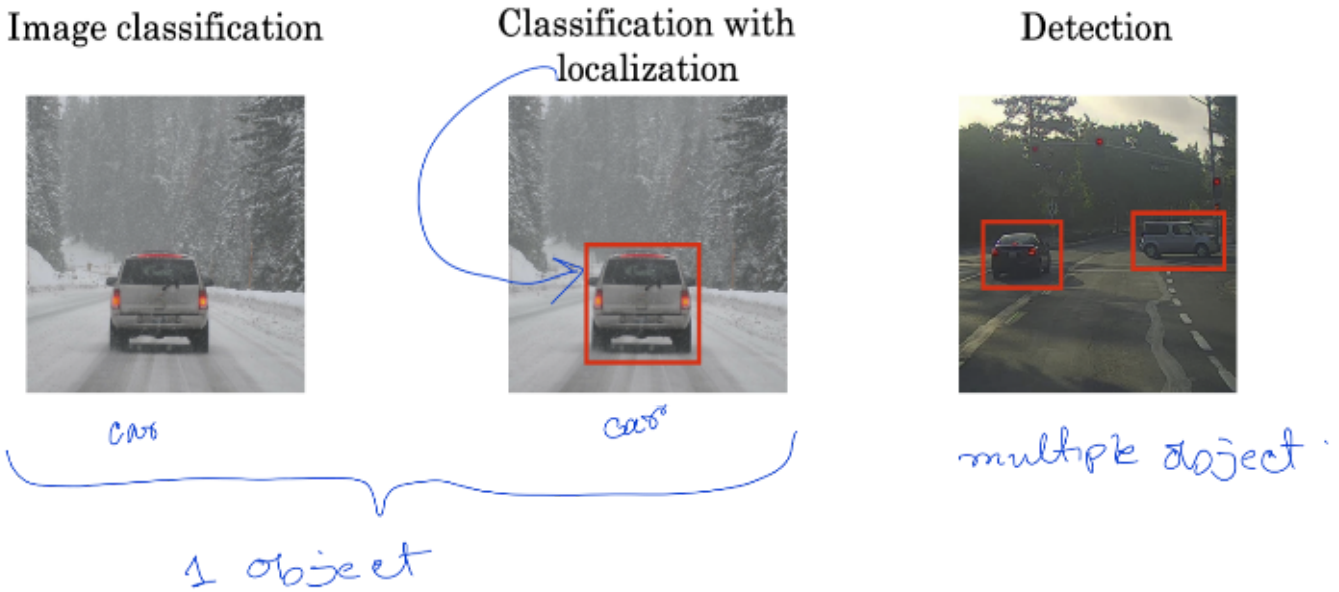
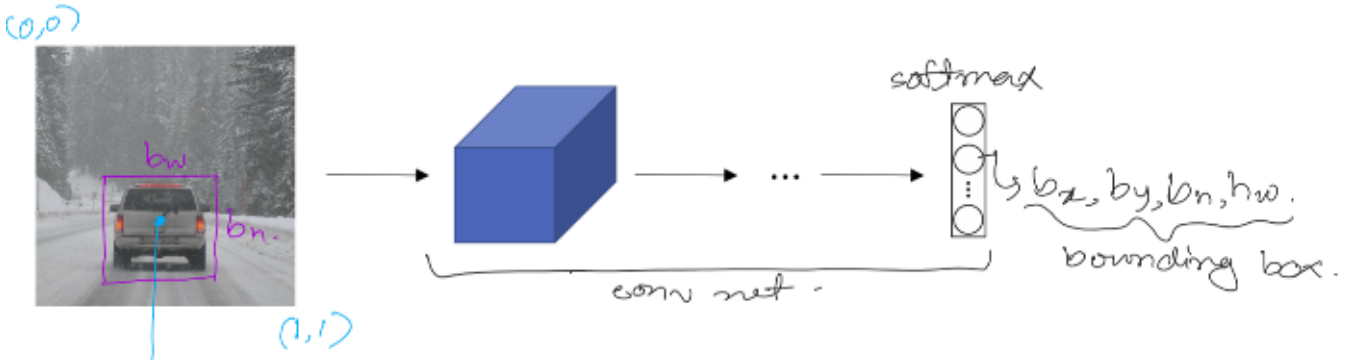
C4W3

