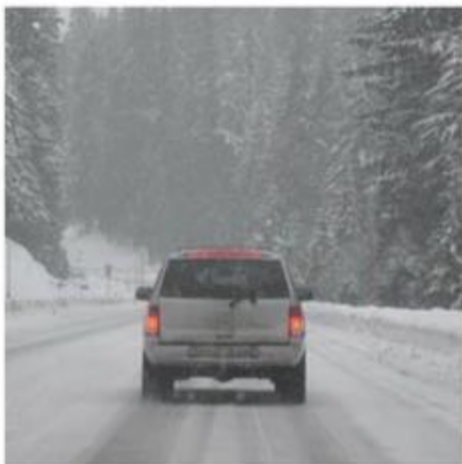


# **Object localization**

# Object localization

- ▶ Q. AVG POOL / Activation function -> Sigmoid / 적은 변수 갯수

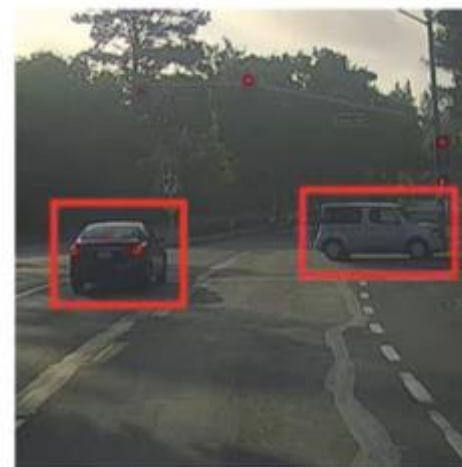
Image classification



Classification with localization



Detection



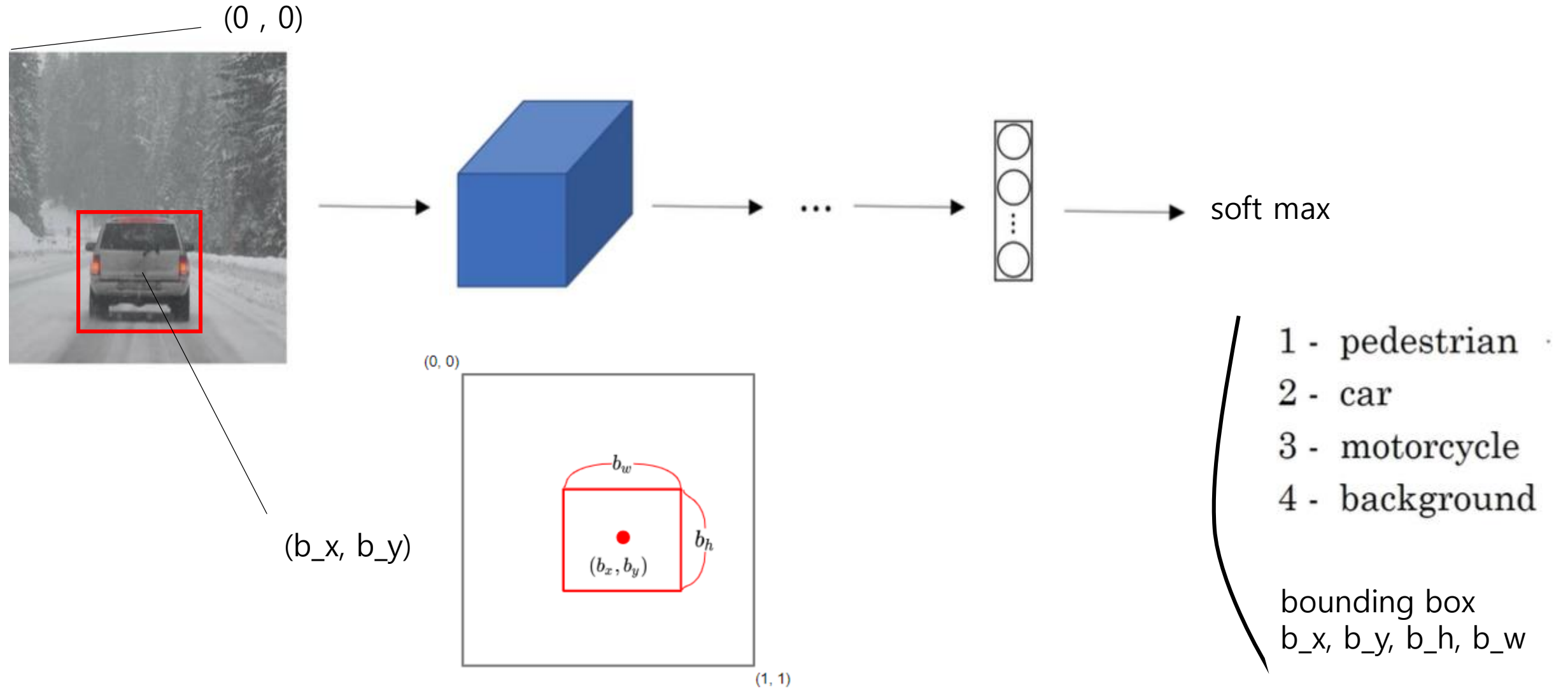
# **Sliding windows detection**

# Sliding windows detection

- ▶ 윈도우가 모든 영역을 지나가면서 conv 연산한다. / 계산비용이 크다. /  
윈도우가 작을 경우 물체를 못 잡을 수도 있다.



# Classification with localization



# Defining the target label 'y'

- ▶ Q. 아래의 이미지일때, bounding box의 좌표는  $b_x, b_y, b_h, b_w$ 로 정의하고, label 'y'를 정의



$$\begin{bmatrix} p_c \\ b_x \\ b_y \\ b_h \\ b_w \\ c_1 \\ c_2 \\ c_3 \end{bmatrix}$$

$p_c$  : object probability : 물체 존재여부의 확률

$b$  : bounding box : 경계상자의 위치

$c$  : classes : 0, 1로 이루어진 물체 클래스의 라벨

- 1 - pedestrian
- 2 - car
- 3 - motorcycle
- 4 - background

$p_c / b_x, b_y, b_h, b_w, / 0, 1, 0$

(background는 object probability가 '0'일 때라서 class에 포함 X)

# **Landmark detection**

# Landmark detection

- ▶ 얼굴인식도 동일하게 얼굴유무, 인식할 좌표 로 클래스를 분류한다.



