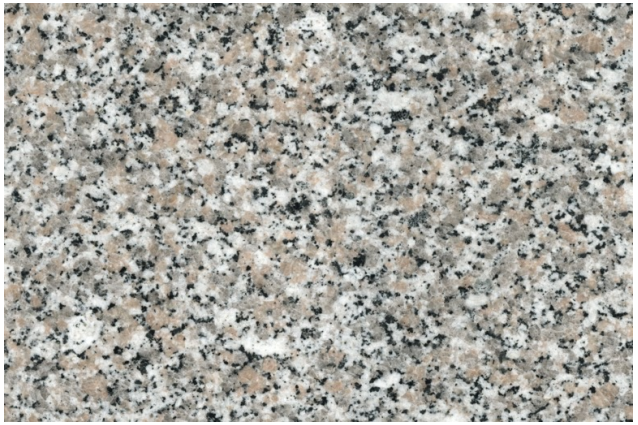
Visual features

- Shape-based features

Visual features

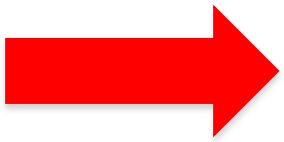
- Texture-based features



Which feature is the best?

- **Example: plant recognition**
  - Plant features: leaf, fruit, flower, root, branch,…
  - Leaf features: shape, vein, margin, texture
  - No single best feature for a given leaf identity → combination of different features
  - No single best presentation for a given feature → multiple descriptors to characterize the feature from different perspectives



- Challenging
- Innovative

Deep learning

Feature engineering

- **Outline:**

  1. Introduction
  2. Feature engineering
  3. Features in visual pattern recognition
  4. **Shape-based feature descriptors**

Shape-based feature descriptor

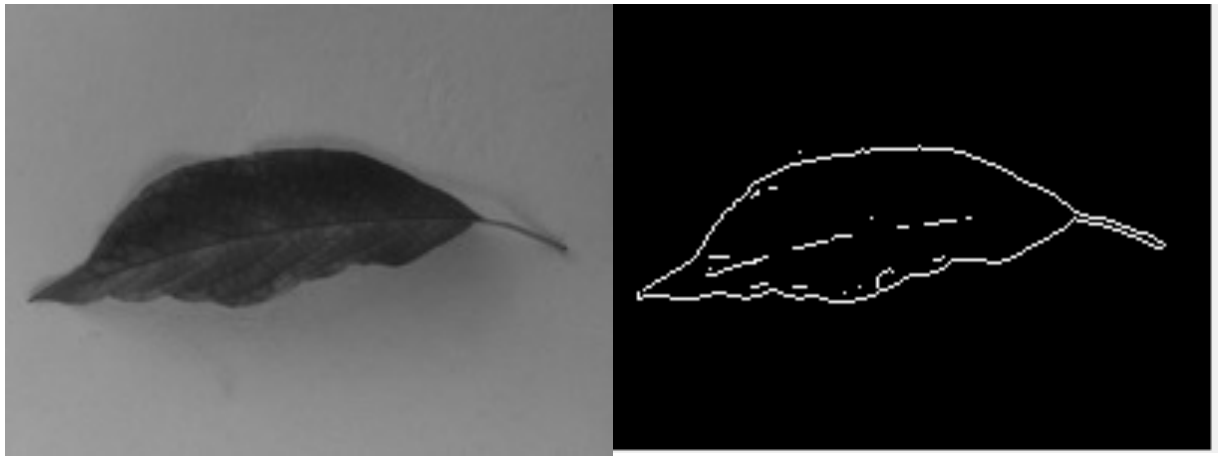
- Shape: important
- Good shape descriptor: invariant to geometrical transformations (rotation, reflection, scaling, translation)
- Types of shape descriptors: simple and morphological shape descriptor (SMSD), contour-based, region-based

Simple and morphological shape descriptor

- Refer to basic geometric properties of the shape
- Basic descriptor: diameter, major axis length, minor axis length, area, perimeter, centroid,…
- Morphological descriptor: aspect ratio, perimeter to area ratio, rectangularity measures, circularity measures,…

Contour-based feature descriptor

- Consider the boundary of a shape and neglect the information contained in the shape interior
- Ex: CCD (centroid contour distance), Fourier descriptor computed on CCD.



