Object Detection

Part2

Presenter: Dae-Yong

Contents

Traditional Object Detection

- Feature: Histogram of Oriented Gradient
- Classifier: Support Vector Machine

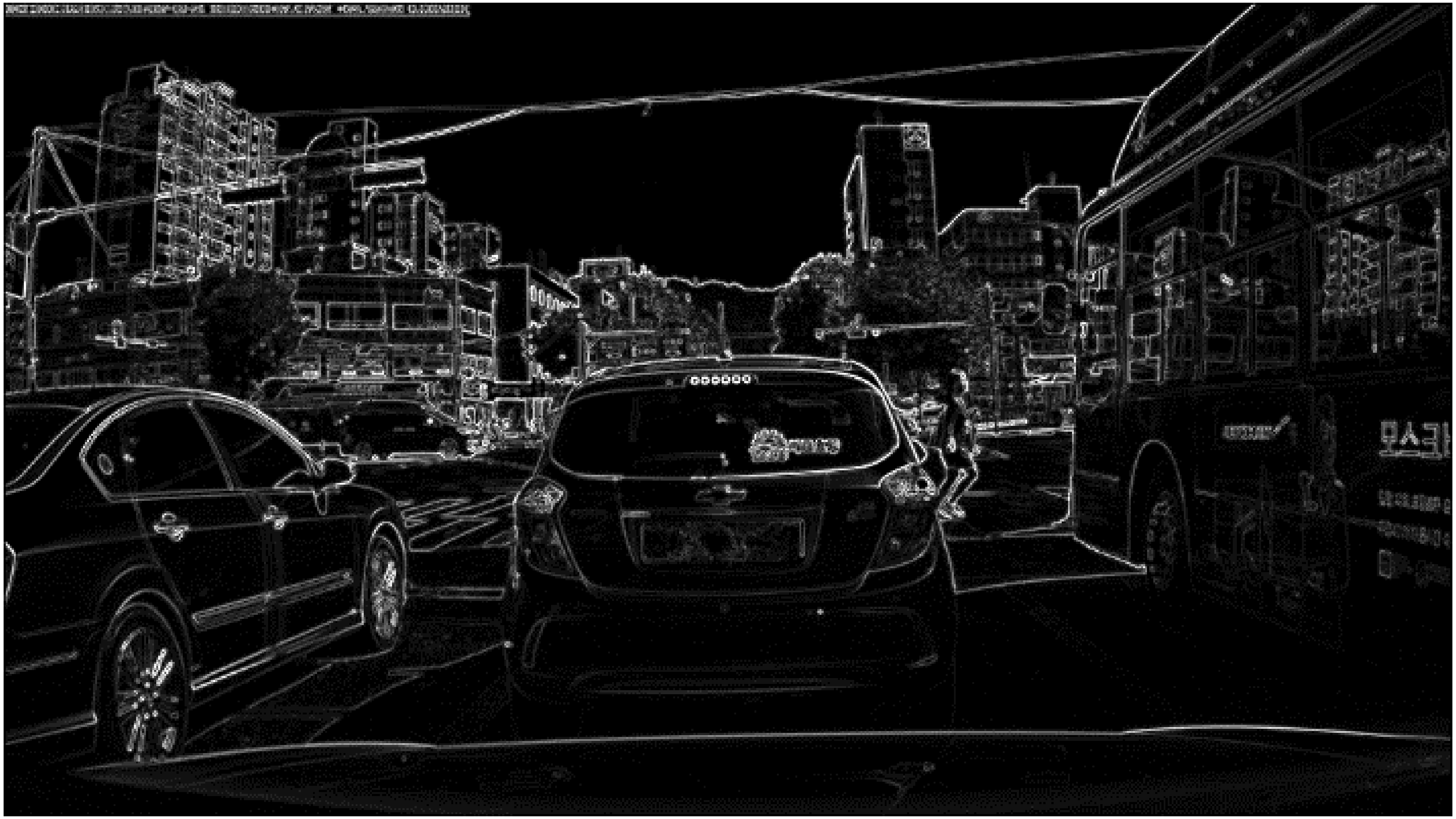
Traditional Object Detection

Traditional Object Detection: Pipeline

Input Image → Feature Extraction → Candidate Generation → Classification → Detection Output