$b_x, b_y, b_h, b_w$

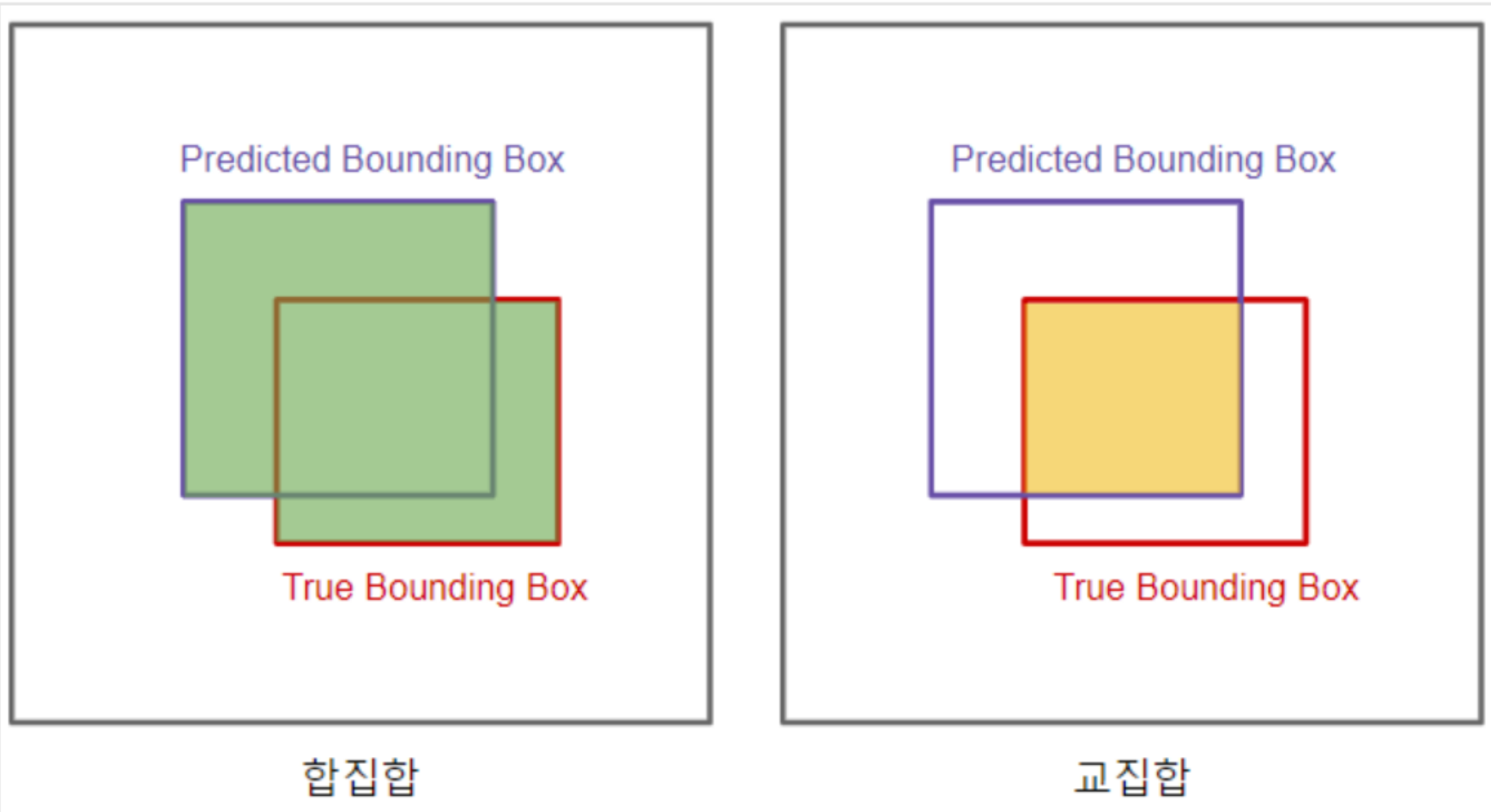




# **Object Detection**

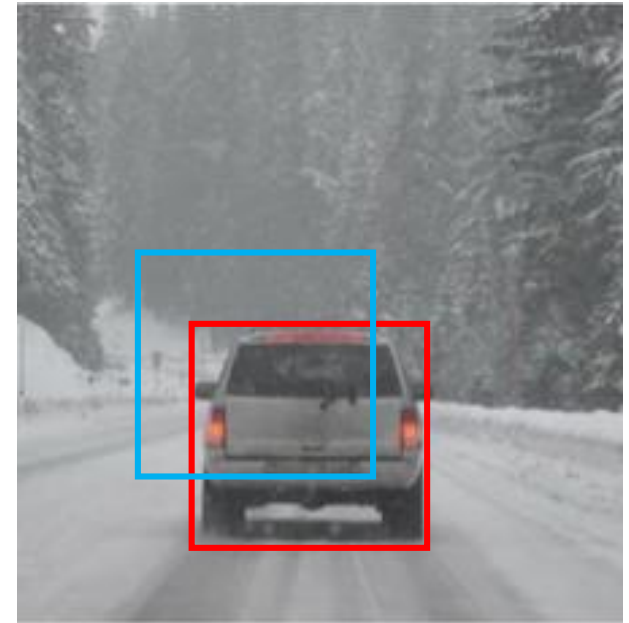
# IOU

- ▶ 통상 IOU가 0.5보다 크면 맞다고 판단



$$\text{IOU} = \frac{\text{의 면적}}{\text{의 면적}}$$

The equation shows a yellow square above the numerator and a green square below the denominator, corresponding to the intersection and union areas shown in the diagram.