Region-based feature descriptor

- Take all the pixels within a shape region into account to obtain the shape representation
- Image moments: statistical descriptor of a shape. Ex: Hu moments
- Local features: select key points in image. Ex: HOG (histogram of oriented gradients), SIFT (scale-invariant feature transform)

Image moments

- Weighted average of image pixel intensities.
- The pixel intensity at location (x,y) is given by s(x,y)
- Binary image: s(x,y) = 0/1
- Simplest moment: $M = \sum_x \sum_y s(x, y)$

➢ The number of white pixels

➢ The area of white region (object) in the image

Hu moments feature descriptor

Central moments:

$$m_{pq} = \sum_x \sum_y (x - \bar{x})^p (y - \bar{y})^q s(x,y) \quad p,q = 0,1,2,3$$

Central normalized moments:

$$M_{pq} = \frac{m_{pq}}{m_{00}^{\frac{p+q}{2}+1}}$$

Centroid of the image:

$$\bar{x} = \frac{\sum_x \sum_y x.s(x,y)}{\sum_x \sum_y s(x,y)}, \bar{y} = \frac{\sum_x \sum_y y.s(x,y)}{\sum_x \sum_y s(x,y)}$$

Hu moments feature (cont)

$$S_1 = M_{20} + M_{02}$$

$$S_2 = (M_{20} - M_{02})(M_{20} + M_{02}) + 4M_{11}M_{11}$$

$$S_3 = (M_{30} - 3M_{12})^2 + (M_{30} - 3M_{21})^2$$

$$S_4 = (M_{30} + M_{12})^2 + (M_{03} + M_{21})^2$$

$$S_5 = (M_{30} - 3M_{12})(M_{30} + M_{12})[(M_{30} + M_{12})^2 - 3(M_{03} + M_{21})^2] + (3M_{21} - M_{03})(M_{03} + M_{21})[3(M_{30} + M_{12})^2 - (M_{03} + M_{21})^2]$$

$$S_6 = (M_{20} - M_{02})[(M_{30} + M_{12})^2 - (M_{03} + M_{21})^2] + 4M_{11}(M_{30} + M_{12})(M_{03} + M_{21})$$

$$S_7 = (3M_{21} - M_{03})(M_{30} + M_{12})[(M_{30} + M_{12})^2 - 3(M_{03} + M_{21})^2] + (M_{30} - 3M_{12})(M_{21} + M_{02})[3(M_{30} + M_{12})^2 - (M_{03} + M_{21})^2]$$

Id	image	S1	S2	S3	S4	S5	S6	S7
1		2.78871	6.50638	9.44249	9.84018	-19.593	-13.1205	19.6797
2		2.67431	5.77446	9.90311	11.0016	-21.4722	-14.1102	22.0012
3		2.67431	5.77446	9.90311	11.0016	-21.4722	-14.1102	22.0012
4		2.65884	5.7358	9.66822	10.7427	-20.9914	-13.8694	21.3202
5		2.66083	5.745	9.80616	10.8859	-21.2468	-13.9653	21.8214
6		2.66083	5.745	9.80616	10.8859	-21.2468	-13.9653	-21.8214

Bài tập tính đặc trưng Hu's moment

- Cho ảnh nhị phân sau:

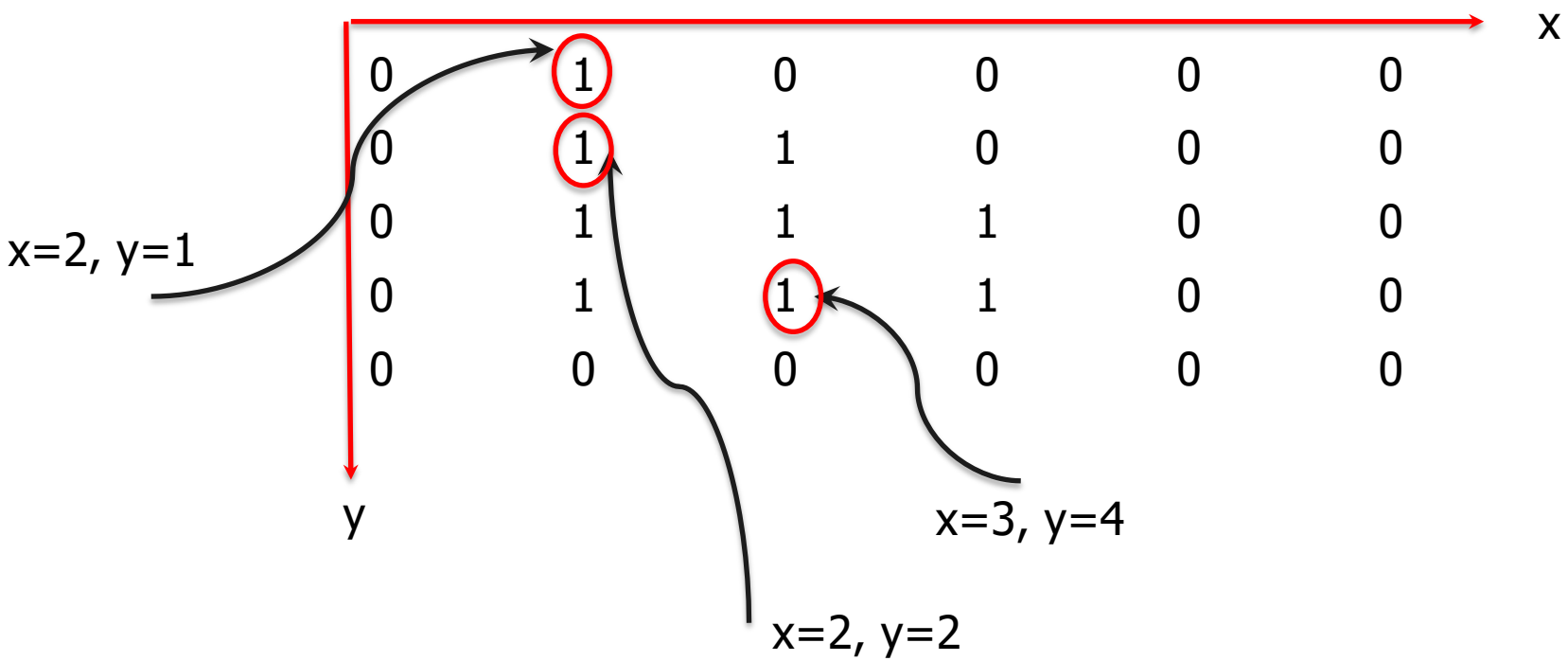
0	1	0	0	0	0
0	1	1	0	0	0
0	1	1	1	0	0
0	1	1	1	0	0
0	0	0	0	0	0

- Tính đặc trưng S_1 (phải tính $m_{00}, \bar{x}, \bar{y}, m_{02}, m_{20}, M_{02}, M_{20}$, suy ra S_1)

bằng cách tính thủ công. Sau đó kiểm tra lại S_1 bằng code.

Sửa bài tập

- Giả sử chọn chiều trục x, y như sau (lưu ý toạ độ gốc là x=1, y=1)



Bài tập về nhà

- **Sửa bài kiểm tra giữa kỳ ngày 9/10/2024:**

- Nhóm 3: lý thuyết
- Nhóm 12B: code

- **Tìm hiểu về bộ mô tả đặc trưng hình dáng:**

- Nhóm 1 → 8: tìm hiểu về đặc trưng HOG
- Nhóm 9 → 16: tìm hiểu về đặc trưng Hu'smoments