Object detection:-

Image localization vs detection:-



Classification with localization:-



Defining target label y:- (supervised learning)

Classes:-

1- pedestrain

need to output:-

bx, by, bw, bh, Class label (1-4)

2 - car

3 - motor bike

4 - background

$y =$ [p_c → if have object. , b_x, b_y, b_w, b_h, c_1, c_2, c_3]

Assuming having only one object per image:-

if a image have car then y will be.

$y =$ [1 → some value, □, □, □, □, 0, 6]

{ no object present.

$y =$ [0, ? → don't care., ?, ?, ?, ?, ?, ?]

Loss:-

If there is an object in the image ($p_c = 1$) loss will be sum of square of all component.

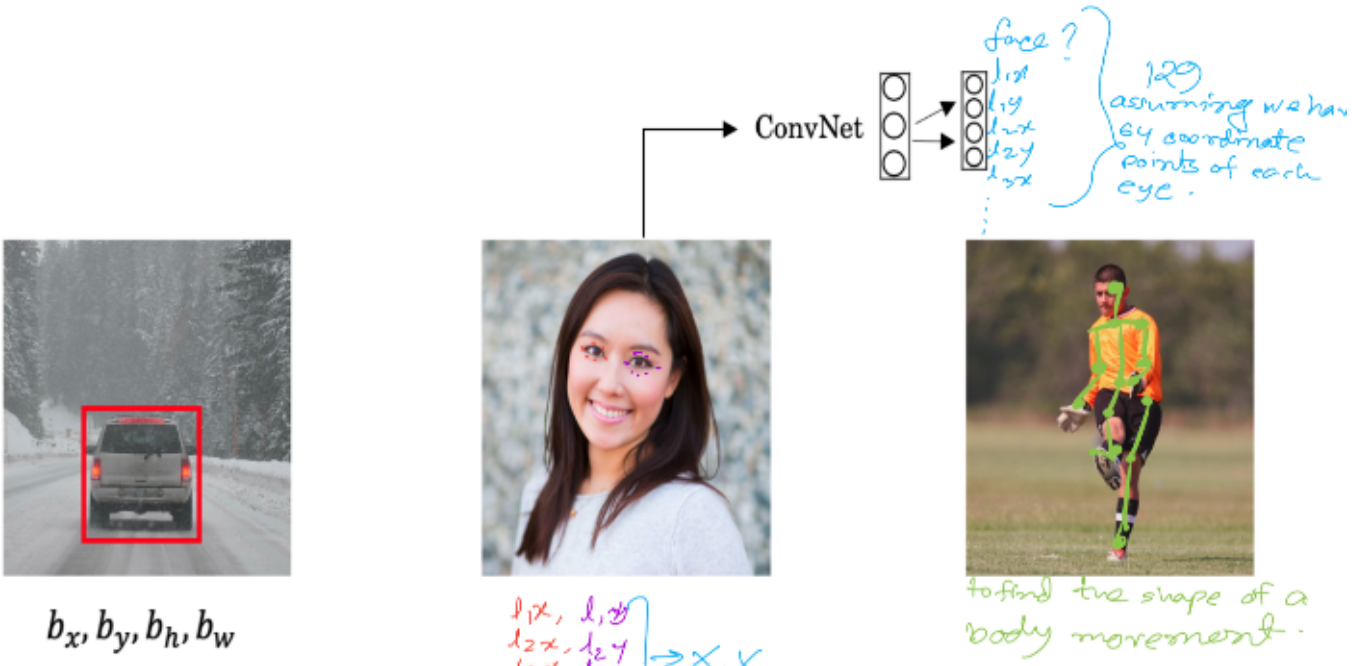
$$\mathcal{L}(\hat{y}, y) = (\hat{y}_1 - y_1)^2 + (\hat{y}_2 - y_2) + \dots + (\hat{y}_8 - y_8)$$

if there is no object ($p_c = 0$)

$$\mathcal{L}(\hat{y}, y) = (\hat{y}_1 - y_1)^2$$

Instead of using the sum of square we can also use the log likelyhood.