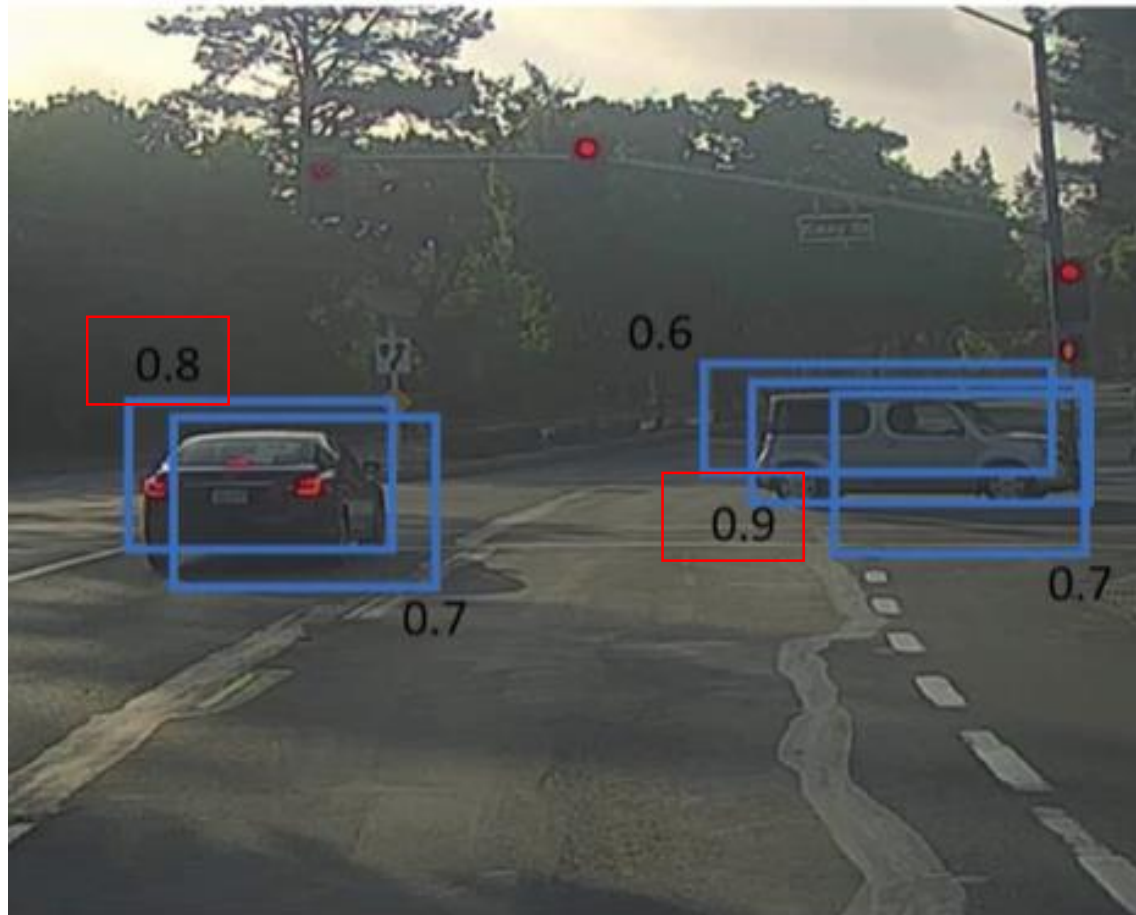
**Q. 하나의 물체에 여러 개의  
bounding box를 인식 할 때 ?**

# **Non-max suppression algorithm**

# Non-max suppression algorithm

▶ 물체 감지가 여러 개 일 때

- (1) 가장 큰 확률값을 가지는 BOX 선택 (특징 임계점 이하의 확률값은 사용하지 않는다. ( ex 0.6 < 제거 )
- (2) 예측값이 가장 큰 box 선택 주변 BOX 감지된 물체와의 IOU가 높은 BOX 제거한다. ( 동일한 물체로 판단)