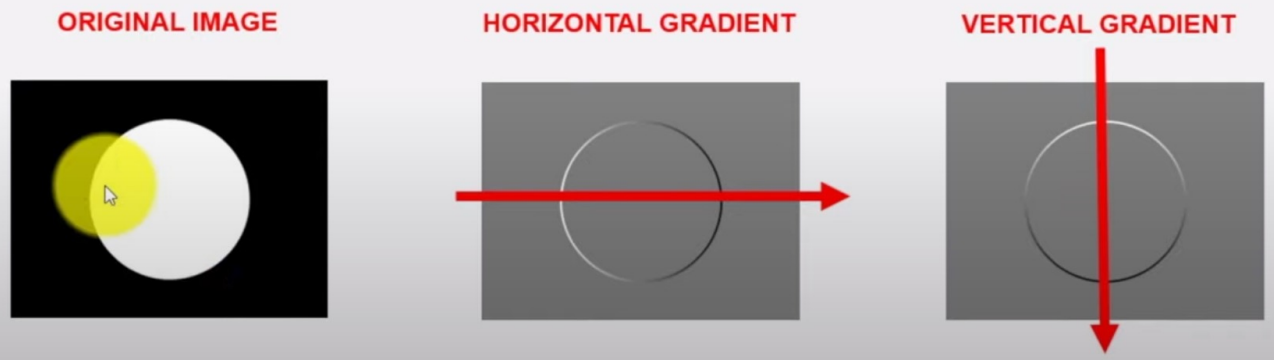
- **Thuyết trình ngày 9/10/2024:**

- Nhóm 4, 5: lý thuyết và code tính đặc trưng HOG dùng CSDL tự xây dựng
- Nhóm 16, 14: lý thuyết và code tính đặc trưng Hu's moments dùng CSDL tự xây dựng (3 loài lá, mỗi loài 50 mẫu)

HOG

https://www.youtube.com/watch?v=XmO0CSsKg88&t=41s

- For function $f(x, y)$, the gradient is the vector (f_x, f_y).
- An image is a discrete function of (x, y) so image gradient can be calculated as well.
- At each pixel, image gradient horizontal (x-direction) and vertical (y-direction) are calculated.
- These vectors have a direction $\text{atan}(\frac{f_y}{f_x})$ and a magnitude $(\sqrt{(f_x^2 + f_y^2)}$
- Gradient values are mapped to 0 - 255. Pixels with large negative change will be black, pixels with large positive change will be white, and pixels with little or no change will be gray.