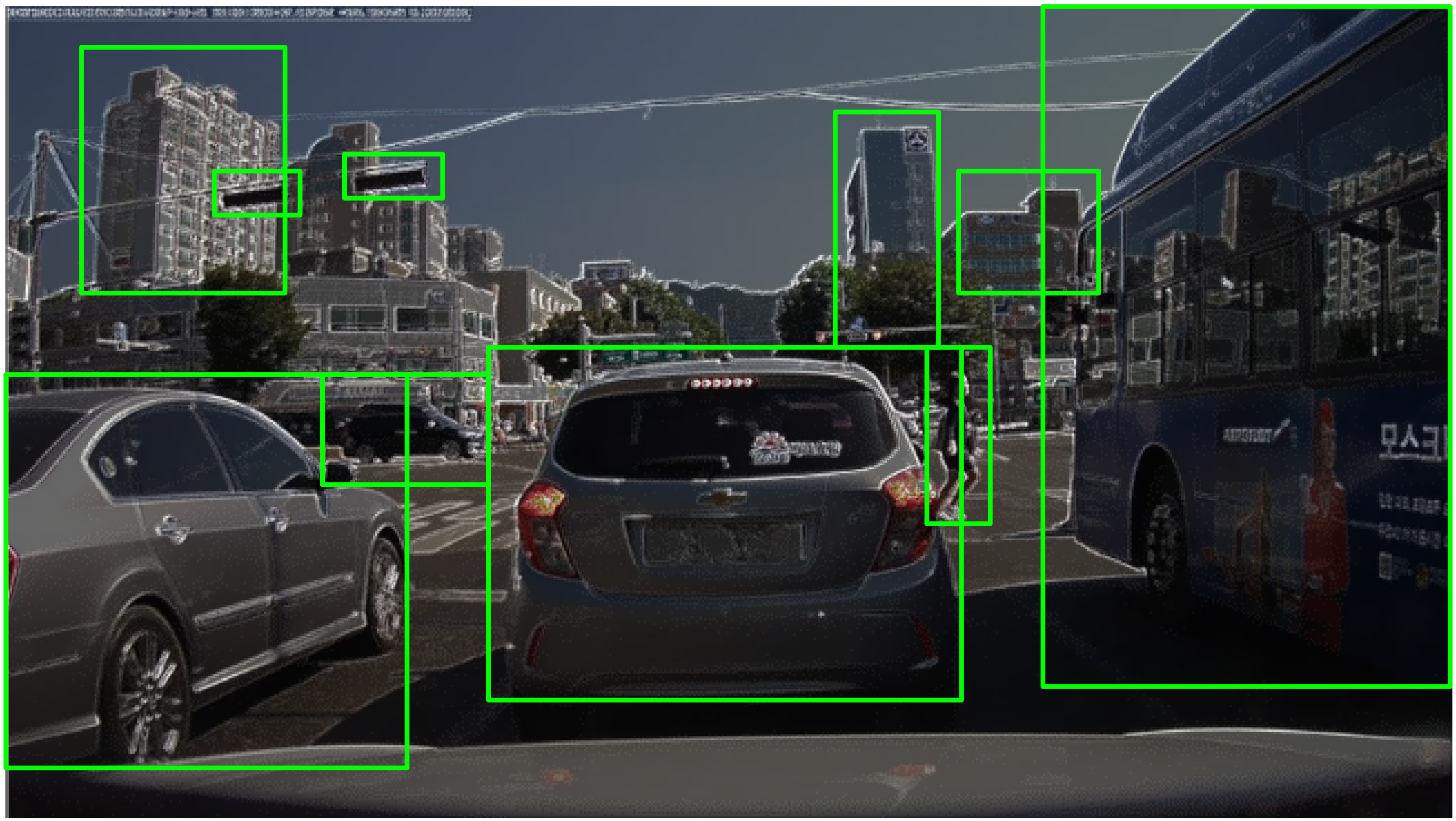
Traditional Object Detection: Pipeline

Input Image → **Feature Extraction** → Candidate Generation → Classification → Detection Output



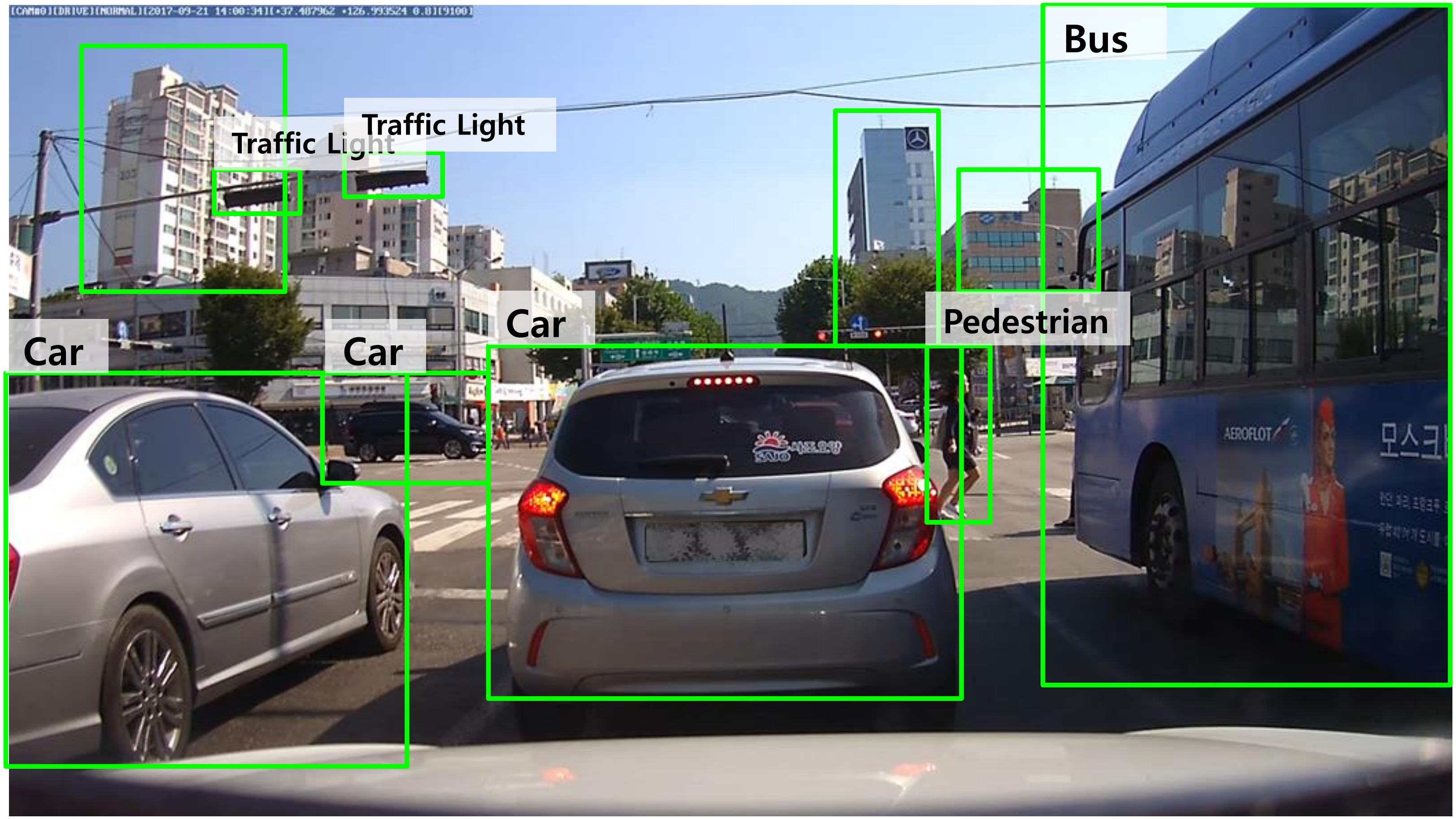
Traditional Object Detection: Pipeline

Input Image → Feature Extraction → **Candidate Generation** → Classification → Detection Output