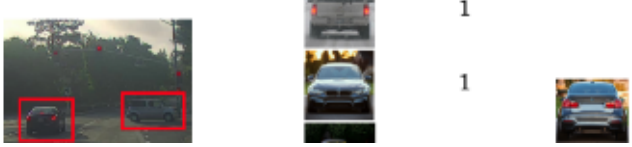
Landmark detection:-

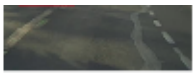


In order to find the eye of a person we need to mark the eye of person face image for training set. Let's we want to find the coordinate of two eyes of a person.

Object detection:-

Let we want to localize cars in a image. One way to do that is sliding window detection. Where we take a small portion of a image and pass it in a convnet. if there are any car in that small window then convnet will return yes. Training set for this model is the cropped version of car image.



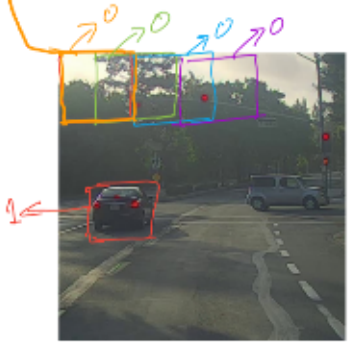


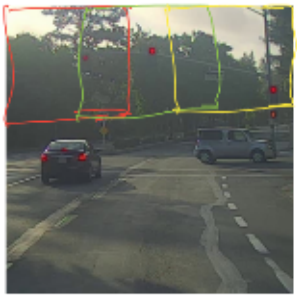
1

0

0

Pass this small region to a convnet whose training sets are