**Q. 물체가 겹쳐 있을 때는?**

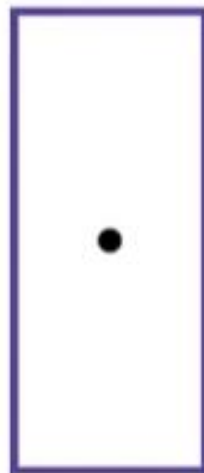
**Anchor box**

# Anchor box

---



Anchor box 1:

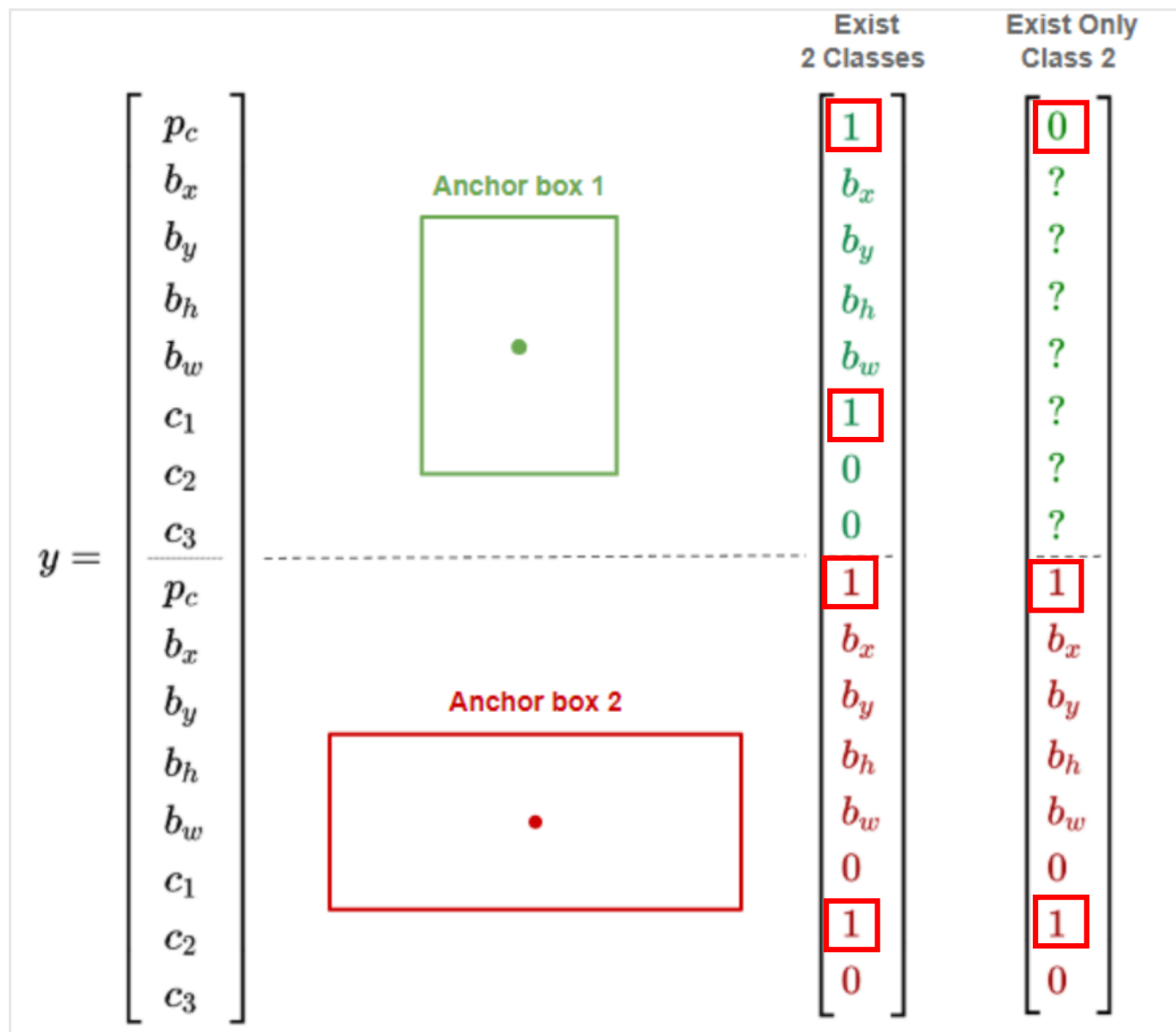


Anchor box 2:



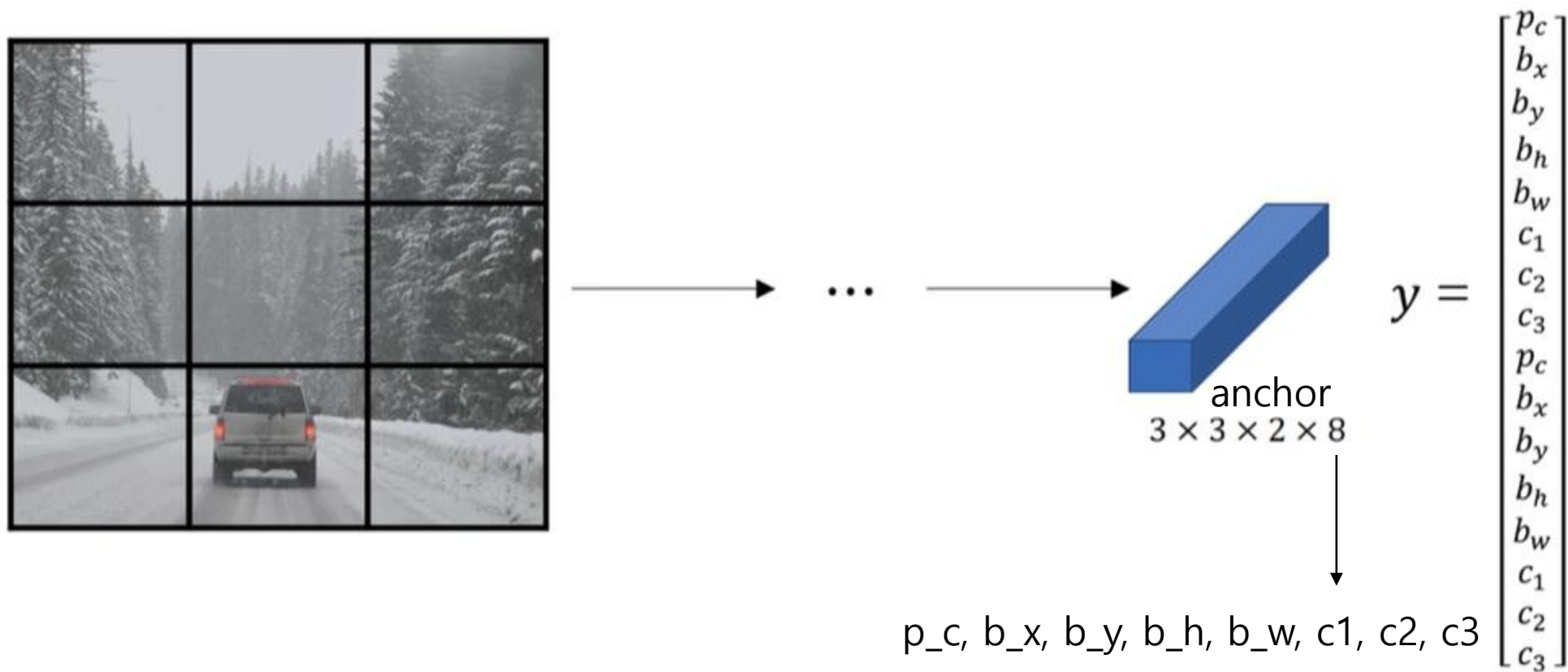


# Anchor box

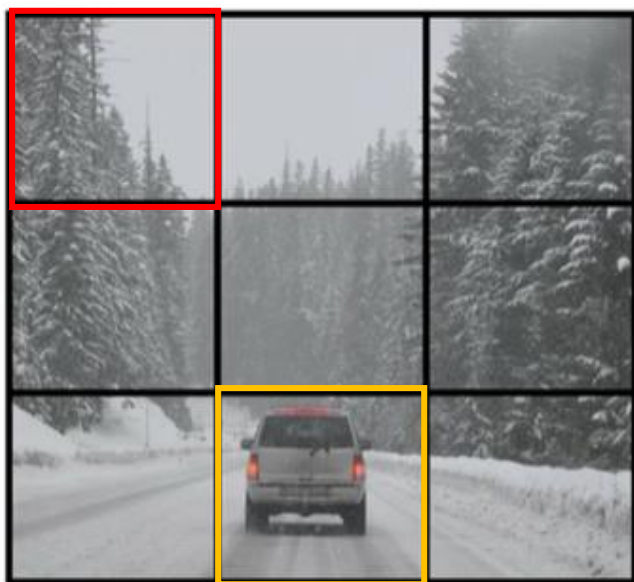


**YOLO**

# YOLO



# YOLO



1 - pedestrian

2 - car

3 - motorcycle

$y =$

$$\begin{bmatrix} p_c \\ b_x \\ b_y \\ b_h \\ b_w \\ c_1 \\ c_2 \\ c_3 \\ p_c \\ b_x \\ b_y \\ b_h \\ b_w \\ c_1 \\ c_2 \\ c_3 \end{bmatrix}$$

0

0

0

1

Anchor

$y$  is  $3 \times 3 \times 2 \times 8$  (3 X 3 X 16)

$p_c, b_x, b_y, b_h, b_w, c_1, c_2, c_3$

# YOLO

---