Traditional Object Detection: Pipeline

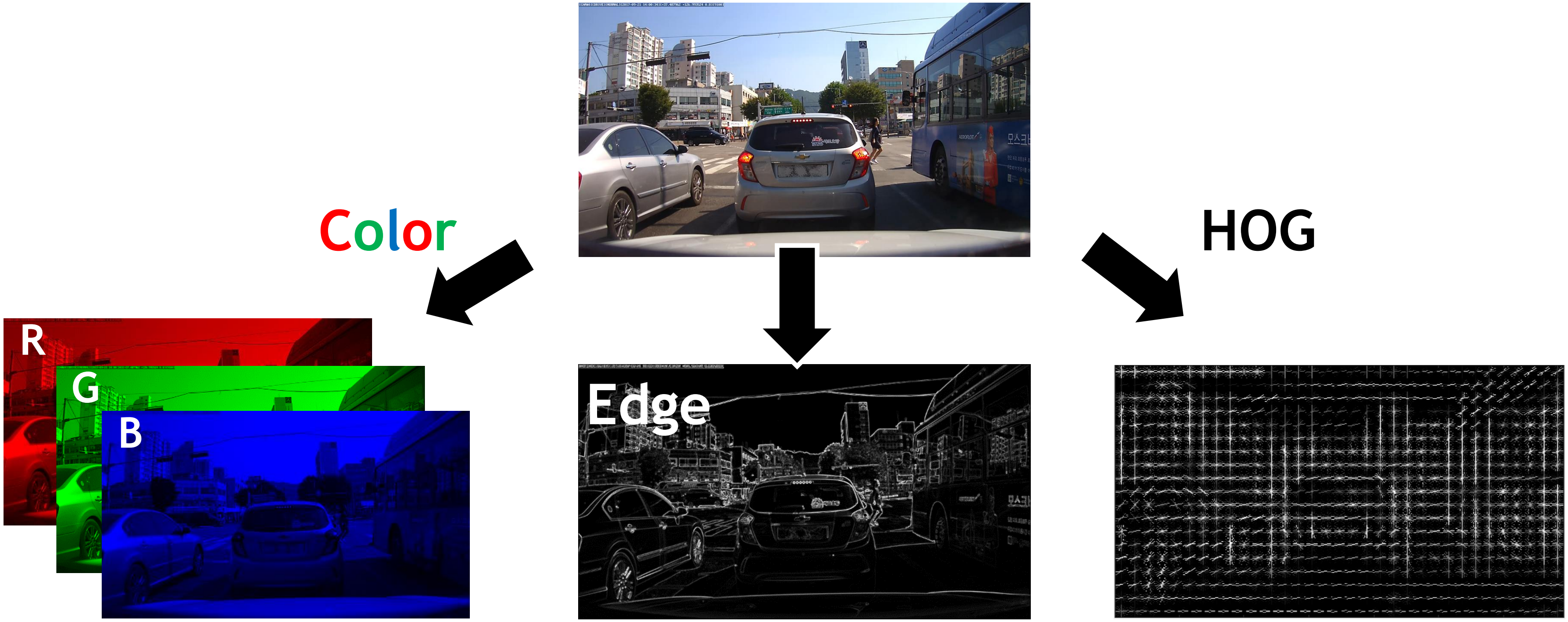
Input Image → Feature Extraction → Candidate Generation → Classification → Detection Output



Traditional Object Detection - Feature Extraction

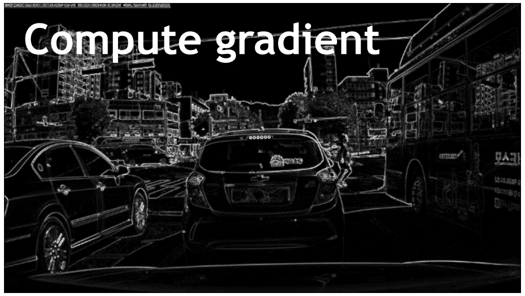
Input Image → **Feature Extraction** → Candidate Generation → Classification → Detection Output

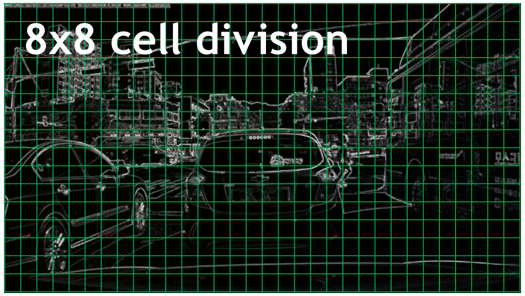
- Real data is too complicate to explain what it is. ➡ **FEATURE**
- Features in Computer Vision
  - Color, Edge, Local Binary Pattern(LBP), Scale Invariant Feature Transform(SIFT), Histogram of Oriented Gradient(HOG), …



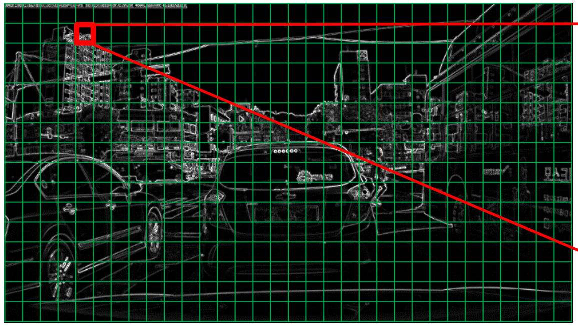
Traditional Object Detection - Feature Extraction