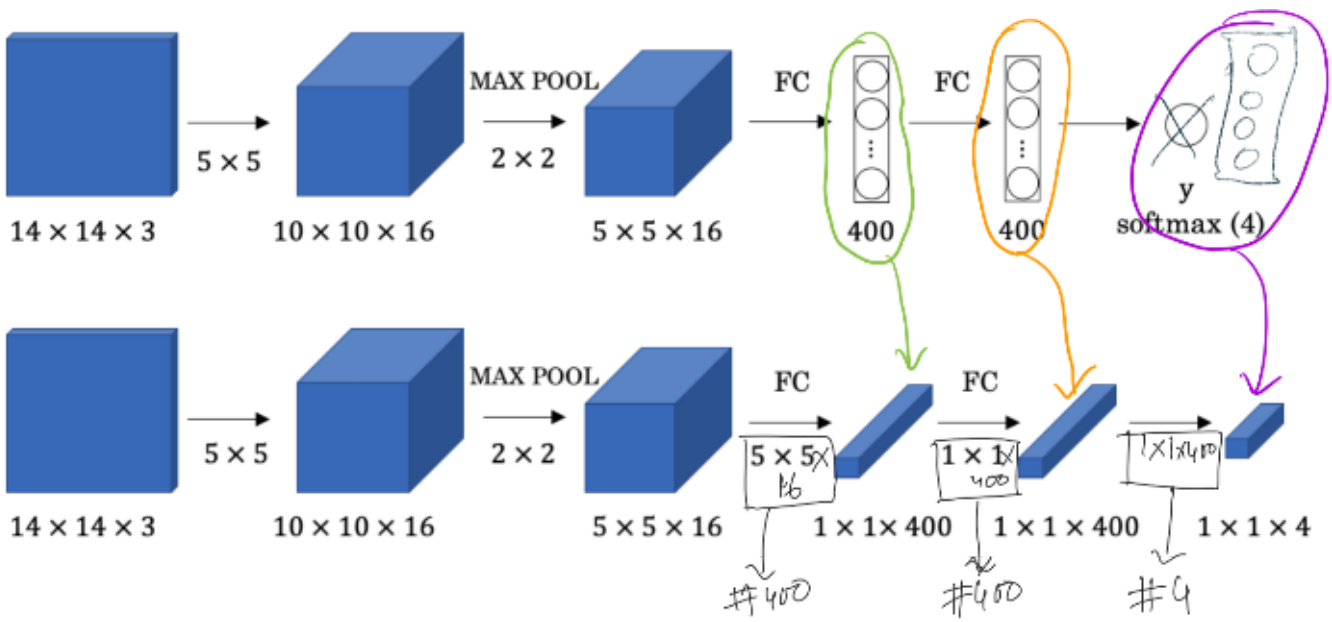


Larger sliding window.

Even larger sliding window size

The computational cost of this model is much higer. We are are using different sliding window size that makes it more costly. However we can reduce the cost by increasing the stride but the model will be inefficient in that case as we might miss some portion of the image.

Turning FC layer into convolutional layers:-



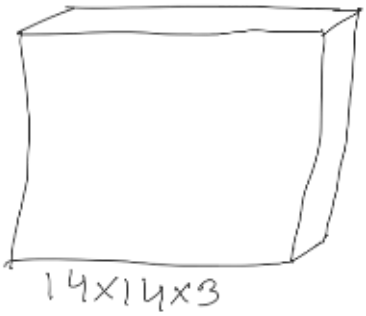
Convolution implementation of sliding window:-

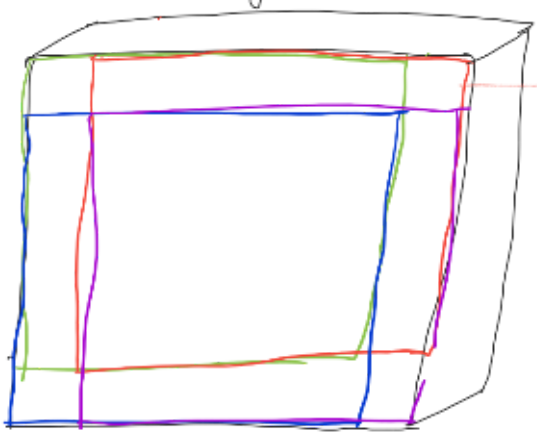
Sliding window of nearby region calculates duplicate information. We can implement convolutional network instead of sliding window to remove the duplicate calculation