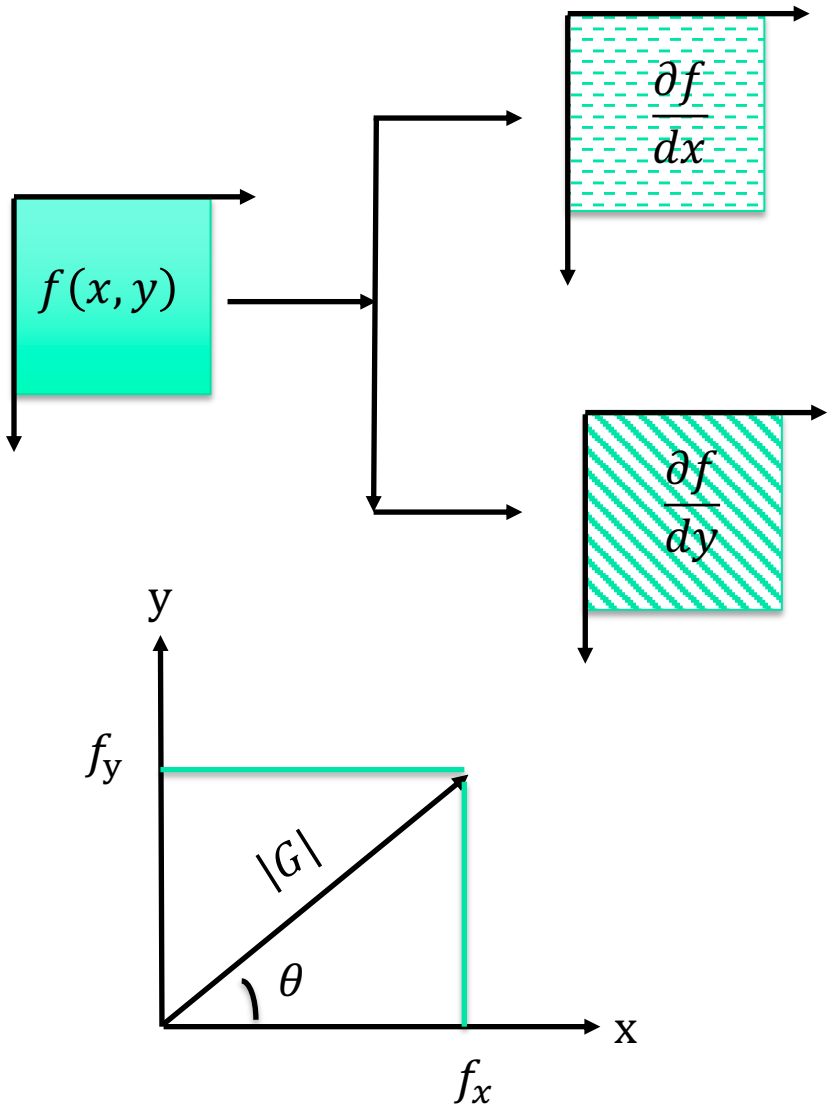


Histogram of Oriented Gradients (HOG)

- What is gradient?
- How to calculate gradients?
- How to calculate histogram using gradients and orientation?
- How to calculate HOG?

What is gradient?

- What is derivative?
  - Function $f(x) \rightarrow f'(x)$
  - Image $f(x,y) \rightarrow (\frac{\partial f}{dx}, \frac{\partial f}{dy})$
- Gradient vector:
  - $\mathrm{G}(\frac{\partial f}{dx}, \frac{\partial f}{dy}) = \mathrm{G}(f_x, f_y) = |\mathrm{G}|e^{j\theta}$
  - $|G| = \sqrt{\mathrm{f_x^2} + \mathrm{f_y^2}}$ and $\theta = \mathrm{arctg}\frac{\mathrm{f_y}}{\mathrm{f_x}}$



Calculating the gradient

- How to calculate derivative?
  - Derivative $f'(x) = \frac{f(x+\Delta x)-f(x)}{\Delta x}$, $\Delta x \to 0$
  - $f'(x) = f(x+1) - f(x)$
  - $f'(x) = f(x+1) - f(x-1)$
- How to calculate gradients of image?
  - Use x-mask and y-mask
  - Use Sobel mask
  - …

x-mask

-1	0	+1

y-mask

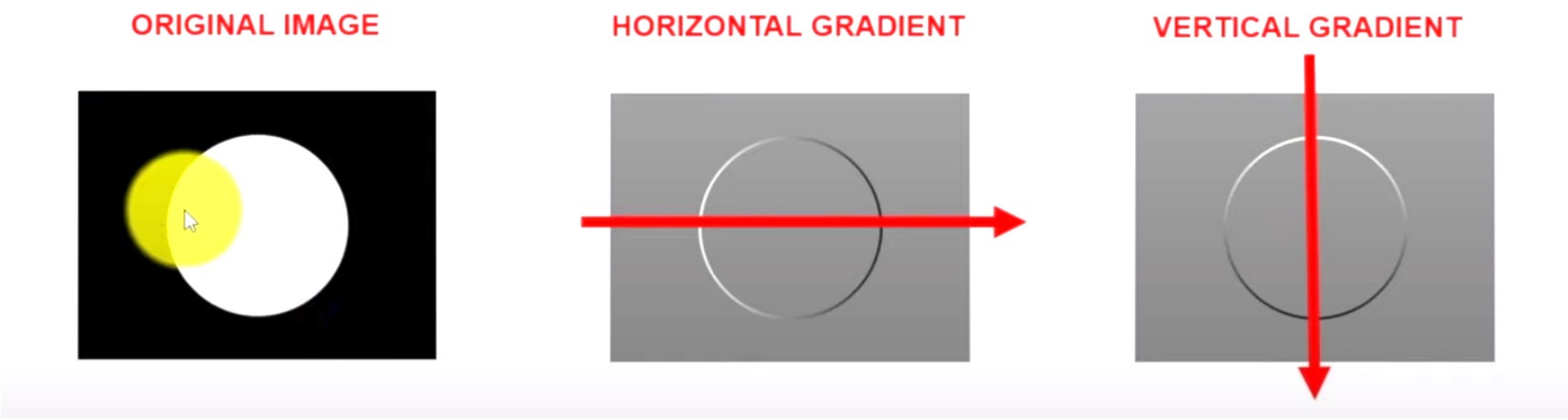
-1
0
+1

-1	0	+1
-2	0	+2
-1	0	+1

+1	+2	+1
0	0	0
-1	-2	-1

Sobel x-mask

Examples of gradients



Introduction to HOG

- HOG focuses on the structure or the shape of an object.
- HOG provides magnitude and direction of the edges
- Orientations are calculated in ‘localized’ regions.
- HOG generate a Histogram for each regions separately by counting the occurrences of gradient orientation in localized portions of an image.