- Histogram of Oriented Gradient Pipeline





Compute Hist. of Oriented Gradient for each 8x8 Cell



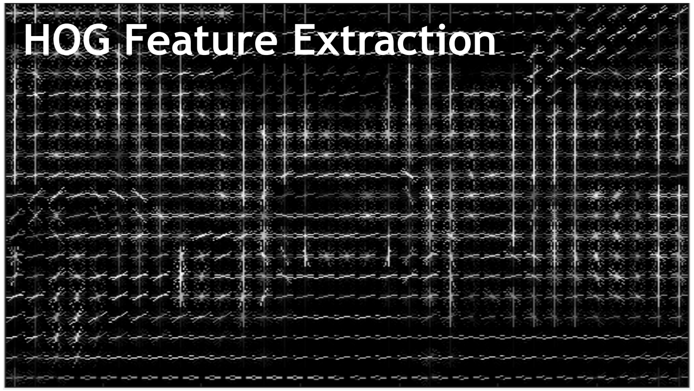
Magnitude

81	85	91	93	90	91	92	91
79	81	80	88	93	90	87	88
110	131	127	129	115	105	119	110
125	110	130	133	138	137	135	135
120	118	122	103	121	129	130	132
5	1	0	4	0	5	2	2
3	7	2	7	2	3	8	1
2	1	7	8	5	4	2	1

Orientation

0	19	18	12	13	48	71	80
1	35	35	37	24	27	21	28
10	3	24	21	29	34	35	27
97	98	91	95	94	88	87	79
91	98	95	97	91	98	94	93
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

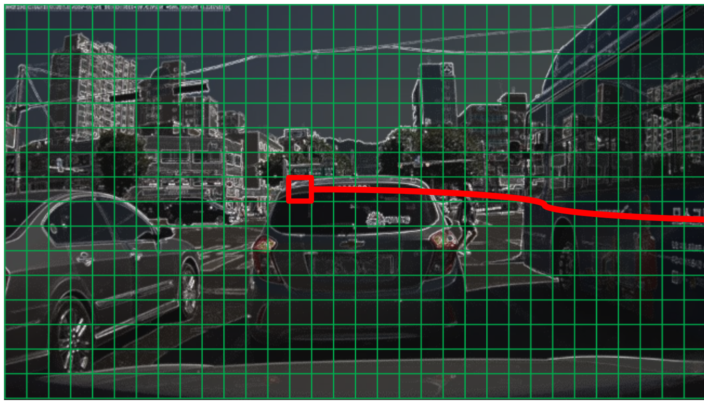
Magnitude	25.5	21	67.4	92.3	11.3	19.1	22.7	11.1	39.1
Orientation	0	20	40	60	80	100	120	140	160



It will be used for describing object

Traditional Object Detection - Feature Extraction

- How to compute Histogram of Oriented Gradient?



Magnitude

81	85	91	93	90	91	92	91
79	81	80	88	93	90	87	88
110	131	127	129	115	105	119	110
125	110	130	133	138	137	135	135
120	118	122	103	121	129	130	132
5	1	0	4	0	5	2	2
3	7	2	7	2	3	8	1
2	1	7	8	5	4	2	1

Orientation

0	19	18	12	13	48	71	80
1	35	35	37	24	27	21	28
10	3	24	21	29	34	35	27
97	98	91	95	94	88	87	79
91	98	95	97	91	98	94	93
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

*Magnitude = $\sqrt{dx^2 + dy^2}$ *(Unsigned) Orientation = $arctan\left(\frac{dy}{dx}\right)$

9 bins

Magnitude									
Orientation	0	20	40	60	80	100	120	140	160

Traditional Object Detection - Feature Extraction

- How to compute Histogram of Oriented Gradient?

Magnitude

81	85	91	93	90	91	92	91
79	81	80	88	93	90	87	88
110	131	127	129	115	105	119	110
125	110	130	133	138	137	135	135
120	118	122	103	121	129	130	132
5	1	0	4	0	5	2	2
3	7	2	7	2	3	8	1
2	1	7	8	5	4	2	1

Orientation

0	19	18	12	13	48	71	80
1	35	35	37	24	27	21	28
10	3	24	21	29	34	35	27
97	98	91	95	94	88	87	79
91	98	95	97	91	98	94	93
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

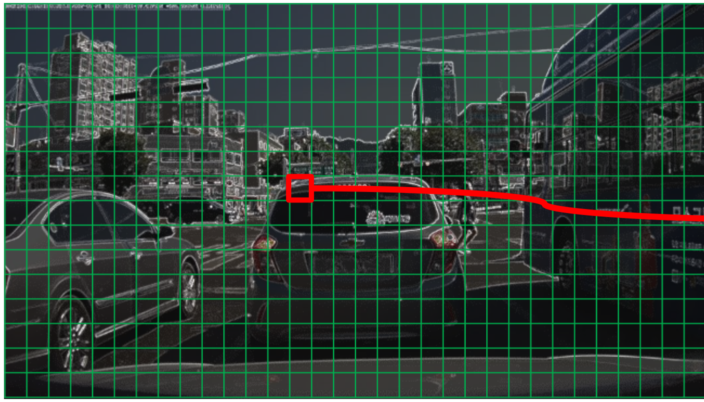
*Magnitude = $\sqrt{dx^2 + dy^2}$ *(Unsigned) Orientation = $arctan\left(\frac{dy}{dx}\right)$

9 bins

Magnitude	81								
Orientation	0	20	40	60	80	100	120	140	160

Traditional Object Detection - Feature Extraction

- How to compute Histogram of Oriented Gradient?