For Example:-

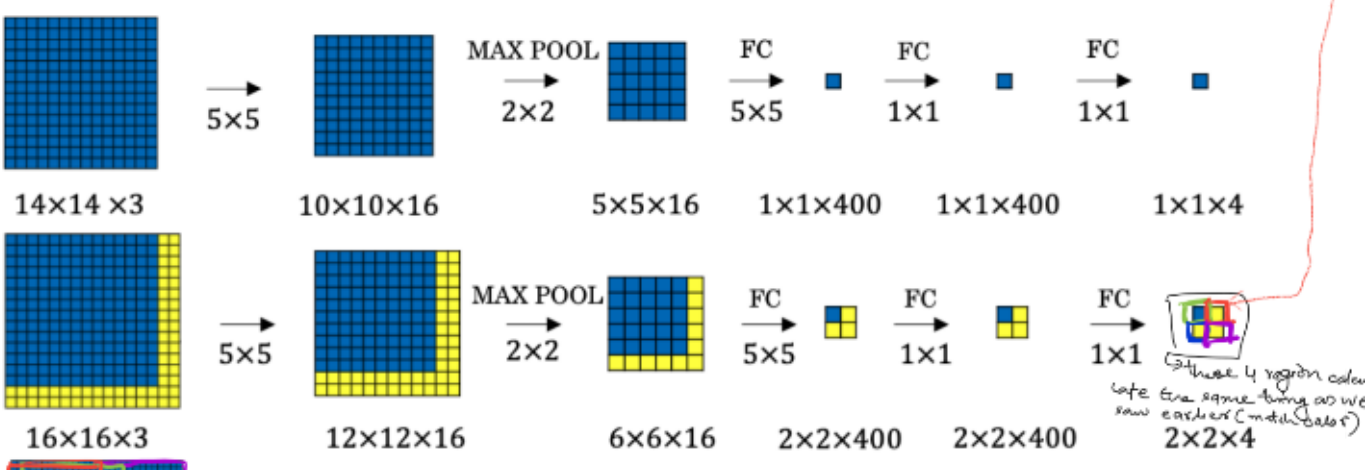
Let,

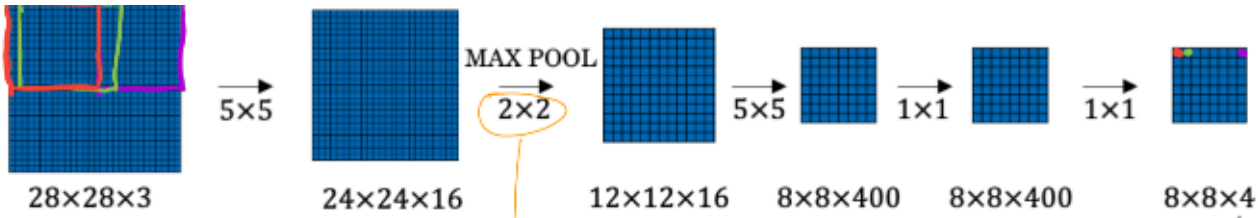
Our trainset is 14x14x3 image. | test set is 16x16x3 so we will use a sliding window of 14x14x3.





As we can see we have 4 different region after implementing 14x14x3 sliding window. And most of them share the same region so that leads duplicate calculations. It doesn't seems to be a problem if our sliding window & test set size doesn't differs much. For the above





If we use stride of two & sliding window of 14x14x3 we get 8x8 result. The red section is the first sliding window the result of that is shown in first block of 8x8x4 matrix. which is calculated in a convolution way.

the problem with this approach is the position of the bounding boxes aren't accurate.