1. 두개의 앵커박스의 예로 들면 각 셀에 2개의 Bounding Box가 생긴다.

# YOLO

---



1. 두개의 앵커박스의 예로 들면 각 셀에 2개의 Bounding Box가 생긴다.
2. 낮은 확률의 예측을 제거한다.

# YOLO

---



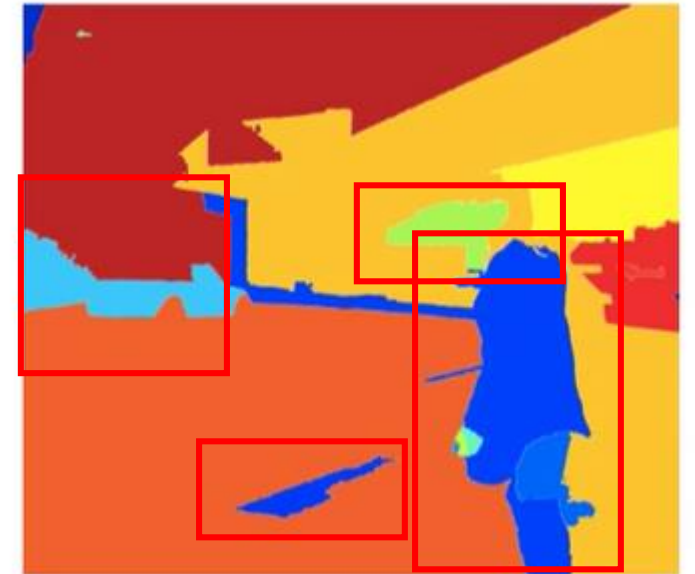
1. 두개의 앵커박스의 예로 들면 각 셀에 2개의 Bounding Box가 생긴다.
2. 낮은 확률의 예측을 제거한다.
3. 예측 하는 각 클래스에 대해서 **Non-max suppression algorithm**를 실행한다.

**R-CNN**



# R-CNN

segmentation algorithm



# Neural style transfer

# Neural style transfer

▶ Gradient descent를 사용해서 초기화된 이미지에서 부터 점점 변화한다.