Magnitude

81	85	91	93	90	91	92	91
79	81	80	88	93	90	87	88
110	131	127	129	115	105	119	110
125	110	130	133	138	137	135	135
120	118	122	103	121	129	130	132
5	1	0	4	0	5	2	2
3	7	2	7	2	3	8	1
2	1	7	8	5	4	2	1

Orientation

0	19	18	12	13	48	71	80
1	35	35	37	24	27	21	28
10	3	24	21	29	34	35	27
97	98	91	95	94	88	87	79
91	98	95	97	91	98	94	93
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

*Magnitude = $\sqrt{dx^2 + dy^2}$

*(Unsigned) Orientation = $arctan\left(\frac{dy}{dx}\right)$

9 bins

Magnitude	85.25	80.75							
Orientation	0	20	40	60	80	100	120	140	160

Traditional Object Detection - Feature Extraction

- How to compute Histogram of Oriented Gradient?

Magnitude

81	85	91	93	90	91	92	91
79	81	80	88	93	90	87	88
110	131	127	129	115	105	119	110
125	110	130	133	138	137	135	135
120	118	122	103	121	129	130	132
5	1	0	4	0	5	2	2
3	7	2	7	2	3	8	1
2	1	7	8	5	4	2	1

Orientation

0	19	18	12	13	48	71	80
1	35	35	37	24	27	21	28
10	3	24	21	29	34	35	27
97	98	91	95	94	88	87	79
91	98	95	97	91	98	94	93
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

*Magnitude = $\sqrt{dx^2 + dy^2}$ *(Unsigned) Orientation = $arctan\left(\frac{dy}{dx}\right)$

9 bins

Magnitude	94.35	162.65							
Orientation	0	20	40	60	80	100	120	140	160

Traditional Object Detection - Feature Extraction

- How to compute Histogram of Oriented Gradient?

Magnitude

81	85	91	93	90	91	92	91
79	81	80	88	93	90	87	88
110	131	127	129	115	105	119	110
125	110	130	$\boxed{133}$	138	137	135	135
120	118	122	103	121	129	130	132
5	1	0	4	0	5	2	2
3	7	2	7	2	3	8	1
2	1	7	8	5	4	2	1

Orientation

0	19	18	12	13	48	71	80
1	35	35	37	24	27	21	28
10	3	24	21	29	34	35	27
97	98	91	$\boxed{95}$	94	88	87	79
91	98	95	97	91	98	94	93
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

*Magnitude = $\sqrt{dx^2 + dy^2}$ *(Unsigned) Orientation = $arctan\left(\frac{dy}{dx}\right)$

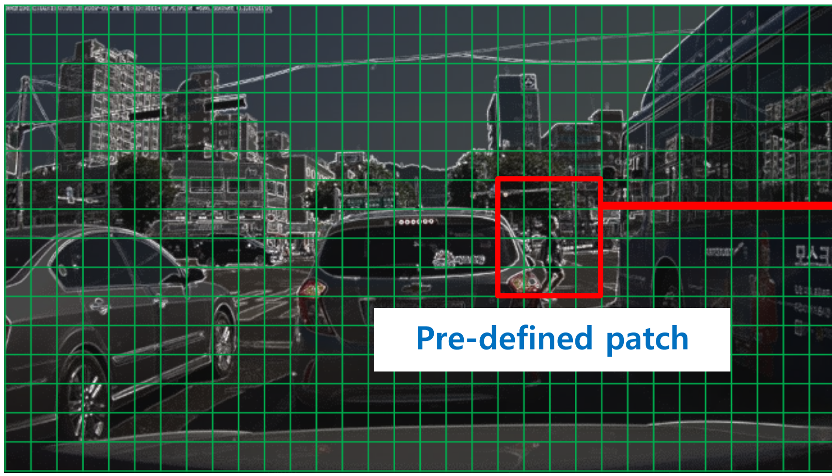
9 bins

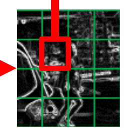
Magnitude	94.35	162.65			33.25	99.75	0	0	0
Orientation	0	20	40	60	80	100	120	140	160

It decreases 64 dimension to 9 dimension (85%↓)

Traditional Object Detection - Feature Extraction

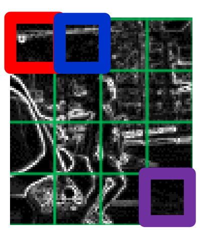
- How to describe TARGET OBJECT using HOG feature?





81	85	91	93	90	91	92	91
79	81	80	88	93	90	87	88
110	131	127	129	115	105	119	110
125	110	130	133	138	137	135	135
120	118	122	103	121	129	130	132
5	1	0	4	0	5	2	2
3	7	2	7	2	3	8	1
2	1	7	8	5	4	2	1

Magnitude	25.5	21	67.4	92.3	11.3	19.1	22.7	11.1	39.1
Orientation	0	20	40	60	80	100	120	140	160

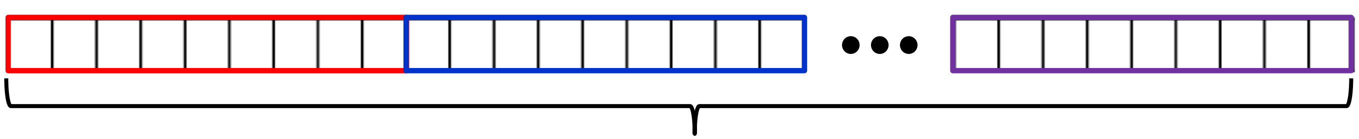


4x4 cells

Magnitude	25.5	21	67.4	92.3	11.3	19.1	22.7	11.1	39.1
Orientation	0	20	40	60	80	100	120	140	160

x16

144-dimensional vector: HOG Feature Descriptor



This **FEATURE vector** describes **PEDESTRIAN**

Traditional Object Detection - Candidate Generation

Input Image → **Feature Extraction** → **Candidate Generation** → **Classification** → **Detection Output**