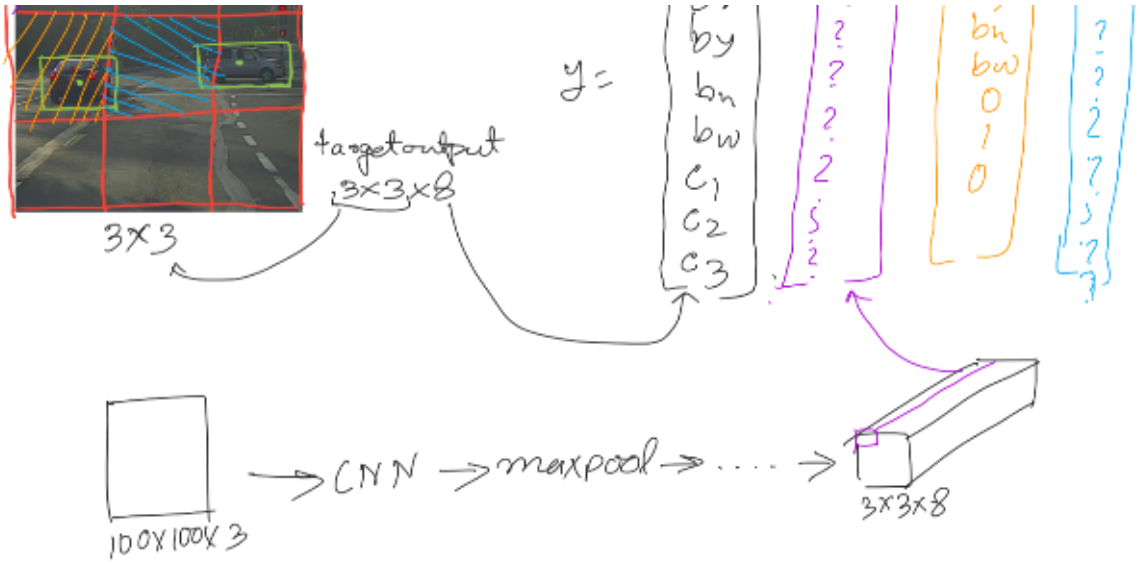
YOLO algorithm (You only look once):-

yolo algorithm solves the previous approach problem. It correctly provide the bounding boxes.

Yolo first make a grid from the picture shown below we show 3x3 grid but in practice the grid size is more dense (19x19). Each of these grid will be evaluated by object detection algorithm as we show in the first algorithm. In the training set if a single object stays in two grid then the object is assigned to the grid where ever the middle point of bounding box reside. The main problem with this approach is if any grid contains more than a object then it cannot detect that

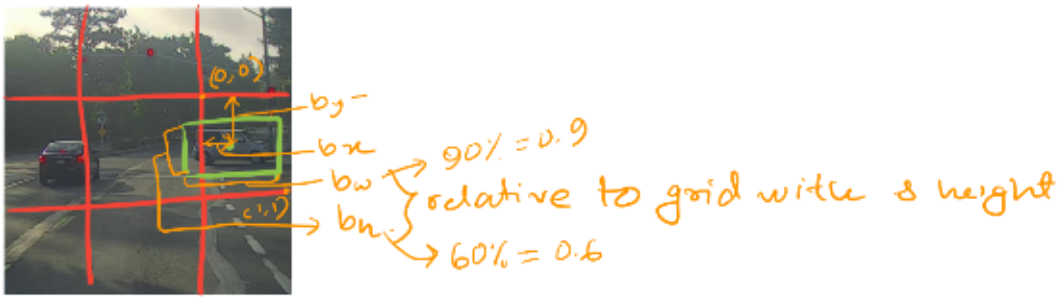
Label for training each grid cell:

$$\begin{bmatrix} P_c \\ b_x \end{bmatrix} \quad \begin{bmatrix} 0 \\ ? \\ ? \end{bmatrix} \quad \begin{bmatrix} 1 \\ b_x \\ b_y \end{bmatrix} \quad \begin{bmatrix} 0 \\ ? \end{bmatrix}$$

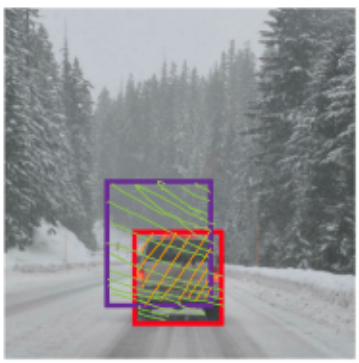


Specify the bounding box:

Bounding boxes is calculated relatively on the grids not on the whole picture.



Evaluating object localization: -



red is the ground truth mask
purple " - detected bounding box.

$$\text{Insertion over union (IoU)} = \frac{\text{size of intersection}}{\text{size of union}}.$$