Content C



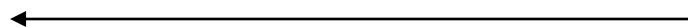
Style S

- $J(G) = \alpha J_{\text{content}}(C, G) + \beta J_{\text{style}}(S, G)$
- $J_{\text{content}}(C, G)$  : 내용 비용, 생성 이미지와 내용 이미지의 내용이 비슷한 정도
- $J_{\text{style}}(S, G)$  : 스타일 비용, 생성 이미지와 스타일 이미지의 비슷한 정도



Generated image G

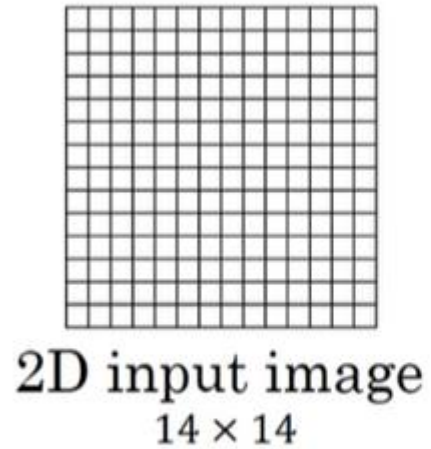
Gradient Descent



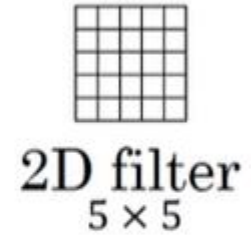
초기 이미지

**1D, 2D, 3D Data**

# 2D, 1D Data

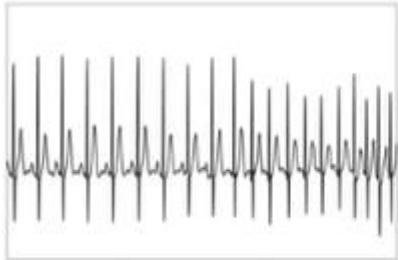


\*



$$(14 \times 14 \times 3) * (5 \times 5 \times 3) / \text{filter} = 16$$

->  $10 \times 10 \times 16$



\*



$$(14 \times 1) * (5 \times 1) / \text{filter} = 16$$

->  $10 \times 16$

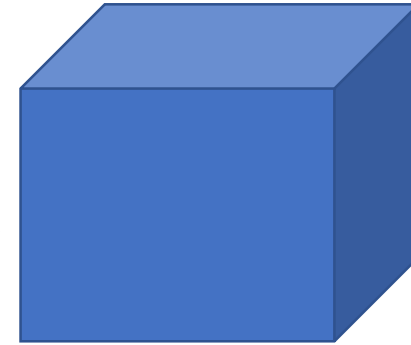
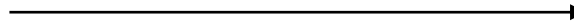
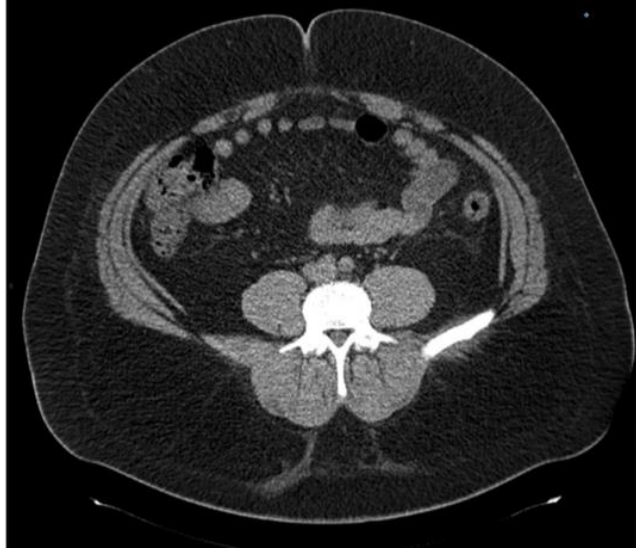
$$(10 \times 16) * (10 \times 16) / \text{filter} = 32$$

->  $6 \times 32$

1	20	15	3	18	12	4	17
---	----	----	---	----	----	---	----

1	3	10	3	1
---	---	----	---	---

# 3D Data



3D data



3D volume

\*



3D filter

$$\begin{aligned} & (14 \times 14 \times 14 \times 1) * (5 \times 5 \times 5 \times 1) / \text{filter} = 16 \\ & \rightarrow 10 \times 10 \times 10 \times 16 \\ & (10 \times 10 \times 10 \times 16) * (5 \times 5 \times 5 \times 16) / \text{filter} = 32 \\ & \rightarrow (6 \times 6 \times 6 \times 32) \end{aligned}$$

채널 수