- What is pre-defined patch?
  - Object candidate
  - **Sliding Window Scheme** / Object Proposal Scheme
  - Designed by **user**

    *Parameters: width, height, stride

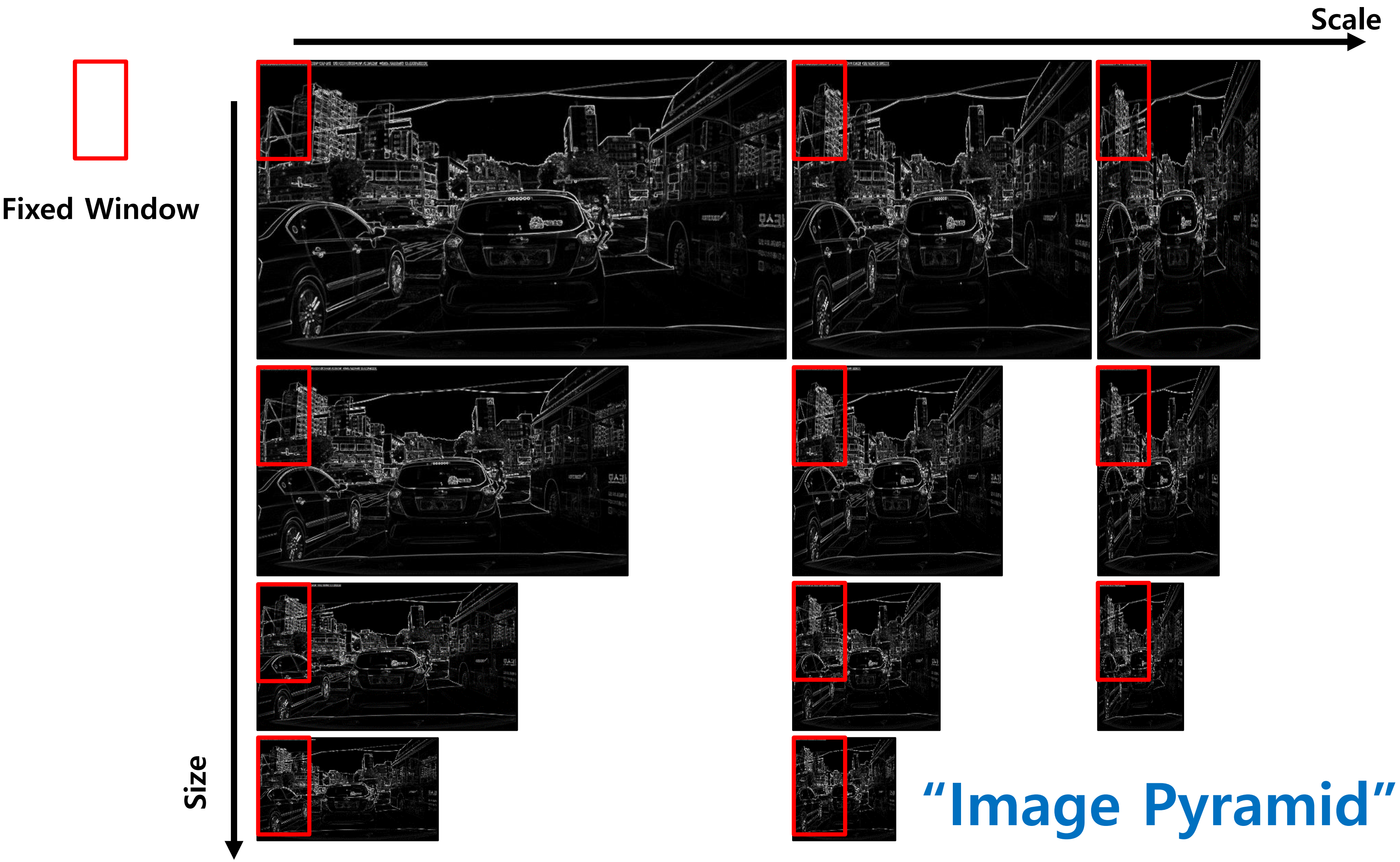
1pixel

Width: 2 pix
Height: 3 pix
Stride: 1 pix

Width: 4 pix
Height: 2 pix
Stride: 2 pix

Traditional Object Detection - Candidate Generation

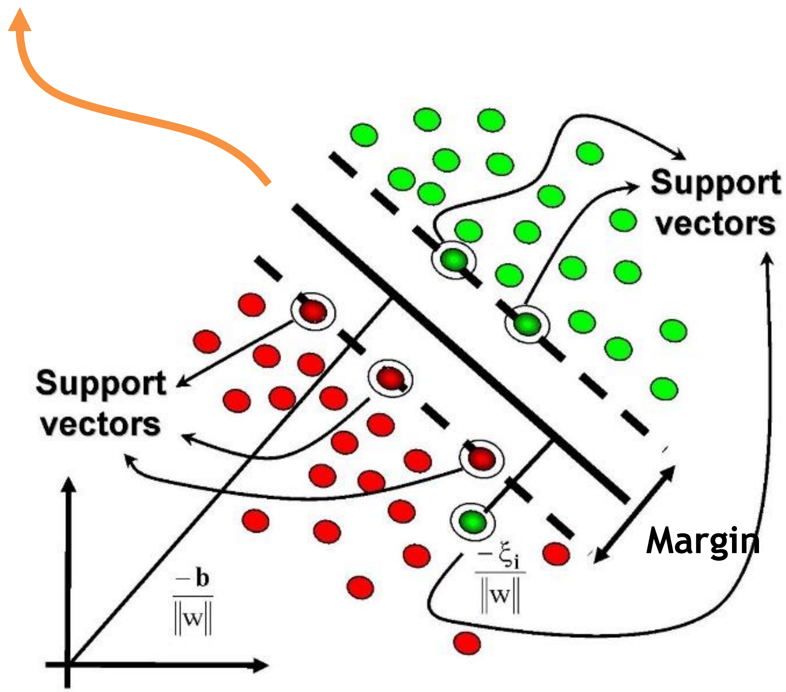
- Another way to consider multiple scale/size of objects