Calculating the HOG

Calculating the HOG

- Step 1: Preprocess the data (ex., 64 x 128)
- Step 2: Calculating gradients (direction x and y)
- Step 3: Calculate the magnitude and orientation
- Step 4: Create histogram of gradients in 8×8 cells (9×1)
- Step 5: Normalize gradients in 16×16 cell (36×1)
- Step 6: Get the features for the complete image

Step 1: preprocess the data

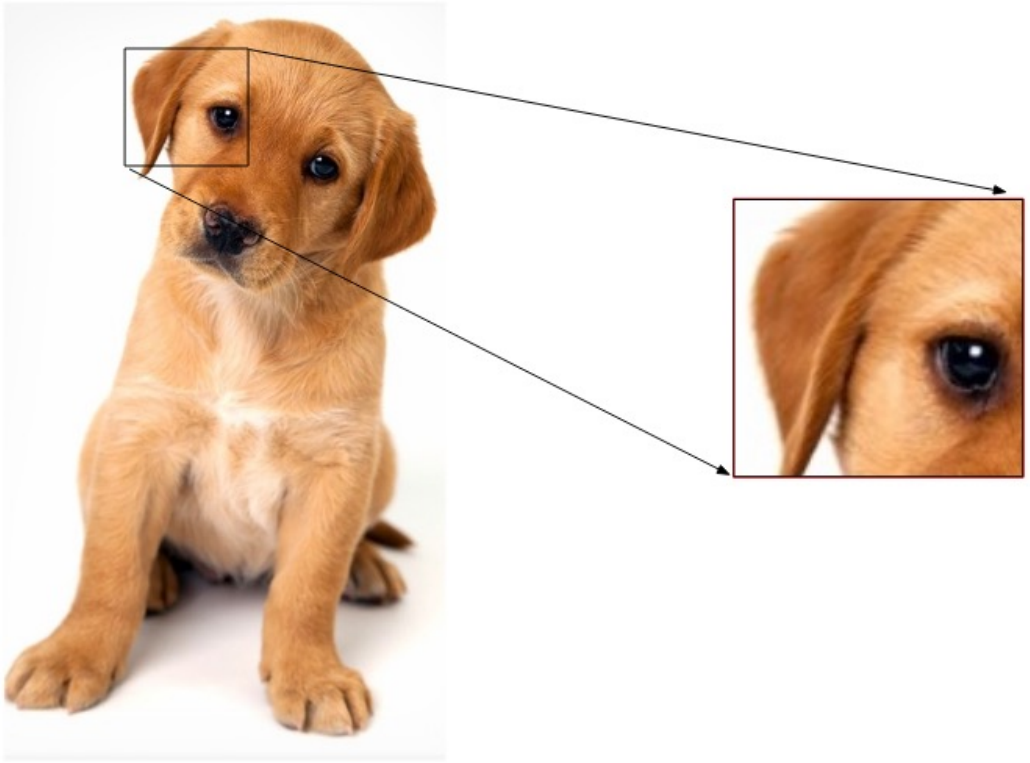
Original Image

Resized Image

64 x 128

Step 2: calculate the gradients

- Take small cell
- Calculate the gradients on cell



Step 2: calculate the gradients

121	10	78	96	125
48	152	68	125	111
145	78	85	89	65
154	214	76	200	66
214	87	45	102	45

- Change in x direction: $f_x = 89 - 78 = 11$
- Change in y direction: $f_y = 76 - 68 = 8$
- Repeat the same process for all the pixels

Step 3: calculate the magnitude and the orientation

121	10	78	96	125
48	152	68	125	111
145	78	85	89	65
154	214	76	200	66
214	87	45	102	45

- Gradient magnitude: $|G| = \sqrt{f_x^2 + f_y^2} = 13.6$
- Orientation: $\theta = \text{arctg}\frac{f_y}{f_x} = 36$
- Repeat the same process for all the pixels

Step 4: create the histogram of gradients

- **Method 1:**

- Take each pixel value
- Find the orientation
- Update the frequency table

We end up with a frequency table that denotes angles and the occurrence of these angles in the image.