$$= \frac{\text{▨}}{\text{▨}}$$

"correct" if $IoU \geq 0.5$

Non max suppression:-

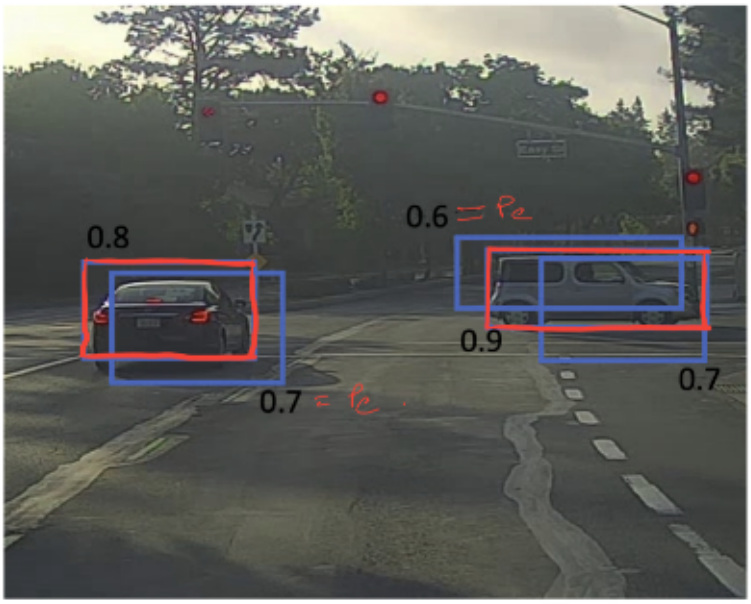
the problem with making this grid approach is any object

can be identified multiple times.

19×19

portion of this car is present in 20 grids. So the car will be recognized 20 times.

Let multiple grid recognizes same car. Each of the grid the their confidence level (P_c) associated with it.



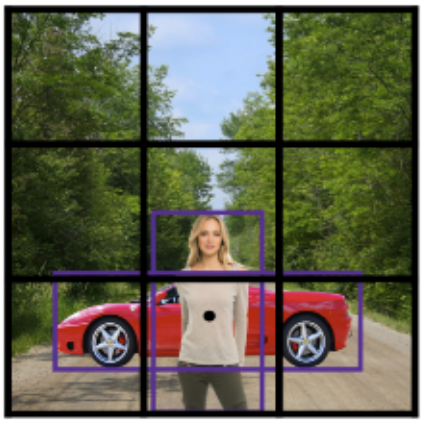
The grid with highest P_c will be considered of having object. The other overlapping grid having larger IoU will be suppressed.

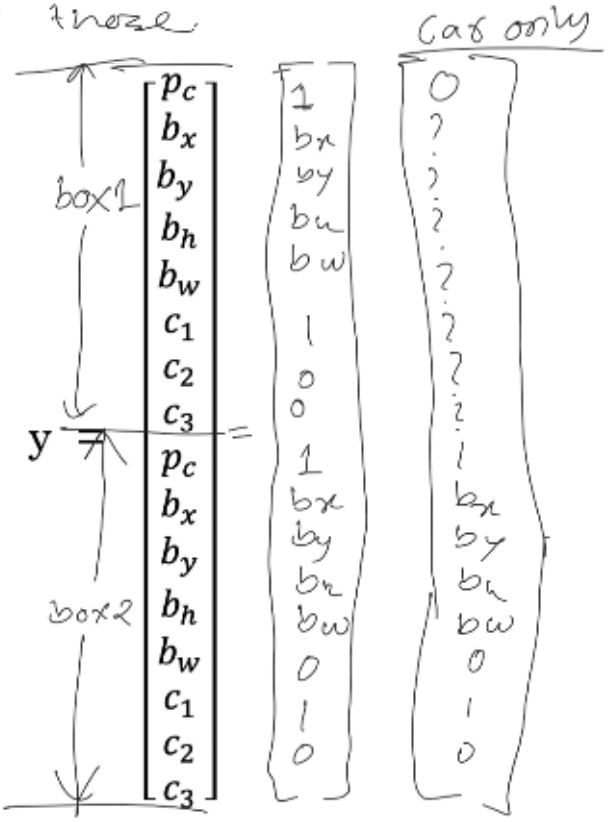
Anchor boxes:-

if we have multiple object in a same grid then

its not possible to detect them with previous approach

Anchor boxes can help to detect these





Anchor box 1: **Anchor box 2:**

→ If same achorbox object (two human) present in the same grid box then this algorithm cannot detect them.

→ If there are three object in the same grid cell this algorithm won't handle that.

→ People choose the size & shape of the anchor boxes by hand

→ we can also use k-means algorithm to choose anchor box