Traditional Object Detection - Classification

Input Image → Feature Extraction → Candidate Generation → **Classification** → Detection Output

- Support Vector Machine (SVM)
  - Goal: Classify POSITIVE(+) – NEGATIVE(-)
  - How: Define HYPERPLANE which maximizes the margin btw two classes



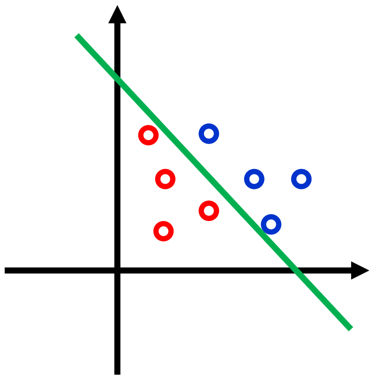
Positive Sample

Negative Sample

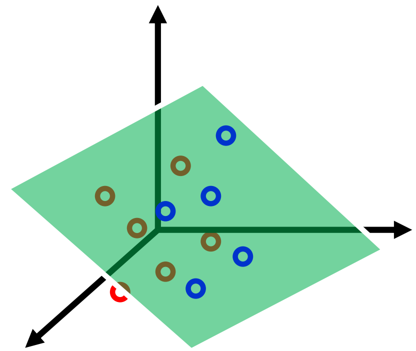
Support Vector Machine (SVM)

- What is the **hyperplane**?

  Def) A subspace of one dimension less than its ambient space



2-dimensional space
>> Hyperplane: line



3-dimensional space
>> Hyperplane: plane

?

n-dimensional space
>> Hyperplane: n-1 dim

$$f(x, y) = ax + by + c \qquad f(x, y, z) = ax + by + cz + d$$

$$\mathbf{w} = \begin{bmatrix} a \\ b \\ c \end{bmatrix} \quad \mathbf{x} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \qquad \mathbf{w}^{\mathrm{T}}\mathbf{x} = 0$$

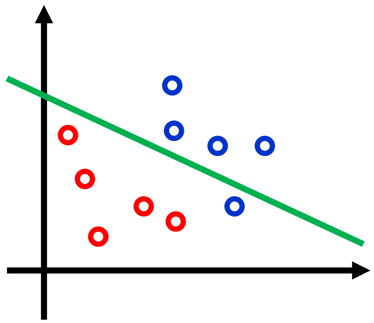
What we know from **training data**

What we need to compute

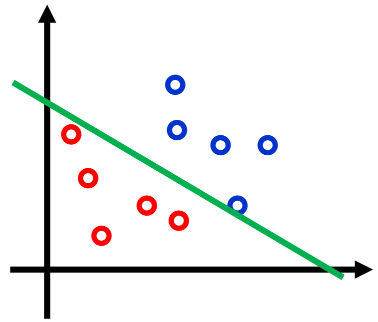
Support Vector Machine (SVM)

- What is the **OPTIMAL** hyperplane?

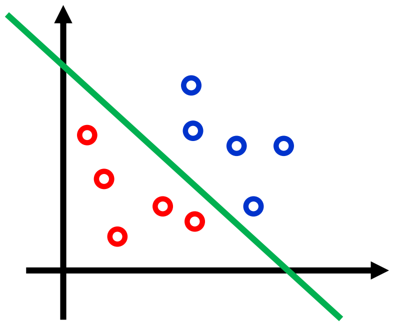
Case 1)



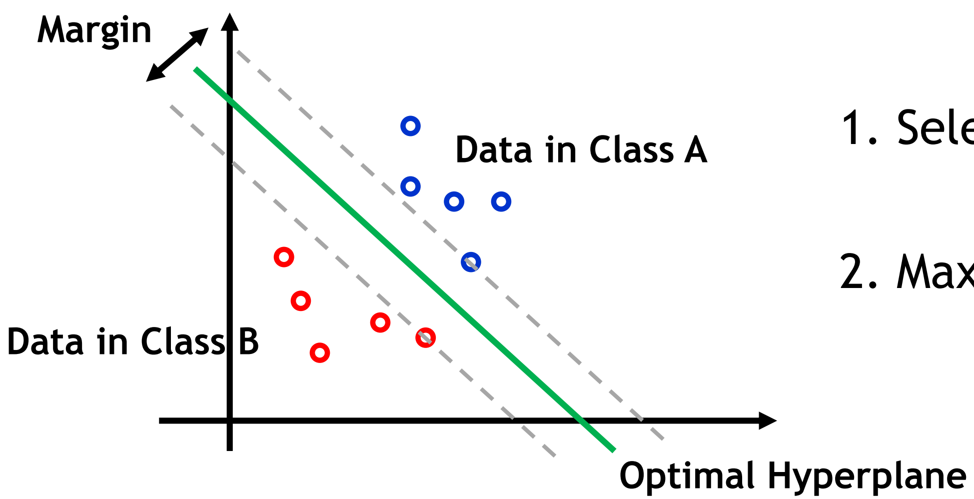
Case 2)



Case 3)