121	10	78	96	125
48	152	68	125	111
145	78	85	89	65
154	214	76	200	66
214	87	45	102	45

Frequency						1									
Angle	1	2	3	4 ...	35	36	37	38	39....	175	176	177	178	179	180

Step 4: create the histogram of gradients

- **Method 2:** similar to the previous method, except that here we have 9 bins.
  - Take each pixel value
  - Find the orientation
  - Store the frequency of orientation in the matrix 9x1

Frequency		1							
Bin	0	20	40	60	80	100	120	140	160

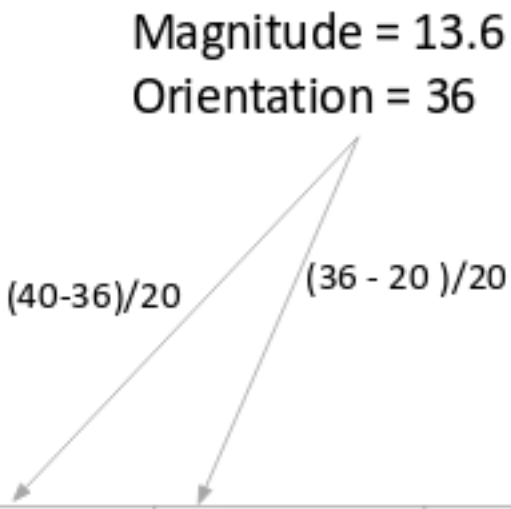
Step 4: create the histogram of gradients

- **Method 3:** use both gradient magnitude and orientation
  - Take each pixel value
  - Find the magnitude and orientation
  - use the gradient magnitude to fill the values in the matrix

Magnitude = 13.6
Orientation = 36

Magnitude		13.6							
Bin	0	20	40	60	80	100	120	140	160

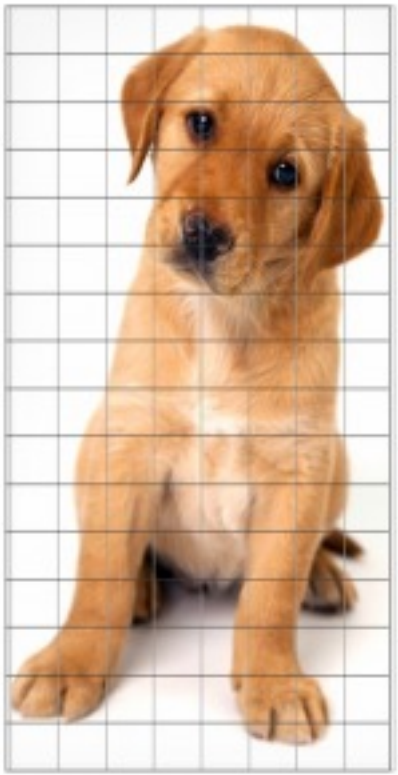
Step 4: create the histogram of gradients

- **Method 4:** add the contribution of a pixel's gradient to the bins on either side of the pixel gradient.
  - Take each pixel value
  - Find the magnitude and orientation
  - 