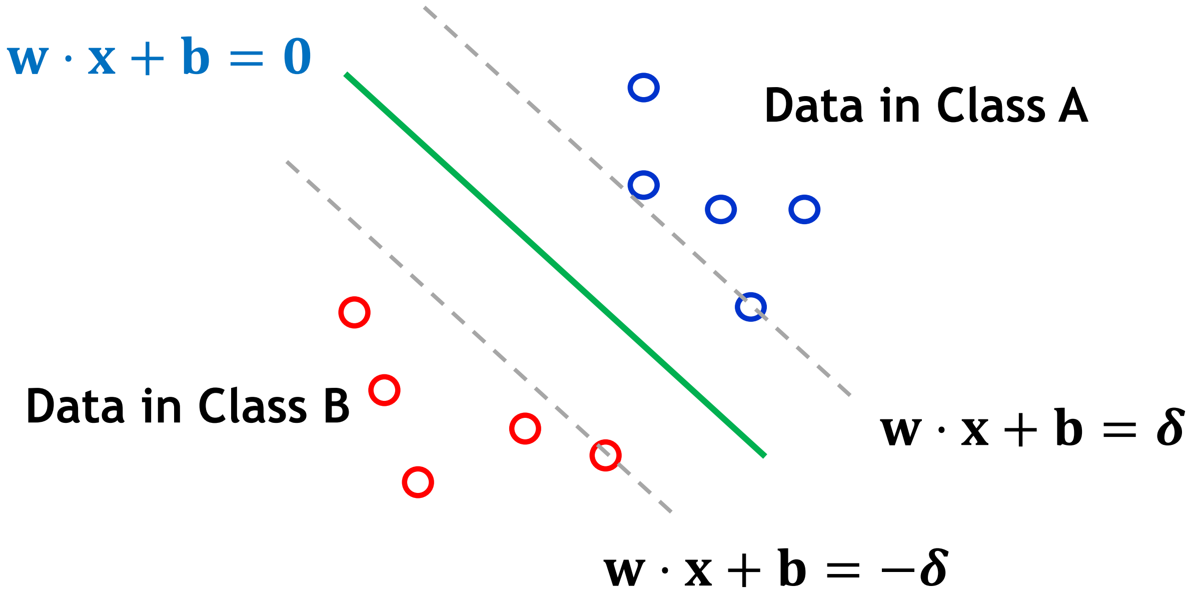
- How to obtain the **OPTIMAL** hyperplane?



1. Select two hyperplanes which separate the data (Gray dashed lines)
2. Maximize their distances (= margin)

Support Vector Machine (SVM)

- Two hyperplane selection



[Constraints]

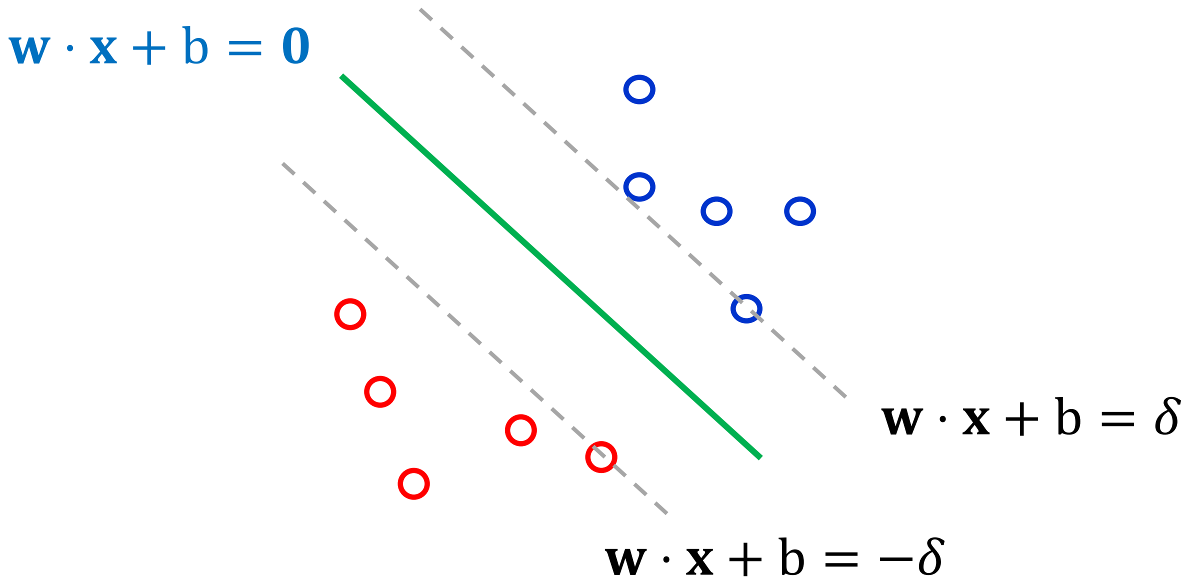
$\mathbf{w} \cdot \mathbf{x} + \mathbf{b} \geq \boldsymbol{\delta}$ for $\mathbf{x}$ having the class A (label: 1)

$\mathbf{w} \cdot \mathbf{x} + \mathbf{b} \leq -\boldsymbol{\delta}$ for $\mathbf{x}$ having the class B (label: -1)

➜ Objective Function: $\mathbf{y}(\mathbf{w} \cdot \mathbf{x} + \mathbf{b}) \geq \mathbf{1}$

Support Vector Machine (SVM)

- Maximize distance between two hyperplanes



Distance between two hyperplanes: $m = \frac{2}{||\mathrm{w}||}$

Maximize $m \leftrightarrow$ Minimize $||\mathbf{w}||$

Minimize $||\mathbf{w}||$ subject to $\mathbf{y}(\mathbf{w} \cdot \mathbf{x} + \mathrm{b}) \geq 1$ → Numerical Optimization

https://www.svm-tutorial.com/2015/06/svm-understanding-math-part-3/

Support Vector Machine (SVM)

- Kernel trick
  - How can we handle data which are not linearly separable?

Practice with Toy Example

COMING SOON

Thank You