Magnitude		(4/20)*13.6	(16/20)*13.6						
Bin	0	20	40	60	80	100	120	140	160

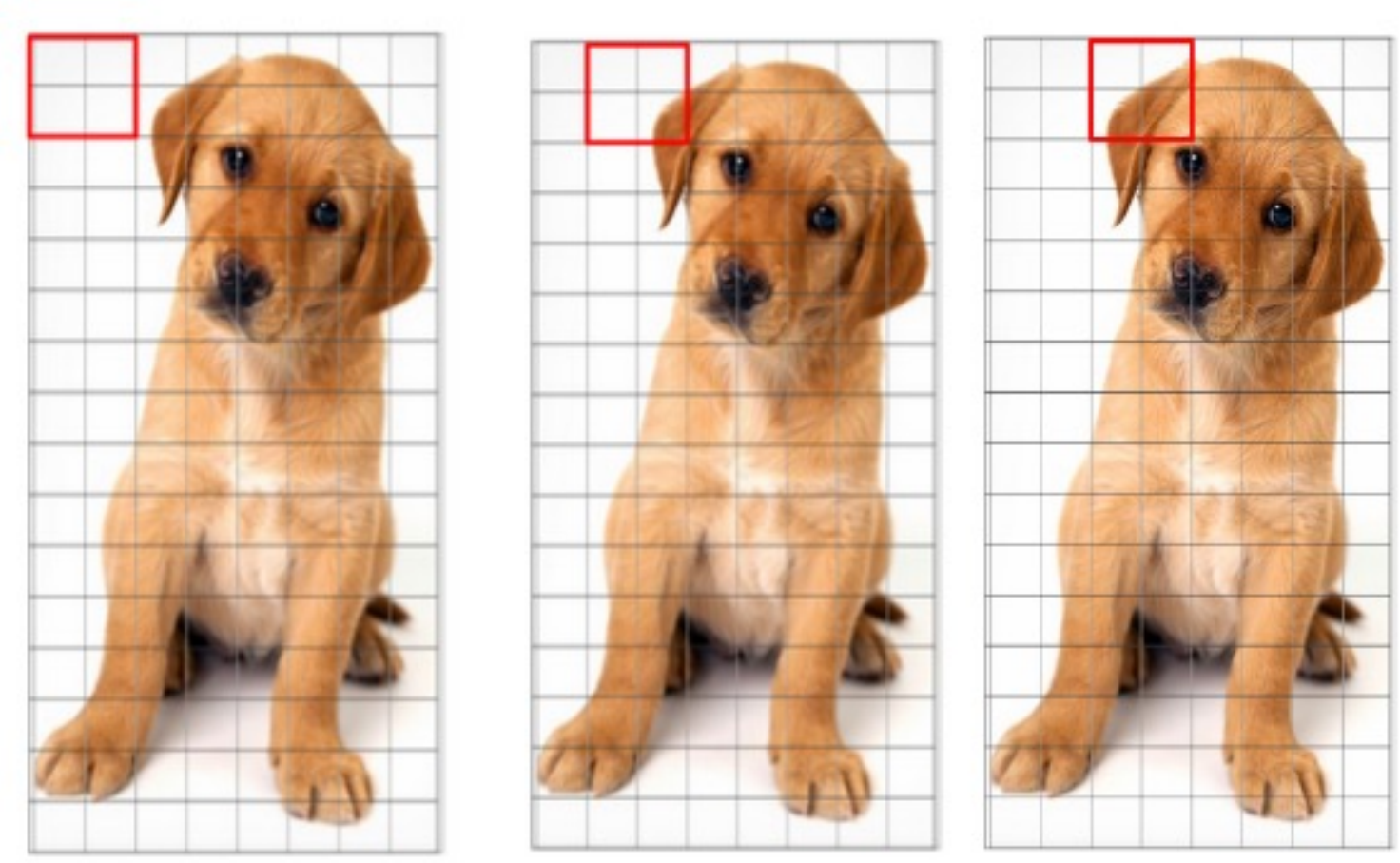
Step 4: create the histogram of gradients

- Divide the image into 8x8 cells
- Apply method 4 to create histogram of gradients for each cell.



Step 5: normalize the histogram

- Combine 4 cells to create 16x16 blocks
- For each block, we get a vector $V = [v_1, v_2, \ldots, v_{36}]$



Step 5: normalize the histogram

- Combine 4 cells to create 16x16 blocks
- For each block, we get a vector $V = [v_1, v_2, \ldots, v_{36}]$
- Calculate k: $k = \sqrt{{v_1}^2 + {v_2}^2 + \ldots + {v_{36}}^2}$
- Normalize the vector:

$$\text{Normalised Vector} = \left(\frac{a_1}{k}, \frac{a_2}{k}, \frac{a_3}{k}, \ldots, \frac{a_{36}}{k} \right)$$

Step 6: get the features

- Combine all features of all blocks
- EX:
  - 7x15 = 105 blocks/image
  - 4 cells/block
  - 9 features/cell
  - Total = 105 x 4 x 9 = 3780 features

  for 64x128 image

https://www.analyticsvidhya.com/blog/2019/09/feature-engineering-images-introduction-hog-feature-descriptor/